Broadcasting in Unreliable Radio Networks

Fabian Kuhn
Faculty of Informatics,
University of Lugano
fabian.kuhn@usi.ch

Nancy Lynch
Computer Science and
Artificial Intelligence Lab, MIT
lynch@csail.mit.edu

Calvin Newport
Computer Science and
Artificial Intelligence Lab, MIT
cnewport@csail.mit.edu

Rotem Oshman
Computer Science and
Artificial Intelligence Lab, MIT
rotem@csail.mit.edu

Andrea Richa
Computer Science and
Engineering, SCIDSE, Arizona
State University
aricha@asu.edu

ABSTRACT

Practitioners agree that unreliable links, which sometimes deliver messages and sometime do not, are an important characteristic of wireless networks. In contrast, most theoretical models of radio networks fix a static set of links and assume that these links are reliable. these links work reliably throughout an execution. This gap between theory and practice motivates us to investigate how unreliable links affect theoretical bounds on broadcast in radio networks.

To that end we consider a model that includes two types of links: *reliable* links, which always deliver messages, and *unreliable* links, which sometimes fail to deliver messages. We assume that the reliable links induce a connected graph, and that unreliable links are controlled by a worst-case adversary. In the new model we show an $\Omega(n \log n)$ lower bound on deterministic broadcast in undirected graphs, even when all processes are initially awake and have collision detection, and an $\Omega(n)$ lower bound on randomized broadcast in undirected networks of constant diameter. This separates the new model from the classical, reliable model. On the positive side, we give two algorithms that tolerate unreliability: an $O(n^{3/2}\sqrt{\log n})$-time deterministic algorithm and a randomized algorithm which terminates in $O(n \log^2 n)$ rounds with high probability.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*computations on discrete structures*;
G.2.2 [**Discrete Mathematics**]: Graph Theory—*graph algorithms*;
G.2.2 [**Discrete Mathematics**]: Graph Theory—*network problems*

General Terms

Algorithms, Theory

Keywords

unreliable networks, broadcast, dual graphs

PODC'10, July 25–28, 2010, Zurich, Switzerland.

1. INTRODUCTION

A fundamental feature of radio networks is the presence of *unreliable* links, which sometimes deliver packets and sometimes do not. Unreliable links can be caused by radio communication *gray zones* [24], multipath propagation, and interference from unrelated networks or electromagnetic devices. As the authors note in [26], something as simple as opening a door can change the connection topology of a network, and it is common in real network deployments to occasionally receive packets from distances significantly longer than the longest reliable link [4]. Unreliable links are so pervasive that virtually every ad hoc radio network deployment of the last five years uses link quality assessment algorithms, such as ETX [13], to cull unreliable connections from those considered by higher-layer protocols. By contrast, many theoretical models of radio networks assume a fixed communication topology consisting only of reliable links.

In this paper, we explore the impact, in terms of algorithmic time complexity, of introducing unreliability into a theoretical model for radio networks. We consider a *dual graph* model, in which there are two types of communication links: *reliable* links that always deliver messages, and *unreliable* links that sometimes deliver messages and sometimes do not. The unreliable links are an abstraction that captures a variety of realistic phenomena. Our goal is to produce a model that is simple enough to be amenable to theoretical analysis, yet still captures the diversity of complex radio behaviors necessary to keep results applicable to real world deployment.

As a first step towards understanding the effects of unreliability we study the fundamental problem of network-wide message broadcast in the dual graph model. Broadcast is a powerful primitive: it can be used to simulate a single-hop network on top of a multi-hop network, greatly simplifying the design and analysis of higher-level algorithms. The broadcast problem has been extensively studied in a variety of models and settings, but mostly in reliable models (see Section 2.2 for an overview of existing work). We show that broadcast in the presence of unreliable links is strictly harder than broadcast in the reliable model. For example, in undirected reliable graphs it is possible to broadcast in $O(n)$ rounds [2, 5], while we show that unreliable links increase the round complexity to $\Omega(n \log n)$ under the same assumptions. For randomized algorithms the stretch is even worse: in the reliable model it is possible to complete a broadcast in $O(D \log(n/D) + \log^2 n)$ rounds with high probability in graphs of diameter D [20], while we show that there is a dual graph network of diameter 2 in which randomized broadcast requires $\Omega(n)$ rounds (this result appeared originally in [22] as a brief announcement). On the other hand, we show that broadcast can still be solved with reasonable efficiency

in the dual graph model: we give an $O(n^{3/2}\sqrt{\log n})$ deterministic algorithm for broadcast in directed dual graphs, and a randomized algorithm that broadcasts in $O(n \log^2 n)$ rounds with high probability. A lower bound from [11] implies that our deterministic algorithm is optimal up to a polylogarithmic factor for *directed* dual graphs; a gap remains for undirected graphs.

2. MODELS FOR RADIO NETWORKS

Many different models for wireless networks have been considered in the literature; we refer the reader to [28, 29] for a comprehensive survey. In this section we introduce our *dual graph model*. Then we briefly review several other models and explain how they compare to the dual graph model.

2.1 The Dual Graph Model

Fix any $n \geq 2$. We define a *dual graph network*, or simply a *network*, to be a pair (G, G') consisting of two directed graphs, $G = (V, E)$ and $G' = (V, E')$, where V is a set of n nodes and $E \subseteq E'$. The set E represents the set of reliable communication links and E' the set of all links, both reliable and unreliable. We assume that V includes a distinguished *source node* s, and that every other node is reachable in G from s. We call a network *undirected* if for every edge (u, v) in E (resp. E'), the edge (v, u) is also in E (resp. E').

We define an algorithm $\mathcal{A}$ to be a collection of n processes, which are either deterministic or probabilistic automata. (See [27] for one possible definition of automata that satisfy our requirements.) We assume that each process $p \in \mathcal{A}$ has a unique identifier ID_p from a totally ordered set I, $|I| = n$. We often write "process i" to mean the process with identifier i.

In order to define how algorithm $\mathcal{A}$ executes on network (G, G'), we must associate processes with graph nodes. Formally, our definition of an execution presupposes a bijection $proc$ from V to $\mathcal{A}$. We assume that an adversary controls the definition of $proc$. The distinction between graph nodes and processes is important for our lower bound results in Sections 4 and 6. However, we generally blur this distinction in our upper bounds in Sections 5 and 7, writing, for example, "node v sends" when we really mean "process $proc(v)$ sends".

An execution of algorithm $\mathcal{A}$ on network (G, G') with a mapping $proc$ proceeds in synchronous rounds, $1, 2, \ldots$. In each round, some input may arrive at each process $proc(v)$ from the external environment. Then $proc(v)$ may or may not send a message. If it sends, its message *reaches* the processes at all of v's outgoing neighbors in G, some arbitrary subset of v's outgoing neighbors in G' that are not outgoing neighbors in G, and v itself. The subset of G'-neighbors that the messages reaches is chosen by the adversary.

When no messages reach a process p, it receives $\perp$, indicating silence. When exactly one message reaches p, it receives the message. When two or more messages reach p, it experiences a *collision*. Collisions can be handled in several ways; we list the possible collision rules in order of decreasing strength (from the algorithmic point of view).

(CR1) If two or more messages reach p (including its own message, if it sends), then p receives $\top$, indicating collision notification.

(CR2) When p sends, it always receives its own message, regardless of whether or not another message reaches it. (This amounts to assuming that a process cannot sense the medium while it is sending.) If two or more messages reach p and p does not send, then it receives collision notification ($\top$).

(CR3) When p sends, it always receives its own message; when two or more messages reach p and p does not send, it hears silence ($\perp$).

(CR4) When p sends, it always receives its own message; when two or more messages reach p and p does not send, it receives either $\perp$ or one of the messages. (Which of these it receives is controlled by the adversary.)

After process p receives, it changes state before beginning the next round.

Another important modelling decision is whether or not all processes start in the same round. Here we consider two rules: the *synchronous start rule* has every process begin in the first round of the execution; the *asynchronous start rule* activates each process the first time it receives a message, either from the environment or from another process.

In our upper bound results, we use the weakest assumptions, that is, collision rule CR4 and asynchronous start; our lower bounds use the strongest assumptions, collision rule CR1 and synchronous start. In each case, this serves to strengthen the results.

The definitions above also apply when $\mathcal{A}$ is probabilistic. But now, in addition, we can define *probability distributions on executions* based on the random choices made by the processes of $\mathcal{A}$. To do this, we specify a particular class of (deterministic or probabilistic) adversaries. Recall that, in general, an adversary may choose the $proc$ mapping, the processes that are reached by each message, and (for collision rule CR4), the particular collision behavior. An adversary class defines precisely what the adversary is allowed to choose and what information is available to it in making its choices. For algorithm $\mathcal{A}$ and any particular adversary in the specified class, we can generate an execution probabilistically using the random choices of the processes of $\mathcal{A}$ together with the adversary's choices. In this way, we obtain a probability distribution on executions. Then for algorithm $\mathcal{A}$ and an entire class of adversaries, we obtain a collection of such probability distributions, one for each adversary in the class. In our lower bound results, we consider very restricted adversaries, whereas our algorithms work with respect to more powerful adversaries.

