Improved Randomized Broadcast Protocols in Multi-hop Radio Networks

Chungki Lee *James E. Burns** *Mostafa H. Ammar*

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280

Abstract

This paper presents a suite of randomized broadcast protocols for the problem of broadcasting a message in multi-hop radio networks. The protocols are compared with the randomized broadcast protocol by Bar-Yehuda et al. The time complexity of one of the randomized broadcast protocols presented in this paper is shown, by simulation, to be much better than those of other protocols in most of the typical cases.

1 Introduction

A radio network is a collection of radios that communicate with each other over radio channels. If all units in a network can hear each other, the radio network is single-hop; otherwise it is multi-hop and repeaters must be used to provide network connectivity. Radio networks have recently received significant attention due to the growing interest in cellular telephones and wireless communication networks. In this paper, the problem of broadcast in multi-hop radio networks is considered. Broadcast is initiated by a single node, called the *source*, which sends a message to all nodes in the network. Broadcast communication is an essential ingredient in many distributed network applications. Despite the broadcast nature of the radio medium, broadcast in multi-hop radio networks requires careful consideration due to the increased potential for collisions.

The broadcast problem in radio networks has been studied extensively in the literature [1, 2, 3, 4, 5, 6, 7]. Chlamtac and Kutten [6] showed that, given a radio network and a designated source, finding an optimal broadcast schedule that uses the minimum number of timeslots is NP-hard. Chlamtac and Weinstein [7] presented a polynomial-time (centralized) algorithm for constructing a broadcast schedule that uses $O(D(\log N)^2)$ timeslots, where N is the number of nodes in the network and D is its diameter. Bar-Yehuda et al. [4] presented a randomized broadcast protocol that runs in expected $O((D + log(\frac{N}{\epsilon})) \log N)$ timeslots to ensure that with probability $1-\epsilon$ all nodes receive the message. For $D = O(1)$, they also showed a $\Omega(N)$ lower bound for deterministic broadcast protocols. Alon et al. [1] presented radio networks with diameter $D = O(1)$ in which every broadcast requires $\Omega((\log N)^2)$ timeslots, using a probabilistic argument.

In this paper, we present a suite of randomized broadcast protocols. Unlike deterministic broadcast protocols in which each node is assumed to know either the complete network topology or its neighbors, randomized broadcast protocols need no topological knowledge of the network except for some upper bounds on its size and its maximum degree. This property makes randomized protocols adaptive to changes in topology which occur throughout the execution, and resilient to non-malicious faults. Also randomized broadcast protocols are conceptually simple and require a minor amount of local computation. The (average) time complexity of one of the randomized broadcast protocols in this paper is shown, by simulation, to be much better than that of the randomized broadcast protocol by Bar-Yehuda et al. [4] in many typical topologies. This paper is organized as follows. In section 2, a model of radio networks with some necessary assumptions is presented. The randomized broadcast protocol by Bar-Yehuda et al. is summarized in section 3. A suite of randomized broadcast protocols is presented in section 4. Section 5 discusses the issue of termination of the protocols. Section 6 compares the various randomized protocols. Section 7 concludes the paper.

2 The Model

A radio network is modeled by a connected, undirected graph whose vertices represent nodes (radios) and whose edges represent two-way communication channels between their incident vertices. Thus, we assume that adjacent radios are of comparable power and are within range of each other. Nodes communicate in synchronous timeslots using radio transmissions. All nodes agree on the beginning of each timeslot (using,

---

*Author's current address: Bellcore, P.O. Box 7040, Red Bank, NJ 07701-5699.

0-8186-3670-X/93 $03.00 © 1993 IEEE

say, a satellite to provide timing signals). The length of a timeslot is assumed to be at least as long as the transmission time of the longest message plus the maximum propagation delay of messages between any pair of nodes.

The properties of radio communication are described by the following rules. In each timeslot, a node can act either as a transmitter or as a receiver, but not both. Thus, a transmitting node cannot directly detect whether or not its transmission is successful. A node receives a message in a timeslot if it acts as a receiver and exactly one of its neighbors transmits. If more than one neighbor of a node transmits, a collision occurs. If a collision occurs, receiving nodes cannot determine reliably that a collision has occurred because collisions are indistinguishable from background noise. Except for collisions, channels are assumed to be error-free.

