Deterministic Broadcasting Time in Radio Networks of Unknown Topology

Dariusz R. Kowalski *
Instytut Informatyki
Uniwersytet Warszawski
Banacha 2, 02-097 Warszawa, Poland
darek@mimuw.edu.pl

Andrzej Pelc †
Département d'Informatique
Université du Québec à Hull
Hull, Québec J8X 3X7, Canada
pelc@uqah.uquebec.ca

Abstract

In a seminal paper [3], Bar-Yehuda, Goldreich and Itai considered broadcasting in radio networks whose nodes know only their own label and labels of their neighbors. They claimed a linear lower bound on the time of deterministic broadcasting in such radio networks, by constructing a class of graphs of diameter 3, with the property that every broadcasting algorithm requires linear time on one of these graphs. Due to a subtle error in the argument, this result is incorrect. We construct an algorithm that broadcasts in logarithmic time on all graphs from [3]. Moreover, we show how to broadcast in sublinear time on all n-node graphs of diameter $o(\log \log n)$. On the other hand, we construct a class of graphs of diameter 4, such that every broadcasting algorithm requires time $\Omega(\sqrt[4]{n})$ on one of these graphs. In view of the randomized algorithm from [3], runnning in expected time $\mathcal{O}(D \log n + \log^2 n)$ on all n-node graphs of diameter D, our lower bound gives the first correct proof of an exponential gap between determinism and randomization in the time of radio broadcasting.

1 Introduction

A radio network is modeled as an undirected connected graph whose nodes are transmitter-receiver devices. An edge e between two nodes means that the transmitter of one end of e can reach the other end. Nodes send messages in synchronous *steps* (time slots), measured by a global clock which indicates the current step number. In every step every node acts either as a *transmitter* or as a *receiver*. A node acting as a transmitter sends a message which can potentially reach all of its neighbors. A node acting as a receiver in a given step gets a message, if and only if, exactly one of its neighbors transmits in this step. The message received in this case is the one that was transmitted. If at least two neighbors v and v' of u transmit simultaneously in a given step, none of the messages is received by u in this step. In this case we say that a *collision* occurred at u. It is assumed that the effect at node u of more than one of its neighbors transmitting is the same as that of no neighbor transmitting, i.e., a node cannot distinguish a collision from silence.

One of the fundamental tasks in network communication is *broadcasting*. Its goal is to transmit a message from one node of the network, called the *source*, to all other nodes. Remote nodes get the source message via intermediate nodes, along paths in the network. In this paper we concentrate on one of the most important and widely studied performance parameters of a broadcasting scheme, which is the total time, i.e., the number of steps it uses to inform all the nodes of the network.

1.1 Our results

In a seminal paper [3], Bar-Yehuda, Goldreich and Itai considered broadcasting in radio networks whose nodes know only their own label and labels of their neighbors. They proved a linear lower bound on the time of deterministic broadcasting in such radio networks, by constructing a class of graphs of diameter 3, with the property that every broadcasting algorithm requires linear time on one of these graphs. Due to a subtle error in the argument, this result is incorrect. Using the same model as in [3], we construct an algorithm that broadcasts in logarithmic time on all graphs from [3]. Moreover, we show how to broadcast in sublinear time on all n-node graphs of diameter $o(\log \log n)$. On the other hand, we construct a class of graphs of diameter 4, such that every broadcasting algorithm requires time $\Omega(\sqrt[4]{n})$ on one of these graphs. In view of the randomized algorithm from [3] runnning in expected time $\mathcal{O}(D \log n + \log^2 n)$ on all n-node graphs of diameter D, our lower bound gives the first correct proof of an exponential gap between determinism and randomization in the time of radio broadcasting, which was claimed in [3].

1.2 Related work

Most of the results concerning broadcasting in radio networks can be divided into two parts: those which assume complete knowledge of the topology of the network at all nodes, or equivalently, dealing with centralized broadcasting for a given network, and those assuming only limited

*This work was done during this author's stay at the Research Chair in Distributed Computing of the Université du Québec à Hull, as a postdoctoral fellow.

†Research supported in part by NSERC grant OGP 0008136 and by the Research Chair in Distributed Computing of the Université du Québec à Hull.

knowledge of the network at all nodes and dealing with distributed broadcasting in arbitrary networks.

Deterministic centralized broadcasting assuming complete knowledge of the network was considered in [7], where a $\mathcal{O}(D \log^2 n)$-time broadcasting algorithm was given for all n-node networks of diameter D. In [14], $\mathcal{O}(D + \log^5 n)$-time broadcasting was proposed. On the other hand, in [1] the authors proved the existence of a family of n-node networks of radius 2, for which any broadcast requires time $\Omega(\log^2 n)$.

The study of deterministic distributed broadcasting in radio networks whose nodes have only limited knowledge of the topology, was initiated in [3]. The authors assumed that nodes know only their own label and labels of their neighbors. Under this scenario, a simple linear-time broadcasting algorithm based on DFS follows from [2]. In [3], the authors constructed a class of n-node graphs of diameter 3, and claimed that every broadcasting algorithm requires time $\Omega(n)$ on one of these graphs. In [15] this result was further strengthened by claiming that every broadcasting algorithm requires time $n - 1$ on one of these graphs. It follows from the present paper that both these results concerning lower bounds are false.

Many authors [4, 5, 8, 9, 10, 11, 12] studied deterministic distributed broadcasting in radio networks under an even weaker assumption that nodes know only their own label (but not labels of their neighbors). In [8] the authors gave a broadcasting algorithm working in time $\mathcal{O}(n)$ for arbitrary n-node networks, assuming that nodes can transmit spontaneously, before getting the source message. In [8, 9, 10, 11, 12] the model of directed graphs was used. Increasingly faster broadcasting algorithms working on arbitrary n-node (directed) radio networks were constructed, the currently fastest being the $\mathcal{O}(n \log^2 n)$-time algorithm from [10]. In [11], a lower bound $\Omega(n \log D)$ on broadcasting time for networks of radius D was shown. On the other hand, in [4, 5, 9, 11] the problem was to find efficient broadcasting algorithms on radio networks of maximum indegree Δ.

Finally, randomized broadcasting algorithms in radio networks were studied, e.g., in [3, 18]. For these algorithms no topological knowledge of the network was assumed. In [3] the authors showed a randomized broadcasting algorithm running in expected time $\mathcal{O}(D \log n + \log^2 n)$. In [18] it was shown that for any randomized broadcasting algorithm and parameters $D \leq n$, there exists an n-node network of diameter D requiring expected time $\Omega(D \log(n/D))$ to execute this algorithm. It should be noted that the lower bound $\Omega(\log^2 n)$ from [1], for some networks of radius 2, holds for randomized algorithms as well.

