Leader Election in Unreliable Radio Networks

Mohsen Ghaffari[1] and Calvin Newport[2]

1 **MIT, Cambridge, MA, USA**
ghaffari@mit.edu
2 **Georgetown University, Washington, DC, USA**
cnewport@cs.georgetown.edu

Abstract

The *dual graph* model describes a radio network that contains both reliable and unreliable links. In recent years, this model has received significant attention by the distributed algorithms community [12, 4, 8, 10, 9, 16, 1, 14]. Due to results in [10], it is known that leader election plays a key role in enabling efficient computation in this difficult setting: a leader can synchronize the network in such a manner that most problems can be subsequently solved in time similar to the classical radio network model that lacks unreliable links. The *feasibility* of efficient leader election in the dual graph model, however, was left as an important open question. In this paper, we answer this question. In more detail, we prove new upper and lower bound results that characterize the complexity of leader election in this setting. By doing so, we reveal a surprising dichotomy: (1) under the assumption that the network size n is in the range 1 to N, where N is a large upper bound on the maximum possible network size (e.g., the ID space), leader election is fundamentally hard, requiring $\tilde{\Omega}(\sqrt{N})$ rounds to solve in the worst-case; (2) under the assumption that n is in the range 2 to N, however, the problem can be solved in only $\tilde{O}(D)$ rounds, for network diameter D, matching the lower bound for leader election in the standard radio network model (within log factors) [7].

1998 ACM Subject Classification C.2.1 Wireless Communication

Keywords and phrases Radio Networks, Leader Election, Unreliability, Randomized Algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.138

1 Introduction

The *dual graph* model is a generalization of the classical radio network model [5, 3] that augments a connected network topology of *reliable* links with an additional set of *unreliable* links. These latter links can come and go on a round-by-round basis as determined links. These latter links can come and go on a round-by-round basis as determined by an unknown adversarial process. The model is motivated by the observation that in real radio networks link quality can change unpredictably due to many different factors (e.g., [17]). Upper bounds proved in this general model are therefore more robust in real world deployments, while lower bounds help formalize the complexity induced by link unreliability.

The dual graph model was introduced in [11] and has since been well-studied in the distributed algorithms community [12, 4, 8, 10, 9, 16, 1, 14]. In this paper, we present new upper and lower bounds for the problem of leader election in this setting. As we elaborate below, by doing so we resolve an important open question from [10] and provide a primitive that enables efficient solutions to many problems in this difficult model.

The Power of Rerandomization. In more detail, the dual graph model describes a radio network consisting of n processes connected by two network topology graphs $G = (V, E)$

43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016).
Editors: Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi;
Article No. 138; pp. 138:1–138:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and $G' = (V, E')$, where the nodes in V correspond to the n processes, E describes reliable links, and E' describes unreliable links. We assume G is connected and use D to describe its diameter. We consider randomized algorithms executing in synchronous rounds. In each round, the network topology is fixed to include all edges in E and some subset of the edges in E', where the latter decision is made by an unknown adversary. Once the topology is fixed, communication follows the standard multi-hop multiple access channel rules (see Section 2).

An important assumption in the dual graph model is the strength of the adversary that selects the unreliable links to include in each round. Early work on this model assumed the adversary was *strongly adaptive* (i.e., it knows the processes' random bits), which led, not surprisingly, to pessimistic lower bounds [11, 12, 4, 8, 16]. In response to these negative results, we argued in [10] that it makes more sense to assume an *oblivious* adversary, as this still allows for unpredictable link behavior, but it removes the unrealistic assumption that link quality can somehow adapt to the processes' private bits.

To validate the claim that the oblivious dual graph model is tractable, we demonstrated that *if* you can synchronize a group of processes with new random bits generated after the execution begins (and therefore unknown to the adversary), these processes can then efficiently communicate. The high-level idea is that transmitters use the shared bits to randomly permute the broadcast probabilities used by the standard *Decay* contention management routine [3] in a manner that is hard for the adversary to predict and therefore is hard for the adversary to impede with its selection of unreliable links.

The key to enabling efficient communication in the dual graph model, therefore, is to efficiently spread shared random bits among the processes in the network. In [10], we took a first step toward this goal by showing that if you are provided a single leader in the network, this leader can generate the bits itself, and then efficiently disseminate them to all processes in $O(D \log n + \log^2(n))$ rounds (with high probability in n)—nearly matching the lower bound of $\Omega(D \log (n/D) + \log^2 n)$ for one-to-all broadcast in the standard radio network model [2, 13, 15]. The high-level idea is that as the new bits spread, the processes that learn them are synchronized and can therefore run the efficient *Decay* variation to efficiently spread them to the next hop.

This strategy reduces the challenge of efficient communication in the dual graph model to electing a single leader. *Whether or not it is possible to efficiently solve leader election in this setting, however, was left as key open question in [10].* In this paper, we answer it.

Our Results. We prove lower and upper bounds that characterize the complexity of leader election in the dual graph model with an oblivious adversary. In doing so, we reveal a surprising dichotomy: (1) under the assumption that the network size n is in the range 1 to N, where N is a large upper bound on the maximum possible network size (e.g., the ID space), leader election is fundamentally *hard*, requiring $\tilde{\Omega}(\sqrt{N})$ rounds to solve in the worst-case; (2) under the assumption that n is in the range 2 to N, however, the problem can be solved in only $\tilde{O}(D)$ rounds, matching the lower bound for leader election in the standard radio network model (within log factors) [7].