A broadcast message is sent by a node, called the source, to all nodes in the network. Our randomized protocols make use of the following quantities:

- N, the number of nodes in the network (It is sufficient to know an upper bound on N).
- Δ, the maximum degree over all nodes in the network (It is sufficient to know an upper bound on Δ).

These quantities are assumed to be globally known and be constant throughout the execution of the protocols. They are used to parameterize the protocols. (It is sufficient for only the source to know the quantities since they could be sent along with the broadcast message.) Also, we assume as in [4] that only a single broadcast by a single source is in progress at any point in time. The protocols require no other knowledge about the network, which allows them to tolerate changes in the network over time.

The performance measures considered are the *expected reception time* (the time until all nodes receive the broadcast message), and the *expected termination time* (the time until the protocol terminates). Another measure of interest in a randomized broadcast protocol is the *success probability* defined as the probability that all nodes receive a copy of the broadcast message before the protocol terminates.

3 The Original Broadcast Protocol

For completeness we first summarize the randomized broadcast protocol by Bar-Yehuda et al. and its properties [4]. The basis for the protocol is a randomized procedure called *Decay*. *Decay* has the property that if $k = 2\lceil \log \Delta \rceil$ and several neighbors of a node v use *Decay* to send messages, then with probability greater than $\frac{1}{2}$, node v receives one of the messages. The procedure is described in Figure 1. In the description k is a parameter and m is the message to be sent. Throughout the paper all logarithms are to base 2.

```
procedure Decay(k, m);
repeat at most k times
  send m to all neighbors;
  flip (binary) coin
until coin = 0
```

Figure 1: Procedure Decay

The randomized broadcast protocol by Bar-Yehuda et al., called **Protocol 0**, makes use of *Decay*. The protocol executed by each node except the source is described in Figure 2. A network is said to execute a broadcast if the source transmits a broadcast message at timeslot 0 and every other node executes the protocol in Figure 2. Note that *Time* in the protocol is the current timeslot. The protocol has the following properties:

1. It terminates within $O((D+(\log \frac{N}{\epsilon}))\log \Delta)$ timeslots.
2. If *threshold* $= \lceil \log \frac{N}{\epsilon} \rceil$ and the network executes a broadcast then each node has received the broadcast message with probability $> 1-\epsilon$, before termination.

```
procedure Protocol 0;
  $k := 2\lceil \log \Delta \rceil$;
  wait until receiving a message, say m;
  do threshold times
    wait until (Time mod k) = 1;
    Decay(k, m);
  od
```

Figure 2: Protocol 0

Note that all participating nodes start executing *Decay* at the same timeslot (i.e., only at one plus integer multiples of $2\lceil \log \Delta \rceil$). Define the ith phase, or *phase* i, to be the duration of time from timeslot $1+((i-1)*(2\lceil \log \Delta \rceil))$ to timeslot $i*(2\lceil \log \Delta \rceil)$. We observe the following features of **Protocol 0**.

1. Whenever a node receives a message during a phase, the node should wait to forward the message until the beginning of the following phase. Thus upon receiving a message for the first time, each node waits for $\lceil \log \Delta \rceil$ timeslots on the average before it starts to forward the message. This feature will delay propagation of the message.
2. During a phase a node can receive the same message several times from a neighbor because each neighbor keeps sending it until $coin = 0$. This continuous transmission by a neighbor helps to resolve

collisions at nodes with more than one sending neighbor during the phase. However, there is some redundancy involved in the continuous transmission. We define a set X to be a *success set* of a node y in a timeslot if every node in X receives a message from y in that timeslot. Note that during a phase the size of the maximal success set is monotonically increasing with each timeslot until y stops sending. Thus, all timeslots of a phase before the final sending timeslot are redundant.

We conjecture that these features degrade the performance of the protocol. The protocols in the following section attempt to eliminate these features in one way or another to improve the performance.

