SIAM J. COMPUT. © 1998 Society for Industrial and Applied Mathematics
Vol. 27, No. 3, pp. 702–712, June 1998 008

AN $\Omega(D \log(N/D))$ LOWER BOUND FOR BROADCAST IN RADIO NETWORKS[*]

EYAL KUSHILEVITZ[†] AND YISHAY MANSOUR[‡]

Abstract. We show that for any *randomized* broadcast protocol for radio networks, there exists a network in which the expected time to broadcast a message is $\Omega(D \log(N/D))$, where D is the diameter of the network and N is the number of nodes. This implies a tight lower bound of $\Omega(D \log N)$ for any $D \leq N^{1-\varepsilon}$, where $\varepsilon > 0$ is any constant.

Key words. radio networks, broadcast, lower bounds

AMS subject classifications. 68Q22, 68M10

PII. S0097539794279109

1. Introduction.

Traditionally, radio networks have received considerable attention due to their military significance. The growing interest in cellular telephones and wireless communication networks has reinforced the interest in radio networks. The basic feature of radio networks, which distinguishes them from other networks, is that a processor can receive a message only from a *single* neighbor at a certain time. If two (or more) neighbors of a processor transmit concurrently, then the processor would not receive either messages.

In many applications, the users of the radio network are mobile, and therefore the topology is unstable. For this reason, it is desirable for radio-networks algorithms to refrain from making assumptions about the network topology, or about the information that processors have concerning the topology. In this work we assume that none of the processors initially have any topological information, except for the size of the network and its diameter.[1] See [Tan81, Gal85, BGI92, BGI91] for a discussion on this model and related models.

We study *broadcast* protocols; those protocols are initiated by a single processor (the *originator*) that has a message M it wishes to propagate to all the other processors in the network. In many of the radio-networks applications (e.g., cellular phones) broadcast is a central primitive which is frequently used, for example, to perform a network-wide search for a user.

Bar-Yehuda, Goldreich, and Itai [BGI92] present a *randomized* broadcast algorithm, that runs in expected $O(D \log N + \log^2 N)$ time slots, where N is the number of processors in the network and D is its diameter. In contrast, they show that for any *deterministic* broadcast algorithm there are networks of constant diameter on which the algorithm needs $\Omega(N)$ time slots.

[*]Received by the editors September 8, 1994; accepted for publication (in revised form) April 3, 1996. An early version of this paper appeared in Proc. 12th ACM Symp. on Principles of Distributed Computing, ACM, New York, 1993, pp. 65–74.

http://www.siam.org/journals/sicomp/27-3/27910.html

[†]Aiken Computation Lab., Harvard University, Cambridge, MA 02138-2901. Current address: Dept. of Computer Science, Technion, Haifa, 32000, Israel (eyalk@cs.technion.ac.il). This research was supported by research contracts ONR-N0001491-J-1981 and NSF-CCR-90-07677.

[‡]Computer Science Dept., Tel-Aviv University, Ramat Aviv, Tel Aviv 69978, Israel and IBM – T. J. Watson Research Center (mansour@cs.tau.ac.il).

[1]Usually, when the topology is unstable, the diameter is unknown to the processors and only a bound on the size of the network is available. However, since we are proving a *lower bound*, this assumption only makes the result stronger.

Alon et al. [ABLP91] made the first step towards proving the optimality of the upper bound of [BGI92]. Their result can be viewed as a graph-theoretic result; they show that there exist networks of diameter $D = 3$ on which any schedule needs at least $\Omega(\log^2 N)$ time slots. This lower bound shows that there are networks on which broadcast requires this many time slots, and it matches the known upper bounds [BGI92, CW87], in the case of constant-diameter networks.

In this work we complete the picture by proving an $\Omega(D \log(N/D))$ lower bound. Our result is of a different nature; we show that for any *randomized* broadcast algorithm and parameters N and D, there is an ordering of the N processors in a network of diameter D such that the expected number of time slots, used by the algorithm, is $\Omega(D \log(N/D))$. For $D \leq N^{1-\varepsilon}$ this gives an $\Omega(D \log N)$ lower bound. Hence, it proves the tightness of the upper bound of [BGI92] for all N and $D \leq N^{1-\varepsilon}$. Moreover, the lower bound holds even if each of the N processors is allowed to use a *different* program (e.g., the processors can use their IDs). In a recent work, Gaber and Mansour [GM95] have shown that for every network, there exists a schedule whose time is $O(D + \log^5 N)$. The scheduler there needs to get the topology of the network in advance, in order to build the schedule. The result of [GM95] shows that the lower bound presented here must rely heavily on the lack of topological knowledge at the processors.