Put another way: the promise that you are not alone in the world renders the problem of leader election vastly more tractable. We are unaware of other similar settings for which the split between $n \geq 1$ and $n \geq 2$ holds such significance. In the standard radio network model, for example, the best known leader election algorithms (which nearly match the relevant lower bounds) work equally well for all network sizes [7].

Our lower bound focuses on the easier problem of *loneliness detection*, in which the goal is for each process in a network to correctly determine whether or not it is alone. We construct

a network of maximum size N that consists of $\approx \sqrt{N}$ *outsider* processes each connected to an *insider* process in G. We then connect the insider processes to satisfy the model requirement that G is connected. The graph G' describing unreliable links is fully connected. We then demonstrate how an oblivious adversary can construct a schedule for adding and removing G' edges between insider and outsider processes that delays some outsider processes from receiving messages for a long period. During this period, these outsiders cannot distinguish this execution from those in networks where they are alone. The challenge in constructing this schedule is that the insider processes might coordinate after the execution begins, creating correlated behavior that is hard to predict when constructing our G' schedule. To sidestep this problem we connect each insider process to a line of length $\approx \sqrt{N}$ such that all these lines connect at their far ends. This allows us to keep insider behavior independent for the $\Omega(\sqrt{N})$ rounds that pass before any two insider processes can learn common information.

Our upper bound, by contrast, leverages a novel strategy that necessitates that $n \geq 2$. In particular, the algorithm works in phases. In each phase, it attempts to elect a single leader by having each process flip a weighted coin and become a candidate leader if it comes up heads. For each weight tried, the processes run a series of *experiments* to detect whether they succeeded in electing a single leader.

At a high-level, in these experiments, each candidate leader begins to grow a *territory* of processes that it synchronizes with new random bits. These bits allow them to communicate efficiently using the rerandomization strategy from [10] described above. For territory A to detect if it neighbors some different territory B (indicating the election failed), processes within each territory watch the success rate of their internal communication. If the collision rate is higher than expected, this is evidence that another territory, using different random bits, exists nearby. This strategy requires that each candidate leader has at least one neighbor to recruit for this testing—generating our requirement that $n \geq 2$. We emphasize that this cooperative approach to interference detection is new in the dual graph literature.

Related Work. A close predecessor to the dual graph model of unreliable radio communication was introduced by Clementi et al. [6]. The model in its current form was identified by Kuhn et al. [11]. It has since been well-studied by the distributed algorithms community [12, 4, 8, 10, 9, 16, 1, 14]. Under the pessimistic assumption of a strongly adaptive adversary controlling the unreliable links, results are known for global broadcast [11, 12], local broadcast [8], and graph structuring algorithms [4, 16]. In [10], we suggested the assumption of an oblivious adversary and produced near optimal bounds for global broadcast. We also proved that local broadcast *could not* be solved in $o(D)$ rounds—establishing the $\tilde{O}(D)$ round strategy for enabling efficient communication presented in this paper as optimal with respect to its dependence on D. Recent work by Lynch and Newport [14] explored a different route to efficient communication in this setting by describing how to enable efficient communication in $o(D)$ time under the *additional assumption* that G and G' satisfy strong geographic constraints. The algorithm described here makes no assumption on the network topologies and the graphs used in our lower bound do not satisfy the constraints of [14].

2 Model and Problem Statement

Model. We study the dual graph model of unreliable networks with an oblivious adversary. In more detail, we model a network consisting of n processes connected by a topology consisting of both *reliable* and *unreliable* links. We describe the reliable links with a connected graph $G = (V, E)$, and the unreliable links with a graph $G' = (V, E')$. We use

D to describe the diameter of G. An algorithm in our setting assigns one computational process to each node in V, and we assume processes are provided no advance information about G or G'. In the following, we will use the terms *node* and *process* interchangeably.

Executions are divided into synchronous rounds. At the beginning of an execution, an oblivious adversary determines which edges from G' to include in the network topology for all upcoming rounds (e.g., it generates an infinite sequence of graphs, $G_1, G_2, ...$, where each $G_i = (V, E_i)$, where E_i includes every edge in E and a subset of edges from E'). The adversary does not have access to the nodes' private random bits in generating this sequence. In each round r, we call the edges in G_r the active edges for the round. At the beginning of the round, each node decides to listen or transmit. A node u receives a message m in r if and only if: (a) u decides to listen; and (b) exactly one neighbor of u in the active edges (i.e., neighbor in G_r) transmits in r and it transmits m. In this case, u learns whether the message came from a reliable neighbor. Put another way, if two or more neighbors transmit, all messages are lost at u due to collision. We assume that collisions cannot be distinguished from silence (i.e., no collision detection).

Problem Statements. The problem of *leader election* requires each node v to output a binary value b_v such that, with high probability, exactly one node outputs 1. This node is called the leader. This problem specification suffices for our lower bound. For our algorithmic result, we strengthen the problem by also requiring that each node other than the leader receive at least one message from the leader. The problem of *loneliness detection* requires that each node v outputs a binary value b'_v such that, with high probability, if v is not alone in the network, $b'_v = 0$, and if v is alone, then $b'_v = 1$.