2.2 Other Models

The standard static model The most common theoretical model for radio networks features a single network graph G, which is static and captures both transmission and interference. A collision occurs at a node when two or more of its neighbors send simultaneously; typically collision rule 3 is assumed, that is, no collision detection is available. The communication graph may be directed or undirected.

For directed graphs with no collision detection and asynchronous starts, the best deterministic upper bound, obtained by combining the algorithms from [20, 12], is $O(n \min\{\log^2 D, \log n\})$; the best lower bound is $\Omega(n \log n / \log(n/D))$ [20]. In [12], a randomized algorithm is given which requires $O(D \log(n/D) + \log^2 n)$ rounds with high probability. This matches the randomized lower bounds of [23, 1], which also hold for undirected networks with synchronous start. In undirected communication graphs with synchronous start it is possible to broadcast in $O(n)$ rounds [2, 5]. This is clearly optimal in n, and [21] shows that this bound is tight even for networks of constant diameter. The $\Omega(n \log n)$ lower bound in Section 6 applies even for undirected graphs with synchronous start, giving a clear separation between the models. The construction may appear superficially similar to the $\Omega(D \log n)$ lower bound of [3], but it differs significantly (the lower bound of [3] does not apply when spontaneous wakeup is allowed).

Explicit-interference models Several works (e.g., [15, 16]) model a network using two graphs, a *transmission graph* G_T and an *interference graph* G_I. It is typically assumed that $G_T \subseteq G_I$. Unlike transmission edges, interference edges can only cause collisions, and messages cannot be conveyed along them. (In contrast, in the dual graph model all edges can convey messages.) A collision occurs at node u when at least one of its G_T-neighbors and at least one of its G_I-neighbors broadcast together. The transmission and interference graphs are both static. A completely different approach is the SINR model [18, 25, 17], in which processes receive messages only when the ratio of the signal to the sum of the noise and other signals exceeds some threshold. The SINR model is geometric: the strength of the signal is assumed to degrade as a function of the distance between the processes. We refer to [30] for a recent treatment of interference in wireless networks.

Models that feature uncertainty The closest model to the dual graph model in the literature is the dynamic-fault model of [11], in which edges of the directed communication graph can fail and recover dynamically during the execution. If one takes G' to be the entire graph and G to be the subgraph induced by edges that never fail, the model of [11] is equivalent to dual graphs, except for one aspect: in [11] it is not assumed that G is connected, and instead the broadcast is only required to reach those processes that are reachable from the source in G. It is shown in [11] that deterministic oblivious algorithms require $\Omega(n^2)$ rounds to broadcast in dynamic-fault graphs; however, the notion of obliviousness used there is a very strong one, and does not allow the behavior of processes to depend on the round in which they first hear the message. In contrast, in Section 5 we give an $O(n^{3/2}\sqrt{\log n})$ broadcast algorithm in which processes use no information *except* the current round and the round in which they first receive the message (and their label). The authors of [11] give a deterministic oblivious algorithm that requires $O(n \min\{n, \Delta \log n\})$ rounds in dynamic-fault graphs of in-degree Δ. This algorithm outperforms ours when $\Delta = o(\sqrt{n}/\log n)$; however, it requires that all processes know (an upper bound on) the in-degree Δ of the interference graph G', whereas our algorithm requires no such knowledge.

In addition, [11] shows an $\Omega(n^{3/2})$ lower bound for non-oblivious deterministic broadcast in directed dynamic-fault graphs. This lower bound carries over to the dual graph model, and implies that the algorithm we give in Section 5 is within $O(\sqrt{n})$ of optimal for directed graphs. In [10], the requirement on broadcast is strengthened to require it to reach all processes, even those that are not connected to the source by a fault-free path. For the stronger broadcast to be possible, it is assumed that in every round there is some functioning link between a process that has the message and a process that does not. This model does not admit a deterministic algorithm, but the authors give an $O(n^2/\log n)$ expected-time randomized algorithm.

Tables 1, 2 summarize the best known upper and lower bounds for broadcast in the classical and dual graph models, assuming synchronous start (SS), asynchronous start (AS), directed (D) or undirected (U) communication graphs. Results shown in bold are presented in the current paper.

3. THE BROADCAST PROBLEM

The broadcast problem requires the dissemination of a message from the process at the distinguished source node s to all processes. We assume that the message arrives at the source process prior to the first round of execution. We assume that the processes treat the message like a *black box*; i.e., that they behave the same regardless of the message contents.

We say that algorithm $\mathcal{A}$ *solves the broadcast problem* in network (G, G') provided that, in any execution of $\mathcal{A}$ in (G, G'), with any assignment $proc$ of processes to nodes, the message eventually arrives at all processes. We say that $\mathcal{A}$ *solves the broadcast problem within k rounds* in network (G, G') provided that, in any execution of $\mathcal{A}$ in (G, G'), with any assignment $proc$ of processes to nodes, the message arrives at all processes within k rounds.

Now consider a probabilistic algorithm $\mathcal{A}$ and a fixed adversary class. Recall that $\mathcal{A}$ generates a collection of probability distributions on executions, one for each adversary in the specified class. For any q, $0 \leq q \leq 1$, we say that probabilistic algorithm $\mathcal{A}$ *solves the broadcast problem* in network (G, G') *with probability* q provided that the following holds: When $\mathcal{A}$ executes in (G, G'), using any adversary in the specified class, with probability at least q, the message eventually arrives at all processes. We say that $\mathcal{A}$ *solves the broadcast problem within k rounds* in (G, G') *with probability* q provided that: When $\mathcal{A}$ executes in (G, G'), using any adversary in the specified class, with probability at least q, the message arrives at all processes within k rounds.

We say that network (G, G') is *k-broadcastable*, where k is a positive integer, if there exist a deterministic algorithm $\mathcal{A}$ and a mapping $proc$ such that, in any execution of $\mathcal{A}$ in (G, G') with $proc$, with collision rule CR1 and synchronous starts, the message arrives at all processes within k rounds. In other words, k-broadcastable captures the intuitive notion that there is a way to resolve the contention in the network such that the message can be propagated to all nodes in k rounds. Note that, if (G, G') is a directed or undirected k-broadcastable network, then the distance from the source to each other node in G must be at most k.

4. BOUNDS FOR 2-BROADCASTABLE NETWORKS

In [22], three of the authors proved the following theorem, which provides a lower bound on the number of rounds required for broadcast in an undirected 2-broadcastable network. The lower bound assumes collision rule CR1 and synchronous starts.

THEOREM 1. *Let $n \geq 3$. There exists a 2-broadcastable undirected network (G, G') such that there do not exist a probabilistic algorithm $\mathcal{A}$ and integer k, $1 \leq k \leq n-3$, where $\mathcal{A}$ solves broadcast within k rounds in (G, G') with probability greater than $k/(n-2)$.*

The lower bound is matched by a deterministic round-robin broadcast strategy, which succeeds (deterministically) in $O(n)$ rounds in (directed or undirected) graphs of constant diameter, and hence, in k-broadcastable networks for any constant k.

5. DETERMINISTIC UPPER BOUND

We describe a deterministic algorithm that solves the broadcast problem in $O(n^{3/2}\sqrt{\log n})$ time. To strengthen the upper bound we assume the weakest assumptions from Section 2: a directed dual graph, Rec. Rule 4, and asynchronous start. For simplicity we assume that $n \geq 3$ and that $\sqrt{n/\log n}$ is a power of 2.

Our algorithm follows the standard broadcast strategy of cycling through *selection objects* of exponentially increasing sizes; c.f., [6, 7]. A selection object is a broadcast schedule for every node, parameterized by the number of nodes participating, which guarantees that if the correct number of nodes participate, each node will be isolated and will be the only node to broadcast in some round. Broadcast algorithms that follow this strategy are typically concerned with isolating all *frontier nodes*, nodes adjacent to some node that does not have the message yet.

	Classical model ($G = G'$)		Dual graphs ($G \neq G'$)	
SS + U	$O(n)$[5]	$\Omega(n)$[21]	$\boldsymbol{O(n^{3/2}\sqrt{\log n})}$	$\boldsymbol{\Omega(n \log n)}$
SS + D	$O(n \min\{\log n, \log^2 D\})$ [20, 12]	$\Omega(n \log n / \log(n/D))$[20]		$\Omega(n^{3/2})$[11]
AS + U				$\boldsymbol{\Omega(n \log n)}$
AS + D				$\Omega(n^{3/2})$[11]

Table 1: Bounds on deterministic broadcast

Classical model ($G = G'$)		Dual graphs ($G \neq G'$)	
$O(n \log(n/D) + \log^2 n)$[12]	$\Omega(n \log(n/D) + \log^2 n)$[23, 1]	$\boldsymbol{O(n \log^2 n)}$	?