2 Model and terminology

We consider undirected graphs whose nodes have distinct labels belonging to the set $\{0, 1, ..., r\}$, where r is polynomial in the number of nodes and is known to all of them. A distinguished node with label 0 is called the *source*. We denote by D the *radius* of the graph, i.e., the distance from the source to the farthest node. (For undirected graphs, the diameter is of the order of the radius.) The jth *layer* of a graph is the set of nodes at distance j from the source. We adopt the communication model used by Bar-Yehuda, Goldreich and Itai [3]. It is summarized in the following definition.

Definition 2.1 [3] *A* broadcast protocol *for radio networks is a multiprocessor (multinode) protocol, the execution of which proceeds in steps (time-slots) (numbered* $0, 1, \ldots$*) as follows:*

1. *In the initial step* 0*, only the source transmits a message called the* source message*.*
2. *In each step each node acts either as a transmitter or as a receiver (or is inactive).*
3. *A node receives a message in a specific step if and only if it acts as a receiver in this step and exactly one of its neighbors acts as a transmitter in that step. The message received in this case is the message transmitted by that neighbor.*
4. *The action of node in a specific step is determined as a function of its initial input (which consists of its own label and the labels of its neighbors) and the (sequence of) messages that it has received in previous steps. All nodes have identical copies of the same program.*
5. *A node may act as a transmitter in a step* > 0 *only if it has received a message in a previous time-slot (there are no "spontaneous" transmissions of nodes other than the source in step 0).*
6. *The broadcast is completed at step* t*, if all nodes have received the source message at one of the steps* $0, 1, \ldots, t$*.*

As in [3], we assume that a node cannot distinguish whether more than one neighbor or no neighbor transmitted in a given step, i.e., we work in the model without collision detection.

3 Logarithmic broadcasting in BGI networks

In [3] the following class of $(n + 2)$-node networks was defined. Let S be a nonempty subset of $\{1, ..., n\}$. The network G_S is a graph (of radius 2) whose nodes are labeled $0, 1, ..., n + 1$. The set of edges of G_S is $E = \{(0, i) : 1 \leq i \leq n\} \cup \{(i, n + 1) : i \in S\}$. Node 0 is the source and node $n + 1$ (the only node in layer 2) is called the *sink*. We refer to all networks G_S as BGI-n networks (see Figure 1).

It was claimed in [3] that for any broadcast protocol there is a BGI-n network on which this protocol works in time $\Omega(n)$. However, the proof in [3] contains the following flaw. The authors fix *predetermined* sets of nodes transmitting in consecutive steps (moves M_i in the hitting game are used as input in procedure find_set), and then construct a network in which some node is not informed during any of these steps. However, they fail to notice that during the broadcasting process, the source may potentially acquire some information, which it may pass to other nodes, thus modifying the sets of nodes transmitting in subsequent steps. So, in fact,

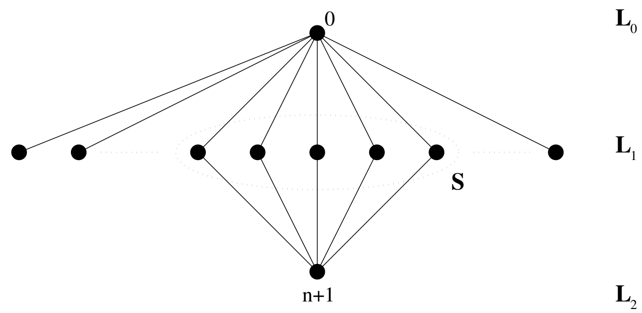

Figure 1. BGI-n **network** G_S.

the proof from [3] works only for oblivious algorithms, in which sets of nodes transmitting in a given step must be fixed in advance.

It turns out that not only the argument from [3] is erroneous but, in fact, the above result itself is incorrect. Below we give an algorithm that broadcasts in all BGI-n networks in time $\mathcal{O}(\log n)$. The technique of selecting one out of many simultaneously transmitting neighbors, which is the main ingredient of this algorithm, will be further used in the construction of a much more involved algorithm which guarantees fast broadcasting in arbitrary networks of small radius.

The main idea of our algorithm is to simulate the collision detection capability in some nodes of the network. (Recall that collision detection is not *a priori* available in our model). In order to get logarithmic broadcasting for BGI-n networks, it is enough to simulate collision detection at the source. To get sublinear broadcasting in all networks of radius $o(\log\log n)$, we will need to simulate this capability in many nodes from different levels.

Let A be a set of neighbors of the source (possibly unknown to it), and let $i \notin A$ be another neighbor of the source. Suppose that nodes in A want to transmit. Our goal is to let the source distinguish whether A has 0,1, or more than 1 element. This can be done using the following 2-step procedure:

Procedure Echo (i, A)

Step 1. Every node in A transmits its label.
Step 2. Every node in $A \cup \{i\}$ transmits its label.

There are 3 possible effects of Procedure Echo (i, A) at the source:

Case 1. A message is received in step 1 and no message in step 2. In this case the source knows that A has 1 node and knows the label of this unique node.
Case 2. No message is received in step 1 and a message (from i) is received in step 2. In this case the source knows that A is empty.
Case 3. No message is received in either step. In this case the source knows that A has at least 2 nodes.

Procedure Echo is used to select one node in the set S of nodes connected to the sink, in a BGI-n network G_S. Once such a unique node is selected and transmits, the sink receives the source message, and broadcast is completed. This is done using the following algorithm. (In the original definition of BGI-n networks, nodes are labeled by consecutive numbers $0, 1, ..., n+1$ but we formulate our algorithm in the more general case, when labels are chosen arbitrarily from a set $\{0, 1, ..., r\}$, where r is polynomial in the number of nodes. W.l.o.g. assume that r is a power of 2.)

Algorithm Binary-Selection-Broadcast

In step 0, the source transmits the source message and the lowest label i of its neighbor (in the original definition of BGI-n networks, $i = 1$). In step 1, node with label i transmits the source message and its degree. If this degree is 2 ($i \in S$), the sink receives the message and broadcast stops. Assume that the degree of i is 1.

All remaining steps 2,3,... are divided into segments of length 3. In the first step of each segment, the source transmits a range R of labels, and orders the execution of Procedure Echo $(i, R \cap S)$ during the last two steps of the segment. (Notice that all nodes from layer 1 know if they are in $R \cap S$.) In the first segment, $R := \{1, ..., r/2\}$. If a range $R = \{x, ..., y\}$ is transmitted in a given segment, the range to transmit in the next segment is chosen according to the three possible effects of Echo $(i, R \cap S)$, described above. In case 1, the sink is informed and broadcast stops. In case 2, $R := \{y + 1, \ldots, y + (y - x + 1)/2\}$. In case 3, $R := \{x, \ldots, (y + x - 1)/2\}$.

