The Cost of Radio Network Broadcast for Different Models of Unreliable Links*

Mohsen Ghaffari
MIT CSAIL
Cambridge, MA
ghaffari@csail.mit.edu

Nancy Lynch
MIT CSAIL
Cambridge, MA
lynch@csail.mit.edu

Calvin Newport
Georgetown University
Washington, DC
cnewport@cs.georgetown.edu

ABSTRACT

We study upper and lower bounds for the global and local broadcast problems in the dual graph model combined with different strength adversaries. The dual graph model is a generalization of the standard graph-based radio network model that includes unreliable links controlled by an adversary. It is motivated by the ubiquity of unreliable links in real wireless networks. Existing results in this model [11, 12, 3, 8] assume an *offline adaptive* adversary—the strongest type of adversary considered in standard randomized analysis. In this paper, we study the two other standard types of adversaries: *online adaptive* and *oblivious*. Our goal is to find a model that captures the unpredictable behavior of real networks while still allowing for efficient broadcast solutions.

For the online adaptive dual graph model, we prove a lower bound that shows the existence of constant-diameter graphs in which both types of broadcast require $\Omega(n/\log n)$ rounds, for network size n. This result is within log-factors of the (near) tight upper bound for the offline adaptive setting. For the oblivious dual graph model, we describe a global broadcast algorithm that solves the problem in $O(D \log n + \log^2 n)$ rounds for network diameter D, but prove a lower bound of $\Omega(\sqrt{n}/\log n)$ rounds for local broadcast in this same setting. Finally, under the assumption of geographic constraints on the network graph, we describe a local broadcast algorithm that requires only $O(\log^2 n \log \Delta)$ rounds in the oblivious model, for maximum degree Δ. In addition to the theoretical interest of these results, we argue that the oblivious model (with geographic constraints) captures enough behavior of real networks to render our efficient algorithms useful for real deployments.

*Research supported in part by: Ford Motor Company University Research Program; AFOSR Contract No. FA9550-13-1-0042; NSF Award No. CCF-1217506; NSF Award No. 0939370-CCF; NSF Award No. CCF-AF-0937274; AFOSR Contract No. FA9550-08-1-0159; NSF Award No. CCF-0726514.

PODC'13, July 22–24, 2013, Montréal, Québec, Canada.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communication; G.2.2 [**Discrete Mathematics**]: Graph Theory—*network problems*

Keywords

radio network; broadcast; dual graph; unreliability

1. INTRODUCTION

Most models used to study algorithms for wireless networks assume *static* links between devices; i.e., the strength of each link remains fixed throughout an execution. This assumption is captured by these models' use of deterministic rules for determining when a message is received. Given a set of transmitters, the receive behavior is fixed. This property is true, for example, of the graph-based *protocol* model (e.g., [4, 2]) and SINR-based *physical* model (e.g., [16]).[1] In real wireless networks, by contrast, links are rarely static. Changes to the environment, interference from unrelated protocols on overlapping spectrum, and even shifting weather conditions can all cause links to exhibit *dynamic* fluctuations in strength (e.g., [18]).

To succeed in establishing a mathematical foundation for the design and analysis of practical wireless network algorithms, the theory community needs to consider wireless models that include dynamic links. A recent series of papers takes up this challenge by studying global broadcast [11, 12], local broadcast [8], and graph algorithms [3] in the context of the *dual graph* model—a generalization of the graph-based protocol model that includes both static and dynamic links.

A defining feature of the dual graph model is that an adversary controls the behavior of the dynamic links (as previously argued [11, 12, 3, 8], simpler assumptions, such as independent loss probabilities, do a poor job of capturing the unpredictable and sometimes highly-correlated nature of dynamic behavior in real networks). Applying the standard terminology from randomized analysis, this existing work on the dual graph model assumes an *offline adaptive adversary.* This adversary type—sometimes also called *strongly adaptive*—can use the execution history and the nodes' random choices for a given round before fixing the link behavior. This definition is strong and therefore, perhaps not surprisingly, most existing radio network results proved with re-

[1]SINR-based models include a *noise parameter*, N, which, in theory could be determined on a per-link basis and potentially change over time. In practice, however, it is almost always treated as a fixed constant.

spect to this adversary type are negative. They show, for example, that both global and local broadcast (defined below) require $\Omega(n)$ rounds [11, 12] in constant-diameter graphs of size n. This is exponentially worse than the $\Theta(\log^2 n)$ round solution of the static protocol model [2, 8]. These lower bounds are not entirely satisfying as they leverage adversary behavior—such as basing link behavior on the outcome of private random choices—that is unrealistically pessimistic.

In this paper, we take the natural next step and derive new upper and lower bounds for global and local broadcast in the dual graph model combined with the two successively weaker adversary types typically considered in randomized analysis: *online adaptive* and *oblivious*. Motivated by the importance of these broadcast problems, our goal is to find a model that can capture the unpredictable behavior of real networks, yet still allow efficient broadcast solutions.

In more detail, the online adaptive adversary—sometimes also called *weakly adaptive*—can use the execution history when deciding the link behavior for a given round, but does not know the nodes' random choices for the round. The oblivious adversary must make all decisions at the beginning of the execution.

The global broadcast problem requires a designated source node to disseminate a message to the entire network. This is arguably the most studied problem in the radio model (see [17]) due to its importance to both theory (it isolates the fundamental difficulty of this setting—multihop contention) and practice (solutions provide key synchronization and search capabilities in real networks). Local broadcast assumes a subset of nodes are provided broadcast messages to deliver to their neighbors in the network graph. In this paper, we focus only on the time for each receiver (i.e., node neighboring a broadcaster) to receive *some* message.[2] This property proves crucial when analyzing local broadcast as a subroutine (e.g., [9]). In addition, this form of the problem reduces to most nontrivial problems in the radio setting as most problems require some communication between nearby nodes to break symmetry. Therefore, lower bounds on local broadcast can extend to many other problems.

Results for the Online Adaptive Dual Graph Model: We begin by proving that there exists a constant-diameter graph of size n such that both global and local broadcast required $\Omega(n/\log n)$ rounds in this graph in the online adaptive dual graph model. Our lower bound uses a simulation-based reduction from an abstract game that leverages the adversary's knowledge of expected behavior.[3]

Results for the Oblivious Dual Graph Model: For the oblivious setting, we first describe a global broadcast algorithm that runs in $O(D \log n + \log^2 n)$ rounds, for network diameter D, which essentially matches the best known results in the static protocol model [2, 10, 7]. The core insight driving this algorithm is that the source can generate a set of random bits after the execution begins (e.g., bits unknown to the adversary) and then include them in the source message. Nodes that have received the message can use the bits to coordinate their probabilistic broadcast behavior. For local broadcast, however, we establish a $\Omega(\sqrt{n}/\log n)$ lower bound. This bound counters the coordination strategy of global broadcast by using the graph topology to isolate key nodes.

Results for the Oblivious Dual Graph Model with Geographic Constraints: Under the assumption of a geographic constraint that generalizes the unit disk graph model (a natural assumption for wireless settings), we describe an algorithm that solves local broadcast in $O(\log^2 n \log \Delta)$ rounds, for maximum degree Δ. This result is only a log-factor slower than the optimal $\Theta(\log n \log \Delta)$ result in the static protocol model [2, 8]. The algorithm use a more involved version of the coordination strategy from our global broadcast bound that leverages the geographic constraint to achieve local coordination. In Figure 1, we summarize all the above results and compare them to existing work.

Contributions.