Table 2: Bounds on randomized broadcast (for any combination of assumptions with no collision detection)

In the reliable model, when a frontier node u is isolated and broadcasts alone, all of u's neighbors receive the message. Thereafter, node u is no longer a frontier node; even if u continues broadcasting, its transmissions cannot interfere with the progress of the message, because all its neighbors already have the message. Thus, in the algorithms of, e.g., [6, 7], nodes continue to cycle through selective families forever, and never stop broadcasting. The different selector sizes are used to ensure that at least one selector matches the size of the frontier, ensuring that all frontier nodes will be isolated.

In the dual graph model the situation is more complicated; there is no clear-cut "frontier". Suppose that node u has some G'-neighbors that have not received the message, but all of its G-neighbors already have the message. Informally, node u no longer contributes to the progress of the algorithm, because the adversary can prevent it from getting the message out to new nodes (its G'-neighbors); in this sense u is no longer a frontier node. However, u can still *interfere* with the progress of the algorithm, because its broadcasts can cause collisions at nodes that do not have the message. Due to this difficulty, we allow processes to participate in each selection object exactly one, limiting the interval during which they can cause interference. This strategy has the additional advantage that nodes eventually stop broadcasting.

In the following, we use the notation $[k, k']$, where $k' \geq k \geq 1$, to indicate the interval $\{k, ..., k'\}$, and use $[k]$, where $k \geq 1$, to indicate $[1, k]$. We continue by defining *Strongly Selective Families* (SSFs), the selection objects used in our algorithm.

DEFINITION 2 ([8]:). *Let $k \leq n$. A family $\mathcal{F}$ of subsets of $[n]$ is (n, k)-strongly selective if for every non-empty subset $\mathcal{Z}$ of $[n]$ such that $|\mathcal{Z}| \leq k$ and for every element $z \in \mathcal{Z}$ there is a set F in $\mathcal{F}$ such that $\mathcal{Z} \cap F = \{z\}$.*

Erdös et. al. provide an upper bound on the size of these objects [14]:

THEOREM 3 ([14]). *For any $n \geq 3$ and for $k \geq 2$, there exist (n, k)-strongly selective families of size $O(\min\{n, k^2 \log n\})$.*

Let $s_{\max} := \log(\sqrt{n/\log n})$. For each $s \in [s_{\max}]$, let $\mathcal{F}_s$ be an (n, k_s)-SSF of size $\ell_s = O(k_s^2 \log n)$, where $k_s = 2^s$. (By [14] we know such families exist.) We fix some total ordering $\mathcal{F}_s[1], ..., \mathcal{F}_s[\ell_s]$ on the ℓ_s sets that comprise each family $\mathcal{F}_s$. Furthermore, we assume that $\mathcal{F}_{s_{\max}}$ is the round robin sequence, which isolates every node in the graph. Thus, $\mathcal{F}_{s_{\max}}$ is an (n, n)-SSF. (We can assume this because $\ell_{s_{\max}} = \Theta(k_{s_{\max}}^2 \log n) = \Theta(n)$.) We now define our algorithm, which we call *strong select*.

The Strong Select Algorithm Assume without loss of generality that processes have a access to a global round counter[1]. The algorithm divides the rounds into contiguous groups of length $2^{s_{\max}} - 1$ called *epochs*. The first round of each epoch is dedicated to the smallest SSF $\mathcal{F}_1$; the next two rounds are dedicated to $\mathcal{F}_2$; the next four rounds to $\mathcal{F}_3$, and so on. In general, we go through 2^{s-1} sets of each SSF $\mathcal{F}_s$ in each epoch.

When a process first receives a message, it waits, for each $s \in [s_{\max}]$, until $\mathcal{F}_s$ cycles back to $\mathcal{F}_s[1]$. It then *participates* in the SSF $\mathcal{F}_s$ for a single iteration, broadcasting in any round in which it is included in the corresponding SSF set. That is, after it starts participating, in round r of epoch e process i broadcasts iff $i \in \mathcal{F}_{\lfloor \log r \rfloor + 1}[\Big((e-2) \cdot 2^{\lfloor \log r \rfloor} + r\Big) \bmod \ell_{\lfloor \log r \rfloor + 1} + 1]$. After participating in one complete iteration of an SSF, the process stops participating in that family. Each process participates in exactly one iteration of each SSF used in the algorithm.

For a given SSF $\mathcal{F}_s$, we use the term *iteration* to describe a complete cycle through $\mathcal{F}_s[1], \ldots, \mathcal{F}_s[\ell_s]$. Note that each iteration of $\mathcal{F}_s$ is spread out over $\ell_s / 2^{s-1}$ epochs. We also remark that in a given epoch it could happen that a process participates in some selector families but not in others, because it is waiting for those other selector families to cycle back to their first set.

Analysis. Fix a network (G, G') and an execution α of the algorithm in the network. Define $f(n)$ to be the log-factor in the size of the SSFs: formally, $f(n)$ is a function such that $f(n) = O(\log n)$ and for each SSF $\mathcal{F}_s$ used by the algorithm, $\ell_s \leq k_s^2 f(n)$.

The proof involves an amortization argument, where (roughly speaking) we show that in every sufficiently long interval the algorithm always makes progress: either many new nodes receive the message for the first time, and a lot of progress is made; or few nodes receive the message for the first time, but then these nodes only have to contend with each other, and they will quickly be isolated and get the message out to other nodes. To formalize this, we define the *density* of an interval $[r, r']$, denoted $\text{den}(r, r')$, to be the number of processes that receive the message for the first time in the interval, divided by $r' - r + 1$:

$$\text{den}(r, r') := \frac{\begin{array}{c}\text{\# processes that receive the message} \\ \text{for the first time during } [r, r']\end{array}}{r' - r + 1}. \quad (1)$$

[1]This can be achieved by having the source node label every message with its local round counter; when a node is awakened by receiving a message, it adopts the round number on that message.

Given an SSF $\mathcal{F}_s$, let $c_s(r, r')$ denote the number of complete iterations of $\mathcal{F}_s$ that fit in the interval $[r, r']$. Finally, we fix two constants that are used throughout the proof: we define a density threshold

$$\rho := \frac{1}{12f(n)2^{s_{\max}}} = \frac{1}{12f(n)\sqrt{n/\log n}},$$

and let T be the smallest round such that $\mathrm{den}(1, T) < \rho$, that is, the round in which the density over the entire execution first drops below ρ. We will eventually show that the algorithm terminates no later than round T.

We begin by showing that each process that participates in one of the last iterations of some SSF ending by round T is isolated.

LEMMA 4. *Consider the last $c := \min\{4, c_s(1, T)\}$ iterations of $\mathcal{F}_s$ in the interval $[1, T]$, for some $s \in [s_{\max}]$. Every process that participates in one of these c iterations broadcasts alone at some point during the iteration.*

PROOF. Let P be the number of processes that participate in one of the c last SSFs. Let $\ell'_s = \ell_s \left(2^{s_{\max}} - 1\right)/2^{s-1}$ be the number of rounds required to complete an iteration of $\mathcal{F}_s$: family $\mathcal{F}_s$ contains ℓ_s sets spread out over $\ell_s/2^{s-1}$ epochs (with 2^{s-1} sets from $\mathcal{F}_s$ in each epoch), and each epoch requires $2^{s_{\max}} - 1$ rounds. Any process that participates in one of these c iterations must receive the message for the first time in the interval $[T', T]$ where $T' = \max\{1, T - 6\ell'_s + 1\}$. Therefore, if we denote by R the number of processes that receive the message for the first time in $[T', T]$, then $P \leq R$. Note also that $\mathrm{den}(T', T) < \rho$, otherwise we would have $\mathrm{den}(1, T') < \rho$ and T would not be minimal. It follows that

$$\begin{aligned} P \leq R &\overset{(1)}{\leq} \mathrm{den}(T, T') \cdot (T - T' + 1) < \rho \cdot 6\ell'_s \\ &= \frac{6k_s^2 f(n)\left(2^{s_{\max}} - 1\right)}{12f(n)2^{s_{\max}} \cdot 2^{s-1}} \overset{(k_s = 2^s)}{<} k_s. \end{aligned}$$

We have shown that the *total* number of participants in any of the last c iterations is less than k_s; therefore, the number of participants in each individual iteration is also less than k_s (because each process participates in just one iteration). From the definition of an SSF, each participant in any of the last c iterations will be selected to broadcast alone in the network. □

LEMMA 5. *No process receives the message for the first time in the interval $[T', T]$, where $T' = \max\{1, T - 1/\rho + 1\}$.*

PROOF. If one or more processes receives a message in this interval, then $\mathrm{den}(T', T) \geq \frac{1}{T - T' + 1} \geq \rho$, contradicting the minimality of T. □

THEOREM 6. *The strong select algorithm solves broadcast in $O(n^{3/2}\sqrt{\log n})$ rounds in any directed (or undirected) network (G, G'), with collision rule 4 and asynchronous starts.*

PROOF. We first show that every process receives the message by the end of round T.