Broadcast in radio networks has received considerable attention in previous works. [CW87] present a deterministic sequential algorithm that, given the network, finds in polynomial time a legal schedule that requires at most $O(D \log^2 N)$ time slots. Broadcast that is based on using a spanning tree was suggested in [CK85a, CK87]. In [BII93] it is shown how to reduce the amortized cost per broadcast by using a breadth-first-search (BFS) tree. Simulation of point to point networks on radio networks is found in [CK85b, ABLP92, BGI91].

An important issue in the study of radio networks is whether collisions can be detected; namely, whether a listener can distinguish between the case when none of its neighbors transmit and the case when two or more of them transmit. In our model it is assumed that the listener cannot distinguish between the two cases (say, it hears noise in both cases). There is another common model in which it is assumed that the two cases are distinguishable (say, if no neighbor transmits, the listener hears silence, while if two or more neighbors transmit, the listener hears noise). A discussion justifying both models can be found in [Gal85, BGI92]. Willard [Wil86] studies a broadcast problem in a single multiaccess channel under this second model (i.e., when collision detection is available). He shows matching upper and lower bounds of $\Theta(\log \log n)$ expected time slots[2] in this model. Our main lemma implies an $\Omega(\log n)$ lower bound for the same problem in our model. Again, this lower bound holds even if the processors use different programs. Hence, we demonstrate a provable exponential gap between these two models.

The rest of this paper is organized as follows: section 2 contains some necessary definitions. Section 3 contains the proof of the main lemma in the *uniform* case, where all the processors use the same program. Section 4 contains the proof of the main lemma in the *nonuniform* case, where processors may use different programs. The

[2]Willard shows an $\Omega(\log \log n)$ lower bound in the single multiaccess channel model. Although this bound applies to a different model, it should be noted that his bound is also significantly restricted by the types of algorithms for which it applies. In particular, he requires independence between the decision whether to transmit in a certain time slot and the decisions made in previous time slots. In our case such a restriction is unacceptable, as the upper bound of [BGI92] has such dependencies. Also, he does not handle the case where each processor uses a different program.

proof for this case is based on a probabilistic reduction to the uniform case. Finally, in section 5, we prove the main theorem. The proof involves constructing a "difficult" network in a probabilistic way.

2. Preliminaries. A *radio network* is described by an undirected graph $G(V, E)$,[3] where $N = |V|$ and D is the diameter of the graph. The nodes of the graph represent processors of the network, and an edge between nodes v and u implies that v can send messages to u (and vice-versa). The *neighborhood* of a node u includes all the nodes v such that there is an edge (u, v) in E.

The time is viewed as divided into slots (or *rounds*). In any given slot, a node (processor) can either transmit some message (a string in $\{0,1\}^*$) or not (i.e., remain silent). A radio network has the property that if two or more nodes in the neighborhood of a node u transmit at the same time slot, then none of the messages is received at u. More formally, we can define the set of possible transmissions as $W = \{0,1\}^* \cup \{\mathit{silent}\}$. If exactly one of the node's neighbors transmits at time t and the message that this neighbor transmits is some $m \in \{0,1\}^*$, then m is received by the node. In any other case (i.e., if either none of the neighbors transmits or more than one neighbor transmits) this node hears *silent*. The *history* of length ℓ of a node is a vector in W^ℓ which consists of its view of the first ℓ rounds.

Each processor P_i in the radio network uses a *probabilistic* program. This program defines whether the processor will transmit at the next time slot j or not. As we are not concerned with the computational power of the processors we can simply view this program as a probability distribution, which may depend on the history. More formally, for each processor P_i and step j there is a probabilistic function $\Gamma_i^j : W^{j-1} \to W$ that, based on the history, determines the action of P_i in step j (i.e., whether it remains silent, or else the value of the message it sends). The *program* of P_i is a collection $\Gamma_i = (\Gamma_i^1, \Gamma_i^2, \ldots)$ that defines the actions of P_i in each step. A *protocol* $\mathcal{P}_{N,D}$ is simply a collection of N such programs, one per processor. A protocol is *uniform* if all processors use the same program. Otherwise, if each processor has a different program, the protocol is *nonuniform*. The above definition allows the protocols to use the values of N and D. On the other hand, the protocol "does not know" the topology of the graph, meaning that the same protocol must work for all graphs of N nodes and diameter D.