□

Theorem 3.1 *Algorithm Binary-Selection-Broadcast completes broadcasting in any BGI-n network in time $\mathcal{O}(\log n)$.*

Proof: Since the size of R transmitted in the ith segment is $r/2^i$, after at most $\log r \in \mathcal{O}(\log n)$ steps, the set $R \cap S$ contains exactly 1 node and hence broadcast is completed in time $\mathcal{O}(\log n)$.

□

4 Sublinear time broadcasting in networks of radius $o(\log\log n)$

In this section we construct a broadcasting algorithm working in time $o(n)$ on all n-node networks of radius $o(\log\log n)$. We will use the following results from the literature. The following two theorems assume that nodes know parameters r and d but do not assume any knowledge of the network topology.

Theorem 4.1 [11] *Consider a radio network modeled by an arbitrary graph (V, E), where V is a subset of $\{1, ..., r\}$. Let A and B be a partition of V, such that all nodes in A have the same message m. Then there exists a protocol working in time $\mathcal{O}(min(r, d\log(r/d)))$, which makes message m known to all nodes $v \in B$ having at least one and at most d neighbors in A.*

Theorem 4.2 [13, 11] *Given a radio network modeled by an arbitrary graph (V, E), where V is a subset of $\{1, ..., r\}$, and in which every node has a (possibly different) message, there exists a protocol working in time $\mathcal{O}(min(r, d^2\log r))$, upon the completion of which, every node of degree at most d learns messages of all its neighbors.*

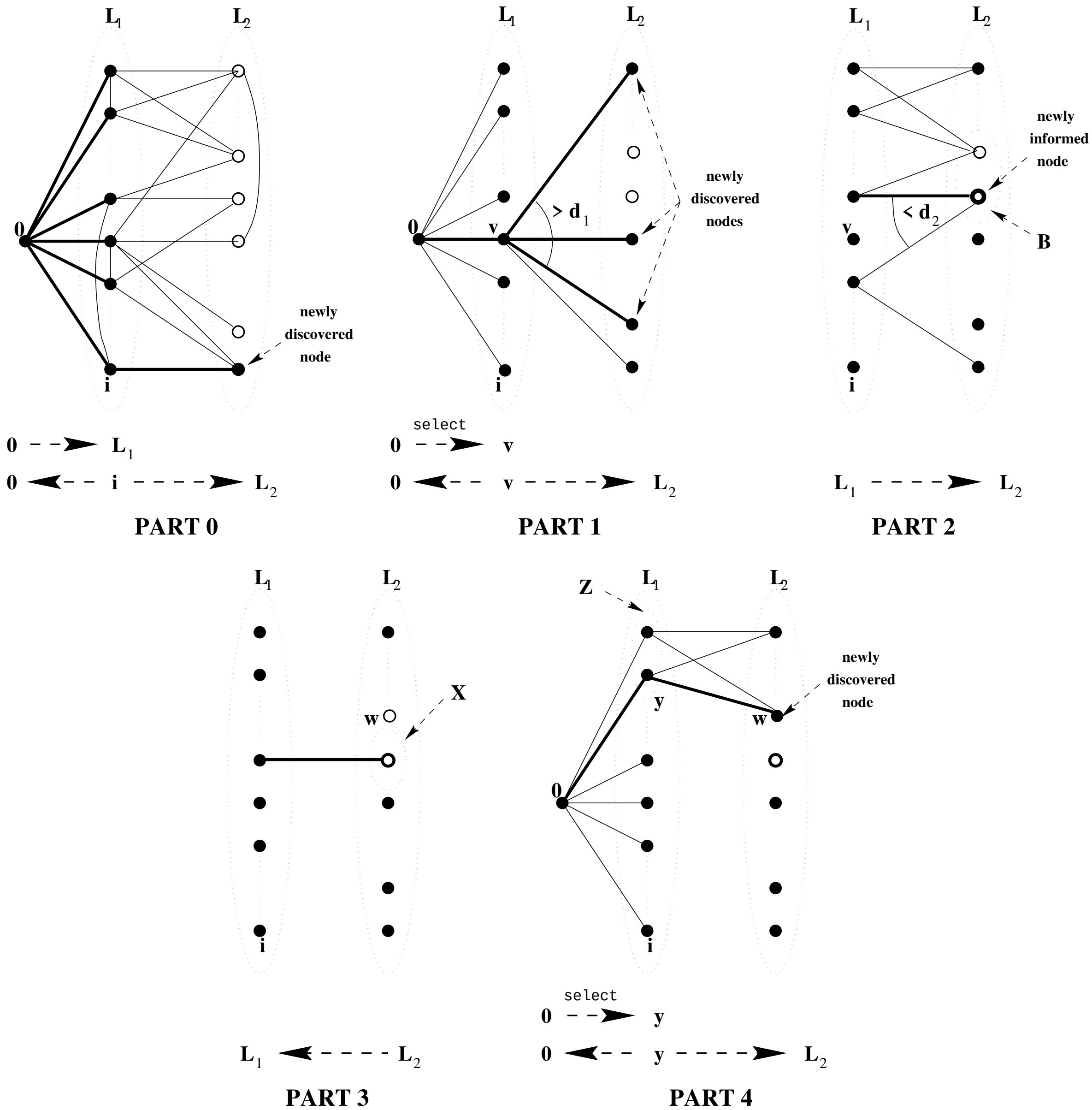

Figure 2. Algorithm $\mathcal{A}_2$.

It should be mentioned that, while the protocols in the above theorems were obtained in a non-constructive way, constructive counterparts of both these results (involving polynomial time local computations) are known, and yield only slightly slower protocols. A constructive counterpart of Theorem 4.1 yielding a $\mathcal{O}(\min(r, d \cdot 2^{(\log\log(r/d))^{\mathcal{O}(1)}}))$-time protocol follows from [16, 19], and a constructive counterpart of Theorem 4.2 yielding a $\mathcal{O}(\min(r, d^2 \log^2 r))$-time protocol follows from [17]. In our algorithm we use protocols from Theorems 4.1 and 4.2 but these constructive counterparts could be used as well, and our resulting broadcasting algorithm would still work in sublinear time.

The next result refers to radio networks of known topology.

Theorem 4.3 **[7]** *Consider a radio network modeled by an arbitrary graph (V, E), where V is a subset of $\{1, ..., r\}$, and assume that all nodes know the topology of the graph. Let A and B be a partition of V, such that all nodes in A have the same message m. Then there exists a protocol working in time $\mathcal{O}(\log^2 r)$, which makes message m known to all nodes $v \in B$ having a neighbor in A.*

4.1 Broadcasting in networks of radius 2

We first describe a sublinear time broadcasting algorithm, working for all networks of radius 2. More generally, in every network G, this algorithm informs all nodes in levels 1 and 2 in sublinear time. At each step of the execution, the source maintains a set DIS of *discovered* nodes: those about which it knows that they received the source message. The algorithm uses the polynomial upper bound r on the number of nodes, and two parameters d_1 and d_2, whose values will be specified later.