3 The Loneliness Detection Lower Bound

Here, we show that detecting whether a node is alone in the network or not requires $\tilde{\Omega}(\sqrt{N})$ rounds, where N is the best known upper bound on n, e.g., the size of the ID space. That is, if all that nodes know about the current number n of nodes in the network is that $n \in [1, N]$, then leader election in the dual graph model requires much more than its analogue in the reliable model, where $\tilde{O}(D)$ round suffices. Note that in the reliable model, detecting loneliness can be solved easily in $O(\log^2 N)$ rounds, with high probability, using the classical Decay protocol. In contrast, as we show here, the same problem in the unreliable model requires $\tilde{\Omega}(\sqrt{N})$ rounds. Formally, we prove the following result:

▶ **Theorem 1.** *Loneliness detection in the oblivious-adversary dual graph model requires $\tilde{\Omega}(\sqrt{n})$ rounds. More specifically, there is a dual graph radio network (G, G') with at most N nodes in which, for any execution of any distributed algorithm $\mathcal{A}$, some nodes cannot distinguish this execution from one in which they are alone.*

We first start with describing the structure of the dual graph radio network (G, G') that proves this lower bound. We then provide intuitive discussions about the adversary's oblivious strategy for each given algorithm $\mathcal{A}$. At the end, we provide the formal attack strategy and prove that it ensures that some of the nodes in the network experience an execution identical to one in the setting where they are alone in the network.

The Lower Bound Network. We first explain the reliable graph G. We have $\lfloor\sqrt{N}\rfloor$ paths, each consisting of $\lfloor\frac{N-1}{\sqrt{N}}\rfloor \approx \sqrt{N}$ nodes. We call these support lines, and we label them 1, 2, ..., $\lfloor\sqrt{N}\rfloor$. For each line, we call one endpoint "right" and the other "left". We connect all

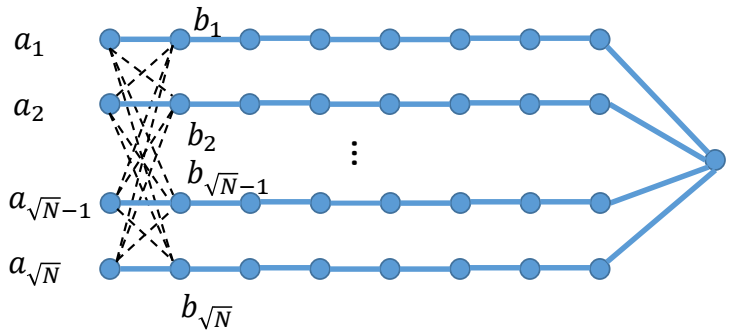

Figure 1 An illustration of the lower bound network. The reliable G-edges are depicted as solid blue lines, and the dashed black lines indicate the unreliable G'-edges.

the right endpoints to one node t. For each line i, we call the first node a_i and the second node b_i. The unreliable graph G' is then defined as a complete bipartite graph between $A = \{a_i | i \in [\lfloor\sqrt{N}\rfloor]\}$ and $B = \{b_i | i \in [\lfloor\sqrt{N}\rfloor]\}$, except for the first path edges a_i–b_i, for each path i, which are reliable and in G. For each i, we refer to a_i and b_i as pairs of each other. Figure 1 shows an illustration of this network.

3.1 Intuitive Discussions about the Adversary's Strategy

The adversary's plan is to ensure that by round $O(\sqrt{N}/\log N)$, with high probability, there are still at least half of the nodes of A which have not received any message from any other node. Thus, for each of them, the execution is identical to one in which the node is alone.

The Adversarial Strategy – Take 1. We start with an insufficient but instructive strategy. For the moment, think of b_i nodes as detached from the rest of their respective lines and let us focus on just the bipartite graph $A \times B$. Let us call a node $v \in A \cup B$ *compromised* if it has received at least one message from some other node of $A \cup B$ or some node in $A \cup B$ has received a message of v. Otherwise, we call v uncompromised.

The adversary uses its knowledge of the algorithm $\mathcal{A}$ to calculate the expected number of nodes in $A \cup B$ that will transmit in each given round. If this expectation is greater than $\Omega(\log N)$, then the adversary will aim for creating a collision for all nodes, by making all edges of G' active. The idea is that, if we have independence between the random decision of nodes in $A \cup B$ transmitting in this round, then due to this high expectation, with high probability, two or more nodes transmit. That would mean all nodes in $A \cup B$ experience collisions and none of them receives a message. Thus, in these high-expectation rounds, no nodes gets compromised, with high probability. On the other hand, if the expected number of transmitting nodes in $A \cup B$ is at most $O(\log N)$, then the adversary will work in the other extreme, and it does not activate any of the unreliable G' edges. Assuming independence between different nodes in $A \cup B$ transmitting, with high probability, at most $O(\log N)$ nodes transmit. This means, with high probability, there are at most $O(\log N)$ many a_i–b_i pairs for which some message exchange happens between a_i and b_i. Hence, in these low-expectation rounds, at most $O(\log N)$ nodes get compromised. Therefore, per round, $O(\log N)$ nodes get compromised, with high probability. That means, even after $O(\sqrt{N}/\log N)$ rounds—for small enough constants—at least half of A remain uncompromised.