A *broadcast* protocol is a protocol that is initiated by a single processor, called *originator*, that holds a message M. Any other processor is inactive (i.e., it remains silent) until receiving a message for the first time. The aim of the protocol is that each processor in the network will receive a copy of the message M.

3. Uniform processors. In this section we prove the main lemma for the uniform case, where all processors use the same program. It shows that if there are n processors[4] arranged in a clique, then there exists a t ($2 \le t \le n$) such that if t processors wish to transmit (we call these t processors the *participants*), then the expected number of rounds (time slots) until a round in which exactly one of them transmits is $\Omega(\log n)$. In fact, we show that this is the case for most of the t's of the form $t = 2^i$. Note that the assumption that the topology is not known to the processors, in the context of this lemma, means that t, the number of processors that

[3]None of the results presented in this work will be changed if the network is a directed one. However, it is common in this area to assume that the network is undirected.

[4]Note that we use here n (and not N) as the number of processors. This will be convenient while using the lemma in the proof of the theorem.

are trying to transmit, is not known to any processor. We can view the scenario as having a family of networks with $n+1$ nodes, composed from a clique of size n and an originator which is connected to t of the nodes in the clique. The (unknown) topology is chosen to be one of these networks.

For a broadcast protocol Π, we call a round *successful* if exactly one processor transmits. Let $E(T_\ell^\Pi)$ denote the expected number of rounds until the *first* successful round, given that the number of participants is 2^ℓ (the expectation is taken over the probabilistic choices of the processors).

LEMMA 1. *Let* Π *be a broadcast protocol, let the network be as above, and let* n *be an upper bound on the number of participants. Then,*

$$E_\ell[E(T_\ell^\Pi)] = \Omega(\log n),$$

where E_ℓ *denotes the expectation when* ℓ *is chosen uniformly from the range* $1 \leq \ell \leq \log n$.

Proof. The first observation that we make is that the lemma deals only with the *first* success. This, in a sense, allows us to get rid of the dependency in the history—we can assume that the (probabilistic) decision as to which rounds a processor tries to transmit is made at the beginning of the protocol. This is done by letting each of the 2^ℓ processors choose whether to transmit in round s or not in the same way as it chooses in the original protocol, when all previous rounds were unsuccessful. Clearly, as far as the *first* success is concerned, this modification has no effect on the protocol. Also, as only the first success is considered, it does not matter what the values of the messages that the processors try to transmit are. Hence, the decision of a processor on whether to transmit in round s may depend on the round number, s, and the probabilistic choices of the processor in the first $s-1$ rounds, but it does *not* depend on choices made by other processors.[5] Therefore, we can think about the processors as if they choose *in advance*, for every round $s = 1, 2, \ldots$, whether they will try to transmit.[6]

For simplicity of notation, we assume that n is a power of 2. Define

$$p_{s,\ell} \overset{\triangle}{=} Pr(\text{failure in rounds } 1, \ldots, s-1 \ \ \textbf{and} \ \ \text{success in round } s | 2^\ell \text{ participants}).$$

As the events described in the definition are disjoint (for *fixed* ℓ and different s's), and assuming that the protocol succeeds with probability 1 (no matter what ℓ is), we have for all ℓ

$$\sum_{s=1}^{\infty} p_{s,\ell} = 1. \tag{1}$$

At some point in the proof below, it will be inconvenient if $p_{s,\ell}$ depends on events that happen in previous rounds. However, we can get rid of this dependency simply by writing

$$p_{s,\ell} \leq Pr(\text{ success in round } s | 2^\ell \text{ participants}). \tag{2}$$

[5]The message M that the processors need to broadcast also influences their decisions. However, it can be thought of as part of the program used by the processors.

[6]To avoid measurability concerns, it is convenient to assume that the protocol is such that s is in the range $1, \ldots, F$, for some *finite* F. If this is not the case, we can always choose F such that the probability of choosing only in the range $1, \ldots, F$ is arbitrarily close to 1. This will cause minor changes in our proof.