Algorithm $\mathcal{A}_2$ (see Figure 2)

Part 0. In step 0, the source transmits the source message, the set L_1 and the lowest label $i \in L_1$. In step 1, node with label i transmits the source message and the set C of its neighbors in L_2. The source sets the set DIS of discovered nodes to $\{0\} \cup L_1 \cup C$ and transmits it in step 2.
Part 1. Using a similar mechanism as in Algorithm Binary-Selection-Broadcast, the source selects a node in L_1 for which the number of undiscovered neighbors in L_2 is maximum. (Every node in L_1 can distinguish its neighbors in L_1 and in L_2). In the next step the selected node transmits the source message and the set of its neighbors in L_2. The source adds them to discovered nodes and transmits the updated set DIS. This process is repeated until there are no nodes in L_1 with more than d_1 undiscovered neighbors in L_2.
Part 2. Apply the protocol from Theorem 4.1 (with $d = d_2$) to the subgraph of G induced by $L_1 \cup B$ and to the partition (L_1, B), where B is the set of undiscovered nodes in L_2. This protocol makes the source message known to all undiscovered nodes from L_2 which have at most d_2 neighbors in L_1.
Part 3. Let X be the set of all nodes from L_2 which received the source message during Part 2 and have at most d_2 neighbors in L_1. Apply the protocol from Theorem 4.2 (for $d = d_1$) to the subgraph of G induced by $X \cup L_1$. The message transmitted by each node contains its label and the source message.
Part 4. All nodes from L_1 check if all their neighbors in L_2 got the source message. Nodes selected in Part 1 know that all their neighbors in L_2 were informed and discovered. Let Z be the set of nodes in L_1 which did not get a message in Part 3 from some of their undiscovered neighbors in L_2. Using a similar mechanism as in Algorithm Binary-Selection-Broadcast, the source selects one node in Z, then this node transmits (alone) the source message and the set of all its neighbors in L_2, and the source adds these neighbors to the set DIS of discovered nodes. After each selection, at least one currently undiscovered node gets the source message. This process continues until all nodes in L_1 know that all their neighbors in L_2 received the source message. □

Theorem 4.4 *Algorithm $\mathcal{A}_2$ completes broadcasting in any n-node network of radius 2 in time $\mathcal{O}(n^{4/5} \log n)$.*

Proof: Correctness is straighforward. The complexity of the algorithm is estimated as follows. Part 0 takes 3 steps. Each selection in Part 1 takes $\mathcal{O}(\log r)$ steps and there are at most n/d_1 selections, hence the entire Part 1 takes $\mathcal{O}((n/d_1) \cdot \log r)$ steps. In view of Theorem 4.1, Part 2 takes $\mathcal{O}(d_2 \cdot \log r)$ steps. In view of Theorem 4.2, Part 3 takes $\mathcal{O}(d_1^2 \log r)$ steps. Each selection in Part 4 takes $\mathcal{O}(\log r)$ steps and there are at most nd_1/d_2 selections, hence the entire Part 4 takes $\mathcal{O}((nd_1/d_2) \cdot \log r)$ steps. Taking $d_1 = \sqrt[5]{n}$ and $d_2 = \sqrt[5]{n^2}$, this totals to $\mathcal{O}(n^{4/5} \log r) = \mathcal{O}(n^{4/5} \log n)$ steps. □

4.2 Extension to arbitrary networks of radius $o(\log \log n)$

Our next result is an algorithm which broadcasts in sublinear time in arbitrary networks of radius $o(\log \log n)$. This algorithm is significantly more involved than $\mathcal{A}_2$, because, in general, a selection of one transmitting node from a layer distant from the source cannot be directly controlled by one node. In order to perform such a selection, a cascade of selection procedures between consecutive layers must be constructed.

Theorem 4.5 *There exists an algorithm broadcasting in time $o(n)$, for all n-node radio networks of radius $o(\log \log n)$.*

The description and the analysis of the algorithm will appear in the full version of the paper.

5 Lower bound

In this section we show that for every deterministic broadcasting algorithm $\mathcal{A}$, there exists a network $G_{\mathcal{A}}$ of radius 2, with at most $2n$ nodes, on which $\mathcal{A}$ requires $\Omega(\sqrt[4]{n})$ steps to complete broadcast. This network is chosen from the family $\mathcal{C}_n$ of networks defined as follows. Every graph $G \in \mathcal{C}_n$ (see Figure 3) consists of the source 0 and two layers $L_1 = \{1, \ldots, n\}$ and $L_2 = \{n+1, \ldots, n+q\}$, where q is the largest odd integer smaller than $\sqrt[4]{n}$. The source is adjacent to all nodes in L_1, and every node in L_1 is adjacent to exactly one node in L_2. These are the only edges in G. If G is fixed, we denote by V the set $\{0\} \cup L_1 \cup L_2$ of all nodes of G. For every node v, we denote by N_v the set of its neighbors.

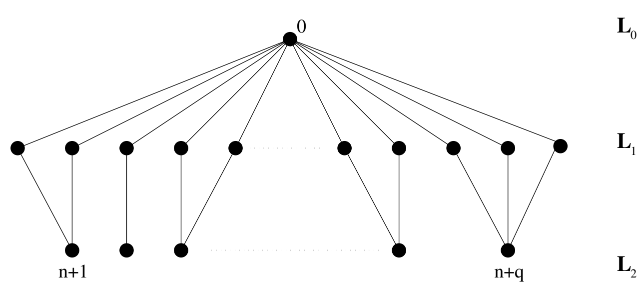

Figure 3. Network $G \in \mathcal{C}_n$.

The idea of the proof is the following. We construct the network step-by-step, using consecutive steps of the fixed broadcasting algorithm $\mathcal{A}$, and *assuming* that particular nodes got particular messages in given steps. In order to express this, we use the notion of *abstract history* of a node, formally defined below. Intuitively, an abstract history of a node v at a given step k consists of a neighborhood of node v and of a sequence of messages received by this node until step k. Since the network is not yet constructed, neighborhoods of some nodes are not determined by step k, and consequently it is not yet known which abstract history will become the real one — the one given by algorithm $\mathcal{A}$ running on the final network. We can ensure that, if a given node had some abstract history up to a certain step, then it would behave in a given way (this is captured by the notion of abstract action function, defined below). Based on that we do the next step of the construction of the network (by determining neighborhoods of two nodes in layer L_2), and simultaneously define abstract histories of nodes in this step. These abstract histories are defined so as to prevent some nodes in layer L_2 of the network from getting any message for a long time. When the construction is finished, we prove that if the algorithm $\mathcal{A}$ runs on the resulting network then the real histories of all nodes are identical to the abstract (assumed) ones, and consequently, some nodes of layer L_2 will indeed fail to receive the source message for $\Omega(\sqrt[4]{n})$ steps.