From a theory perspective, these results provide a complete characterization of broadcast in unreliable radio networks with respect to the classic adversary types—identifying the key relationships between adversary power and the efficiency of these important primitives. From a practical perspective, we argue that our efficient upper bounds for the oblivious adversary are well-suited for real world deployment. The oblivious model can capture the unpredictable behavior observed in real networks. At the same time, its obliviousness to the algorithm's ongoing execution is a tolerable concession, as the adversary is playing the role of environmental forces that are independent of the protocols running in the network. Our best algorithms, therefore, can provide practitioners significantly more robustness without sacrificing efficiency.

Related Work.

The static graph-based radio network model was introduced by Chlamtac et al. [4], and has since been extensively studied (see [17] for examples). In this paper, following recent naming conventions, we call it the *protocol* model. Bar-Yehuda et al. [2] described a randomized distributed broadcast solution that runs in $O(D \log(n) + \log^2 n)$ rounds in the protocol model. This result was later slightly improved to $O(D \log(n/D) + \log^2 n)$ rounds [10, 7], which is optimal [1, 15]. For local broadcast, a slight tweak to the strategy of [2] provides a local broadcast solution the runs in $O(\log n \log \Delta)$ rounds [8]. The dual graph model was introduced by Clementi et al. [5], where it was called the *dynamic fault* model. We independently reintroduced the model in [11] with the *dual graph* name. In this model combined with an offline adaptive adversary, global and local broadcast now require $\Omega(n)$ rounds, even in constant diameter graphs [11]. The closest matching upper bounds require $O(n \log^2 n)$ rounds for global broadcast [12], $O(n)$ rounds for local broadcast.[4] See Figure 1 for a summary of these results and how they compare to the new results in this paper. Other problems, including building graph structures [3] and deterministic broadcast [13] have also been studied in the dual graph model with this strong adversary. A related lit-

[2]The other property studied with respect to these algorithms is the time for a sender to successfully deliver its message to all its receivers. We do not study that property here because we recently proved a strong lower bound in the static protocol model, leaving us no gap to close in our dual graph variants [8].

[3]We note that a similar bound was independently discovered, using a different method, by Cornejo, Ghaffari, and Haeupler. Their version is unpublished.

[4]Local broadcast can always be solved in $O(n)$ rounds using round robin broadcasting on the n node ids.

	Global Broadcast	Local Broadcast
DG + Offline Adaptive	$\Omega(n)$ [11] / $O(n \log^2 n)$ [13]	$\Omega(n)$ [11] / $O(n \log n)$ [8]
DG + Online Adaptive	$\Omega(n/\log n)$	$\Omega(n/\log n)$
DG + Oblivious	$O(D \log n + \log^2 n)$	General Graphs: $\Omega(\sqrt{n}/\log n)$ Geo. Graphs: $O(\log^2 n \log \Delta)$
No Dynamic Links	$\Theta(D \log(\frac{n}{D}) + \log^2 n)$ [10, 1, 15]	$\Theta(\log n \log \Delta)$ [2, 8]

Figure 1: All results in the second and third rows are new results proved in this paper. Existing results are described in the first and last rows along with the relevant citations.

erature studies distributed algorithms in the *dynamic network* model, in which the entire communication topology can change from round to round under different constraints (see [14] for a good survey). These results, however, assume reliable communication among neighbors in each round. The dynamic model studied in this paper, by contrast, assumes concurrent communication yields collisions—making it well-suited for describing radio networks. The exception is the work of Clementi et al. [6], which studies a dynamic network model that preserves the standard radio network collision rules. In this model, they show a tight bound of $\Theta(n^2/\log n)$ rounds for solving broadcast with a strong adversary constrained only to keep the graph connected in a useful manner in each round.

2. MODEL AND PROBLEMS

We define the *dual graph* model, which describes randomized algorithms executing in a synchronous multihop radio network with both static and dynamic links. The model describes the network topology with two graphs on the same vertex set: $G = (V, E)$ and $G' = (V, E')$, where $E \subseteq E'$, and the $n = |V|$ nodes in V correspond to the wireless devices. An *algorithm* in this model consists of n randomized *processes*. An execution of an algorithm in a given network (G, G') begins with an adversary assigning each process to a node in the graph. This assignment is unknown to the processes. To simplify notation, we use the terminology *node* u, with respect to an execution, to refer to the process assigned to node u in the graph in a given execution. Executions then proceed in synchronous rounds. In each round r, each node decides whether to transmit a message or receive based on its randomized process definition. The communication topology in r consists of the edges in E plus *some subset* of the edges in $E' \setminus E$. This subset, which can change from round to round, is determined by an adversary that we call a *link process* (see below). Once a topology is fixed for a given round, we use the following communication rules: a node u receives a message m from v in r, if and only if: (1) u is receiving; (2) v is transmitting m; and (3) v is the only node transmitting among the neighbors of u in the communication topology *fixed by the link process for* r. Notice, the dual graph model is a strict generalization of the static protocol model. In more detail: for $G = G'$, the model is the same as the protocol model described with respect to topology G.

For a given $G' = (V, E')$ and $u \in V$, we use $N_{G'}(u)$ to describe the neighbors of u in E', and define $\Delta = \max\{|N_{G'}(u)| : u \in V\}$. We assume n and Δ are known to processes in advance. Some of our bounds require a geographic constraint on the G and G'. In these cases, we assume the same constraint introduced in [3], which itself is a generalization of the *unit disk graph* property. In more detail, our constraint assumes the existence of a constant $r \geq 1$, such that we can embed the nodes in our graph in a Euclidean plane with distance function d, and, $\forall u, v, u \neq v$: if $d(u, v) \leq 1$ then (u, v) is in G, and if $d(u, v) > r$, (u, v) is not in G'. This constraint says that close nodes can communicate, far away nodes cannot, and for nodes in the *grey zone* in between, the behavior is dynamic and unpredictable. We call a dual graph network topology that satisfies this property a *geographic graph*.

Adversary Types.

In our model, the choice of which edges from $E' \setminus E$ to include in the communication topology each round is determined by an adversary called a link process. In this paper, we study the three classical definitions of such adversaries from the randomized analysis literature: offline adaptive, online adaptive, and oblivious. In more detail, the *offline adaptive* link process, when making a decision on which links to include in a given round r, can use knowledge of: the network topology; the algorithm being executed; the execution history through round $r - 1$; and the nodes' random choices for round r. The *online adaptive* link process weakens this definition such that it no longer learns the nodes' random choices in r before it makes its link decisions for r. The *oblivious adversary*, by contrast, must make all of its link decisions at the beginning of the execution—though it can still make use of the network topology and algorithm description in generating this behavior.

In this paper, when we refer to the "⟨*adversary type*⟩ *dual graph model*", we mean the dual graph model combined with link processes that satisfy the ⟨*adversary type*⟩ constraints.

Global and Local Broadcast.

We study the global and local broadcast problems. The global broadcast problem assumes a designated source node is provided a message. The problem is solved when it has disseminated the message to the entire network. The local broadcast problem assumes some subset of nodes $B \subseteq V$ are provided a message. Let R be the set of nodes with at least one neighbor in B by G. The problem is solved when every node in R has received at least one message from a neighbor in B. Both problems assume G is connected. When we say a randomized algorithm solves one of these problems, we require that it solves it with high probability (i.e., probability at least $1 - 1/n$).