The next claim gives a bound on the sum of the success probabilities in a given round. Intuitively it says that you cannot have high probability of success in (a fixed) round s for more than a few values of 2^ℓ. This would imply that since the number of participants is unknown, $\Omega(\log n)$ rounds would be required to reach a success for all numbers of participants. Formally, we make the following claim.

CLAIM 2. *For any s,*

$$\sum_{\ell=1}^{\log n} Pr(\textit{success in round } s \mid 2^\ell \textit{ participants}) < 2.$$

Proof. Fix s. As already discussed, we assume that the processors make all their choices in advance. The *history* of choices of a processor is a string in $\{0,1\}^{s-1}$, where the value of the ith bit means trying ("1") or not trying ("0"). Define

$$q(s) \stackrel{\triangle}{=} Pr(\text{trying in round } s) = \sum_{\text{history } h} Pr(h) \cdot Pr(\text{trying in round } s|h).$$

Note that $q(s)$ does *not* depend on ℓ. We assume, without loss of generality, that $q(s) > 0$ (rounds with $q(s) = 0$ can be omitted from the protocol). Recall that a successful round is one in which exactly one processor is trying to transmit. Therefore,

$$Pr(\text{success in round } s \mid 2^\ell \text{ participants}) = 2^\ell \cdot q(s) \cdot (1-q(s))^{2^\ell - 1}.$$

We get

$$\begin{aligned}\sum_{\ell=1}^{\log n} Pr(\text{success in round } s|2^\ell \text{ participants}) &= \sum_{\ell=1}^{\log n} 2^\ell q(s)(1-q(s))^{2^\ell-1}\\ &= q(s)\sum_{\ell=1}^{\log n} 2^\ell (1-q(s))^{2^\ell-1}\\ &\le 2\cdot q(s)\sum_{j=1}^{n-1}(1-q(s))^j\\ &= 2\cdot q(s)\cdot\frac{1-(1-q(s))^n}{q(s)} < 2\end{aligned}$$

which completes the proof of the claim. ◻

Let k be a parameter (to be fixed later). We are interested in $\sum_{s=1}^{k} p_{s,\ell}$, which is intuitively the probability that, given that there are 2^ℓ participants, the algorithm succeeds in one of the first k rounds. Using equation (2) and Claim 2, we get

$$(3) \qquad \sum_{\ell=1}^{\log n}\sum_{s=1}^{k} p_{s,\ell} \le \sum_{s=1}^{k}\sum_{\ell=1}^{\log n} Pr(\text{success in round } s|2^\ell \text{ participants}) < 2k.$$

By definition,

$$E_\ell[E(T_\ell^\Pi)] = \sum_{\ell=1}^{\log n}\frac{1}{\log n}\sum_{s=1}^{\infty} p_{s,\ell}\cdot s \ge \frac{k}{\log n}\sum_{\ell=1}^{\log n}\sum_{s=k}^{\infty} p_{s,\ell}.$$

By equation (1), this equals

$$\frac{k}{\log n}\sum_{\ell=1}^{\log n}\left(1-\sum_{s=1}^{k-1}p_{s,\ell}\right),$$

which, by equation (3), is greater than

$$\frac{k}{\log n}\cdot(\log n-2(k-1)).$$

By choosing $k=\frac{1}{4}\log n$, we have that

$$E_\ell[E(T_\ell^\Pi)]\geq\frac{1}{8}\log n+\frac{1}{2},$$

which completes the proof of the lemma.[7] $\Box$

4. Nonuniform processors.

In this section we prove the main lemma for the nonuniform case, where the n processors may use *different* programs. The main idea of the proof is to "reduce" the nonuniform case to the uniform one, and use the result of the previous section (Lemma 1).

LEMMA 3. *Let Π be a protocol for n distinct processors $P_1,\ldots,P_n$ that run (possibly) different programs. Let $E(T_\ell^\Pi)$ denote the expected number of rounds until the* first *successful round, given that a* random *set of 2^ℓ processors participates (the expectation is taken over the choice of the set and the probabilistic choices made by the processors). Then*

$$E_\ell[E(T_\ell^\Pi)]=\Omega(\log n),$$

where ℓ is chosen uniformly from the range $1\leq\ell\leq\log n$.