Fix a deterministic broadcasting algorithm $\mathcal{A}$. For this algorithm, running on any network, we define the following objects.

Histories and message format. H_k denotes the history of computation of algorithm $\mathcal{A}$ until the end of step k. This is the set $\{H_k(v) : v \in V\}$, where $H_k(v)$ is the history of computation at node v, until the end of step k. For any v and k, $H_k(v)$ is a sequence of *messages* $(M_0(v), M_1(v), \ldots, M_k(v))$. Messages are defined inductively, as follows. $M_0(v)$ is either the triple $(\emptyset, \emptyset, \emptyset)$, called the *empty message*, or the triple $(0, N_0, source\_message)$. $M_l(v)$ (for $l = 1, \ldots, k$) is the empty message if node v did not get any message in step l. Otherwise, it is a triple consisting of:

- the label of node w from which node v received a message in step l,
- the set N_w,
- history $H_{l-1}(w)$.

Notice that we restrict attention to messages conveying the entire history of the transmitter. If a particular protocol requires transmitting specific information, the receiver can deduce this information from the received history, since programs of all nodes are the same. History $H_k(v)$ containing only empty messages is called the *empty history*.

Action function and sets of transmitters. Given algorithm $\mathcal{A}$, we denote by $\pi(v, N_v, H_{k-1}(v))$ the action of node v in step k, if its set of neighbors is N_v and its history until the end of step $k-1$ is $H_{k-1}(v)$. The values of the function π can be 1 or 0: if the value is 1, node v is sending the message $(v, N(v), H_{k-1}(v))$ in step k, otherwise it is receiving in step k. Under a fixed history H_{k-1}, we define the set of neighbors of v transmitting in step k as follows: $T_k(v) = \{w \in N_v : \pi(w, N_w, H_{k-1}(w)) = 1\}$.

We construct a network $G_{\mathcal{A}} \in \mathcal{C}_n$ on which $\mathcal{A}$ will work inefficiently. The construction is by induction on steps of the algorithm $\mathcal{A}$. The set of nodes $\{0\} \cup L_1 \cup L_2$, where $L_1 = \{1, \ldots, n\}$ and $L_2 = \{n+1, \ldots, n+q\}$, as well as all edges $\{(0, i) : i = 1, \ldots, n\}$ are given in the beginning. At each step of the induction some edges between nodes from L_1 and L_2 are added. Since at each stage of the construction, only a part of the network $G_{\mathcal{A}}$ is specified, the new edges are constructed using algorithm $\mathcal{A}$ and some *abstract history* $\hat{H}_k$ until this step. Abstract histories at each node are parametrized with a nonempty set A representing a possible neighborhood of v to be constructed in a later step.

Abstract objects. Let $v \in V$ and $A \subseteq V$. An abstract history $\hat{H}_k(v, A)$ of node v, assuming that its neighborhood is A, is defined as a sequence $(\hat{M}_0(v, A), \hat{M}_1(v, A), \ldots, \hat{M}_k(v, A))$ of *abstract messages*. $\hat{M}_0(v, A) = M_0(v)$, and $\hat{M}_l(v, A)$, for $l > 0$, is either the empty message or is of the format $(w, B, \hat{H}_{l-1}(w, B))$, for some $w \in V$ and $B \subseteq V$. We will construct the abstract history step by step, in parallel with the construction of network $G_{\mathcal{A}}$. Notice that, in general, abstract histories and abstract messages are not necessarily linked to any particular protocol.

We also define the *abstract action function* $\hat{\pi}(v, A, \hat{H}_{k-1}(v, A))$ as an extension of the action function π described above: if $\pi(v, A, \hat{H}_{k-1}(v, A))$ is defined for some v and A then $\hat{\pi}(v, A, \hat{H}_{k-1}(v, A)) = \pi(v, A, \hat{H}_{k-1}(v, A))$. Otherwise, $\hat{\pi}(v, A, \hat{H}_{k-1}(v, A)) = 0$.

We now define sets of *abstract transmitters*. First consider a node v with neighborhood N_v fixed at the end of step k of the construction, and assume that neighborhoods N_w of all nodes $w \in N_v$ are also fixed. Under a fixed abstract history $\hat{H}_{k-1}$, we define the set of abstract transmitters $\hat{T}_k(v, N_v) = \{w \in N_v : \hat{\pi}(w, N_w, \hat{H}_{k-1}(w, N_w)) = 1\}$.

Now define sets of abstract transmitters for nodes whose neighborhood is not yet fixed. Let S_k denote the set of all nodes j in L_2 for which the neighborhood N_j is not fixed until the end of step k of the construction. Let $R_k = L_1 \setminus \bigcup_{j \in L_2 \setminus S_k} N_j$. (Additionally, let $S_0 = L_2$ and $R_0 = L_1$.) For these nodes we define the sets of abstract transmitters in step k as follows:

if $v \in R_k$ then, for any $j \in S_k$, $\hat{T}_k(v, \{0, j\}) = \{0\}$ if $\hat{\pi}(0, L_1, \hat{H}_{k-1}(0, L_1)) = 1$, and $\hat{T}_k(v, \{0, j\}) = \emptyset$, otherwise;

if $v \in S_k$ and $R \subseteq L_1$ then $\hat{T}_k(v, R) = \{i \in R : \hat{\pi}(i, \{0, v\}, \hat{H}_{k-1}(i, \{0, v\})) = 1\}$.

We now describe the construction of the graph $G_{\mathcal{A}}$. We begin by defining the abstract history $\hat{H}_0$. $\hat{H}_0(v, A) = (\hat{M}_0(v, A))$, for all nodes v and sets A, where $\hat{M}_0(v, A)$ is the empty message for all $v \notin L_1$, and $\hat{M}_0(v, A) = (0, L_1, source\_message)$, for all $v \in L_1$.

In step 1 of the construction we will use the following procedure:

Procedure FIRST-STEP-SELECTION(j, R, S)

- Choose some node $j \in S$ such that the size of $X = \hat{T}_1(j, R)$ is maximal and put two nodes from X to N_j (or one if X has one element, or nothing if X is empty), then remove these nodes from R. Remove j from S.

- Modify N_j as follows:
if $N_j = \emptyset$ **then** put an arbitrary $i \in R$ to N_j and remove i from R;
while there exists a node $i \in R$ such that $\hat{T}_1(v, R) = \{i\}$, for some $v \in S$
do put i into N_j and remove i from R.

Step 1 of the construction

The goal of step 1 of the construction is choosing two nodes j_1' and j_1 in L_2, together with their neighborhoods, in such a way that if some node from R_1 transmits in the first step of algorithm $\mathcal{A}$ then at least two nodes from $N_{j_1'} \cup N_{j_1}$ also do, and hence node 0 does not receive any message. This property is essential for defining $\hat{H}_1(0, L_1)$.