3. ONLINE ADAPTIVE DUAL GRAPH MODEL

Previous work proved the existence of constant-diameter graphs where both global and local broadcast require $\Omega(n)$ rounds in the offline adaptive dual graph model [11]. Here we prove a similar bound holds when we weaken the adver-

sary to the online adaptive model. This result demonstrates that the difficulty of broadcasting in an adaptive dual graph model is not dependent on the strong assumption that the link process knows random choices in advance. Formally:

THEOREM 3.1. *There exists a constant-diameter dual graph network such that every algorithm requires $\Omega(n/\log n)$ rounds to solve global and local broadcast in this network in the online adaptive dual graph model.*

The proof of our theorem reduces an abstract game, called *β-hitting*, to broadcast. We show that an efficient broadcast solution allows a player to efficiently win the β-hitting game by simulating the solution in a specific type of constant-diameter network we call *dual clique*. We then leverage an existing bound on β-hitting to bound broadcast in the dual clique network.

The β-Hitting Game: The game is defined for integer $\beta > 0$. There is a player represented by a probabilistic automaton $\mathcal{P}$. At the beginning of the game, an adversary chooses a *target* value, $t \in [\beta]$, which it keeps secret from the player. The $\mathcal{P}$ automaton executes in rounds. In each round, it can output a guess from $[\beta]$. The player wins the game when $\mathcal{P}$ outputs t. The only information it learns in other rounds is that it has not yet won the game. In previous work [11], we bound this game as follows:

LEMMA 3.2 (ADAPTED FROM [11]). *Fix some $\beta > 3$ and $k, 1 \leq k \leq \beta - 2$. There does not exist a player that solves β-hitting in k rounds with probability greater than $k/(\beta - 1)$.*

The Dual Clique Network: Partition the n nodes in V into two equal sized sets, A and B. Connect the nodes in A (resp. B) to form a clique in G. Connect a single node $t_A \in A$ to a single node $t_B \in B$, forming a *bridge* between the two cliques. Let G' be the complete graph over all nodes. Notice, this graph has constant diameter. It is also a geographic graph (which strengthens our lower bound).

We now proceed with the proof of our main theorem:

PROOF OF THEOREM 3.1. Fix some broadcast algorithm $\mathcal{A}$. The bulk of our proof is dedicated to proving the following claim: if $\mathcal{A}$ solves either global or local broadcast in $f(n)$ rounds, then we can construct a player $\mathcal{P}_\mathcal{A}$ that solves the β-hitting game in $O(f(2\beta)\log\beta)$ rounds, with probability at least $1 - 1/\beta$. Once established, our theorem statement follows directly from this claim and Lemma 3.2: if $f(n) = o(n/\log n)$, then $\mathcal{P}_\mathcal{A}$ would solve β-hitting in $o(\beta)$ rounds with probability at least $1 - 1/\beta$—violating Lemma 3.2 which established the necessity of $\Omega(\beta)$ rounds to achieve this success probability.

We proceed argue the following claim. Our strategy is to have $\mathcal{P}_\mathcal{A}$ simulate a collection of 2β nodes in a dual clique network of size $n = 2\beta$. The player uses the behavior of the simulated nodes to specify its guesses for the hitting game. Below, we begin by describing the network it simulates, then its rules for determining guesses, and finally we prove these rules solve the game with the needed time complexity.

Simulated Network: To simulate nodes in the dual clique network the player must assign the processes to nodes in the graph. In more detail, let $\{1, ..., 2\beta\}$ be the ids of the $n = 2\beta$ nodes we simulate. The player places nodes 1 to β in clique A and $\beta + 1$ to 2β in clique B. Let t be the target for this instance of the hitting game. The network the player will simulate assigns node t to t_A and $t + \beta$ to t_B. That is, the nodes on either side of the bridge have their ids correspond to the target for the hitting game. Of course, *the player does not know t in advance*, but we will prove that its simulation remains consistent with this particular instantiation of the dual clique network. For the remainder of the proof, we call this assignment of ids to the dual clique network our *target* network.

Guess Generation Rules: We now describe how to connect the player's simulation of the target network with its guesses or the β-hitting game. Fix some simulated round. The player begins by simulating the broadcast behavior of its simulated nodes in this round. If $\mathcal{A}$ is a global broadcast algorithm, it assumes that node $1 \in A$ is the global broadcast message source; if it is local broadcast, it puts all nodes in A in the broadcast set. Based on this broadcast behavior, the player will generate a series of guesses for the hitting game. If none of these guesses solve the game, it will then finish its simulation of the round by simulating the receive behavior, and then moving on to the next simulated round. Therefore, every simulated round generates a variable number of hitting game guesses. Here we describe how it uses the simulated broadcast behavior to generate its guesses. In the next piece of the proof we will describe how it simulates the receive behavior if its guesses fail.

Let X_A be the nodes from A that broadcast in our simulated round, and let X_B be the nodes from B that broadcast in the round. Let $X = X_A \cup X_B$. Finally, let S be the state of the nodes at the *beginning* of this round. This state includes the execution history through $r-1$, but does not include the random bits the nodes will use in round r. It captures the information an online adaptive link process can use to make link decisions in this round. We have our player calculate the expected value of $|X|$ given S. If $\mathbb{E}[|X| \mid S] > c \log \beta$, for a constant $c \geq 1$ which we will fix later, then the player labels the round *dense*, otherwise it labels the round *sparse*.

Now we are ready to generate our guesses. If the round is dense and, once the player simulates the broadcast behavior, $X = \{i\}$: then the player guesses, one by one, all values from 1 to β (guaranteeing that it will win the hitting game). If the round is dense and $|X| \neq 1$: then the player makes no guesses. If the round is sparse: the player guesses the ids in X, one by one, subtracting β from the ids in $X \cap X_B$ to transform them into values in $[\beta]$. In other words, the player's guesses are a combination of the *expected* and *actual* broadcast behavior in the simulated round.

Simulating Receive Behavior: If none of the guesses generated with the above process win the hitting game, the player must now conclude its simulated round by simulating the receive behavior in a manner that is valid for our target network and the online adaptive link process constraints. To do so, the player must first decide which edges from $G' \setminus G$ to include in the communication topology (that is, it plays the role of the link process). In our simulation, the player uses the rule to determine this decision: if the rounds in dense, it includes all G' edges in the topology; if the round is sparse, it includes no G' edges between A and B. At this point, we must be careful to confirm that these link process decision made by the player satisfy the constraints of an online adaptive link process. If, for example, the player used more information than is allowed by an online adaptive link process, it would be simulating a network more difficult than what is expected by $\mathcal{A}$ and therefore $\mathcal{A}$ is not required

to work correctly. As mentioned above, however, an online adaptive link process *is* able to determine if a round is dense or sparse as this requires only the state at the beginning of the round, not the random choices made during the round.

After fixing the G' topology, the link process has almost enough information to simulate the receive behavior. The only piece of the graph topology it is missing is the edge between in G between t and $t+\beta$ (as it does not know t, it does not know the endpoints of this edge). Our approach is to have the player simulate receive behavior under the assumption that there is *no* edge in G between A and B. We will later argue that the behavior it simulates in this incomplete network is consistent with what would occur in our target network that includes this edge between the cliques.

Proving the Validity of the Simulation: We now argue by induction on the simulated round number that this simulation remains a valid simulation of $\mathcal{A}$ in our target network until the player wins the hitting game. For our hypothesis, assume that the first $r > 0$ rounds of the simulation are valid, and the player has not yet won the hitting game. For our step, we will show that in simulated round $r+1$, either the player wins the hitting game or this round is also valid.