Proof. As argued in the previous section, as only the first successful round is considered, each program can be thought of as a "schedule"—a choice of a subset of rounds in which the processor will transmit. Processor P_i chooses its schedule from a distribution μ_i.

We now define, based on the (possibly different) programs used by $P_1,\ldots,P_n$, a new program that will be used by each of L *uniform* processors $Q_1,\ldots,Q_L$: processor Q_j chooses (uniformly) at random $1\leq i\leq n$ and simulates the program of processor P_i. Namely, it chooses a schedule s with probability $\frac{1}{n}\sum_{i=1}^{n}\mu_i(s)$, where $\mu_i(s)$ is the probability that processor P_i chooses the schedule s. We denote by $c(Q_j)$ the processor P_i that Q_j chose to simulate. We emphasize that all the Q_j's run the *same* program (i.e., they are uniform), and that different Q_j's may choose to simulate the same processor P_i (we will choose L "small enough" so that this will happen only with a "small" probability).

The following claim says that, given that all the $c(Q_j)$'s are distinct for $Q_1,\ldots,Q_{2^\ell}$, then the probability distribution of the schedules chosen by the Q_j's is the same as that of a *random* set of 2^ℓ processors P_i.

CLAIM 4. *Let $Q=\{Q_1,\ldots,Q_{2^\ell}\}$. For every $Q_j\in Q$, let $c(Q_j)$ be a random processor P_i. If $\forall j_1\neq j_2\,:\,c(Q_{j_1})\neq c(Q_{j_2})$, then $P=\{c(Q_j)|Q_j\in Q\}$ is a random*

[7] In the original version of this paper [KM93], we proved a slightly better lower bound of $\frac{1}{4}\log n$; however, the proof here is simpler.

set of 2^ℓ processors (in $P_1, \ldots, P_n$), and the following holds: for every choice of 2^ℓ schedules $\vec{s}_{2^\ell} = (s_1, \ldots, s_{2^\ell})$,

$$Pr[\vec{s}_{2^\ell} | \textit{processors } Q \textit{ run }] = Pr[\vec{s}_{2^\ell} | \textit{processors } P \textit{ run }].$$

The following claim is the main tool in the reduction from the nonuniform case to the uniform case.

CLAIM 5. *Let Q be as above and let $Q' = \{Q'_1, \ldots, Q'_{2^\ell}\}$ be a set of 2^ℓ processors. Each processor Q'_j runs the program of Q_j at the odd steps and the* [BGI92] *program at the even steps. (Note that the* [BGI92] *program is also a uniform protocol, and therefore, so is the program run by the processors Q'.) Let β_ℓ be the probability that $\forall j_1 \neq j_2$, $c(Q_{j_1}) \neq c(Q_{j_2})$. Let $T_\ell^{Q'}$ be the random variable indicating the time of first success when the 2^ℓ identical programs in Q' run, and recall that T_ℓ^Π is the random variable indicating the time of first success when a random subset of 2^ℓ distinct programs $P_{i_1}, \ldots, P_{i_{2^\ell}}$ run. Then,*

$$E[T_\ell^{Q'}] \leq 2\beta_\ell E[T_\ell^\Pi] + 8(1 - \beta_\ell)\log n.$$

In the above claim we mixed the given (unknown) protocol with the [BGI92] protocol. This is because we have no guarantee about the running time of the simulation, in the case when some Q_j's choose to simulate the same P_i. For example, a protocol that lets processor P_i transmit at time slot i would not terminate if all the Q_j simulate the same processor P_i.

Proof. Let *unique* be the event that $\forall Q_{j_1}, Q_{j_2} \in Q'$, $c(Q_{j_1}) \neq c(Q_{j_2})$. Then,

$$E[T_\ell^{Q'}] = E[T_\ell^{Q'} | \textit{unique}] \cdot Pr[\textit{unique}] + E[T_\ell^{Q'} | \textit{not unique}] \cdot Pr[\textit{not unique}].$$

By definition, $Pr[\textit{unique}] = \beta_\ell$. By Claim 4,

$$E[T_\ell^{Q'} | \textit{unique}] \leq 2E[T_\ell^\Pi],$$

where the additional factor of 2 is due to the interleaving of the two protocols. In the case when the choices of $c(Q_j)$ are not unique, we cannot use the properties of the original protocol. However, we can use the fact that the [BGI92] protocol has the expected time until the first success of at most $4\log n$. Therefore,

$$E[T_\ell^{Q'} | \textit{not unique}] \leq 8\log n,$$

which completes the proof of the claim. □

The next claim says that with "high probability" the choices $c(Q_j)$ are unique.