1. Initialize $R := L_1$ and $S := L_2$.

2. Execute FIRST-STEP-SELECTION(j_1', R, S);
$R_1' := R$;
Execute FIRST-STEP-SELECTION(j_1, R, S);
$R_1 := R$ and $S_1 := S$.

3. We construct the abstract history $\hat{H}_1$. History $\hat{H}_0$ is fixed, hence it is enough to define $\hat{M}_1(v, A)$, for all v, A. If $\hat{\pi}(v, A, \hat{H}_0(v, A)) = 1$ then $\hat{M}_1(v, A)$ is the empty message (transmitting nodes should not receive messages). Otherwise, $\hat{M}_1(v, A)$ is defined as follows:

   - Since nodes in $(L_1 \setminus R_1) \cup (L_2 \setminus S_1)$ have fixed neighborhoods, and also their neighbors have fixed neighborhoods, for each $v \in (L_1 \setminus R_1) \cup (L_2 \setminus S_1)$ we define $\hat{M}_1(v, N_v) = (w, N_w, \hat{H}_0(w, N_w))$, if $\hat{T}_1(v, N_v) = \{w\}$, and we define $\hat{M}_1(v, A)$ as the empty message in all other cases.
   - We define $\hat{M}_1(v, A)$ as the empty message, for all nodes $v \in S_1$ and all sets A.
   - We define $\hat{M}_1(0, L_1)$ as follows:
     - if $|\hat{T}_1(j_1', N_{j_1'}) \cup \hat{T}_1(j_1, N_{j_1})| \neq 1$ then $\hat{M}_1(0, L_1)$ is empty;
     - if $\hat{T}_1(j_1', N_{j_1'}) \cup \hat{T}_1(j_1, N_{j_1}) = \{i\}$ then $\hat{M}_1(0, L_1) = (i, N_i, \hat{H}_0(i, N_i))$. (Note that N_i is already defined at this point).

     $\hat{M}_1(0, A)$ is defined as the empty message, for all $A \neq L_1$.
   - For every $v \in R_1$ and $j \in S_1$, if $\hat{T}_1(v, \{0, j\}) = \{0\}$ then $\hat{M}_1(v, \{0, j\}) = (0, L_1, \hat{H}_0(0, L_1))$, and $\hat{M}_1(v, A)$ is the empty message in all other cases.

□

This concludes the first step of the construction. For any $k \geq 1$, the following invariant will be preserved after step k of the construction.

Invariant after step k

The following objects are defined:
sets $S_l \subseteq L_2$, for $l = 0, 1, ..., k$;
sets $R_l \subseteq L_1$, for $l = 0, 1, ..., k$;
sets $R_l' \subseteq L_1$, for $l = 1, ..., k$;
nodes $j_l', j_l \in S_{l-1} \setminus S_l$, for $l = 1, ..., k$;

The following properties hold:

1. Neighborhoods of nodes in $\{0\} \cup (L_1 \setminus R_k) \cup (L_2 \setminus S_k)$ are defined.
2. Histories $\hat{H}_k(v, A)$ are defined, for all nodes v and all sets A.
3. For all sets A, histories $\hat{H}_k(j, A)$ are empty for all $j \in S_k$, and histories $\hat{H}_{k-1}(j, A)$ are empty for all $j \in S_{k-1} \setminus S_k$.
4. For all nodes $j \in S_k \cup \{j_k\}$ and steps $l \leq k$, we have $|\hat{T}_l(j, R_k')| \neq 1$.
5. For all nodes $j \in S_k$ and steps $l \leq k$, we have $|\hat{T}_l(j, R_k)| \neq 1$.
6. For all nodes $j \in S_{k-1} \setminus S_k$ and steps $l < k$, we have $|\hat{T}_l(j, N_j)| \neq 1$.

□

In the description of step $k+1$ of the construction, we will use the following procedure:

Procedure $(k+1)$th-STEP-SELECTION(j, R, S)

- Choose some node $j \in S$ such that the size of $X = \hat{T}_{k+1}(j, R)$ is maximal and put two nodes from X to N_j (or one if X has one element, or nothing if X is empty), then remove these nodes from R. Remove j from S.
- Modify N_j as follows:
if $N_j = \emptyset$ **then** put an arbitrary $i \in R$ to N_j and remove i from R;
  - **set** `stop` := 0,
  - **while** `stop` = 0 **do**
    - **set** `stop` := 1,
    - **while** there exists a node $i \in R$ such that $\hat{T}_l(j', R) = \{i\}$, for some $l \leq k+1$, $j' \in S$ **do** put i into N_j and remove i from R,
    - **while** there exists a node $i \in N_j$ such that $\hat{T}_l(j, N_j) = \{i\}$, for some $l \leq k$
do find another node $i' \in \hat{T}_l(j, R)$ (if it exists), put i' into N_j and remove i' from R, **set** `stop` := 0;

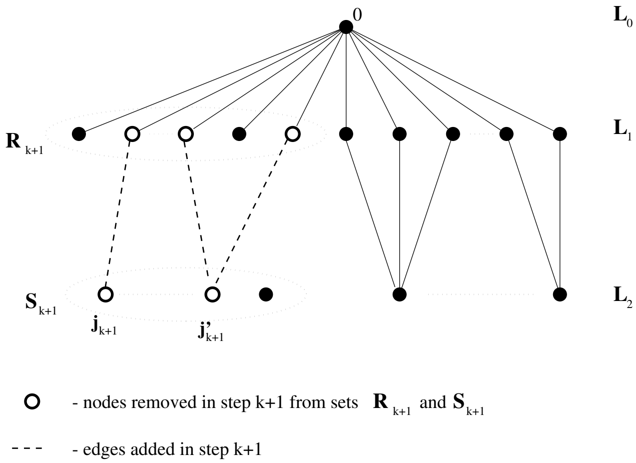

Figure 4. Step $k+1$ of the construction of $G_{\mathcal{A}} \in \mathcal{C}_n$.

Step $(k+1)$ of the construction (see Figure 4)

1. Execute $(k+1)$th-STEP-SELECTION(j'_{k+1}, R, S);
   $R'_{k+1} := R$;
   Execute $(k+1)$th-STEP-SELECTION(j_{k+1}, R, S);
   $R_{k+1} := R$ and $S_{k+1} := S$.