By our inductive hypothesis, we know the broadcast behavior simulated at the beginning of round $r+1$ is valid. If the corresponding guesses win the hitting game, we are done. Assume, therefore, that the guesses do not win. We consider the possible cases for the simulated receive behavior and argue all case are valid for our target network.

If the round is dense, then we know $|X| \neq 1$ (if $|X| = 1$ then the player would have guesses all values and won the game in the preceding guess phase). In this case, the player would simulate no messages being received, as the G' edges included in the topology make it a complete graph (if multiple messages broadcast, all receive a collision). The fact that the player did not know the edge between t and $t+\beta$ does not matter here.

If the round is sparse, then the player simulates receive behavior as if the two cliques are isolated in the communication topology. This receive behavior will be the same as in our target network (which includes an edge between t and $t+\beta$) if neither t nor $t+\beta$ broadcast. On the other hand, if t or $t+\beta$ *do* broadcast, our simulated receive behavior would no longer necessarily be valid. However, in this case, we would have guessed t during the preceding guesses and won the hitting game.

Proving that a Valid Simulation Wins the Hitting Game: Having established that our simulation is valid, we must now argue that a valid simulation of our broadcast algorithm will lead our player to eventually win the game. By assumption, $\mathcal{A}$ solves either local or global broadcast in the dual clique network with any online adaptive link process. Because we argued that our simulation of $\mathcal{A}$ is valid, it too will eventually solve broadcast (or win the hitting game before it gets a chance to finish). Notice, if $\mathcal{A}$ solves broadcast, it follows that at least one message must pass between A and B. (Recall from above: for global broadcast, we put the source in A, and for local broadcast, we placed all nodes in A, including the endpoint of the bridge, in the broadcast set.) Given the link process behavior used in our simulation, there are only two cases where this is possible: (1) a round in which just a single node broadcasts; or (2) a sparse round in which t or $t+\beta$ broadcast. In both cases, when we get to such a round, we will guess t and win the game.

Bounding the Number of Guesses Per Simulated Round: Because $\mathcal{A}$ solves broadcast with high probability, we have established that our player simulating $\mathcal{A}$ will also solve the game with high probability. In particular, with probability at least $1-1/(2\beta)$ (we are substituting 2β for n in the definition of high probability). Our final step is to bound *how long* this takes. We know that $\mathcal{A}$ finishes in $f(2\beta)$ rounds. But because we can have multiple guesses per simulated round, the number of rounds required by our player to win the hitting game might be much longer. We will show that with sufficient probability the player never need more than $O(\log\beta)$ guesses per simulated round.

Returning to the guessing rules, we see that there are two cases where the player might have to guess multiple values for a given simulated round. The first case is occurs when only a single node broadcasts during a dense round. If this occurs, then the player guesses all β values. Recall, however, that if a round is dense, then $\mathbb{E}[|X| \mid S] > c\log\beta$. For a sufficiently large constant c, we can apply a Chernoff bound to prove that the probability that there is 1 node broadcasting is less than $1/(2\beta)^4$.

The second case with multiple guesses occurs when the round is sparse. In this instance, all broadcasters are guessed. For a sparse round, however, the expected number of broadcasters is low; i.e., $\mathbb{E}[|X| \mid S] \leq c\log\beta$. Plugging in the c we used for case 1, there exists some other smaller constant c' such that the probability there are more than $c'\log\beta$) broadcasters in a sparse round is also less than $1/(2\beta)^4$. By a union bound, the probability we have to guess more than $c'\log\beta$ values in a given round is less than $1/(2\beta)^3$. We can assume w.l.o.g. that our broadcast algorithm requires no more than $(2\beta)^2$ rounds,[5] so by another union bound, the probability that we have too many broadcasters in *any* simulated round is bounded by $1/(2\beta)$.

We can now piece together our two failure probabilities. We know that $\mathcal{A}$ solves global broadcast in $f(n)$ rounds with probability at least $1-1/n$. Therefore, it will solve broadcast in our target network in $f(2\beta)$ rounds with probability at least $1-1/(2\beta)$. It follows that it fails to solve broadcast with probability no more than $1/(2\beta)$. As shown above, it fails to bound its guesses per round to $O(\log\beta)$ also with probability no more than $1/(2\beta)$. Therefore, the probability that it solves the game in $O(f(2\beta)\log\beta)$ rounds, is at least $1-1/(2\beta)+1/(2\beta) = 1-1/\beta$, as required by our claim. □

4. OBLIVIOUS DUAL GRAPH MODEL

Having just proved that broadcast cannot be solved efficiently in the online adaptive setting, we turn our attention to the oblivious dual graph model. As argued in the introduction, this model is important because the adversary is powerful enough to replicate the unpredictable behavior of real radio networks. Upper bounds proved in this model, therefore, should still hold in real deployment.

4.1 Global Broadcast Upper Bound

We begin by describing an algorithm that solves global broadcast in $O(D\log n+\log^2 n)$ rounds in the oblivious dual graph model. Notice, this bound matches the well-known solution of Bar-Yehuda et al. [2] in the protocol model. In fact, our new upper bound is based on the classic result

[5]We can always solve broadcast among 2β nodes in $(2\beta)^2$ rounds by doing round robin broadcast 2β times.

of [2]. This existing global broadcast algorithm is based on a *decay* subroutine that has nodes with the message cycle (in a coordinated manner) through the $\log n$ probabilities: $\{1/2, 1/4, ..., 2/n, 1/n\}$. For each potential receiver, one of these probabilities is appropriate for the number of broadcasters it neighbors. This strategy works well for advancing the message in the protocol model, but it can be attacked by an oblivious adversary because the fixed schedule of broadcast probabilities allows it to calculate in advance the expected broadcast behavior, and choose dynamic link behavior accordingly (as the online adaptive adversary did in our bound from Section 3). In our new protocol, we sidestep this attack by having the source generate random bits at the beginning of the execution, and then append them to the broadcast message. Nodes that have received the message can use these bits to permute the order in which they visit the decay probabilities. From the perspective of the adversary, these permutations are random, thwarting his attack. The formal description follows (to simplify notation, we assume that log is base-2 and n is a power of 2):

Permuted Decay Subroutine.

The *permuted decay* subroutine, used by our global broadcast algorithm, is called with a broadcast message m, a string S of $\gamma \log n \log\log n$ *permutation bits*, and an integer parameter $\gamma \geq 1$. The routine runs for $\gamma \log n$ rounds. During each round, it selects a value $i \in [\log n]$ using $\log\log n$ new bits from S. It then broadcasts m with probability 2^{-i}.

Global Broadcast Algorithm.

Our global broadcast algorithm works as follows. The source, provided message m', creates a new message $m = \langle m', S\rangle$, where S is a collection of $32\log^2 n \log\log n$ bits generated with uniform and independent randomness *after* the execution begins. In the first round, the source broadcasts m to its neighbors. At this point, the source's role in the broadcast is finished. For every other node u, on first receiving a message $\langle m', S\rangle$ in round r, it waits until the first round $r' \geq r$, where r' *mode* $16\log n = 0$, and then calls *permuted-decay*$(m, 16, s)$, $2\log n$ times in a row, where each time s includes $16\log n \log\log n$ new bits from S.