CLAIM 6. *Let β_ℓ be the probability that $\forall j_1 \neq j_2$, $c(Q_{j_1}) \neq c(Q_{j_2})$, and assume that $2^\ell \leq n^{1/4}$. Then,*

$$\beta_\ell > 1 - \frac{1}{\sqrt{n}}.$$

Proof. Note that

$$Pr[j_1 \neq j_2 \text{ and } c(Q_{j_1}) = c(Q_{j_2})] = \frac{1}{n}.$$

Therefore,

$$\beta_\ell = Pr[\forall j_1 \neq j_2 \ : \ c(Q_{j_1}) \neq c(Q_{j_2})] \geq 1 - \binom{2^\ell}{2}\frac{1}{n}.$$

Since $2^\ell \leq n^{1/4}$ the lemma follows. □

Let $L = n^{1/4}$. By Claims 5 and 6,

$$E[T_\ell^{Q'}] \leq 2\beta_\ell E[T_\ell^\Pi] + (1-\beta_\ell)8\log n \leq 2E[T_\ell^\Pi] + \frac{8\log n}{\sqrt{n}}$$

or

$$E[T_\ell^\Pi] \geq \frac{1}{2}E[T_\ell^{Q'}] - \frac{4\log n}{\sqrt{n}}.$$

We now take the expectation over all values $1 \leq \ell \leq \log L$ and get

$$E_\ell[E[T_\ell^\Pi]] \geq \frac{1}{2}E_\ell[E[T_\ell^{Q'}]] - \frac{4\log n}{\sqrt{n}}.$$

By Lemma 1,

$$E_\ell[E[T_\ell^{Q'}]] = \Omega(\log L) = \Omega(\log n),$$

which implies that

$$E_\ell[E[T_\ell^\Pi]] = \Omega(\log n),$$

as desired. □

5. Main theorem.

In this section we prove the main theorem. We show that for every broadcast algorithm that does not know the topology of the network, for every N and every D, there exist networks of N processors and diameter D such that the expected running time of the algorithm (until all processors receive the message) is $\Omega(D\log(N/D))$. This implies a similar lower bound for the *worst case* running time, when a small probability of error is allowed (which is the scenario in which the upper bound of [BGI92] is described).

Given an algorithm and the values N and D, we construct a network as follows. Let $n = N/D$, and assume for simplicity that n is a power of 2. We construct a complete layered network of $D + 2$ layers. The first layer (layer 0) contains one node, s, which will be the originator of the broadcast. Each of the next D layers (layers $1, 2, \ldots, D$) consists of $n_i = 2^{\ell_i} \leq n$ nodes, where ℓ_i is chosen uniformly (and independently for each layer i) in the range $1, \ldots, \log n$. The last layer contains all the other nodes (so that the total number of nodes will be N). Each node in layer i is connected to all nodes in layers $i-1$ and $i+1$. (See Figure 1.)

Recall that the topology of the network is not known to the processors. (If the topology was known, then an efficient uniform protocol would be to let a processor at layer i broadcast with probability $1/n_i$, with expected time $O(D)$. A nonuniform protocol that knows the topology simply lets one node in each layer transmit.) The algorithm can depend, however, on other information that the processors have, in particular, the history, the number of steps, etc. (As mentioned, other information which is independent of the graph, such as the message M to be broadcast, the