4 New Broadcast Protocols

In this section a suite of new randomized broadcast protocols is presented.

4.1 Protocol 1

The first protocol, **Protocol 1**, is the same as **Protocol 0** except that after a node receives a message for the first time, the node tries to forward the message from the next timeslot without waiting. Thus the protocol described in Figure 3 also uses the procedure *Decay*. However, unlike **Protocol 0** the protocol does not require that all participants start executing *Decay* simultaneously. So the concept of a phase does not apply here. It seems that immediate transmission of the message by the node upon receiving it will speed up propagation of the message and allow more concurrent transmission of messages (i.e., we can make better use of spatial reuse of transmission timeslots). Thus we conjecture that **Protocol 1** is better than **Protocol 0** in most typical cases.

```
procedure Protocol 1;
  $k := 2\lceil \log \Delta \rceil$;
  wait until receiving a message, say $m$;
  $t := Time$;
  $t_0 := (t + 1)$ mod $k$;
  do threshold times
    wait until $(Time$ mod $k) = t_0$;
    $Decay(k, m)$;
  od
```

Figure 3: Protocol 1

4.2 Protocol 2

The second protocol we consider, **Protocol 2**, is also a variation of **Protocol 0**. The protocol tries to avoid an undesirable feature of **Protocol 0**, i.e., (possibly) continuous transmission by a node during a phase until it tosses 0. That is, unlike **Protocol 0**, a node flips a binary coin before it sends a message. If the value of the coin is 1, the node sends the message and keeps quiet during the remaining timeslots of the phase. Thus each node sends a message at most once during a phase. Note that in **Protocol 0** each node sends a message at least once during a phase. The protocol uses a new procedure *Decay2* described in Figure 4. As proved in our technical report [9], *Decay2* is a randomized protocol with the property that if $k = 2\lceil \log \Delta \rceil$ and several neighbors of a node execute *Decay2* simultaneously to send messages, the node receives one of the messages with probability $\geq \frac{1}{2}$.

```
procedure $Decay2(k, m)$;
do at most $k$ times
  flip (binary) coin
  if $coin = 1$ then send $m$ to all neighbors; exit
od
```

Figure 4: Procedure Decay2

The protocol executed by each node except the source is described in Figure 5. If the source transmits a broadcast message at timeslot 0 and every other node executes the protocol, the following properties (proved in our technical report [9]) hold:

1. A broadcast terminates within $O((D + (\log \frac{N}{\epsilon})) \log \Delta)$ timeslots.
2. If $threshold = \lceil \log \frac{N}{\epsilon} \rceil$, each node has received the message with probability $> 1 - \epsilon$, before termination.

Note that all participating nodes start executing *Decay2* at the same timeslot.

```
procedure Protocol 2;
  $k := 2\lceil \log \Delta \rceil$;
  wait until receiving a message, say $m$;
  do threshold times
    wait until $(Time$ mod $k) = 1$;
    $Decay2(k, m)$;
  od
```

Figure 5: Protocol 2

Unlike **Protocol 0**, if $d \leq \Delta$ neighbors of a node start executing *Decay2* simultaneously, the node can receive a message exactly once from a neighbor during a phase. Also every node is given an opportunity to send until it actually sends during a phase. Notice that a node can receive the same message several times from different neighbors during a phase. **Protocol 2** requires that after a node has received the message during a phase, the node waits until the next phase to start sending. Again it seems that immediate transmission of the message by the node will speed up propagation of the message and allow multiple simultaneous transmissions of messages to be received. The protocol in the following subsection eliminates the wait.