Assume for contradiction that some node has not received the message by round T. Since all nodes are reachable from the source in G, there exist two nodes i, j such that i has the message by round T and j does not, and $(i, j) \in E$. This means that process i has not been isolated prior to round T; we will show that process i cannot have received the message prior to round T, deriving a contradiction. The proof involves repeatedly using Lemma 4 to show that process i cannot have received the message by the last iteration of selector families of decreasing size, pushing forward the round in which process i first received the message until eventually we exceed round $T' = T - 1/\rho$, obtaining a contradiction to Lemma 5.

Formally, we show by backwards induction on s that for all $s = s_{\max}, \ldots, 1$, process i did not receive the message by round $T - 2\ell'_s$. Here, as in the proof of Lemma 4, we define $\ell'_s = \ell_s(2^{s_{\max}} - 1)/2^{s-1}$ to be the number of rounds required for a complete iteration of $\mathcal{F}_s$. Note that $T - 2\ell'_s$ may be negative, in which case the claim trivially holds.

Induction base: for $s = s_{\max}$, suppose that $T - 2\ell'_{s_{\max}} \geq 0$ and suppose by way of contradiction that node i received the message by round $T - 2\ell'_{s_{\max}}$. Since $\mathcal{F}_{s_{\max}}$ cycles back every $\ell'_{s_{\max}}$ rounds, node i started participating in $\mathcal{F}_{s_{\max}}$ no later than round $T - \ell'_{s_{\max}}$; by round T it has had enough time to participate in a full iteration of $\mathcal{F}_{s_{\max}}$. However, recall that $\mathcal{F}_{s_{\max}}$ is an (n, n)-SSF; any node that participates in a full iteration of $\mathcal{F}_{s_{\max}}$ is isolated. Since we assumed that i has not been isolated by round T, it cannot have received the message by round $T - 2\ell'_{s_{\max}}$.

Induction step: suppose that node i did not receive the message by round $T - 2\ell'_s$, and suppose by way of contradiction that i received the message by round $T - 2\ell'_{s-1} \geq 0$. Observe that since $\ell'_{s-1} = 2^{2(s-1)}f(n)(2^{s_{\max}} - 1)/2^{s-2}$ and $\ell'_s = 2^{2s}f(n)(2^{s_{\max}} - 1)/2^{s-1}$, we have $\ell'_s = 2\ell'_{s-1}$: two iterations of $\mathcal{F}_{s-1}$ fit inside every iteration of $\mathcal{F}_s$. Since process i did not get the message by round $T - 2\ell'_s = T - 4\ell'_{s-1}$, and we assumed for contradiction that it got it by round $T - 2\ell'_{s-1}$, it participates in one of the last $\min\{4, c_{s-1}(1, T)\}$ iterations of $\mathcal{F}_{s-1}$. From Lemma 4, process i is isolated, yielding a contradiction. This concludes the induction.

We have shown that process i did not get the message by round $T - 2\ell'_1 = T - 8f(n)\sqrt{n/\log n} > T - 1/\rho$. Since we assumed that i did get the message prior to round T, it follows that i got the message for the first time in the interval $[\max\{1, T - 1/\rho + 1\}, T]$, contradicting Lemma 5. This completes the first part of the proof; we can now conclude that every process receives the message no later than round T.

To conclude the proof, consider the interval $[1, X]$, where we define $X = n/\rho = 12n^{3/2}f(n)/\sqrt{\log n} = O(n^{3/2}\sqrt{\log n})$. If $\mathrm{den}(1, X) \geq \rho$, then n processes receive the message during the interval $[1, X]$. On the other hand, if $\mathrm{den}(1, X) < \rho$, then by definition $T \leq X$, so again all processes receive the message no later than round X. In both cases the broadcast is complete by round X, and the algorithm terminates in $O(n^{3/2}\sqrt{\log n})$ rounds. □

A Note on Constructive Solutions The (n, k)-SSFs of size $O(\min\{n, k^2 \log n\})$ used in *strong select* are derived from an existential argument [14]. The smallest-size constructive definition of an (n, k)-SSF, from a 1964 paper by Kautz and Singelton [19], is of size $O(\min\{n, k^2 \log^2 n\})$. Replacing the SSFs in our algorithm with the variant from [19] would increase our time complexity by only a $\sqrt{\log n}$-factor.

6. DETERMINISTIC LOWER BOUNDS

In this section, we present two lower bounds for deterministic broadcast algorithms. For both algorithms, we assume collision rule CR1 and synchronous starts. The following bound is a straightforward adaptation of the result presented as Theorem 4.2 of [9]:

THEOREM 7. *There exists a $\sqrt{n}$-broadcastable directed network (G, G'), such that every deterministic algorithm $\mathcal{A}$ that solves the broadcast problem in (G, G') has an execution in which it takes $\Omega(n^{3/2})$ rounds until the message arrives at all processes.*

It follows that our upper bound in Section 5 is tight to within a factor of $O(\sqrt{\log n})$. However, this lower bound construction depends heavily on the fact that the network is directed. If the graph

were undirected, processes could provide feedback to their neighbors when they receive the message; this would break the reduction to the SSF lower bound which is at the core of the lower bound from [9].

We proceed with an $\Omega(n \log n)$ lower bound that handles *undirected* networks. It is unknown whether this bound is tight.

THEOREM 8. *There exists an undirected network (G, G'), such that every deterministic algorithm $\mathcal{A}$ that solves the broadcast problem in (G, G') has an execution in which it takes $\Omega(n \log n)$ rounds until the message arrives at all processes.*

In the following proof, we say a process is *about to be isolated* after a given finite execution if it will send in the next round, and is the only process that will do so.

PROOF. Let the set V of nodes be $\{0, 1, \ldots, n-1\}$, where 0 is the source node. We assume for simplicity that $n-1$ is a power of 2, $n-1 \geq 4$. We divide the nodes into *layers* L_k, $k = 0, \ldots, \frac{n-1}{2}$, where $L_0 = \{0\}$ and for each k, $1 \leq k \leq \frac{n-1}{2}$, $L_k = \{2k-1, 2k\}$.

We construct a dual graph (G, G') with vertex set V. The reliable graph, G, is a complete layered graph, with edge set E given by:

$$\{\{0, u\} \mid u \in \{1, 2\}\} \cup \{\{u, v\} \mid \exists k : u, v \in L_k \text{ and } u \neq v\}$$
$$\cup \{\{u, v\} \mid \exists k : u \in L_k \text{ and } v \in L_{k+1}\}.$$

The unreliable graph, G', is the complete graph over V: $E' = \{\{u, v\} \mid u \neq v\}$. Note that by design, when process $proc(u)$ trasmits, where $u \in L_k$, its message can reach the processes at any subset of the nodes that includes L_{k-1} (if $k > 0$) and L_{k+1} (if $k < \frac{n-1}{2}$).

We assume that the identifier set I includes a distinguished identifier i_0 that is assigned to node 0, that is, that $proc(0) = i_0$.

We construct an execution α and mapping $proc$ in stages numbered $1, 2, \ldots, \frac{n-1}{4}$. At Stage k, $1 \leq k \leq \frac{n-1}{4}$, the construction assigns processes to the nodes ($2k-1$ and $2k$) in layer L_k, and constructs a longer prefix α_k of α. For any k, let A_k be the set of identifiers of processes that are assigned to nodes in layers $L_0, \ldots, L_k$, by the end of Stage k. Our construction will ensure that, by the end of α_k, exactly the processes with identifiers in A_k have received the broadcast message. Moreover, α_k ends with some process in A_k about to be isolated.

As a base case for this construction, in Stage 0 we construct an execution α_0 in which all G'-edges are used in every round, ending with the first round after which i_0 is about to be isolated. There must be some such round, since otherwise no process other than process i_0 will ever receive the message. We define $A_0 = \{i_0\}$. Note that by the end of α_0, only i_0 has the message, because it has not yet sent alone.