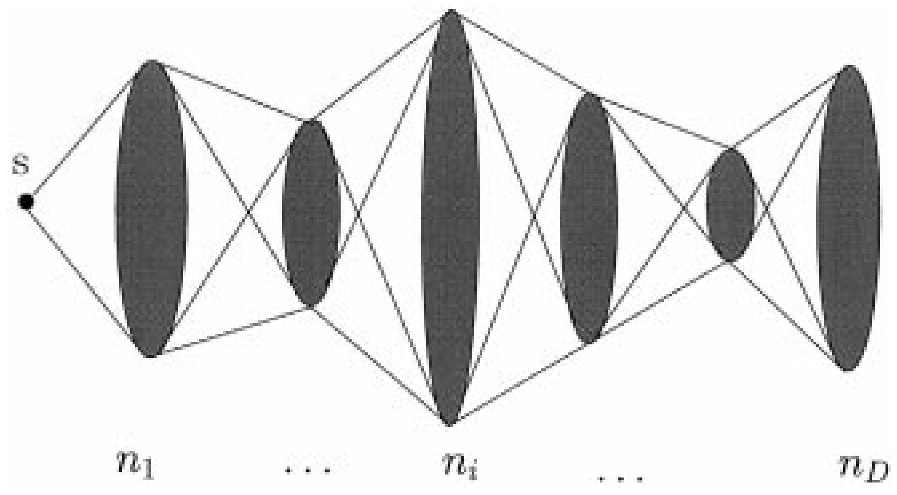

FIG. 1. *Structure of the network.*

processors' IDs, or the value of a clock, can be thought of as encoded into the programs of the processors.)

We discuss the uniform case, in the sense that all the processors at layer i have the same protocol. The extension to the nonuniform case employs the techniques of the previous section, and the proof is the same but the notation becomes cumbersome. (In particular, in the nonuniform case, at each layer i we will choose not only n_i but also a random set of n_i processors.) The main property of this construction is the following. For all i and all runs of the protocol, all the processors in layer i have the same view; every message received at one of these processors is received by all other processors at the same time. Therefore, the broadcast progresses in a layer-by-layer fashion. Moreover, this implies that all the processors in layer i choose schedules according to the same distribution μ (the choice of μ depends on the history, but all the processors of layer i share the same history), which allows us to use Lemma 1.

Finally, before going into the details, we make one more assumption that makes our argument simpler. We give the processors of layer i, at the time they get the first message from a processor in layer $i-1$, all the other messages they will get from layer $i-1$ in the future, as well as the actual values of $\ell_1, \ldots, \ell_{i-1}$. As this extra information can only help the processors to make the broadcast faster, we are allowed to make this assumption.

Let t_i be the random variable indicating the number of rounds from the time the processors of layer i get the message (and become active) until their success (the first time that a single processor in layer i transmits). We need to show that for some choice of $\ell_1, \ldots, \ell_D$ we get $E_\Pi(\sum_{i=1}^{D} t_i) = \Omega(D \log(N/D))$, where the expectation is taken over the random choices of the algorithm Π. Certainly, it is enough to show that $E_{\ell_1,\ldots,\ell_D,\Pi}(\sum_{i=1}^{D} t_i) = \Omega(D \log(N/D))$. By linearity of expectation, we get

$$E_{\ell_1,\ldots,\ell_D,\Pi}\left(\sum_{i=1}^{D} t_i\right) = \sum_{i=1}^{D} E_{\ell_1,\ldots,\ell_D,\Pi}(t_i).$$

So all we have to bound now is $E_{\ell_1,\ldots,\ell_D,\Pi}(t_i)$. Clearly, the choice of $\ell_{i+1}, \ldots, \ell_D$ has no influence on the expectation of t_i; i.e.,

$$E_{\ell_1,\dots,\ell_D,\Pi}(t_i) = E_{\ell_1,\dots,\ell_i,\Pi}(t_i).$$

Also, by the discussion above, with every history (which depends on the random choices made in the first $i-1$ layers, including the choice of $\ell_1,\dots,\ell_{i-1}$) we can associate a probability distribution μ used by the processors in layer i to choose their schedules. (Note that since we assume that the processors of layer i get all the future information with the first message, they can make all their random choices at this time.) Therefore, we can write

$$E_{\ell_1,\dots,\ell_i,\Pi}(t_i) = \sum_{b_1,\dots,b_{i-1}} E_{\ell_i,\Pi}(t_i|\ell_1 = b_1,\dots,\ell_{i-1} = b_{i-1}) \cdot Pr[\ell_1 = b_1,\dots,\ell_{i-1} = b_{i-1}]. \tag{4}$$