The Problem in the Take 1 Strategy. The key difficulty with the above strategy is that, whether different nodes of $A \cup B$ transmit or not in a given round are not independent. This creates a challenge for concentration arguments around the expectation. For instance, there can be dependency between nodes that are compromised as there might be some communication that happened between them. Furthermore, note that the adversary cannot know which nodes are compromised in each round, as that depend on the random decisions of nodes with respect to transmitting or not.

The Adversarial Strategy – Take 2. The adversary will base its decision on the expected number X^r of uncompromised nodes that transmit, in each given round. Notice that the adversary does not know which nodes are compromised. Hence, in this expectation, whether a node is compromised or not is also a random event. Note that $X^r = \sum_i x^r_{a_i} + x^r_{b_i}$ where, for instance, $x^r_{a_i}$ is an indicator random variable for the event of a_i remaining uncompromised up to round r and then transmitting in round r.

The reason that we can focus the attack on only uncompromised nodes is that the behavior of compromised nodes will not impact the new nodes that get compromised, with high probability. Particularly, if $X^r = \Omega(\log N)$ and we have independence, then with high probability, two or more uncompromised nodes transmit. Thus activating all G' edges makes every node in $A \cup B$ experience collision. On the other hand, if $X^r = O(\log N)$ and we have independence, then with high probability, at most $O(\log N)$ uncompromised nodes transmit. Hence, if no G'-edge is activated, only nodes which their pair transmitted can be compromised. Thus, per round, at most $O(\log N)$ new nodes get compromised. Hence, even after $O(\sqrt{N}/\log N)$ rounds, at least half of the nodes in A remain uncompromised.

The Problem in the Take 2 Strategy. It remains to explain how the adversary computes the probabilities of the indicator random variables $x^r_{a_i}$ and $x^r_{b_j}$. Furthermore, the events of these variables are not precisely independent, because there is dependency between different nodes becoming compromised. For instance, if a node transmits alone in a high-expectation round in $A \cup B$, all nodes are compromised. Also, if a node gets compromised in a low-expectation round, so does its pair. We need to explain that we have sufficient independence between the random variables to use concentration arguments.

3.2 The Formal Adversarial Strategy and its Analysis

We now describe our exact adversarial strategy for determining the active edges of G' per round. We describe this process algorithmically. We later prove that for any loneliness detection algorithm $\mathcal{A}$, under this oblivious adversarial strategy, at least half of the A-nodes remain uncompromised. the schedule produced by this process works well.

In more detail, let $T = c\sqrt{n}/\log n$, for a sufficiently small constant $c > 0$, to be fixed later. As mentioned before, the adversary uses only two choices regarding which edges of G' are active: The first option is include no extra G'-edges. We call these rounds *sparse*. The second option is to include all G' edges. We call these rounds *dense*. We can, therefore, describe the adversary's behavior with a binary array S of size T, where $S[r] = 0$ indicates round r will be sparse and $S[r] = 1$ indicates it will be dense.

Oblivious Link Schedule Generation for Duration T

// `Initialization`
α is a positive constant that we use in defining our sparse/dense threshold
A is binary array of size T initialized to 0 in all positions
$N(u)$, for $u \in A \cup B$, is a real value initialized to 1
$P_r(u)$, for $u \in A \cup B$ and $r \in [T]$, is the probability node u transmits in r if it is still uncompromised at r (this depends on the definition of $\mathcal{A}$)
// `Assigning Values to Array` S
for $r = 1$ to T:
  $x \leftarrow \sum_{u \in A \cup B} N(u) \cdot P_r(u)$
  if $x \geq \alpha \log n$ **then**
    $S[r] \leftarrow 1$ // `round` r `is dense` (so we assume no compromises)
  else
    $S[r] \leftarrow 0$ // `round` r `is sparse`
    foreach $u \in A \cup B$
      $N(u) \leftarrow N(u) \cdot [(1 - P_r(u)) \cdot (1 - P_r(v)) + P_r(u) \cdot P_r(v)]$
        // `updating the probability that` u `is compromised, where` v `is the pair of` u `in the respective path.`

We must first verify that the above schedule can be generated by the oblivious adversary prior to the execution. The only non-trivial information in this effort is how the adversary generates $P_r(u)$ in advance, which we discuss next.

Computing Transmission Probabilities of Uncompromised Nodes in $A \cup B$. For $u \in A$, this follows directly from the definition of the algorithm: given the algorithm and a single node u, it is straightforward to calculate the probability that the algorithm broadcasts in any round r, given that node u is isolated.

For $u \in B$, where $u = b_i$ the adversary computes $P_r(u)$ by recursively simulating the following modified graph under a fixed pattern of delivering messages. The graph is simply the line starting from b_i and ending with the left endpoint of the path. The simulation is done (recursively, round by round) under the setting that whenever $S[r] = 1$, node b_i does not receive any message even if it is listening and its left neighbor transmits, whenever $S[r] = 0$, the reception of b_i is the same as it when this detached path is running alone. Simulating this detached line under this fixed reception strategy to b_i allows us to compute the $P_r(b_i)$ for each round $r \leq T$. This is because, if b_i is not compromised, the only difference between this simulated ditached line and the line in our network would be the connection to node t at the far right end. However, that connection cannot affect the behavior of b_i in the first $T = c\sqrt{N}/\log N$ rounds, as it is $\sqrt{N}$ hops away from b_i and due to the synchronous rounds, the causal dependency propagates with a speed of one hop per round at most.