THEOREM 4.1. *The algorithm described above solves global broadcast in $O(D\log n + \log^2 n)$ rounds in the oblivious dual graph model.*

To prove our theorem, we note that our global broadcast algorithm is the same as the algorithm in [2] with the exception that we replaced decay with permuted decay. Fortunately, the existing proof from [2] treats the decay subroutine as a black box, requiring only that a node receives a message from some broadcasting neighbor with probability greater than 1/2 after each call to the subroutine. To prove our above theorem, therefore, it is sufficient to prove that the same property holds for permuted decay. We accomplish this with the below lemma (which, technically, replaces part *ii* of Theorem 1 of [2]):

LEMMA 4.2. *Fix some node u, constant $\gamma \geq 16$, message m, string s of $16\log n \log\log n$ bits generated with uniform and independent randomness after the execution begins, and sets I_G and $I_{G'}$, where I_G is a non-empty subset of u's G neighbors and $I_{G'}$ is a subset of u's G' neighbors. Assume the nodes in $I = I_G \cup I_{G'}$ call* permuted-decay(m, γ, s) *during the same round and all other neighbors of u remains silent. Node u will receive m from a node in I during this instance of* permuted-decay *with probability greater than* 1/2.

PROOF. Fix some round r during the decay instance. Let $I_r \subseteq I$ be the subset of nodes in I connected to u in the topology selected by the oblivious link process for this round. Notice, by definition $I_G \subseteq I_r$, so $I_r \neq \emptyset$. If $|I_r| > 1$, then let our *target* $i' = \lfloor \log |I_r| \rfloor$. Otherwise, let $i' = 1$. All nodes that called decay will select the same i as they all use the same permutation bits from m to select i. By assumption, the choice of i is random and, because the bits were generated after the execution begins, independent of I. Therefore, with probability at least $1/\log n$, $i = i'$.

In a round where $i = i'$, we can bound the probability p_u that u receives a message from I, as follows: $p_u = \sum_{v,\in I_r} \frac{1}{2^{i'}}(1 - \frac{1}{2^{i'}})^{|I_r \setminus \{v\}|} > \frac{|I_r|}{2^{i'}}(\frac{1}{4})^{\frac{|I_r|}{2^{i'}}} \geq \frac{1}{16}$.

For the final step of the reduction, we note that $i' \in (\log |I_r|/2, \log |I_r|]$, and used the largest possible value for i' for the first term and the smallest for the second, to ensure we end up with a lower bound.

Combining our two observations, we see that u receives the message in any given round with probability greater than $1/(16\log n)$ Therefore, the probability that it fails for γ consecutive rounds of the subroutine is bounded as $(1 - \frac{1}{16\log n})^{\gamma \log n} < e^{-\gamma/16} < 1/2$ as required by our lemma. □

4.2 Local Broadcast Lower Bound

The strategy used by our global broadcast upper bound in Section 4.1 does not directly apply to the local broadcast setting, as we can no longer assume that all nodes needing to broadcast are coordinated by a common broadcast message. Here we show that *no* strategy can achieve efficient local broadcast in the oblivious model, by proving a $\Omega(\sqrt{n}/\log n)$ lower bound. This result establishes a strict separation between these two problems in this model:

THEOREM 4.3. *There exists a dual graph network such that every algorithm requires $\Omega(\sqrt{n}/\log n)$ to solve local broadcast in this network in the oblivious dual graph model.*

We use the same proof structure (and much of the proof argument) from our lower bound for the online adaptive model in Section 3. That is, we reduce the β-hitting game to solving local broadcast in a particular type of dual graph network, by showing how solve β-hitting by simulating a local broadcast algorithm in this network. Because we have a weaker adversary, however, our task is complicated. We are now required to use a different network type and more involved simulation strategy.

The Bracelet Network: Select from among the n nodes two non-intersecting subsets of $\sqrt{n/2}$ nodes each: $A = \{a_1, a_2, .., a_{\sqrt{n/2}}\}$ and $B = \{b_1, b_2, .., b_{\sqrt{n/2}}\}$. As in the dual clique network, connect some $a_t \in A$ to some $b_t \in B$ in G. For each $a_i \in A$, construct a line of length $\sqrt{n/2}$ that contains a_i, labeling the nodes as $a_i, a_{i,2}, a_{i,3}, ..., a_{i,\sqrt{n/2}}$. Connect each consecutive pair in the line in G. Do the same for each b_i in B (now labeling the nodes $b_i, b_{i,2}, b_{i,3}$, and so on). We call each such line a *band*. Because we require our graphs to be connected in G, also connect the endpoints of these bands (i.e., each node labelled $a_{i,\sqrt{n/2}}$ or $b_{i,\sqrt{n/2}}$) into a clique in G. Finally, add G' edges between every pair

$(a_i, b_j), a_i \in A, b_j \in B$. Notice, this yields $2(\sqrt{n/2})^2 = n$ total nodes. We call this network a *bracelet* network, as it is defined by $\sqrt{n/2}$ bands in G edge, which are connected at one end by a single G edge (between a_t and b_t), forming a clasp.

Isolated Broadcast Functions: Our reduction argument leverages the following insight: nodes in A and B must behave independently of each other until common information can reach both. Due to the length of their bands, this takes a while. During this time, therefore, an oblivious adversary can do a good job of estimating their broadcast behavior. To formalize this strategy, we define a *support sequence* to be a bit sequence of sufficient size to contain all the bits needed for the nodes in a band in the bracelet network to resolve their random choices for $\sqrt{n/2}$ rounds. We can represent this information as a bit string of length $(\delta n)/2$, where δ is the maximum number of random bits needed by a node in a single round (δ bits per round for $\sqrt{n/2}$ rounds for $\sqrt{n/2}$ nodes in a band yields $(\delta n)/2$ total bits). We also say an execution of an algorithm in the bracelet network is *isolated* with respect to a node $u \in A \cup B$ through round $r > 0$, if node u only receives messages from its neighbor in the band through the first r rounds.

Leveraging these definitions, we prove the existence of a useful formalism:

LEMMA 4.4. *Fix some algorithm $\mathcal{A}$ and node $u \in A \cup B$ from the bracelet network. We can construct an* isolated broadcast function $f_{\mathcal{A},u} : \{0,1\}^{(\delta n)/2} \times \{1, ..., \sqrt{n/2}\} \to \{0,1\}$ *that satisfies the following property: for any support sequence γ and round $r \leq \sqrt{n/2}$: $f_{\mathcal{A},u}(\gamma, r) = 1$ if and only if node u would broadcast in round r of an isolated execution where u's band uses the random bits described by γ.*

PROOF. We can construct $f_{\mathcal{A},u}$ by simulating node u in the bracelet network as follows: For a given γ and r, we can calculate $f_{\mathcal{A},u}(\gamma, r)$ by running an r round simulation of nodes in u's band, where: in the first round of the simulation, initialize all nodes in u's band with the first round bits from γ; in the second round, simulate nodes $u, u_2, u_3, ..., u_{\sqrt{n/2}-1}$; in the third, $u, u_2, u_3, ..., u_{\sqrt{n/2}-2}$; and so on, until round r. The simulated behavior of u in round r determines the output of the broadcast function for that round. This approach drops each node from the simulation right before its externally observable behavior (e.g., messages sent) could possibly be influenced by a node from outside the band (recall that the definition of our network connects the endpoints of the bands together in a clique). We can only maintain the simulation for $\sqrt{n/2}$ rounds, because, at that point, information from outside u's band could have made it u, affecting its behavior in a way we cannot capture in a simulation that know only about the behavior of the band. □

The key property regarding isolated broadcast functions is that they are independent. That is, the behavior of one function provided a randomly generated support sequence is independent of the behavior of another function provided a different randomly generated support sequence. This independence is a direct consequence of the graph topology (and would be impossible to achieve if we had to satisfy a geographic constraint). The following lemma proves, therefore, that calling these functions multiple times with different random bits should generate similar outcomes:

LEMMA 4.5. *Fix a constant $x \geq 1$ and an array F of $k \geq 1$ isolated broadcast functions. Construct two arrays S_1 and S_2, each consisting of k support sequences generated with uniform and independent randomness. For $r \in [\sqrt{n/2}]$, let $Y_r^1 = \sum_i^k F[i](S_1[i], r)$ and $Y_r^2 = \sum_i^k F[i](S_2[i], r)$. The following two properties hold with probability at least $1 - n^{-x}$: For every $r \in [\sqrt{n/2}]$: (1) if $Y_r^1 > 16(x+1)\ln n$ then $Y_r^2 \geq 2$; and (2) if $Y_r^1 \leq 16(x+1)\ln n$ then $Y_r^2 \leq 64(x+1)\ln n$.*

PROOF. Consider some isolated broadcast function $F[i]$ and round r, $i \in [k], r \in [\sqrt{n/2}]$. Notice that for a randomly generated support sequence γ, $F[i](\gamma, r)$ behaves as an indicator variable X_i where $Pr[X_i = 1] = p_i$ and $Pr[X_i = 0] = 1 - p_i$, for some probability p_i we can determine based on the definition of the broadcast function. Crucially, we note that X_i and X_j, for $i \neq j$, defined with independent support sequences, are independent. Let $Y = \sum_i^k X_i$; i.e., the outcome of a trial where we call each function in F with a randomly generated support seqeuence. Notice that Y_r^1 and Y_r^2 describe the value of Y for two different trials. To prove our theorem statement, therefore, it is sufficient to show that it is unlikely that Y will differ by too much between any pair of trials. We turn to Chernoff to aid in this effort. To simplify notation, in the following, let $c = 16(x+1)$.

We begin by bounding the probability that property 1 fails to hold. Let $\mu = \mathbb{E}[Y]$. There are two cases to consider. In the first case, μ is closer to Y_r^1 then Y_r^2. It follows that $\mu > (c/2)\ln n$ and Y_r^2 is less than half the expectation. A Chernoff Bound tells us that: $Pr[Y_r^2 < (1/2)\mu] \leq e^{(-\mu)/8} < e^{(-c/16)\ln n} = n^{-(x+1)}$.

The second case is where μ is closer to Y_r^2 then Y_r^1. Here, we know $\mu \leq (c/2)\ln n$ and Y_r^1 is at least a factor of 2 greater than the expectation. Applying a Chernoff Bound to this direction tells us: $Pr[Y_r^1 \geq 2\mu] \leq e^{(-\mu)/3} < e^{(-c/6)\ln n} < n^{-(x+1)}$.

In other words, for any given round r, the probability that we violate property 1 is no more than $n^{-(x+1)}$. A union bound over $\sqrt{n/2}$ rounds gives us a final probability of failure in at least one round that is less than n^{-x}, as needed.

The argument for property 2 proceeds symmetrically, except now our μ is bounded around $2c\ln n$ in our argument. This only decreases the probability of violating the properties, so the final probability upper bound of n^{-x} still holds. □

Now we are ready to prove our main theorem:

PROOF OF THEOREM 4.3. At a high-level, we apply the same argument as in Theorem 3.1. If $\mathcal{A}$ terminates quickly in the bracelet network, then we can create a player $\mathcal{P}_{\mathcal{A}}$ that simulates $\mathcal{A}$ in this network, to solve the β-hitting game quickly. The bound on β-hitting from Lemma 3.2 provides our bound for $\mathcal{A}$. Instead of recreating the entire proof from Theorem 3.1, we will focus here only where things differ.

In more detail, the player simulates in $\mathcal{A}$ in the bracelet network with a_t and b_t corresponding to the hitting game guess. It will use the behavior of nodes in its simulation to determine guesses for the hitting game. The player does not know a_t and b_t, but, as in the previous proof, we will show that this lack of knowledge does not matter, as the player will win the hitting game before this piece of the topology can affect its simulation.

In simulating each round of $\mathcal{A}$, the player must simulate the oblivious link process. To do so, it first constructs the isolated broadcast functions corresponding to all $2\sqrt{n/2}$

nodes in $A \cup B$ (as in Lemma 4.4). Let F be an array of these functions. It then generates, with uniform and independent randomness, a support sequence for each function. Let S be the array consisting of these $2\sqrt{n/2}$ support sequences, one for each function in F. Using F and S, the player then labels each of the first $\sqrt{n/2}$ rounds as *dense* or *sparse* as follows: if the number of functions in F that output 1 in the round, when called with the sequences in S, is greater than $c \ln n$ (for some constant c we will define later), then the round is dense, otherwise it is sparse.

Once these labels are determined the player chooses the topology and generates as in the proof of Theorem 3.1. Notice, the simulation of the link process here satisfies the constraints of an oblivious adversary, as the isolated support functions can be constructed and simulated for all rounds before the execution begins. We are arguing, in other words, that the behavior of these functions (which capture the broadcast behavior of nodes in $A \cup B$) cannot change much based on the random bits supplied as input (which correspond to the random choices of nodes in a particular execution). This allows an oblivious adversary to simulate the behavior of these specific nodes, for a constrained number of rounds, before the execution begins.

Applying Lemma 4.5, we can show that, with high probability, that for the first $\sqrt{n/2}$ rounds: if multiple nodes in $A \cup B$ broadcast if the round was dense, and $O(\log n)$ broadcast if the round was sparse. These are the key properties needed by the argument in Theorem 3.1 to prove that: (a) the player never needs more than $O(\log n)$ hitting game guesses per simulated round; and (b) the player's lack of knowledge of a_t and b_t will not affect the validity of is simulation in any round before it has won the hitting game. We conclude the proof by selecting our constants carefully such that, after the needed union bounds, we end up with sufficiently high probability as success (see the full version of the proof of Theorem 3.1 for an example of working through these values). □

4.3 Local Broadcast Upper Bound

In Section 4.2, we proved a negative result: $\Omega(\sqrt{n}/\log n)$ rounds are necessary to solve local broadcast in the oblivious model. Notice, however, that this proof argument relied on local neighborhoods with large independence numbers—a property that is unlikely to occur in topologies generated by omnidirectional wireless broadcast. With this in mind, we assume in this section the geographic constraint defined in Section 2. Under this constraint, we describe an algorithm that solves the problem in $O(\log^2 n \log \Delta)$ rounds, which is within a log-factor of the optimal solution in the static protocol model.

Local Broadcast Algorithm.

Our algorithm executes in two stages: *initialization* and *broadcast*. The initialization stage locally disseminates shared randomness to coordinate nearby nodes. The broadcast stage uses these shared bits to efficiently solve local broadcast.

In more detail, the initialization stage divides rounds into $\log \Delta$ *phases*, each consisting of $O(\log^2 n)$ rounds. All nodes begin the stage *active*. During the first round of a given phase $i \in [\log \Delta]$, each node that is still active elect itself a *leader* with probability $2^{-(\log \Delta - i + 1)}$ (i.e., we use the probabilities: $1/\Delta, 2/\Delta, ..., 1/4, 1/2$, as the phases advance). Each leader then generates a *seed* consisting of $O(\log^3 n (\log \log n)^2)$ bits, selected with uniform and independent randomness. It then *commits* to this seed. During the remaining $O(\log n)$ rounds of the phase, each leader broadcasts its seed in each round with probability $1/\log n$. At the end of the phase, the leaders become *inactive*. Any node that was active but *not* a leader during the phase, and that received at least one seed, will commit to the first seed it received and become inactive as well. The only nodes remaining for the next phase, therefore, are those that were active, not a leader, and did not receive a seed message. If a node ends the initialization stage still active, it generates its own seed and commits to it. Therefore, at the end of this stage, all nodes have committed.

