A Faster Distributed Radio Broadcast Primitive (Extended Abstract)

Bernhard Haeupler
Carnegie Mellon University
haeupler@cs.cmu.edu

David Wajc
Carnegie Mellon University
wajc@cs.cmu.edu

ABSTRACT

We present a faster distributed broadcasting primitive for the classical radio network model.

The most basic distributed radio network broadcasting primitive - called Decay - dates back to a PODC'87 result of Bar-Yehuda, Goldreich, and Itai. In any radio network with some informed source nodes, running Decay for $O(d \log n + \log^2 n)$ rounds informs all nodes at most d hops away from a source with high probability. Since 1987 this primitive has been the most important building block for implementing many other functionalities in radio networks. The only improvements to this decades-old algorithm are slight variations due to [Czumaj, Rytter; FOCS'03] and [Kowalski and Pelc, PODC'03] which achieve the same functionality in $O(d \log \frac{n}{d} + \log^2 n)$ rounds. To obtain a speedup from this, d and thus also the network diameter need to be near linear, i.e., larger than $n^{1-\epsilon}$.

Our new distributed primitive spreads messages for d hops in $O(d \frac{\log n \log \log n}{\log d} + \log^{O(1)} n)$ rounds with high probability. This improves over Decay for any super-polylogarithmic $d = \log^{\omega(1)} n$ and achieves near-optimal $O(d \log \log n)$ running time for $d = n^{\epsilon}$. This also makes progress on an open question of Peleg.

CCS Concepts

•**Theory of computation** → **Distributed algorithms;** *Graph algorithms analysis;*

Keywords

radio networks; broadcast; d-hop broadcast problem; decay

PODC'16, July 25–28, 2016, Chicago, IL, USA.
© 2016 ACM. ISBN 978-1-4503-3964-3/16/07...$15.00
DOI: http://dx.doi.org/10.1145/2933057.2933121

1. INTRODUCTION

This paper presents a faster distributed broadcasting routine for the classical radio network setting.

The Radio Network Setting.

In the classical radio network setting [3] a multi-hop wireless network is modeled by an undirected graph G. Communication operates in synchronous rounds in which each node can either listen or broadcast some message to all of its neighbors, typically consisting of $O(\log n)$ bits. This message can be received by all listening neighboring nodes, unless multiple transmissions interfere at a receiver. This is modeled by stipulating that a listening node will receive a message sent by a neighboring node if and only if no other neighbor sends at the same time.

The radio network model thus captures both the inherent broadcast nature of wireless transmissions as well as the additional intricacies caused by half-duplex media access and interference. The question of how to coordinate and schedule transmissions in radio networks to most efficiently disseminate information is an interesting and important one. Of particular interest are distributed solutions in which nodes do not need to know the network topology. The 2007 survey of Peleg, "Time-Efficient Broadcasting in Radio Networks"[16], gives an excellent summary and overview of the intense study of this question over the last decades.

The fundamental broadcast problem we study in this paper is the following.

Definition 1.1 (d-hop broadcast)**.** *The d-hop broadcast problem consists of a network $G = (V, E)$ and a subset of nodes $S \subseteq V$ which are informed about a message M. The goal is to transmit M to all nodes in the d-hop neighborhood of S.*

Decay: A Basic Broadcasting Routine.

The first and most basic distributed broadcasting primitive for the radio network setting is a simple algorithm of Bar-Yehuda, Goldreich and Itai published in 1987 called *Decay* [2]. In a network with some informed nodes, Decay uses a simple exponential backoff strategy to neighbors of informed nodes. In particular, in round $i = 0, 1, \ldots, \log_2 n - 1$ every informed node broadcasts independently with probability 2^{-i}. A simple two-line proof shows that after $\log n$ such rounds any node neighboring an informed node will be informed with constant probability. Repeating the above $O(d + \log n)$ times gives the following guarantee:

Lemma 1.1 (Decay [2])**.** *Decay solves the d-hop broadcast problem in the radio network model in $O(d \log n + \log^2 n)$ rounds with high probability.*

Table 1: d-hop broadcast problem

Algorithm	Rounds	Reference
Decay	$O(d \log n + \log^2 n)$	[2]
Modified Decay	$O(d \log \frac{n}{d} + \log^2 n)$	[5, 11]
Divide and Chatter	$O(d \frac{\log n \log \log n}{\log d} + \log^{O(1)} n)$	**This work**

Table 2: One-to-all broadcast problem

Bound	Notes	Reference
$\Omega(D + \log^2 n)$	holds even for offline schedule computation	[1]
$\Omega(D \log \frac{n}{D})$	assuming conditional wake-up	[13]
$O(D \log n + \log^2 n)$		[2]
$O(D \log \frac{n}{D} + \log^2 n)$		[5, 11]
$O(D + \log^2 n)$	assuming known topology	[12, 10]
$O(D + \log^6 n)$	assuming collision detection	[10]
$O(D \frac{\log n \log \log n}{\log D} + \log^{O(1)} n)$		**This work**

Over the last 29 years, work on multi-hop radio network settings has extensively used this basic primitive. Many algorithms implementing higher functionalities, for example, leader election [2, 9] repeatedly run Decay as the most basic communication subroutine for various values of d. One important property of the Decay algorithm is its distributed nature. In particular, nodes do not need to know anything about the topology to be able to execute it.