Analysis. Next we analyze the performance of any algorithm $\mathcal{A}$ when executed in our network with the G' edges following the adversarial strategy captured by the array S as generated above.

▶ **Lemma 2.** *With high probability, in each of the first $T = c\sqrt{N}/\log N$ rounds, at most $O(\sqrt{N})$ new nodes get compromised.*

Proof. By induction on r, we show that with high probability, (1) in each dense round, all nodes in $A \cup B$ experience collision and thus no new node gets compromised, and (2) in each sparse round, at most $O(\log N)$ nodes get compromised. This directly proves the lemma.

The claim is trivially correct for round $r = 1$ because in this round, all nodes are uncompromised and $N(u) = 1$ for all of them. So, the parameter x in the adversary's calculations is the correct expectation of the number of transmitting nodes in $A \cup B$ in this round. Since we have independence between these events in the first round, Chernoff bound gives that with high probability, dense rounds are in collision for all nodes in $A \cup B$, and sparse rounds compromise at most $O(\log N)$ nodes.

Next, we prove the inductive step. Suppose that the claim is true for all rounds $r' \leq r-1$. Note that the claim implies that, modulo an inverse-polynomially small term (for the negligible change of the claim being broken), in round r, the variable $N(u)$ in the pseudo-code correctly captures the probability of node u remaining uncompromised. This is because, the calculation ignores the dense rounds, in which we know the u would not get compromised there, and it takes the transmission probability of the node and its pair into account in sparse rounds, as only the transmission of exactly one of these two nodes can compromise u.

Define N_r be random variable indicating the set of uncompromised nodes in r. Also, define the random variable X_r to be the number of uncompromised nodes in $A \cup B$ that transmit in r. Let $Y_r(u)$ be the indicator variable that is 1 if and only if u is uncompromised and transmits in r. We clearly have $E[X_r] = \sum_{u \in A \cup B} Y_r(u) = \sum_{u \in A \cup B} P_r(u) \cdot \Pr[u \in N_r]$.

In the following, we use the notation x_r to refer to the value of x in the pseudo-code run by our oblivious scheduler algorithm in round r. By the argument above, and thanks to the hypothesis of induction, we know that the computations of $N(u)$ for the probability of not being compromised was correct up to round $r-1$.

▶ **Claim 3.** *By way of induction, assume that in every dense round $r' \leq r-1$, all nodes of $A \cup B$ experience collision and thus, none of them receives a message. It follows that for each $v, u \in A \cup B$ where $(u, v) \notin G$, and each $r \leq T$, the events $Y_r(u)$ and $Y_r(v)$ are independent.*

Claim Proof. Given the inductive assumption that collisions are enforced in dense rounds $r' \leq r-1$, for each $u \in A \cup B$, we have the following: If it is in A, until u is compromised, it sees an execution identical to those in which it is alone. Also, if it is in B, until u is compromised, it experiences an execution identical to those in which the related detached line is under the reception adversary discussed above when computing $P_r(u)$. Particularly, u has received no information from outside its support line, and therefore its transmission decisions are independent of nodes in other lines.

Hence, the only way for a node to be compromised for the first time is if the node u or its pair transmit during a sparse round, in the past. However, these transmissions (prior to being compromised) are independent events between nodes of different pairs (i.e., lines). ◀

By the independence provided by the above claim, we can prove that the inductive assumption of having collisions in dense rounds is correct also for the next round r. This is because, under the inductive assumption that dense rounds prior to round r were all in collisions, the calculations $N(u)$ in the pseudo-code correctly captures the probability of a node u being uncompromised up to and including round r. This is equal to the probability of remaining uncompromised up to round $r-1$, which we know was correctly computed by indcution, and then having either none of u and its pair v transmit, or both transmit. Now, given that these events $Y_r(u)$ for different $u \in A \cup B$ are independent, except for each depending on one other event in the same set, we can finish the argument as follows: we use the standard extensions of Chernoff bound to settings with bounded (constant) dependency-graph degree and conclude the following: if the expectation $E[X_r]$ is $\Omega(\log N)$, then with high probability, $X_r \geq 2$. Hence, with high probability, the dense round r also is in collision. Similarly, if the expectation $E[X_r]$ is $O(\log N)$, then with high probability

$X_r \leq O(\log N)$ and thus, in round r, at most $O(\log N)$ new nodes get compromised. This finishes the proof of the inductive proof and thus also the proof of the lemma. ◀

4 The Algorithm

Here, we explain a distributed algorithm that computes a leader in $\tilde{O}(D)$ rounds, in the dual graph model with an oblivious adversary, when nodes are given the promise that $n \geq 2$.

▶ **Theorem 4.** *There is a distributed algorithm that in any dual graph network (G, G') with an oblivious adversary, and where $n \geq 2$ and G has diameter D, elects a leader in at most $O(D \log^4 N)$ rounds, with probability at least $1 - 1/N^c$, for any desired constant $c \geq 2$.*

4.1 Outline

Suppose that all nodes know an upper bound D on the diameter of the reliable graph G, that is within a constant factor of the diameter. We will explain how to remove this assumption using the standard doubling technique, at the end of this section.