It remains to bound the expression $E_{\ell_i,\Pi}(t_i|\ell_1 = b_1,\dots,\ell_{i-1} = b_{i-1})$. As mentioned, we allow the processors at layer i to have access to $b_1,\dots,b_{i-1}$ (the actual values of $\ell_1,\dots,\ell_{i-1}$). Therefore, we need to evaluate $E_{\ell_i,\Pi_i}(t_i)$, where Π_i is the protocol at layer i, with the additional information about the lower layers. By Lemma 1, for each such Π_i,

$$E_{\ell_i,\Pi_i}(t_i) \geq c \log n$$

for some constant c. Therefore, for every $b_1,\dots,b_{i-1}$, we have

$$E_{\ell_i,\Pi}(t_i|\ell_1 = b_1,\dots,\ell_{i-1} = b_{i-1}) \geq c \log n,$$

which by (4), implies

$$E_{\ell_1,\dots,\ell_i,\Pi}(t_i) \geq c \log n.$$

This implies

$$E_{\ell_1,\dots,\ell_D,\Pi}\left(\sum_{i=1}^{D} t_i\right) = \Omega(D \log n) = \Omega(D \log(N/D)),$$

which completes the proof of our main theorem.

THEOREM 7. *For any nonuniform broadcast protocol, for every number of processors N and every diameter D, there exists a network in which the expected time to complete a broadcast is $\Omega(D \log(N/D))$.*

When $D \leq N^{1-\varepsilon}$, the above proof shows a lower bound of $\Omega(D \log N)$. Combining our result with the results of Alon et al. [ABLP91] and Bar-Yehuda, Goldreich, and Itai [BGI92], we have the following tight result.

COROLLARY 8. *For any nonuniform broadcast protocol, for every number of processors N and every diameter D, there exists a network in which the time to complete a broadcast is $\Omega(\log^2 N + D \log(N/D))$. Furthermore, there is a (uniform) protocol that requires only $O(\log^2 N + D \log N)$ expected time (which is tight for $D \leq N^{1-\varepsilon}$).*

Note that unlike [ABLP91] we show that for any protocol there exists a network for which the lower bound holds, while they prove that there exists a network on which any protocol requires the lower bound.

Acknowledgments. We wish to thank Oded Goldreich and the anonymous referees for their very useful comments.

REFERENCES

[ABLP91] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, *A lower bound for radio broadcast*, J. Comput. System Sci., 43 (1991), pp. 290–298.

[ABLP92] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, *Single round simulation on radio networks*, J. Algorithms, 13 (1992), pp. 188–210.

[BGI91] R. Bar-Yehuda, O. Goldreich, and A. Itai, *Efficient emulation of single-hop radio network with collision detection on multi-hop radio network with no collision detection*, Distrib. Comput., 5 (1991), pp. 67–71.

[BGI92] R. Bar-Yehuda, O. Goldreich, and A. Itai, *On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization*, J. Comput. System Sci., 45 (1992), pp. 104–126.

[BII93] R. Bar-Yehuda, A. Israeli, and A. Itai, *Multiple communication in multi-hop radio networks*, SIAM J. Comput., 22 (1993), pp. 875–887.

[CK85a] I. Chlamtac and S. Kutten, *On broadcasting in radio networks—problem analysis and protocol design*, IEEE Trans. Comm., COM-33 (1985), pp. 1240–1246.

[CK85b] I. Chlamtac and S. Kutten, *A spatial reuse TDMA/FDMA for mobile multi-hop radio networks*, in INFOCOM, 1985, pp. 389–394.

[CK87] I. Chlamtac and S. Kutten, *Tree-based broadcasting in multihop radio networks*, IEEECOM, C-36 (1987), pp. 1209–1223.

[CW87] I. Chlamtac and O. Weinstein, *The wave expansion approach to broadcasting in multihop radio networks*, in INFOCOM, pp. 874–881, 1987.

[Gal85] R. Gallager, *A perspective on multiaccess channels*, IEEE Trans. Inform. Theory, 31 (1985), pp. 124–142.

[GM95] I. Gaber and Y. Mansour, *Broadcast in radio networks*, in Proc. 6th ACM-SIAM Symposium on Discrete Algorithms, SIAM, 1995, pp. 577–585.

[KM93] E. Kushilevitz and Y. Mansour, *An $\Omega(D \log(N/D))$ lower bound for broadcast in radio networks*, in Proc. 12th ACM Symp. on Principles of Distributed Computing, 1993, pp. 65–74.

[Tan81] A. S. Tanenbaum, *Computer Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[Wil86] D. E. Willard, *Log-logarithmic selection resolution protocols in a multiple access channel*, SIAM J. Comput., 15 (1986), pp. 468–477.