Now we describe Stage $k+1$, $0 \leq k \leq \frac{n-1}{4} - 1$, which assigns processes to the two nodes ($2k+1$ and $2k+2$) in layer L_{k+1}, and extends α_k to α_{k+1}. For each pair of processes $\{i, i'\} \subseteq I - A_k$, we define an extension $\beta_{i,i'}$ of α_k, in which we assign processes i and i' to L_k, arbitrarily assigning one of the two processes to $2k+1$ and the other to $2k+2$. We first define $\beta_{i,i'}$ for any $\{i, i'\}$, and then describe how we choose the particular pair i, i' that is used to construct α_{k+1}. For convenience we number the rounds of $\beta_{i,i'}$ after α_k as $0, 1, \ldots$.

In round 0 of $\beta_{i,i'}$, we know that exactly one process sends, and it belongs to A_k. The adversary allows this message to reach (and so, to be received by), exactly the processes in $A_k \cup \{i, i'\}$ (by using the appropriate G' edges). Thereafter, we use the following adversary rules to determine where messages reach. Collisions are handled according to CR1, our strongest rule.

1. If more than one process sends, then all messages sent reach everywhere, and all processes receive $\top$.
2. If a single process $j \in A_k$ sends alone, then its message reaches exactly the processes with ids in $A_k \cup \{i, i'\}$, so exactly these receive it.
3. If a single process $j \in I - (A_k \cup \{i, i'\})$ sends alone, then the message reaches all processes, so they all receive it.
4. If either i or i' sends alone, then the message reaches all processes, so they all receive it. (We include this rule for completeness; this case will not arise within the number of rounds we will consider.)
5. If no process sends, then all processes receive $\perp$.

These rules are designed so that, until either i or i' sends alone, only the nodes in $A_k \cup \{i, i'\}$ will have the broadcast message. It is easy to verify that the adversary can *always* follow the rules above regardless of the process assignment to nodes $2k+3, \ldots, n-1$ (which we have not yet committed to at this point).

Having defined $\beta_{i,i'}$ for all possible pairs $\{i, i'\}$, we must choose the pair $\{i, i'\}$ that will actually be assigned to layer L_k and used to define α_{k+1}. We do this by constructing a sequence of *candidate sets* of process identifiers, $C_0, C_1, \ldots, C_{\log(n-1)-2}$, where $C_0 = I - A_k$, and each candidate set in the sequence is a subset of the previous one. Informally speaking, the identifiers in each C_ℓ are the candidates that remain after we take into account behavior through round ℓ. The process ids i and i' will be elements of $C_{\log(n-1)-2}$.

We begin with $C_0 = I - A_k$ and construct the remaining candidate sets inductively. Observe that $|C_0| = |I - A_k| \geq \frac{n-1}{2}$, because we apply this construction for only $\frac{n-1}{4}$ stages and add only two processes to A_k at each stage.

We maintain the following inductive property for each candidate set C_ℓ (where $0 \leq \ell \leq \log(n-1) - 2$).

Property $P(\ell)$.

(1) $|C_\ell| \geq \frac{n-1}{2^{\ell+1}}$.

(2) Let $j \in I$, and let $\{i_1, i_1'\}$ and $\{i_2, i_2'\}$ be two pairs of elements of C_ℓ. Suppose that j is either in neither subset or in both. Then process j receives the same values (either $\perp$, $\top$, or an actual message) in rounds $1, \ldots, \ell$ of $\beta_{i_1,i_1'}$ and $\beta_{i_2,i_2'}$.

(3) Let $i, i' \in C_\ell$. Then neither i nor i' sends alone at any of rounds $1, \ldots, \ell$ of $\beta_{i,i'}$.

Part (1) of $P(\ell)$ will be used to ensure that we can extend Stage k to $\Omega(\log n)$ rounds. Part (2) ensures that neither of the processes assigned to layer L_k learns the identity of the other process, and also that none of the processes assigned to layers greater than k learns the identities of the processes assigned to layer k. Part (3) says that the candidates that remain after round ℓ have not yet sent alone, after α_k.

Suppose we already have a set $C_{\log(n-1)-2}$ satisfying $P(\log(n-1) - 2)$. Conditions (1) and (3) together imply that there exist $i, i' \in C_{\log(n-1)-2}$ such that neither i nor i' sends alone in any of rounds $1, \ldots, \log(n-1) - 2$ of $\beta_{i,i'}$. We arbitrarily choose one such pair $\{i, i'\}$, and define α_{k+1} to be the prefix of $\beta_{i,i'}$ ending at the first time when either i or i' is about to be isolated; this extends α_k by at least $\log(n-1) - 2$ rounds.

Inductive construction of $C_0, \ldots, C_{\log(n-1)-2}$. Property $P(0)$ is clearly true for C_0. Suppose we have already constructed C_ℓ, where $0 \leq \ell \leq \log(n-1) - 3$, such that $P(\ell)$ holds, and let us construct $C_{\ell+1}$. We begin by defining two sets:

- $S_{\ell+1}$ is the set of remaining candidates $i \in C_\ell$ such that if we assign i to layer L_k, then i will send in round $\ell+1$. Formally, $S_{\ell+1}$ is defined to be the set of ids $i \in C_\ell$ such that for some $i' \in C_\ell$, $i' \neq i$, process i sends in round $\ell+1$ of $\beta_{i,i'}$. (By Part 2 of $P(\ell)$, this set is equivalent to what what we obtain if we replace "for some i'" with "for every i'".)
- $N_{\ell+1}$ is the set of remaining candidates $i \in C_\ell$ that will send in round $\ell+1$ if we do *not* assign them to layer L_k. That is, $N_{\ell+1}$ is the set of nodes such that for some $j, j' \in C_\ell$ where $i \notin \{j, j'\}$, process i sends in round $\ell+1$ of $\beta_{j,j'}$. (As above, by Part 2 of $P(\ell)$, this also holds if we replace "for some j, j'" with "for every j, j'".)

Note that for every $i \in C_\ell - (S_{\ell+1} \cup N_{\ell+1})$, process i will not send in round $\ell+1$ *regardless* of whether or not it is assigned to layer L_k.

Now we are ready to define $C_{\ell+1}$. We consider cases based on the sizes of $S_{\ell+1}$ and $N_{\ell+1}$.

Case I: $|N_{\ell+1}| \geq 2$, that is, there are at least two processes that would send in round $\ell+1$ if they are not assigned to layer L_k.

In this case we omit two such processes from the candidate set: we define $C_{\ell+1} := C_\ell - \{j, j'\}$, where j, j' are the two smallest elements of $N_{\ell+1}$.

Case II: $|N_{\ell+1}| \leq 1$ and $|S_{\ell+1}| \geq \frac{|C_\ell|}{2}$. Then we set $C_{\ell+1} := S_{\ell+1}$.

Case III: $|N_{\ell+1}| \leq 1$ and $|S_{\ell+1}| < \frac{|C_\ell|}{2}$. Then we set $C_{\ell+1} := C_\ell - (S_{\ell+1} \cup N_{\ell+1})$.

That is, if at least two processes would send in round $\ell+1$ if they *did not* receive the message in round 0, then we omit two such processes from the new candidate set. This guarantees that, in the remaining executions we will consider, they will not receive the message in round 0 and will therefore send in round $\ell+1$, so everyone will receive $\top$ in round $\ell+1$.

On the other hand, if at most one process would send in round $\ell+1$ if it did not receive the message in round 0, then we determine the candidates based on the number of processes that would send in round $\ell+1$ if they *did* receive the message in round 0. If at least half would send in round $\ell+1$, we include exactly those that would send. This ensures that, in the remaining executions, at least two of these will receive the message in round 0 and will send in round $\ell+1$, again causing everyone to receive $\top$ in round $\ell+1$.

The remaining case is where at most one process would send in round $\ell+1$ if it did not receive the message in round 0, and strictly fewer than half would send in round $\ell+1$ if they did receive the message in round 0. In this case, we include exactly those that would not send if they received the message, omitting the possible single process that would send if it did not receive the message. This ensures that, in the remaining executions, the processes that receive the message at slot 0 will not send at slot $\ell+1$. Other processes, however, may send at slot $\ell+1$.