The broadcast stage has each node in B (e.g., node with a message to broadcast to its neighbors) execute the permuted decay subroutine from Section 4.1, $O(\log^2 n)$ times in a row. We call each call to decay an *iteration*. For each iteration, each node in B (i.e., node with a local broadcast message) decides to participate in the iteration with probability $1/\log n$. It use $\log \log n$ bits from its seed to make this random choice, so all nodes with the same seed make the same participation decision for each iteration. If a node decides to participate, it runs permuted decay providing it the needed $\Theta(\log n \log \log n)$ bits also from its seed. Therefore, all nodes with same seed will run permuted decay with the same permutation bits. If a node decides not to participate in an iteration, it does nothing until the next iteration begins.

THEOREM 4.6. *The algorithm described above solves local broadcast in $O(\log^2 n \log \Delta)$ rounds in geographic graphs in the oblivious dual graph model.*

To prove our theorem, we first highlight a property of the geographic graphs first established in [3]. Given such a geographic dual graph, we can partition the nodes into *regions* $\mathcal{R} = \{R_1, R_2, ..., R_n\}$, such that all nodes in the same region are connected in G, and for any given region R_i, the number of *neighboring* regions (i.e, regions that contain a G'-neighbor of a node in R_i) is bounded by some constant γ_r, dependent on the value of r. We will use this region decomposition in the remainder of our analysis.

We begin our analysis by focusing on the initialization stage. In the following, for a given phase j and region $R_i \in \mathcal{R}$, let $a_i(j)$ describe the number of active nodes in R_i at the beginning of phase j, and $\ell_i(j)$ describe the number of leaders elected in R_i in the first round of j. We say R_i is *active* in phase j if $\ell_i(j) > 0$. We say a given phase j is *good* if for all $R_i \in \mathcal{R}$, $\ell_i(j) < c \log n$, where c is some constant we fix later. We use p_j, for phase j, to describe the leader election probability associated with this phase. We use $P_{i,j} = a_i(j)p_j$ to describe the leader election probability sum for R_i in j.

The following three lemmas establish that the initialization phase provides the needed density of seeds.

LEMMA 4.7. *With probability at least $1 - 1/n^7$, the following holds for every good phase j and region $R_i \in \mathcal{R}$ that is active in j: At the end of phase j, every node in R_i has committed to a seed and is inactive.*

PROOF. By assumption, the phase is good. It follows that there are no more than $\gamma_r c \log n = O(\log n)$ leaders

in this phase within range of nodes in R_i. We also assume that R_i is active in this phase, so we know it includes at least one node, $u \in R_i$, this is a leader. In each round, u broadcasts with probability $1/\log n$. In each such round, it succeeds in delivering its message to all nodes that are still active in R_i with probability p, bounded as: $p \geq 1/\log n(1-1/\log n)^{\gamma_r c \log n} > 1/(4^{\gamma_r c}\log n) = \Omega(1/\log n)$.

The phase lasts for $c' \log^2 n$ rounds. Therefore, the probability that u fails in every round of the phase is bounded as $(1-p)^{c' \log^2 n} < e^{-\frac{c'}{4^{\gamma_r c}} \log n}$, which, for sufficiently large constant c', is less than $1/n^9$. A union bound over all regions (of which there can be no more than n) and all good phases, provides us with a result that holds with probability $1/n^7$, as required by the lemma. □

Next, we leverage this property to prove that good phases are ubiquitous. The intuition behind the following result is that before the leader election probabilities can get large enough to elect more than $c \log n$ leaders in a region (eliminating goodness in the network), that region passes through a phase where the probability is *just right* for us to apply the previous lemma and render all nodes inactive.

LEMMA 4.8. *With probability at least $1-1/n^3$, every initialization stage phase is good.*

PROOF. Fix some phase j. Let region R_i be the region that maximizes $P_{i,j}$. Recall that $c \geq 1$ is the constant used in the definition of good. We begin the proof by establishing the following claim: there exists a constant $c', 1 \leq c' < c$, such that if $P_{i,j} \leq c' \log n$, then phase j is good.

To establish this claim, we first note that region i has the largest expected value of ℓ for this phase. To bound this expectation, let A_i be the active nodes in R_i at the beginning of j, and let X_u, for each $u \in A_i$ be the indicator variable that is 1 if u elects itself leader in this phase, and is otherwise 0. It follows:

$$\mathbb{E}[\ell_i(j)] = \mathbb{E}[\sum_{u \in A_i} X_u] = \sum_{u \in A_i} \mathbb{E}[X_u] = P_{i,j} \leq c' \log n.$$

Chernoff tells us that if we fix constant c to be appropriately large compared to c', then the probability that $\ell_i(j) > c \log n$ is less than n^{-6}. Notice, the expectation for the other regions is no larger than the expectation for region i, therefore this same bound applies for any region during this phase. Finally, a union bound tells us that this holds for every region and every phase, with probability at least $1-n^{-4}$. Moving forward, assume this property holds.

We are left to prove that the probability a leader election sum ever exceeds $c' \log n$ is small. To do so, let us consider the first region to exceed this sum. Say, region R_i in phase $j+1$. Notice, the leader election sum in R_i can at most double between j and $j+1$ (the election probability doubles between phases but the number of active nodes can never increase). For R_i to exceed $c' \log n$ in some phase $j+1$, therefore, it must first spend phase j with $c'/2 \log n < P_{i,j} \leq c' \log n$. We will now show that the probability R_i survives phase j without having all nodes become inactive is small.

In more detail, by our above assumption, we know j is a good phase (as $j+1$ would be the first phase where a leader election sum exceeded $c' \log n$). Given that $P_{i,j} > c'/2 \log n$, for a sufficiently large constant c', a Chernoff bound tells us that the probability $\ell_i(j) < 1$ is small—say less than n^{-7}. (It is here that we can fix our constant value c' which allows us to fix constant c which we use in our definition of goodness.) This implies that the region is active in this phase. Our above assumption tells us that the phase is also good. We can, therefore, apply Lemma 4.7, which tells us that if the phase is active and good then all nodes in R_i will become inactive during j with probability also at least $1-n^{-7}$. If this event occurs, of course, then the leader election sum is 0 for all future phases. By a union bound, the probability that the sum exceeds $c' \log n$ in $j+1$ is less than n^{-6}. A union bound over all regions and phases tells us that the probability that any region ever exceeds $c' \log n$ is bounded as n^{-4}.

We are left to combine, with a union bound, two failure probabilities: (1) that our first claim—if the election sums are small the phase is good—fails; (2) that our second claim—the election sums are always small—fail. Both these failure probabilities are less than n^{-4}, so the probability that at least one fails is less than n^{-3}. It follows that the probability that all phases are good is at least $1-1/n^3$, as required. □

If all phases are good, we can conclude by arguing that we never generate more than $O(\log n)$ seeds per region, which, in turn, restricts the total number of unique seeds neighboring any given node to be $\leq \gamma_r \cdot O(\log n) = O(\log n)$:

LEMMA 4.9. *With probability at least $1-1/n^2$, at the end of the initialization phase, every node has committed to a seed, and no node neighbors more than $O(\log n)$ unique seeds in G'.*

PROOF. By the definition of the algorithm, every node ends up with a unique seed (as it will generate its own seed if it gets to the end of the stage without having committed). We turn our attention, therefore, to the bound on the number of nearby seeds. Let us first consider the seeds generated during the stage (as oppose to the seeds generated at the end of the stage). Lemma 4.8 tells us that every phase is good, with probability at least $1-1/n^3$. It follows that no more than $c \log n$ leaders are ever elected in a single region in a single phase, with this same probability. Lemma 4.7, however, tells us that the if a region is active during a good phase, all nodes in this region are inactive for the remainder of the stage, with probability at least $1-1/n^6$. Combining these two observations (and, with a union bound, their respective probabilities), it follows that with probability at least $1-1/n^2$, the total number of leaders ever elected in a region is bounded by $c \log n$.