2. We construct the abstract history $\hat{H}_{k+1}$. History $\hat{H}_k$ is fixed, hence it is enough to define $\hat{M}_{k+1}(v, A)$, for all v, A. If $\hat{\pi}(v, A, \hat{H}_k(v, A)) = 1$ then $\hat{M}_{k+1}(v, A)$ is the empty message (transmitting nodes should not receive messages). Otherwise, $\hat{M}_{k+1}(v, A)$ is defined as follows:

   - Since nodes in $(L_1 \setminus R_{k+1}) \cup (L_2 \setminus S_{k+1})$ have fixed neighborhoods, and also their neighbors have fixed neighborhoods, for each $v \in (L_1 \setminus R_{k+1}) \cup (L_2 \setminus S_{k+1})$ we define $\hat{M}_{k+1}(v, N_v) = (w, N_w, \hat{H}_k(w, N_w))$, if $\hat{T}_{k+1}(v, N_v) = \{w\}$, and we define $\hat{M}_{k+1}(v, A)$ as the empty message in all other cases.
   - We define $\hat{M}_{k+1}(v, A)$ as the empty message, for all nodes $v \in S_{k+1}$ and all sets A.
   - We define $\hat{M}_{k+1}(0, L_1)$ as follows:
     - if $|\bigcup_{j \in L_2 \setminus S_{k+1}} \hat{T}_{k+1}(j, N_j)| \neq 1$ then $\hat{M}_{k+1}(0, L_1)$ is empty;
     - if $\bigcup_{j \in L_2 \setminus S_{k+1}} \hat{T}_{k+1}(j, N_j) = \{i\}$ then $\hat{M}_{k+1}(0, L_1) = (i, N_i, \hat{H}_k(i, N_i))$. (Note that N_i is already defined at this point).

     $\hat{M}_{k+1}(0, A)$ is defined as the empty message, for all $A \neq L_1$.
   - For every $v \in R_{k+1}$ and $j \in S_{k+1}$, if $\hat{T}_{k+1}(v, \{0, j\}) = \{0\}$ then $\hat{M}_{k+1}(v, \{0, j\}) = (0, L_1, \hat{H}_k(0, L_1))$, and $\hat{M}_{k+1}(v, A)$ is the empty message in all other cases.

□

Lemma 5.1 *The invariant after step k is preserved, for all $k \geq 1$ such that S_k and R_k are nonempty.*

Proof: The validity of the invariant after step 1 follows from the exit conditions in procedure FIRST-STEP-SELECTION(j'_1, R, S) and procedure FIRST-STEP-SELECTION(j_1, R, S). Assume that the invariant holds after step k and that S_{k+1} and R_{k+1} are nonempty. We prove that it holds after step $k+1$.

All required objects are defined by the construction in step $k+1$, using nonemptiness of S_{k+1} and R_{k+1}. It remains to prove the six properties.

1. This follows from property 1 of the invariant after step k and from the construction of j'_{k+1}, j_{k+1}, and of their neighborhoods during part 1 of the construction in step $k+1$.

2. $\hat{H}_{k+1}$ was defined in part 3 of step $(k+1)$th of the construction.

3. The fact that $\hat{H}_{k+1}(v, A)$ is empty for all nodes $v \in S_{k+1}$ and all sets A, follows from the assumption in part 2 of step $(k+1)$ of the construction. The fact that $\hat{H}_k(j'_{k+1}, A)$ and $\hat{H}_k(j_{k+1}, A)$ are empty (j'_{k+1}, j_{k+1} are the only elements of $S_k \setminus S_{k+1}$) follows from part 3 of the invariant after step k.

4. We prove that, for all nodes $j \in S_k \cup \{j_k\}$ and steps $l \leq k+1$, we have $|\hat{T}_l(j, R'_{k+1})| \neq 1$. This follows from the exit conditions of the external and of the first internal loop in procedure $(k+1)$th-STEP-SELECTION(j'_{k+1}, R, S). The execution of the external loop ends iff value `stop` becomes 1, which means that the condition in the second internal loop is always false in the last turn of the external loop. Hence the condition of the first internal loop must be false at the end of the last turn of the external loop. This implies $|\hat{T}_l(j, R'_{k+1})| \neq 1$, for all nodes $j \in S_{k+1}$ and steps $l \leq k+1$.

5. We prove that for all nodes $j \in S_{k+1}$ and steps $l \leq k+1$ we have $|\hat{T}_l(j, R_{k+1})| \neq 1$. This follows by an argument similar as above applied to procedure $(k+1)$th-STEP-SELECTION(j_{k+1}, R, S).

6. The property $|\hat{T}_l(j, N_j)| \neq 1$, for all nodes $j \in S_k \setminus S_{k+1}$ and steps $l < k+1$, follows from the exit condition of the second internal loop in procedure $(k+1)$th-STEP-SELECTION(j'_{k+1}, R, S) and procedure $(k+1)$th-STEP-SELECTION(j_{k+1}, R, S). Observe that the existence of i' in the second internal loop follows from part 5 of the invariant after step k and from (the just proven) part 4 of the invariant after step $k+1$.

□

Lemma 5.2 *The inductive construction of the network can be carried out for at least $\frac{q-1}{2}$ steps, where q is the largest odd integer smaller than $\sqrt[4]{n}$.*

Proof: Let $k \leq (q-1)/2$. Sets S_k are decreased by two nodes during one step, hence $S_k \neq \emptyset$, since $|S_0| = q$.

Claim. Sets R_k are decreased by at most $2q^3$ nodes during one step, at most q^3 for each of the chosen nodes j'_k, j_k.

This can be computed by analyzing loops in procedures kth-STEP-SELECTION(j'_k, R, S) and kth-STEP-SELECTION(j_k, R, S). Every turn of each of the internal loops increases the neighborhood $N_{j'_k}$ (resp. N_{j_k}) by at most one element, and consequently decreases R by at most one element.

Consider the first procedure. During the first internal loop, at most $kq \leq q^2/2$ nodes can be added to $N_{j'_k}$, since each action makes one set $\hat{T}_l(j, R)$ empty, where $l \leq k$, and $n+1 \leq j \leq n+q$. During the second internal loop, at most $k-1 \leq q/2$ nodes can be added to $N_{j'_k}$, since each action makes one set $\hat{T}_l(j'_k, N_{j'_k})$ of size at least 2, where $l \leq k-1$. There may be at most $k-1 \leq q/2$ executions of the external loop, since every execution of the external loop increases the size $|\hat{T}_l(j'_k, N_{j'_k})|$ to at least 2 for some $l \leq k-1$, while performing the second internal loop. When $|\hat{T}_l(j'_k, N_{j'_k})| \neq 1$, for all $l \leq k-1$, the execution of the external loop stops. Hence $N_{j'_k}$ is bounded by $(q^2/2 + q/2) \cdot (q/2) \leq q^3$ and consequently R is decreased at the rate of at most q^3 nodes per step. The same is true for the second procedure. Hence R_k is smaller than R_{k-1} by at most $2q^3$ nodes, which concludes the proof of the Claim.

Since $q < \sqrt[4]{n}$, we have $R_k \neq \emptyset$ for all $k \leq (q-1)/2$. It follows that the construction can be carried out for $(q-1)/2$ steps. □

Using Lemma 5.2, the construction of the network $G_{\mathcal{A}}$ can be now concluded as follows. All nodes in $R_{(q-1)/2}$ are made adjacent to the only node in $S_{(q-1)/2}$. It follows from the construction that $G_{\mathcal{A}}$ belongs to the class $\mathcal{C}_n$ defined in the beginning of this section.