CLAIM 9. *Property $P(\ell+1)$ holds for $C_{\ell+1}$. That is,*

1. $|C_{\ell+1}| \geq \frac{n-1}{2^{\ell+2}}$.
2. *Let $j \in I$, and let $\{i_1, i_1'\}$ and $\{i_2, i_2'\}$ be two pairs of elements of $C_{\ell+1}$. Suppose that j is either in neither subset or in both. Then process j receives the same values (either $\perp$, $\top$, or an actual message) in rounds $1, \ldots, \ell+1$ of $\beta_{i_1,i_1'}$ and $\beta_{i_2,i_2'}$.*
3. *Let $i, i' \in C_{\ell+1}$. Then neither i nor i' sends alone at any of rounds $1, \ldots, \ell+1$ of $\beta_{i,i'}$.*

PROOF. For Part 1, note that $|C_\ell| \geq \frac{n-1}{2^{\ell+1}}$, by Part 1 of $P(\ell)$. If $|C_\ell|$ is even, the result then follows by easy calculations based on the three cases in the definition of $C_{\ell+1}$ from C_ℓ. If $|C_\ell|$ is odd, then the calculation is straightforward for Cases 1 and 2(a). The argument for Case 2(b) is slightly more involved. We know that $|C_\ell| \geq \frac{n-1}{2^{\ell+1}}$. We know that $\frac{n-1}{2^{\ell+1}}$ is even, because $\ell \leq \log(n-1) - 3$. Since $|C_\ell|$ is odd, we have $|C_\ell| \geq \frac{n-1}{2^{\ell+1}} + 1$. Also, since $|S_{\ell+1}| < \frac{|C_\ell|}{2}$, we have $|S_{\ell+1}| \leq \frac{|C_\ell|-1}{2}$. So we have

$$|C_{\ell+1}| = |C_\ell| - |S_{\ell+1}| - 1 \geq |C_\ell| - \frac{|C_\ell|-1}{2} - 1 = \frac{|C_\ell|-1}{2}.$$

By the lower bound on C_ℓ, the right-hand side is

$$\geq \frac{(\frac{n-1}{2^{\ell+1}} + 1) - 1}{2} = \frac{n}{2^{\ell+2}},$$

as needed.

Part 3 follows from Part 3 of $P(\ell)$ and the cases in the definition of $C_{\ell+1}$.

In remains to show Part 2; for this, fix j, i_1, i_1', i_2, i_2' as in the hypotheses. Part 2 of P_ℓ implies that j receives the same values in the first ℓ rounds; we consider what happens in round $\ell+1$. We consider cases as in the definition of $C_{\ell+1}$.

Case I: $|N_{\ell+1}| \geq 2$**.** Then in both $\beta_{i_1,i_1'}$ and $\beta_{i_2,i_2'}$, two processes in $N_{\ell+1}$ do not receive the message in round 0 and so send at round $\ell+1$. It follows that j receives $\top$ in round $\ell+1$ in both executions.

Case II: $|N_{\ell+1}| \leq 1$ **and** $|S_{\ell+1}| \geq \frac{|C_\ell|}{2}$**.** Then both i_1 and i_1' send in round $\ell+1$ in $\beta_{i_1,i_1'}$ and both i_2 and i_2' send in round $\ell+1$ in $\beta_{i_2,i_2'}$, so again j receives $\top$ in round $\ell+1$ in both executions.

Case III: $|N_{\ell+1}| \leq 1$ **and** $|S_{\ell+1}| < \frac{|C_\ell|}{2}$**.** Here we must carefully consider which processes send in round $\ell+1$. We know that neither i_1 nor i_1' sends in round $\ell+1$ of $\beta_{i_1,i_1'}$, and neither i_2 nor i_2' sends in round $\ell+1$ of $\beta_{i_2,i_2'}$. Also, we know that each process in A_k chooses whether/what to send based on its own state after α_k, its receipt of the message in round 0, and whatever values it receives in rounds $1, \ldots, \ell$. All of this information is the same in $\beta_{i_1,i_1'}$ and $\beta_{i_2,i_2'}$, using Part 2 of Property $P(\ell)$ (here, each element of A_k is always in neither of the two sets). Therefore, it behaves the same in round $\ell+1$ of both executions.

We now consider two sub-cases.

Subcase IIIa: $|N_{\ell+1}| = 0$**.** Then no process in $I - (A_k \cup \{i_1, i_1'\})$ sends in round $\ell+1$ of $\beta_{i_1,i_1'}$, and no process in $I - (A_k \cup \{i_2, i_2'\})$ sends in round $\ell+1$ of $\beta_{i_2,i_2'}$. Since neither i_1 nor i_1' sends in round $\ell+1$ in $\beta_{i_1,i_1'}$, and neither i_2 nor i_2' sends in round $\ell+1$ in $\beta_{i_2,i_2'}$, it follows that in this subcase, no process in $I - A_k$ sends in round $\ell+1$ of $\beta_{i_1,i_1'}$ or $\beta_{i_2,i_2'}$.

We are left to consider the processes in A_k. If no process in A_k sends in round $\ell+1$ then j receives $\perp$ in both β executions. If two or more processes in A_k send in round $\ell+1$, then by the adversary rules, both messages reach all processes, so j receives $\top$ in both executions. If exactly one process in A_k sends, then by the adversary rules, the message reaches exactly the processes in $A_k \cup \{i_1, i_1'\}$ in $\beta_{i_1,i_1'}$, and reaches exactly the processes in $A_k \cup \{i_2, i_2'\}$ in $\beta_{i_2,i_2'}$. Since j is either in both sets $A_k \cup \{i_1, i_1'\}$ and $A_k \cup \{i_2, i_2'\}$ or neither, the message reaches j either in both executions or in neither execution. Thus, either j receives the message in both executions, or it receives $\perp$ in both executions.

Subcase IIIb: $|N_{\ell+1}| = 1$**.** Then a single process $n_1 \in N_{\ell+1}$ sends in round $\ell+1$ of both β executions. This follows because

we have explicitly omitted n_1 from $C_{\ell+1}$, ensuring that it does not receive the message in round 0 in β_{i_1,i'_1} or β_{i_2,i'_2}, which implies that it sends in round $\ell+1$. By the adversary rules, we know that n_1's message reaches j in both executions.

Now we consider the processes in A_k. If no process in A_k sends in round $\ell+1$, then j receives the message from n_1 in round $\ell+1$ in both executions. If one or more processes from A_k sends, then by the adversary rules, their messages reach all processes. So then j receives $\top$ in both executions (because the A_k message(s) collide with the n_1 message).

Combined, these cases establish Part 2 of $P(\ell+1)$, thus completing the proof of the claim. □

Claim 9 implies that $P(\log(n-1)-2)$ holds for $C_{\log(n-1)-2}$. Therefore, there exist two identifiers $i, i' \in C_{\log(n-1)-2}$ such that neither i nor i' sends alone at any of the first $\log(n-1)-2$ slots of $\beta_{i,i'}$. (Use Part 1 to show that $|C_{\log(n-1)-2}| \geq 2$, and Part 3 to show that the processes in this set do not send alone.) We then define α_{k+1} to be the prefix of $\beta_{i,i'}$ that ends just before the first round where either i or i' sends alone. This gives us an extension of at least $\log(n-1)-2$ slots. Note that only processes in $A_k \cup \{i, i'\}$ have the broadcast message by the end of α_{k+1}.

For the entire construction, we begin with α_0 and construct successive extensions $\alpha_1, \alpha_2, \ldots, \alpha_{\frac{n-1}{4}}$. Since only two new processes receive the message in each stage, by the end of $\alpha_{\frac{n-1}{4}}$, some processes have still not received the message. The resulting execution is $\Omega(n \log n)$ rounds long, which yields our lower bound.

7. RANDOMIZED UPPER BOUND

In this section we give a simple randomized algorithm for broadcast, which completes in $O(n \log^2 n)$ rounds with high probability. We assume a directed communication graph and collision rule CR4, the weakest rule.

The randomized algorithm we describe is symmetric: all processes execute the same algorithm (and in particular, they do not use unique identifiers). For simplicity in notation, in this section we assume that the processes are indexed by $1, \ldots, n$.

Algorithm Harmonic Broadcast Nodes begin participating immediately after they receive the message. If node v receives the broadcast message for the first time in round t_v, then in all rounds $t > t_v$ it transmits the message with probability $p_v(t)$, given by

$$\forall t > t_v \quad : \quad p_v(t) := \frac{1}{1 + \lfloor \frac{t-t_v-1}{\mathcal{T}} \rfloor}.$$

Hence, for the first $\mathcal{T}$ rounds after receiving m, nodes transmit the message with probability 1; in the next $\mathcal{T}$ rounds the message is transmitted with probability $1/2$, then the probability becomes $1/3$, and so on. The parameter $\mathcal{T} \geq 1$ in the algorithm is an integer parameter that will be fixed later. For $t \leq t_v$, we define $p_v(t) := 0$. For convenience, we assume that the sender s receives m at time 0, i.e., $t_s = 0$ and s starts broadcasting m in round 1.

Analysis. For every $t \geq 1$, we define

$$P(t) := \sum_{v \in V} p_v(t) \tag{2}$$

to be the sum of the transmitting probabilities in round t. We say that round t is *busy* if $P(t) \geq 1$, and otherwise we say that round t is *free*.