4.3 Protocol 3

The third protocol, **Protocol 3**, is the same as **Protocol 2** except that after a node receives a message for the first time, the node tries to forward the message from the next timeslot without waiting. Thus the protocol uses the procedure *Decay*2 again and is described in Figure 6. However, unlike **Protocol 2** the protocol does not require that all participants start executing *Decay*2 simultaneously. We conjecture that this protocol is the best among the protocols in arbitrary networks.

```
procedure Protocol 3;
  $k := 2\lceil \log \Delta \rceil$;
  wait until receiving a message, say $m$;
  $t := Time$;
  $t_0 := (t + 1)$ mod $k$;
  do threshold times
    wait until ($Time$ mod $k$) = $t_0$;
    $Decay2(k, m)$;
  od
```

Figure 6: Protocol 3

4.4 Protocol 4

The fourth protocol, **Protocol 4**, uses a simple strategy. That is, after a node receives a message for the first time, the node sends the message with probability $\frac{1}{2\lceil \log \Delta \rceil}$ for a certain number of times from the following timeslot. To guarantee the termination of the protocol at non-source nodes, we limit the number of times to a *threshold* value. Due to the random nature of the protocol, only if the threshold value is infinite is the protocol guaranteed to broadcast a message before termination. The protocol is described in Figure 7. In the protocol at most $\frac{\Delta}{2\lceil \log \Delta \rceil}$ neighbors of a node on the average can send a message during a timeslot, which helps to reduce the number of collisions. Note that $\frac{\Delta}{2\lceil \log \Delta \rceil}$ is relatively very small compared with Δ.

```
procedure Protocol 4;
  $k := 2\lceil \log \Delta \rceil$;
  wait until receiving a message, say $m$;
  do threshold times
    generate a uniform random number, $r$;
    if $r < \frac{1}{k}$
    then send $m$ to all neighbors
    else keep quiet;
  od
```

Figure 7: Protocol 4

4.5 Determining Threshold Values

The *success probability* of each protocol is strongly dependent on the value of the *threshold* parameter. For **Protocol 0** it has been shown that if *threshold* $= \lceil \log \frac{N}{\epsilon} \rceil$ then *success probability* $> 1 - \epsilon$. With the exception of **Protocol 2**, we have not yet been able to analytically determine *threshold* values required to achieve given success probabilities for our protocols.

For **Protocol 2** it can be shown (see [9]) that with *threshold* $= \lceil \log \frac{N}{\epsilon} \rceil$, *success probability* $> 1 - \epsilon$. For the other protocols (**Protocol 1, 3, and 4**) and after considerable experimentation we use the following thresholds

- **Protocols 1 & 3**: *threshold* $= \lceil \log \frac{N}{\delta} \rceil$
- **Protocol 4**: *threshold* $= 4\lceil \log \frac{N}{\delta} \rceil$

where δ is a small constant. Our simulation results indicate that for $\delta = 0.01$ and with the above thresholds, success probabilities are > 0.99.

5 Termination Time Analysis

We prove that if the reception time of any protocol is known, the termination time can be computed easily. Our results in the next section show the average reception time for the various protocols in several network environments. We prove only **Theorem 2**. Other theorems can be proved similarly (See [9]).

Theorem 1. *Let t_0 be the reception time of* **Protocol 1 (Protocol 3)** *in a given network with N nodes and at most Δ degree. Assume that the source initiates a broadcast at timeslot 0 in* **Protocol 1 (Protocol 3)**. *Let k be $2\lceil \log \Delta \rceil$. Then the termination time of* **Protocol 1 (Protocol 3)** *is $t_0 + k\lceil \log(\frac{N}{\epsilon}) \rceil$.*

Theorem 2. *Let t_0 be the reception time of* **Protocol 2** *in a given network with N nodes and at most Δ degree. Assume that the source initiates a broadcast at timeslot 0 in* **Protocol 2**. *Let k be $2\lceil \log \Delta \rceil$. Then the termination time of* **Protocol 2** *is $t_0 + (k - (t_0 - 1)$ mod $k)$ mod $k + k\lceil \log(\frac{N}{\epsilon}) \rceil$.*

Proof: Let x be a node that received a broadcast message last at timeslot $t_0 - 1$. Then node x will terminate last because every other node starts **Protocol 2** not later than node x. After it received the message during a phase, it waits for the beginning of the next phase. The number of the timeslots to wait is the difference between the first timeslot of the next phase and the timeslot when it received a message. Since phases always start at timeslot one plus integer multiples of k, it is $(k - (t_0 - 1) \ mod \ k) \ mod \ k$. Then node x will execute the procedure *Decay2* $\lceil \log(\frac{N}{\epsilon}) \rceil$ times. This implies the theorem. □