We will try $O(\log N)$ estimates of the form $\eta = 2^i$ for n, where $i \in [1, \log N]$. For each of these, we run $O(\log N)$ experiments, where in each experiment each node is marked with probability $1/\eta = 2^{-i}$. Each marked node is considered as a candidate for leadership and an experiment is *successful* if and only if there is exactly one candidate. It is easy to see that, with high probability, there is at least one experiment in which there is exactly one marked node. The challenge is to identify experiments with exactly one candidate.

We run the $O(\log^2 N)$ experiments in parallel. Particularly, we divide time into epochs of length $O(\log^2 N)$ and the j^{th} round of each epoch belongs to the j^{th} experiment. Let us focus on one experiment. The problem we need to solve in a single experiment is to detect whether in this experiment, there is exactly one candidate or not, in $\tilde{O}(D)$ rounds.

For now, we focus on one fixed arbitrary experiment. In this experiment, we perform a special broadcast procedure starting from each of the candidates. At any point in time, we call the set of nodes that receive (only) the identifier of a candidate v the *territory* of v. Nodes that have received two or more candidate identifiers in this experiment are called *in conflict*, and nodes that have not received any candidate identifier in this experiment are called *leaderless*. We define the *conflict* and *leaderless* territories accordingly.

We first explain in Section 4.2 the simple broadcast scheme that starts from each candidate and tries to grow its territory. The guarantee will be that, if the candidate is alone, the message reaches all nodes in $O(D \log^2 N)$ rounds. In Section 4.3, we then explain how to detect whether the broadcast has already reached all nodes or not. The guarantee will be that an experiment terminates if and only if there is exactly one candidate, and in that case, it terminates in $\Theta(D \log^2 N)$ rounds. Hence, once an experiment terminates, if we wait for a (sufficiently large) constant factor more time, all those experiment that can terminate do so. At the end, among these successful experiments, the one with the smallest experiment-number wins. That means the single candidate of that experiment is elected as the leader.

4.2 Growing Territories

Each candidate v initiates a special single-message broadcast procedure. Included in this message is the identifier of the candidate v, as well as $O(\log^2 N \log \log N)$ bits of randomness. The broadcast procedure then proceeds in D phases, where in each phase, we perform the following procedure:

Rerandomized Decay Protocol. Each phase consists of $L = \alpha \log^2 N$ consecutive rounds, for a large enough constant α. In each round of the phase, all nodes that have received the message of a candidate by the start of the phase use the randomness shared in the message to pick a shared random number $j \in_{\mathcal{U}} [1, 2 \log N]$. The details of this part will be discussed shortly. Then, each node decides to transmit with probability 2^{-j}, using its own private randomness, and remains silent otherwise. At the end of the phase, each node only keeps the messages that it received from its reliable neighbors, or those that it had before. Note that the former uses the assumption that each node is able to distinguish successful receptions from reliable and unreliable neighbors. This will later simplify the process of removing the assumption of knowing an upper bound on the diameter D.

We note that the above is an adaptation of the classical Decay procedure[3], with the key change being in the publicly random choice of the transmission probability. This property helps crucially in defeating the oblivious adversary.

We show that if we have a single candidate, after $O(D \log^2 N)$ rounds, its message reaches all nodes, w.h.p. This follows immediately from the following simple lemma, which particularly implies that if there is only one candidate, its broadcast grows at a speed of one G-hop per round, thus reaching all nodes in $O(D \log^2 N)$ rounds.

▶ **Lemma 5.** *Suppose that node w has at least one G-neighbor that had received the message of a candidate v by the start of the phase. Furthermore, assume that no G'-neighbor of w has received a message from any candidate other than v. Then, regardless of the behavior of the adversary, w.h.p., node w receives the message of candidate v during this phase.*

Proof. Consider each round r of the phase and let G^r be the graph fixed by the adversary for round r. Let d^r be the number of active neighbors of w in round r that have received the message of candidate v, by the start of the phase. Notice that all potentially-active neighbors of w that transmit have received the message of candidate v and act according to the randomness bits received in the related message. Particularly, this shared randomness, they pick a common $j \in_{\mathcal{U}} [1, 2 \log N]$ for this round r. With probability $\frac{1}{2 \log N}$, the random value $j \in_{\mathcal{U}} [1, 2 \log N]$ will be such that $j = \lceil \log d^r \rceil$. If that happens, the probability that exactly one active neighbor of w transmit in round r is at least $\frac{d^r}{2^j}(1 - \frac{1}{2^j})^{d^r - 1} \geq 1/5$. Thus, per round, the probability that node w receives the message of candidate v from one of its active neighbors is at least $\frac{1}{10 \log N}$. By a simple Chernoff bound, this means that during the $\alpha \log^2 N$ rounds of the phase, node w receives the message of v with high probability. ◀

4.3 Detecting Loneliness for Candidates

Note that if in an experiment, we have two or more candidates, their territories will grow and eventually reach each other. At that point, the territories can obstruct each other from growing and covering all nodes. The key task will be to determine if the broadcast of a candidate has reached all nodes or not.

We call a node u in the territory of candidate v a *boundary node* of this territory if u has at least one G-neighbor outside the territory of v. Notice that, if the broadcast of v has not reached all nodes, then there is at least one boundary node in the territory of node v. On the other hand, if v was the only candidate, once the broadcast reaches all nodes, there will not be any boundary nodes.