Now we consider the seeds generated by uncommitted nodes at the end of the stage. Let R_i be a region that contains one such uncommitted node. there cannot be more than $2c' \log n$ nodes left uncommitted in R_i at the end of the stage, where $1 \leq c' < c$ is the constant we defined in the proof of Lemma 4.8. If there were *more* nodes left in R_i, then the leader election sum in the final phase would have been strictly more than $c' \log n$. If we dive into the proof of Lemma 4.8, however, we see that it works, in part, by establishing that the the leader election sum never exceeds $c' \log n$. Therefore, if we assume this lemma holds, we can assume no election sum ever exceeds this bound

Because each node can neighbor at most $\gamma_r = O(1)$ regions, and each region has $\max\{c \log n, 2c' \log n\} = O(\log n)$ nodes, the needed bound on nearby unique seeds holds probability $1-1/n^2$, as needed. □

We conclude by proving our main theorem:

PROOF OF THEOREM 4.6. The time complexity follows from the definition of the algorithm, as both the initialization and broadcast stages require $O(\log^2 n \log \Delta)$ rounds. Lemma 4.9 tells us that every node ends the initialization stage with a seed, and no node neighbors more than $O(\log n)$ unique seeds in G' (with probability at least $1 - 1/n^2$). Assume this lemma holds. Next, consider some node $u \in R$. It follows that u has a neighbor $v \in B$ in G such that v has a broadcast message. Let $\mathcal{S} = \{S_1, S_2, ..., S_k\}$ be a partition of the nodes in $N_{G'}(u) \cap B$ such that all nodes in S_i have the same seed from the initialization stage. Notice, by our above assumption, $k = O(\log n)$.

Let $S_i \in \mathcal{S}$ be the partition that includes v. Consider a particular decay iteration. The probability that u receives a message from S_i during this iteration is $p_{solo}p_{succ}$, where p_{solo} is the probability that S_i is the only partition from $\mathcal{S}$ to participate in this iteration, and p_{succ} is the probability that u gets a message during an iteration preconditioned on the assumption that only S_i participates . We can bound this quantity as: $p_{solo}p_{succ} > (1/\log n)(1-1/\log n)^k \cdot (1/2) > 1/(2\log n)4^{-O(\log n)/\log n} = \Omega(1/\log n)$, where we get $p_{succ} > 1/2$ from Lemma 4.2 in Section 4.1.

The broadcast stage consists of $O(\log^2 n) = c'' \log^2 n$ iterations of decay, for some constant c''. We note that for a sufficiently large constant c'', the probability that u fails to receive a message in all k iterations is less than n^{-3}. A union bound tells us that the probability that at least one node in R fails to receive a message is less than n^{-2}. Finally, using another union bound, we combine this probability with the probability that Lemma 4.9 fails to hold, which tells us that local broadcast fails with probability less than $1/n$. □

5. CONCLUSION

In this paper, we proved upper and lower bounds for global and local broadcast in unreliable radio networks where the link behavior is controlled by either an online adaptive or oblivious adversary. Existing results were proved with respect to an offline adaptive adversary. This paper fills in the gaps for these weaker adversary types—finding the thresholds at which efficiency becomes possible. In terms of future work, it remains an interesting open question to explore other problems—such as rumor spreading, leader election, or graph algorithms—under these weaker dual graph variants. It is also interesting to explore the impact of this style of unreliable behavior in SINR-style radio network models.

6. REFERENCES

[1] N. Alon, A. Bar-Noy, N. Linial, and D. Pelegi. A Lower Bound for Radio Broadcast. *Journal of Computer and System Sciences*, 43(2):290–298, 1991.

[2] R. Bar-Yehuda, O. Goldreigch, and A. Itai. 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(1):104–126, 1992.

[3] K. Censor-Hillel, S. Gilbert, F. Kuhn, N. Lynch, and C. Newport. Structuring Unreliable Radio Networks. In *Proceedings of the ACM Conference on Distributed Computing*, 2011.

[4] I. Chlamtac and S. Kutten. On Broadcasting in Radio Networks–Problem Analysis and Protocol Design. *IEEE Transactions on Communications*, 33(12):1240–1246, 1985.

[5] A. Clementi, A. Monti, and R. Silvestri. Round Robin is Optimal for Fault-Tolerant Broadcasting on Wireless Networks. *Journal of Parallel and Distributed Computing*, 64(1):89–96, 2004.

[6] A. E. Clementi, A. Monti, F. Pasquale, and R. Silvestri. Broadcasting in Dynamic Radio Networks. In *Proceedings of the ACM Conference on Distributed Computing*, 2007.

[7] A. Czumaj and W. Rytter. Broadcasting algorithms in radio networks with unknown topology. *Journal of Algorithms*, 60:115–143, 2006.

[8] M. Ghaffari, B. Haeupler, N. Lynch, and C. Newport. Bounds on Contention Management in Radio Networks. In *Proceedings of the International Conference on Distributed Computing*, 2012.

[9] M. Khabbazian, D. Kowalski, F. Kuhn, and N. Lynch. Decomposing Broadcast Algorithms using Abstract MAC Layers. In *Proceedings of the International Workshop on the Foundations of Mobile Computing*, 2010.

[10] D. Kowalski and A. Pelc. Broadcasting in Undirected Ad Hoc Radio Networks. *Distributed Computing*, 18(1):43–57, 2005.

[11] F. Kuhn, N. Lynch, and C. Newport. Brief Announcement: Hardness of Broadcasting in Wireless Networks with Unreliable Communication. In *Proceedings of the ACM Conference on Distributed Computing*, 2009.

[12] F. Kuhn, N. Lynch, C. Newport, R. Oshman, and A. Richa. Broadcasting in Unreliable Radio Networks. In *Proceedings of the ACM Conference on Distributed Computing*, 2010.

[13] F. Kuhn, N. Lynch, and R. Oshman. Distributed Computation in Dynamic Networks. In *Proceedings of the Symposium on Theory of Computing*, 2010.

[14] F. Kuhn and R. Oshman. Dynamic Networks: Models and Algorithms. *ACM SIGACT News*, 42(1):82–96, 2011.

[15] E. Kushilevitz and Y. Mansour. An Ω(D\log(N/D)) Lower Bound for Broadcast in Radio Networks. *SIAM Journal on Computing*, 27(3):702–712, 1998.

[16] T. Moscibroda and R. Wattenhofer. The Complexity of Connectivity in Wireless Networks. In *Proceedings of the IEEE International Conference on Computer Communications*, 2006.

[17] D. Peleg. Time-Efficient Broadcasting in Radio Networks: a Review. *Distributed Computing and Internet Technology*, pages 1–18, 2007.

[18] K. Srinivasan, M. Kazandjieva, S. Agarwal, and P. Levis. The β-Factor: Measuring Wireless Link Burstiness. In *Proceedings of the Conference on Embedded Networked Sensor System*, 2008.