Theorem 3. *Let t_0 be the reception time of* **Protocol 4** *in a given network with N nodes and at most Δ degree. Assume that the source initiates a broadcast at timeslot 0 in* **Protocol 4**. *Then the termination time of* **Protocol 4** *is $t_0 + 4\lceil \log(\frac{N}{\epsilon}) \rceil$.*

6 Reception Time Analysis

Determining reception times of our protocols is very difficult for arbitrary graphs, so we first consider some graphs with simple structure that lend themselves to analysis. While these are not very realistic models of radio networks, they do help us to understand the behavior of the protocols in extreme conditions. We then consider graphs that are more representative of realistic radio networks.

The network topologies considered are lines, complete binary trees, meshes, and geometric graphs. These topologies give a broad selection of different diameters and degrees of nodes. Note that the time complexity of the protocols strongly depends on the diameter of the network and degrees of nodes. The maximum degrees and diameters of these topologies with N nodes are summarized in Table 1. The maximum degree and diameter of geometric graphs (defined below) are obtained empirically.

Topology	*Maximum Degree*	*Diameter*
line	2	$N-1$
binary tree	3	$\lfloor \log N \rfloor$
mesh	4	$O(\sqrt{N})$
geometric graph	$O(\log N)$	$O(\sqrt{\frac{N}{\log N}})$

Table 1: Maximum Degrees and Diameters of Network Topologies

The reception time of the protocols is the time (the number of timeslots) until every node receives the message from the source. Where possible we derive average reception times analytically. In other situations we resort to simulation. For a network of a given size except geometric graphs, 100 runs with different random number seeds are made and the performance measures of those runs are averaged. In geometric graphs, for a given size, 100 different networks are generated, a run is made per network, and the performance measures of those runs are averaged.

6.1 Graphs of Small Degree

For graphs of small degree, we consider

- a line of N nodes with the source at one end,
- a complete binary tree with the source being the root,
- a mesh of N (even) nodes with the source at the upper left corner.

6.1.1 Lines

Line networks are not very realistic for radio networks, but provide a simple structure that lends itself to precise analysis. We prove only **Theorem 6** (See [9] for details of the proof of other theorems).

Theorem 4. *Assume that the graph corresponding to a network is a line with N nodes with the source at one end, where $N > 2$. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 0** *receive the message with probability one by time $2(N-2)$.*

Theorem 5. *Assume that the graph corresponding to a network is a line with N nodes with the source at one end, where $N > 1$. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 1** *receive the message with probability one by time $N-1$.*

Theorem 6. *Assume that the graph corresponding to a network is a line with N nodes with the source at one end, where $N > 1$. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 2** *receive the message on the average by time $\frac{1}{3}(8N - 13)$.*

Proof: A neighbor of the source receives a message at timeslot 0. Let us calculate how many phases are needed to forward a message one distance down the line network. At the start of the phase each node with a message sends the message with probability $\frac{1}{2}$. If it did not send, it sends the message with probability $\frac{1}{2}$ in the second (last) timeslot of the phase. So the probability that a node sends a message during a phase is

$$\frac{1}{2} + (\frac{1}{2})(\frac{1}{2}) = \frac{3}{4}.$$

Thus the probability that $i \geq 1$ phases are needed for a node to send a message successfully is $(\frac{1}{4})^{i-1}(\frac{3}{4})$.

Therefore the expected number of phases needed for a node to send a message successfully is

$$\sum_{i=1}^{\infty} i(\frac{1}{4})^{i-1}(\frac{3}{4}) = \frac{4}{3}.$$

Since the duration of a phase is 2, the expected time that all nodes receive a message is

$$1 + (\frac{4}{3})2(N-2) = \frac{1}{3}(8N-13).$$

$\square$

Theorem 7. *Assume that the graph corresponding to a network is a line with N nodes with the source at one end, where* $N > 1$*. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 3** *receive the message on the average by time* $2N - 3$.