The boundary nodes have the responsibility for detecting whether the broadcast has reached all nodes or not. They will deliver their indications to the candidate, which then decides whether the experiment was successful or not. The existence of boundary nodes will

be taken as an indication that the territory does not cover all nodes. Let us first explain the key idea for detecting boundary nodes.

Boundary Detection. As the lower bound from the previous section suggests, even the rudimentary problem of each node receiving a single message from any nearby node can be hard. Thus we cannot rely on the boundary nodes receiving a message from a node outside their territory. We turn the problem around. Instead of insisting on actually receiving a message from outside the territory, we take lack of a frequent receptions from the node's own territory as an indication for collisions, implying that the node is potentially a boundary node. More concretely, the following two are indications for the bound w being boundary and thus that the related broadcast has not reached all nodes: (1) if w receives a message from the territory of a different leader, or more cruicially (2) if w *does not receive* messages from its own territory frequently enough. Let us make this formal.

The boundary detection algorithm consists of a single phase—i.e., $L = \alpha \log^2 N$ rounds. Each node that has received a message from exactly one candidate v performs one phase of its Rerandomized Decay Protocol, exactly as explained above. On the other hand, all other nodes (those in conflict or leaderless) transmits a special NOISE message in all of the rounds of the phase. A node will consider itself boundary if it receives messages from its own territory in less than $\frac{\alpha \log N}{15}$ of the rounds of this phase. The next two lemmas, the second of which is the key lemma in this whole approach, explain why this criterion correctly identifies the boundary nodes:

▶ **Lemma 6.** *If a node w is not alone in its territory and it does not have a G'-neighbor out of its territory, then throughout the phase, it will receive at least $\frac{\alpha \log N}{15}$ messages from its own territory.*

Proof. The proof is similar to Lemma 5. We easily see that per round, w receives a message from its own territory with probability at least $\frac{\log N}{10}$. Thus, by a Chernoff bound, node w receives messages from its own territory in at least $\frac{\alpha \log N}{15}$ rounds, with high probability. ◀

▶ **Lemma 7.** *[Key Lemma] If w has at least one G-neighbor out of its territory, then w.h.p., it will either receive a message from a different territory, or it receives less than $\frac{\alpha \log N}{25}$ messages from its own territory.*

Proof. First note that if w has a G-neighbor in the conflict or leaderless territory, it will not receive any message from its own territory and hence it detects that it is a boundary. Next, suppose that each G-neighbor of w has received a message from exactly one marked node.

Consider a given round r of the phase, and fix the edges incident on w made active for this round by the adversary. Let A_r be the set of active neighbors of w in its own territory, and let B_r be the set of active neighbors of w in other territories. Note that by the assumption that w has at least one G-neighbor out of its territory, B_r is nonempty. Suppose that the set B_r is composed of nodes of t territories, $B_r = B_r^1 \cup B_r^2 \cup \ldots B_r^t$, where without loss of generality, B_r^t is not the conflict or leaderless territory. Define p_r to be the probability of the event that no node in B_r^1 to B_r^{t-1} transmits. We have the following simple observations regarding the transmission probabilities in these sets:

(1) The probability that no node in A_r transmits is at least $1/(2e)$.
(2) The probability that exactly one node of A_r transmits is in at most $\frac{5}{\log N}$.
(3) The probability that no node in any of B_r^1 to B_r^t transmits is at most p_r.
(4) The probability that exactly one node of B_r transmits, and it is in B_r^t, is at least $\frac{p_r}{10 \log N}$.

Note that the events of different territories are independent. Hence, by (2) and (3), we have that the probability that w receives a message from its own territory is at most $\frac{5p_r}{\log N}$. Also, by (1) and (4), we have that the probability that w receives a message from another territory is at least $\frac{p_r}{20e \log N}$.

Therefore, over all the L rounds of the phase, the expected number of messages that w receives from other territories is at least $\sum_{r=1}^{L} \frac{p_r}{20e \log N}$. If $\sum_{r=1}^{L} \frac{p_r}{20e \log N} \geq 5 \log N$, then with high probability, w receives a message from another territory. On the other hand, if $\sum_{r=1}^{L} \frac{p_r}{20e \log N} \leq 5 \log N$, the expected number of messages that w receives from its own territory is at most $\sum_{r=1}^{L} \frac{5p_r}{\log N} \ll \frac{\alpha \log N}{30}$, as the constant α is chosen to be large enough. Thus, in this case, with high probability, the number of rounds in which w receives a message from its own territory is less than $\frac{\alpha \log N}{25}$. This completes the proof. ◀

Delivering the Boundary Signal to the Candidate. If for a candidate v, there is a node u in the territory of v for which the condition of Lemma 7 is satisfied, then v knows that its experiment was not successful. We next describe how to let the candidate v know whether there is such a "boundary" node u in its territory. The basis for this again will be mimicking *collision detection*, where lack of frequent receptions from a node's own territory is regarded as a collision.

We now explain how we deliver the indicator about the existence of boundaries to the candidate, and how they react. Note that the broadcast algorithm, explained in Section 4.2, grows the territory at a speed of one G-hop per phase, when there is a single candidate, and at a (potentially) slower speed, when there are two or more. Hence, all nodes of the territory of a candidate v are within its $O(D)$ hops in G. After one phase of boundary detection, if the territory of v does not include all nodes, there will be at least one node u in this territory that satisfies the condition of Lemma 7. Thus, this node provides an excuse for not announcing the experiment successful, and potentially continuing the broadcast.