The histories $\hat{H}_k$ in consecutive steps of the construction were abstract histories defined in order to continue the construction in subsequent steps. The next lemma shows that the actual histories $H_k(v)$ of all nodes v of network $G_{\mathcal{A}}$ obtained by running algorithm $\mathcal{A}$ on this network, are identical to abstract histories $\hat{H}_k(v, N_v)$.

The proof of the following lemma will appear in the full version of the paper.

Lemma 5.3 *Let $k \leq (q-1)/2$ be a step of the execution of algorithm $\mathcal{A}$ on network $G_{\mathcal{A}}$. Then $H_k(v) = \hat{H}_k(v, N_v)$, for all nodes v of network $G_{\mathcal{A}}$.*

Theorem 5.1 *For any deterministic broadcasting algorithm $\mathcal{A}$, there exists a network $G_{\mathcal{A}}$ of radius 2, with at most $2n$ nodes, for which this algorithm requires time $\Omega(\sqrt[4]{n})$.*

Proof: Network $G_{\mathcal{A}}$ constructed above has $n+1+q \leq 2n$ nodes, since q is the largest odd integer smaller than $\sqrt[4]{n}$. It has radius 2 by construction. Let $k = (q-1)/2$. By Lemma 5.2, S_k is nonempty. By Lemma 5.1 and part 3 of the invariant after step k, histories $\hat{H}_k(j, N_j)$ are empty, for all $j \in S_k$. By Lemma 5.3, histories $H_k(j)$ are empty, for all $j \in S_k$. Hence no node in S_k receives the source message by step k of algorithm $\mathcal{A}$. It follows that algorithm $\mathcal{A}$ requires time $\Omega(\sqrt[4]{n})$ to broadcast on network $G_{\mathcal{A}}$. □

Using the above technique we can prove the following more general result.

Corollary 5.1 *For any deterministic broadcasting algorithm $\mathcal{A}$ and any parameters $D \leq n$, there exists an n-node network of radius D, for which this algorithm requires time $\Omega(\sqrt[4]{nD^3})$.*

6 Conclusion

In this paper we studied deterministic broadcasting time in radio networks whose nodes know only their immediate neighborhood. We presented an algorithm for broadcasting in sublinear time in all networks of radius $o(\log \log n)$ and we proved a lower bound $\Omega(\sqrt[4]{n})$ on broadcasting time even in networks of radius 2. In view of the randomized algorithm from [3] runnning in expected time $\mathcal{O}(D \log n + \log^2 n)$ on all n-node graphs of diameter D, our lower bound proves an exponential gap between time of deterministic and randomized broadcasting in radio networks.

The main problem that remains open is the following. Is there a deterministic broadcasting algorithm running in sublinear time on all networks with sublinear radius, if nodes know only their immediate neighborhood? If complete knowledge of the network is available, the positive answer follows from [14].

Another open problem is to establish minimum broadcasting time for networks of constant radius. In particular, it remains open, whether our lower bound $\Omega(\sqrt[4]{n})$, established for the class $\mathcal{C}_n$ of networks (whose members have radius 2 and a fairly simple structure) can be improved. The best upper bound that we know on broadcasting time for networks of this class is $\mathcal{O}(\sqrt{n} \log n)$. This could suggest a possibility of improving our lower bound.

References

[1] N. Alon, A. Bar-Noy, N. Linial and D. Peleg, A lower bound for radio broadcast, Journal of Computer and System Sciences 43 (1991), 290-298.

[2] B. Awerbuch, A new distributed depth-first-search algorithm, Information Processing Letters 20 (1985), 147-150.

[3] R. Bar-Yehuda, O. Goldreich, and A. Itai, On the time complexity of broadcast in radio networks: an exponential gap between determinism and randomization, Journal of Computer and System Sciences 45 (1992), 104-126.

[4] D. Bruschi and M. Del Pinto, Lower bounds for the broadcast problem in mobile radio networks, Distr. Comp. 10 (1997), 129-135.

[5] I. Chlamtac and A. Faragó, Making transmission schedule immune to topology changes in multi-hop packet radio networks, IEEE/ACM Trans. on Networking 2 (1994), 23-29.

[6] I. Chlamtac and S. Kutten, On broadcasting in radio networks - problem analysis and protocol design, IEEE Transactions on Communications 33 (1985), 1240-1246.

[7] I. Chlamtac and O. Weinstein, The wave expansion approach to broadcasting in multihop radio networks, IEEE Trans. on Communications 39 (1991), 426-433.

[8] B.S. Chlebus, L. Gąsieniec, A. Gibbons, A. Pelc and W. Rytter, Deterministic broadcasting in unknown radio networks, Distributed Computing 15 (2002), 27-38.

[9] B.S. Chlebus, L. Gąsieniec, A. Östlin and J.M. Robson, Deterministic radio broadcasting, Proc. 27th Int. Coll. on Automata, Languages and Programming (ICALP'2000), LNCS 1853, 717-728.

[10] M. Chrobak, L. Gąsieniec and W. Rytter, Fast broadcasting and gossiping in radio networks, Proc. 41st Symposium on Foundations of Computer Science (FOCS'2000), 575-581.

[11] A.E.F. Clementi, A. Monti and R. Silvestri, Selective families, superimposed codes, and broadcasting on unknown radio networks, Proc. 12th Ann. ACM-SIAM Symposium on Discrete Algorithms (SODA'2001), 709-718.

[12] G. De Marco and A. Pelc, Faster broadcasting in unknown radio networks, Information Processing Letters 79 (2001), 53-56.

[13] P. Erdös, P. Frankl, and Z. Füredi, Families of finite sets in which no set is covered by the union of r others, Israel Journal of Math. 51 (1985), 79-89.

[14] I. Gaber and Y. Mansour, Broadcast in radio networks, Proc. 6th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'1995), 577-585.

[15] F.K. Hwang, The time complexity of deterministic broadcast radio networks, Discrete Applied Mathematics 60 (1995), 219-222.

[16] P. Indyk, Explicit constructions of selectors and related combinatorial structures, with applications, Proc. 13th Ann. ACM-SIAM Symposium on Discrete Algorithms (SODA'2002), 697-704.

[17] W.H. Kautz and R.R.C. Singleton, Nonrandom binary superimposed codes, IEEE Trans. on Information Theory 10 (1964), 363-377.

[18] E. Kushilevitz and Y. Mansour, An $\Omega(D \log(N/D))$ lower bound for broadcast in radio networks, SIAM J. on Computing 27 (1998), 702-712.

[19] O. Reingold, S. Vadhan and A. Widgerson, Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors, Proc. 41th Symp. on Foundations of Computer Science (FOCS'2000), 3-13.