Theorem 8. *Assume that the graph corresponding to a network is a line with N nodes with the source at one end, where* $N > 1$*. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 4** *receive the message on the average by time* $2N - 3$.

The performance of all protocols is shown in Figure 8. Note that the reception time of **Protocol 1** is optimal because the time is the same as the trivial lower bound (i.e., the diameter of the network). Also note that the reception times of **Protocol 0**, **Protocol 3**, and **Protocol 4** are almost the same. The reception time of **Protocol 2** is the worst.

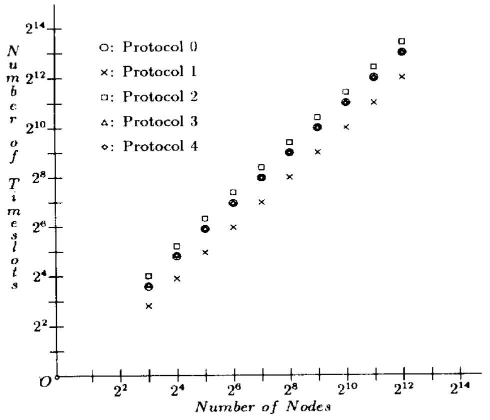

Figure 8: Average Reception Times in Line Networks

6.1.2 Complete Binary Trees

Next we consider a complete binary tree of N nodes with the source as the root, where N is a power of 2 minus one. For this network, when a node sends a message to its children, no collisions occur at them. Thus both **Protocol 0** and **Protocol 1** can be analyzed exactly. It is easy to prove the following theorems (see [9]).

Theorem 9. *Assume that the graph corresponding to a network is a complete binary tree with N nodes, where* $N = 2^i - 1$ *and* $i > 1$*. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 0** *receive the message with probability one by time* $4\lfloor \log N \rfloor - 6$.

Theorem 10. *Assume that the graph corresponding to a network is a complete binary tree with N nodes, where* $N = 2^i - 1$ *and* $i > 1$*. Also suppose that the source initiates a broadcast at timeslot 0. Then all nodes using* **Protocol 1** *receive the message with probability one by time* $\lfloor \log N \rfloor$.

Obviously **Protocol 1** is optimal. Note that the performance of **Protocol 0** is worse than that of **Protocol 1** by the $k = 2\lceil \log \Delta \rceil = 4$ factor because although a node receives a message at the first timeslot of a phase, the node waits until the next phase. Unlike **Protocol 0** (**Protocol 1**) in which a node sends a message at the first timeslot of a phase, the node sends a message with probability $\frac{1}{2}$ in **Protocol 2** (**Protocol 3**). Therefore because no collisions occur when a node sends a message to its children, the performance of **Protocol 2** and **Protocol 3** is worse than that of **Protocol 0** and **Protocol 1**, respectively. However, note that **Protocol 3** performs a little better than **Protocol 0**. **Protocol 4** is the worst. Figure 9 shows these results obtained from either analysis or simulation.

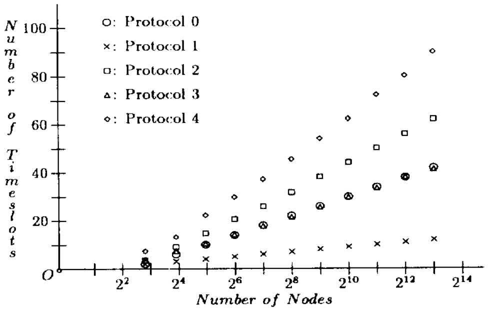

Figure 9: Average Reception Times in Tree Networks

6.1.3 Meshes

Next we consider a mesh of N nodes with the source at the upper left corner, where N is a power of 2. This topology is interesting in the sense that it has a regular structure but, if collisions occur, only at most 4 nodes are involved and so it is relatively easy to resolve randomly.