To deliver this information to the candidate, we run $O(D)$ additional phases of broadcast. Here, if a node has noticed a boundary (or is in conflict or leaderless), it constantly transmits NOISE in all the rounds of the phase. Otherwise, the node broadcasts simply according to its rerandomized decay protocol, using the shared randomness provided in the candidate's message. At the end of each phase, each node w believes that there is a boundary node in its territory if it believed so before or if receives a message from outside territory or a NOISE message, or more critically, if it does not receive messages from the node's own territory frequently enough (less than $\frac{\alpha \log N}{25}$ times). If this happens, node w constantly transmits NOISE in the rounds of the next phases. The following lemma follows by straightforward application of Lemma 7.

▶ **Lemma 8.** *If there is at least one boundary in the territory of candidate v, then by the end of these $O(D)$ phases, candidate v will be informed about it. Moreover, if the territory includes all nodes, then v will not receive such an indication.*

Finally, if after all these $\Theta(D)$ phases, a candidate v received no indication of a boundary, it considers this experiment successful. It then initiates a "*successful experiment*" announcement. Other candidate do not initiate any announcement. We run a broadcast from the candidates, in $\tilde{O}(D)$ rounds, spreading this success announcement. If the experiment was successful, then with high probability, the candidate is alone and thus, this success announcement gets delivered to all nodes of the network.

Removing the knowledge of diameter. Recall that above, we assumed that nodes know a constant factor upper bound D on the diameter of graph G. Here, we remove this assumption using a standard doubling method.

Effectively, we try 2-factor estimates of diameter. We divide the time into $\log N$ scales, where the k^{th} is made of $\tilde{O}(2^k)$ consecutive rounds. In the k^{th} scale, we imagine 2^k to be the upper bound on diameter. We thus grow the territories up to a radius of 2^k by running the broadcast (from scratch) for 2^k phases. Then we have one boundary detection phase, and then we spend 2^k additional phases to deliver the indicator about existence of boundaries to the candidate. A final set of 2^k spreads the potential announcements of successful experiments. Note that since the length of the scales form a geometric sum, and as when 2^k reaches D we will have the first successful experiment, we will observe the first successful experiment in $O(D)$ phases, considering all the scales. In the other experiments prior to this scale, the broadcasts will not be able to reach all nodes, as it spreads at a speed of one G-hop per round. Hence, those experiments will not be announced successful.

Consider the first scale for which some candidate announces a successful experiment. In this scale, we take the candidate of these successful experiments that has the smallest experiment number as the global leader.

References

1. Mohamad Ahmadi, Abdolhamid Ghodselahi, Fabian Kuhn, and Anisur Rahaman Molla. The cost of global broadcast in dynamic radio networks. In *Proceedings of the International Conference on Principles of Distributed Systems*, 2015.
2. N. Alon, A. Bar-Noy, N. Linial, and D. Peleg. A Lower Bound for Radio Broadcast. *Journal of Computer and System Sciences*, 43(2):290–298, 1991.
3. Reuven Bar-Yehuda, Oded Goldreigch, and Alon 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.
4. Keren Censor-Hillel, Seth Gilbert, Fabian Kuhn, Nancy Lynch, and Calvin Newport. Structuring unreliable radio networks. *Distributed Computing*, 27(1):1–19, 2014.
5. I. Chlamtac and S. Kutten. On Broadcasting in Radio Networks–Problem Analysis and Protocol Design. *IEEE Transactions on Communications*, 33(12):1240–1246, 1985.
6. A. E. F. 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.
7. Mohsen Ghaffari and Bernhard Haeupler. Near optimal leader election in multi-hop radio networks. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2013.
8. Mohsen Ghaffari, Bernhard Haeupler, Nancy Lynch, and Calvin Newport. Bounds on contention management in radio networks. In *The International Symposium on Distributed Computing (DISC)*, 2012.
9. Mohsen Ghaffari, Erez Kantor, Nancy Lynch, and Calvin Newport. Multi-message broadcast with Abstract MAC layers and unreliable links. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2014.
10. Mohsen Ghaffari, Nancy Lynch, and Calvin Newport. The cost of radio network broadcast for different models of unreliable links. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2013.
11. Fabian Kuhn, Nancy Lynch, and Calvin Newport. Brief announcement: Hardness of broadcasting in wireless networks with unreliable communication. In *Proceedings of the ACM Symposium on the Principles of Distributed Computing (PODC)*, 2009.

12 Fabian Kuhn, Nancy Lynch, Calvin Newport, Rotem Oshman, and Andrea Richa. Broadcasting in unreliable radio networks. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2010.

13 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.

14 Nancy Lynch and Calvin Newport. A (truly) local broadcast layer for unreliable radio networks. In *Proceedings of the ACM Conference on Distributed Computing*, 2015.

15 Calvin Newport. Lower bounds for radio networks made easy. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2014.

16 Calvin Newport. Lower bounds for structuring unreliable radio networks. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2014.

17 Calvin Newport, David Kotz, Yougu Yuan, Robert S Gray, Jason Liu, and Chip Elliott. Experimental Evaluation of Wireless Simulation Assumptions. *Simulation*, 83(9):643–661, 2007.