We begin by bounding the number of busy rounds in any execution from above. We define the *wake-up pattern* of an execution to be a non-decreasing sequence $W = t_1 \leq t_2 \leq \cdots \leq t_n$ of round numbers, where $t_1 = 0$, and t_i is the round in which the i^{th} node receives the message. (That is, t_2 is the round in which the first node that is not the source receives the message, and so on.) Note that the wake-up pattern of an execution determines the broadcasting probabilities of the nodes in every round; therefore, to reason about broadcast probabilities it is sufficient to reason about all possible wake-up patterns (including ones that cannot occur in any execution of the algorithm).

LEMMA 10. *Let $B(n)$ be the maximum number of busy rounds induced by any wake-up pattern. Then there is a wake-up pattern for which rounds $1, \ldots, B(n)$ are all busy.*

PROOF. Let $W = t_1 \leq \cdots \leq t_n$ be a wake-up pattern that maximizes the number of busy rounds, and among those wake-up patterns that maximize the number of busy rounds, minimizes the number of free rounds before the last busy round. We argue that this wake-up pattern has no free rounds between the busy rounds, that is, rounds $1, \ldots, B(n)$ are all busy rounds.

For the sake of contradiction, suppose that there is a free round before the last busy round, and let r_0 be the last free round before the last busy round. By definition, $P(r_0) < 1$, and since round r_0+1 must be busy, we also have $P(r_0+1) \geq 1$. The sum of the broadcast probabilities can only increase from one round to the next if some new node receives the message for the first time; thus, there is some node $i_0 \in [n]$ such that $t_{i_0} = r_0$.

Consider the alternative wake-up pattern $W' = t'_1 \leq \cdots \leq t'_n$, where $t'_i = t_i$ if $i < i_0$ and otherwise $t'_i = t_i - 1$. Let us use $P(t), P'(t)$ to denote the sum of the probabilities induced by wake-up patterns W and W' in round t, respectively. Further, let $p_x(t)$ be the sending probability in round t of a node that first receives the message in round x (as defined in the algorithm). Because the wake-up patterns W, W' are the same up to round $r_0 - 2$, we have $P(t) = P'(t)$ for all $t < r_0$. For $t \geq r_0$, we have

$$P'(t) = \sum_{i=1}^{n} p_{t'_i}(t) = \sum_{i=1}^{i_0-1} p_{t_i}(t) + \sum_{i=i_0}^{n} p_{t_i}(t+1)$$
$$\geq \sum_{i=1}^{n} p_{t_i}(t+1) = P(t+1).$$

Therefore, if round $t > r_0$ is busy for W, then round $t-1$ is busy for W', and the total number of busy rounds in W' is at least the same as in W. Furthermore, round r_0 (which was free for W) is busy for W', because round r_0+1 is busy for W. It follows that W' has fewer free rounds before the last busy round than W does, but it has at least as many busy rounds, contradicting the choice of W. (Recall that W was chosen to be a wake-up pattern that maximizes the total number of busy slots, and among these wake-up patterns, minimizes the number of free time slots before the last busy slot.) □

The following lemma bounds the total number of busy rounds induced by any wake-up pattern.

LEMMA 11. *The total number of busy rounds for any wake-up pattern is at most $n \cdot \mathcal{T} \cdot H(n)$.*

PROOF. Consider an arbitrary n-node wake-up pattern $W = t_1 \leq \cdots \leq t_n$. We show that there has to be a free round by time $t_f(n) := n \cdot \mathcal{T} \cdot H(n)$ where $H(n) = \sum_{i=1}^{n} 1/i$, $H(0) = 1$ denotes the harmonic sum. Together with Lemma 10, this implies the claim.

We prove that there is a free time round by time $t_f(n)$ by induction on n. For $n = 1$ the claim is immediate.

Thus, let $n > 1$. For $i \in [n]$, let v_i be the node that wakes up (receives the message) at time t_i, and let τ_i be the first free round when using the i-node wake-up pattern $t_1, \ldots, t_i$ (that is, the prefix of W in which nodes $v_{i+1}, \ldots, v_n$ are never awakened). By the induction hypothesis, $\tau_i \leq t_f(i)$ for all $i < n$. We want to show that $\tau_n \leq t_f(n)$.

Let us first consider the case where $t_{i+1} \geq \tau_i$ for some $i \in [n-1]$. In this case, round τ_i remains free when we consider the complete wake-up pattern W; thus, $\tau_n = \tau_i \leq t_f(i) \leq t_f(n)$.

Next, consider the case where $t_{i+1} \leq \tau_i - 1 \leq t_f(i) - 1$ for all $i \in [n-1]$. For any $i \in [n]$, at time $t_f(n)$, the sending probability of node v_i is

$$p_{v_i}\big(t_f(n)\big) = \frac{1}{1 + \left\lfloor \frac{t_f(n) - t_i - 1}{\mathcal{T}} \right\rfloor} \leq \frac{1}{1 + \left\lfloor \frac{t_f(n) - (t_f(i-1)) - 1) - 1}{\mathcal{T}} \right\rfloor} < \frac{1}{(n-i+1)H(n)}.$$

For the sum of transmitting probabilities, we therefore obtain

$$P\big(t_f(n)\big) = \sum_{i=1}^{n} p_{v_i}\big(t_f(n)\big) < \sum_{i=1}^{n} \frac{1}{(n-i+1)H(n)} = \frac{H(n)}{H(n)} = 1.$$

Hence, round $t_f(n)$ is free, as required. □

We say that a process is *isolated* in a round if it is the only process transmitting in that round. In the following, we show that a process that broadcasts in a free round is isolated with high probability, and that as soon as the number of free rounds since a process received the message is large enough, that process is isolated with high probability.

LEMMA 12. *Let $t \geq 1$ be a free round and assume that node v transmits in round t with probability $p_v(t)$. The probability that v is isolated in round t is at least $p_v(t)/4$.*

PROOF. Because t is a free round, all transmitting probabilities are smaller than 1 and thus for all $u \in V$ we have $p_u(t) \leq 1/2$. Let q be the probability that none of the nodes in $V \setminus \{v\}$ send in round t. We have

$$q = \prod_{u \in V \setminus \{v\}} (1 - p_u(t)) \geq \prod_{u \in V \setminus \{v\}} \left(\frac{1}{4}\right)^{p_u(t)} > \left(\frac{1}{4}\right)^{P(t)} > \frac{1}{4}.$$

In the last two steps we used the fact that for $0 \leq x \leq 1/2$ it holds that $1 - x \geq (1/4)^x$, and that $P(t) < 1$, because t is a free round. The probability that v is isolated in round t is $p_v(t) \cdot q > p_v(t)/4$. □

LEMMA 13. *Consider a node v, and let t_v be the time when v first receives the message. Further, let $t > t_v$ be such that at least half of the rounds $t_v + 1, \ldots, t$ are free. If $\mathcal{T} \geq 12\ln(n/\epsilon)$ for some $\epsilon > 0$, then with probability larger than $1 - \epsilon/n$ there exists a round $t' \in [t_v + 1, t]$ such that v is isolated in round t'.*

PROOF. Let $\tau = t - t_v$. Note that $\tau \geq 2\mathcal{T}$ because v sends with probability 1 in the first $\mathcal{T}$ rounds (a nd hence the first $\mathcal{T}$ rounds are not free). In round t, the transmitting probability of v is

$$p_v(t) = \frac{1}{1 + \left\lfloor \frac{\tau - 1}{\mathcal{T}} \right\rfloor} \geq \frac{1}{1 + \frac{\tau - 1}{\mathcal{T}}} = \frac{\mathcal{T}}{\mathcal{T} + \tau - 1}. \qquad (3)$$

Because the transmitting probability is non-increasing, by Lemma 12, for every free round $t' \in [t_v + 1, t]$, the probability that v is isolated is larger than $\frac{\mathcal{T}}{4(\mathcal{T}+\tau-1)}$. Let q be the probability that there is no free round $t' \in [t_v + 1, t]$ in which v transmits alone. As there are at least $\lceil \tau/2 \rceil$ free rounds, the probability q is bounded by

$$q < \left(1 - \frac{\mathcal{T}}{4(\mathcal{T} + \tau - 1)}\right)^{\lceil \tau/2 \rceil} < e^{-\frac{\mathcal{T} \cdot \tau}{8(\mathcal{T}+\tau-1)}} < e^{-\frac{\mathcal{T}}{8} \cdot \frac{2}{3}} \leq e^{-\frac{12\ln(n/\epsilon)}{12}} = \frac{\epsilon}{n}.$$

The first inequality follows from Lemma 12 and from (3); the second inequality follows because for all $x \in \mathbb{R}$ we have $(1 - x) < e^{-x}$. Finally, the third and fourth inequalities follow from $\tau \geq 2\mathcal{T}$ and from the fact that $\mathcal{T} \geq 12\ln(n/\epsilon)$, respectively. □

Finally, we are ready to prove the main theorem, showing that the broadcast completes in $O(n \log^2 n)$ rounds with probability at least $1 - n^{-O(1)}$.