All protocols have been simulated on (square or almost square) meshes. As shown in Figure 10, the reception time of **Protocol 1** is the best. The reception time of **Protocol 3** is almost as good as that of **Protocol 1**. Observe that the reception time of **Protocol 0** and **Protocol 2** is much worse than that of **Protocol 1** and **Protocol 3**, respectively. When the size of the network becomes larger, the gap between them grows larger. Thus it shows that the wait until the next phase in **Protocol 0** and **Protocol 2** degrades the performance in meshes. Also simulation shows that when the size of the network becomes larger, the performance of **Protocol 1** seems to approach the trivial lower bound, the diameter of the network. Simulation shows that there is little performance difference between **Protocol 0** and **Protocol 2**.

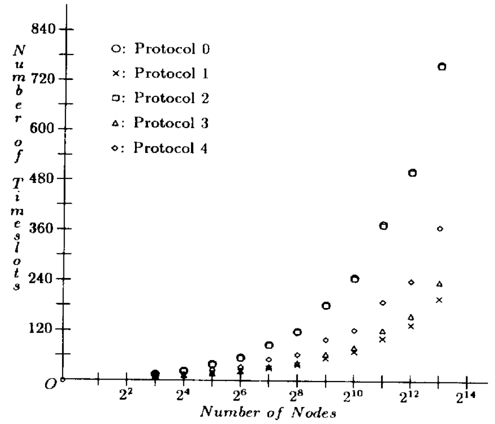

Figure 10: Average Reception Times in Mesh Networks

6.2 Graphs of Intermediate Degree

A family of graphs with maximum degree of $O(\log N)$, **geometric graphs**, is considered. The class of **geometric graphs** was first defined by Johnson, et al. [8]. These graphs are defined in terms of two parameters, N and d. To randomly generate a graph from the class, N points are chosen uniformly and independently from the unit square. A pair of points are connected in the graph if and only if they are within Euclidean distance d of each other. Geometric graphs are a good representation for radio networks in that they simulate a random dispersement of radios of equal power in open terrain (without shadowing objects such as mountains).

We want to choose d so that graphs tend to be connected, but are not too dense (as d increases, we approach the complete graph case). By choosing d to be $\frac{1}{N} + \sqrt{\frac{\log N}{2N}}$, the graph is empirically known to be connected with an expectation of about $\frac{1}{2}$. For these graphs, the expected diameter is approximately $2\sqrt{\frac{N}{\log N}}$ and there are about $N \log N$ edges. For points not too close to the boundary, the expected degree is approximately $\log N$.

All protocols have been simulated on geometric graphs. As shown in Figure 11, the reception time of **Protocol 3** is the best. Observe that the reception time of **Protocol 0** and **Protocol 2** is much worse than that of **Protocol 1** and **Protocol 3**, respectively. When the size of the network becomes larger, the gap between them grows larger. Thus it shows that the wait until the next phase in **Protocol 0** and **Protocol 2** degrades the performance in geometric graphs. Note that the performance of **Protocol 3** is asymptotically better than that of **Protocol 1**. It is interesting that **Protocol 4** performs better than both **Protocol 0** and **Protocol 2**.

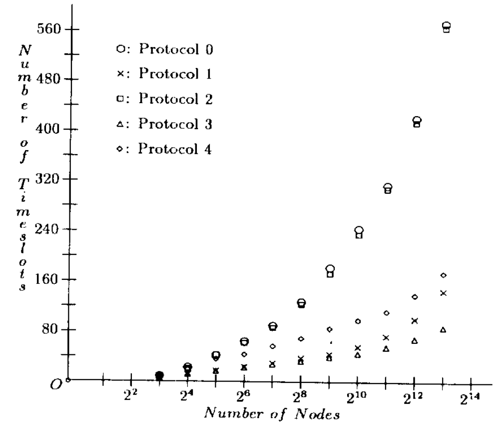

Figure 11: Average Reception Times in Geometric Networks

6.3 The Relationship between Reception and Termination

As proved in section 5, the termination time of a protocol is the reception time plus an offset (an execution time of the protocol and waiting time if applicable). Specifically, we observe the following relationship between the reception and the termination time:

- In line, complete binary tree, and mesh networks, the termination times of the protocols are related in the same manner as their reception times. That is, let R_i and R_j be a reception time of **Protocol** i and **Protocol** j, where $0 \leq i \neq j \leq 4$, respectively. Also let T_i and T_j be a termination time of

Protocol i and **Protocol** j, where $0 \leq i \neq j \leq 4$, respectively. Then if $R_i > R_j$ then $T_i > T_j$. This is due to the fact that the offset (i.e., $T_i - R_i$) is relatively small compared to R_i in **Protocol** i, where $0 \leq i \leq 4$.

- As shown in Figure 12, for geometric graphs the termination times are not related in the same manner as their reception times. That is, **Protocol 4** which has the third lowest reception time terminates first. This is due to the fact that the offset of **Protocol 4** is relatively small compared to the offsets of **Protocol 1** and **3**. However, when the size of the network becomes much larger, it seems that the gap between **Protocol 3** and **Protocol 4** becomes narrower.

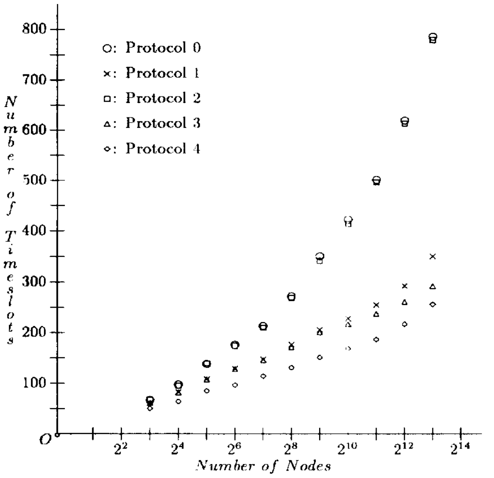

Figure 12: Average Termination Times in Geometric Networks

7 Conclusion

In many applications, the nodes in a radio network are mobile, and therefore the topology is dynamically changing. For this reason, it is desirable for protocols of radio networks not to make assumption about the network topology, or about the information that nodes have concerning the topology. Bar-Yehuda et al. are the first to present a randomized broadcast protocol in this direction. In this work we also assume that none of the nodes have initially any topological information, except for the size of the network and its maximum degree. Also in many of the radio network applications broadcast is a central primitive that is frequently used, for example to perform a network-wide search for a particular node. Therefore there is a strong need for efficient broadcast protocols in radio networks.

In this paper we have presented a suite of randomized broadcast protocols and compared them with the randomized broadcast protocol by Bar-Yehuda et al. In any network considered, we found out that at least one of the new protocols is better than the protocol by Bar-Yehuda et al. Simulation shows that **Protocol 1** is the best choice in networks with small (constant) degree. This seems to be due to the fact that in those networks fewer collisions occur during the execution of the protocol. Also simulation shows that, in many typical topologies, the performance of **Protocol 3** is best or the second best. Therefore we conclude that **Protocol 3** is the best randomized broadcast protocol in most of the typical cases.

References

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

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

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

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

[5] Bar-Yehuda, R., Israeli, A., and Itai, A., "Multiple communication in multi-hop radio networks," In Proc. 8th ACM Symp. on Principles of Distributed Computing, August 1989, pp. 329-338.

[6] Chlamtac, I., and Kutten, S., "On broadcasting in radio networks - problem analysis and protocol design," *IEEE Transactions on Communications* COMM-33, 12(1985), pp. 1240-1246.

[7] Chlamtac, I., and Weinstein, O., "The wave expansion approach to broadcasting in multihop radio networks," In Proc. INFOCOM, April 1987, pp. 874-881.

[8] Johnson, D.S., Aragon, C.R., McGeoch, L.A., and Schevon, C., "Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning," *Operations Research* 37 (1989), No.6, pp. 865-892.

[9] Lee, C., Burns, J.E., and Ammar, M.H., "Improved Randomized Broadcast Protocols in Multi-hop Radio Networks," Tech. Rep. GIT-CC-93/14, February 1993, Georgia Institute of Technology, Atlanta, GA 30332.