THEOREM 14. *If $\mathcal{T} = \lceil 12\ln(n/\epsilon) \rceil$ for some $\epsilon > 0$, the algorithm solves broadcast by time $T = 2 \cdot n \cdot \mathcal{T} \cdot H(n)$ with probability at least $1 - \epsilon$. For $\epsilon = n^{-O(1)}$, we get $T = O(n \log^2 n)$.*

PROOF. For any node v, let t_v be the round in which v first receives the message, or ∞ if v never receives the message. Let t'_v be the first round after t_v in which the number of free rounds greater than t_v is equal to the number of busy rounds after t_v. By Lemma 13, node v has been isolated by round t_v with probability at least $1 - \epsilon/n$. By a union bound argument, the probability that *every* node v has been isolated by t'_v (assuming t'_v is finite) is at least $1 - \epsilon$. We will show that whenever this event occurs, all nodes receive the message before the first time in which the total number of free rounds in the execution equals the total number of busy rounds. Together with Lemma 11, this proves the theorem.

Let τ be the first round in which over the entire interval $[1, \tau]$, the number of free rounds equals the number of busy rounds, and suppose by way of contradiction that every node v was isolated no later than round t'_v (if round t'_v is finite) but some node has not received the message. Let $U \subseteq V$ be the non-empty set of nodes that have not received the message by round $\tau - 1$. Since G is broadcastable, there exists a directed edge (v, u) where $u \in U$ and $v \in V \setminus U$. If we can show that $t'_v \leq \tau$, then by our assumpion that v is isolated by round t'_v, process u receives the message by round τ, contradicting the choice of u.

To that end, assume by way of contradiction that $t'_v > \tau$ (or t'_v is infinite), that is, the number of free rounds in the interval $[t_v, \tau]$ is smaller than the number of busy rounds. By choice of τ we know that the number of free rounds in the interval $[1, \tau]$ is at least the number of busy rounds in the interval $[1, \tau]$. It follows that the number of free rounds in $[1, t_v]$ exceeds the number of busy rounds in $[1, t_v]$, contradicting the minimality of τ. □

8. CONCLUSION

In this paper we introduce dual graphs, a new model for radio networks. Unlike most traditional models for radio networks, the dual graph model allows for *dynamic* interference and unreliable communication. Like traditional models, the dual graph model includes a graph G of reliable communication links; but in addition, unreliable links are represented in the form of a second graph G', whose edges can be deployed against the algorithm by a worst-case adversary. Algorithms for the dual graph model are therefore resilient to interference and noise.

In the current paper we showed that for the broadcast problem, resilience to link failures comes at the cost of higher round complexity: a lower bound of $\Omega(n \log n)$ holds for a setting in which

the reliable model admits an $O(n)$-round deterministic algorithm. Our deterministic upper bound, at $O(n^{3/2}\sqrt{\log n})$ rounds, does not yet match this lower bound; nevertheless, we gave reasonably efficient deterministic and randomized algorithms for broadcast.

A significant part of the difficulty comes from the fact that the network topology is unknown to the processes at the time of the broadcast. In future work we intend to explore *repeated* broadcast in dual graphs, where we hope to improve long-term efficiency by learning the topology of the graph. Topology control in dual graphs is another interesting area for future research.

9. REFERENCES

[1] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg. A lower bound for radio broadcast. *J. Comput. Syst. Sci.*, 43(2):290–298, 1991.

[2] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time-complexity of broadcast in radio networks: an exponential gap between determinism randomization. In *PODC '87: Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, pages 98–108, New York, NY, USA, 1987. ACM.

[3] D. Bruschi and M. Del Pinto. Lower bounds for the broadcast problem in mobile radio networks. *Distrib. Comput.*, 10(3):129–135, 1997.

[4] K.-W. Chin, J. Judge, A. Williams, and R. Kermode. Implementation Experience with MANET Routing Protocols. *SIGCOMM Computer Communication Review*, 32(5):49–59, 2002.

[5] B. S. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, and W. Rytter. Deterministic broadcasting in unknown radio networks. In *Symposium on Discrete Algorithms*, pages 861–870, 2000.

[6] M. Chlebus, L. Gasieniec, A. Ostlin, and J. Robson. Deterministic broadcasting in radio networks. In *the International Colloquium on Automata, Languages and Programming (ICALP)*, 2000.

[7] M. Chrobak, L. Gasieniec, and W. Rytter. Fast broadcasting and gossiping in radio networks. *Journal of Algorithms*, 43:177–189, 2002.

[8] A. Clementi, A. Monti, and R. Silvestri. Selective families, superimposed codes, and broadcasting on unknown radio networks. In *the annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 709–718, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[9] A. Clementi, A. Monti, and R. Silvestri. Round robin is optimal for fault-tolerant broadcasting on wireless networks. *Journal of Parallel Distributed Computing*, 64:89–96, 2004.

[10] A. E. F. Clementi, A. Monti, F. Pasquale, and R. Silvestri. Broadcasting in dynamic radio networks. *J. Comput. Syst. Sci.*, 75(4):213–230, 2009.

[11] A. E. F. Clementi, A. Monti, and R. Silvestri. Round robin is optimal for fault-tolerant broadcasting on wireless networks. *J. Parallel Distrib. Comput.*, 64(1):89–96, 2004.

[12] A. Czumaj and W. Rytter. Broadcasting algorithms in radio networks with unknown topology. *J. Algorithms*, 60(2):115–143, 2006.

[13] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. *Wireless Networks*, 11(4):419–434, 2005.

[14] P. Erdos, P. Frankl, and Z. Furedi. Familes of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51:79–89, 1985.

[15] F. Galčík. Centralized communication in radio networks with strong interference. In *SIROCCO '08: Proceedings of the 15th international colloquium on Structural Information and Communication Complexity*, pages 277–290, Berlin, Heidelberg, 2008. Springer-Verlag.

[16] F. Galčík, L. Gasieniec, and A. Lingas. Efficient broadcasting in known topology radio networks with long-range interference. In *PODC '09: Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 230–239, New York, NY, USA, 2009. ACM.

[17] O. Goussevskaia, T. Moscibroda, and R. Wattenhofer. Local broadcasting in the physical interference model. In M. Segal and A. Kesselman, editors, *DIALM-POMC*, pages 35–44. ACM, 2008.

[18] P. Gupta and P. Kumar. The capacity of wireless networks. *IEEE Transactions on information theory*, 46:388–404.

[19] W. Kautz and R. Singleton. Nonrandom binary superimposed codes. *IEEE Transactions on Information Theory*, 10(4):363–377, 1964.

[20] D. R. Kowalski and A. Pelc. Broadcasting in undirected ad hoc radio networks. *Distrib. Comput.*, 18(1):43–57, 2005.

[21] D. R. Kowalski and A. Pelc. Time complexity of radio broadcasting: adaptiveness vs. obliviousness and randomization vs. determinism. *Theor. Comput. Sci.*, 333(3):355–371, 2005.

[22] F. Kuhn, N. Lynch, and C. Newport. Brief announcement: Hardness of broadcasting in wireless networks with unreliable communication. In *The Annual ACM Symposiun on Principles of Distributed Computing (PODC)*, pages 330–331, 2009.

[23] E. Kushilevitz and Y. Mansour. An $\Omega(D\log(N/D))$ lower bound for broadcast in radio networks. *SIAM J. Comput.*, 27(3):702–712, 1998.

[24] H. Lundgren, E. Nordstr
"o, and C. Tschudin. Coping with Communication Gray Zones in IEEE 802.11b Based Ad Hoc Networks. In *the International Workshop on Wireless Mobile Multimedia*, 2002.

[25] T. Moscibroda and R. Wattenhofer. The complexity of connectivity in wireless networks. In *INFOCOM*. IEEE, 2006.

[26] C. Newport, D. Kotz, Y. Yuan, R. Gray, J. Liu, and C. Elliott. Experimental Evaluation of Wireless Simulation Assumptions. *Simulation*, 83(9):643, 2007.

[27] C. Newport and N. Lynch. Modeling Radio Networks. In *the International Conference on Concurrency Theory (CONCUR)*, 2009.

[28] D. Peleg. Time-efficient broadcasting in radio networks. In *DISC '07: Proceedings of the 21st international symposium on Distributed Computing*, pages 3–4, Berlin, Heidelberg, 2007. Springer-Verlag.

[29] S. Schmid and R. Wattenhofer. Algorithmic models for sensor networks. In *IPDPS*. IEEE, 2006.

[30] P. von Rickenbach, R. Wattenhofer, and A. Zollinger. Algorithmic models of interference in wireless ad hoc and sensor networks. *IEEE/ACM Trans. Netw.*, 17(1):172–185, 2009.