Related Work: Upper and Lower Bounds.

Subsequent work by Alon et al.[1] showed that Decay's additive polylogarithmic overhead of $\Theta(\log^2 n)$ is necessary and thus optimal even if d is constant and the network topology is known and nodes know which other nodes are already informed. On the positive side, both [5] and [11] showed in 2003 that one can achieve a small improvement over the $d \log n$ term. In particular, by choosing slightly different broadcast probabilities a running time of $O(d \log \frac{n}{d} + \log^2 n)$ rounds suffices to solve the d-hop broadcast problem. For $d \leq n^{1-\epsilon}$ this results in the same $O(d \log n)$ running time as Decay but for near-linear diameters and values of $d = n^{1-\omega(1)}$ this improves over Decay all the way to a linear $O(d)$ running time for linear diameter networks and $d = \Theta(n)$. A lower bound by Kushilevitz and Mansour [13] for the wake-up problem shows this to be tight if one makes the extra assumption that uninformed nodes are not allowed to communicate (until they are informed). Without this assumption only the trivial $\Omega(d)$ lower bound exists and there is nothing that precludes an $O(d + \log^2 n)$ distributed algorithm for the d-hop broadcast problem. On the other hand, except for near-linear d, no improvement on the classical Decay procedure has been put forward over the last three decades. In fact, no better solution than Decay is known for the d-hop broadcast problem, even if one just asks about the existence of a centralized schedule which can utilize the full knowledge of the topology.

Our Result: A Faster d-hop Broadcasting Routine.

In this work we give a new distributed broadcasting routine which solves the d-hop broadcast problem faster than Decay:

Theorem 1.2. *There is a randomized algorithm which solves the d-hop broadcast problem in the radio network model in $O(d \cdot \frac{\log n \log \log n}{\log d}) + \log^{O(1)} n$ rounds with high probability.*

This improves over Decay for any d which is only super-polylogarithmic, i.e., $d = \log^{\omega(1)} n$. It also achieves a near optimal $O(d \log \log n)$ running time for any $d \geq n^{\epsilon}$. As the Decay protocol, our algorithm is a randomized distributed routine succeeding with high probability and operating with $\log n$ bit messages and without collision detection. That is, a node which is broadcast to by several neighbors receives no message, nor does it receive indication of having been broadcast to by multiple neighbors.

The one-to-all broadcast problem.

Beyond providing a faster solution for the (distributed) d-hop broadcast problem, which can be used to replace the Decay type protocols [2, 5, 11] as a communication primitive, our result also makes progress on an open question asked in the survey of Peleg [16]. Peleg asked about the round complexity of informing all nodes of a network about a message which initially starts in a single node. This can be seen as a simple special case of the d-hop broadcast problem where d is set to equal the diameter D of the network and where a single source node is initially informed. A long line of work [7, 6, 8, 12, 4] resulted in a tight bound of $\Theta(D + \log^2 n)$ for the complexity of this problem (and efficient deterministic algorithms for constructing these schedules) for the case that the topology is known. A similar $O(D + \log^6 n)$ bound was proven more recently [10] for distributed algorithms in which nodes can detect collisions. If one disallows collision detection and adds the requirement that nodes need to be woken up first (by receiving a message) before they can take part in the communications, then the $\Theta(D \log \frac{n}{D} + \log^2 n)$ bound of the modified Decay protocol is tight. This line of research leaves the standard distributed setting without collision detection and without wake-up requirement as the only as-of-yet unresolved setting. Our solution is the first to show that the $O(D \log \frac{n}{D} + \log^2 n)$ complexity of Decay is not tight for this setting and can be improved. The exact complexity however remains open. Tables 1 and 2 outline our results and prior work.

Our Approach and Technical Contributions.

Our algorithm differs significantly from Decay.

From the wake-up lower bound of [13] we know nodes cannot efficiently coordinate if they only start acting after receiving a message. On the other hand, the results of [10] and [4] suggest that significantly better schedules can be constructed around a node if it knows a bit more about its neighborhood.

Our high-level idea is thus very simple. We first compute a suitable clustering of the network which can be done locally and thus in sub-diameter many rounds (more generally, $O(d)$ time). Within each cluster we then compute the fast schedules of [10], which can be done in sub-diameter time if cluster diameters are small. The final schedule is then a fully-oblivious schedule which simply interleaves the fast intra-cluster schedules with the Decay routine to allow for crossing between clusters. Low-diameter decompositions are particularly attractive clusterings in this context because they guarantee that: as their name suggest, the clusters they output have small diameter, and each shortest path does not pass through too many clusters, leading to a negligible overhead for the Decay cost of crossing between clusters.

While this approach sounds nice and easy there are several technical difficulties in making it work:

1) The low-diameter decomposition algorithm needs to work in the radio network model, which is much more restrictive than the usual CONGEST setting. The low-diameter decomposition furthermore needs to produce connected clusters with strong diameter guarantees.

2) When using and constructing intra-cluster schedules in all clusters simultaneously, neighboring transmissions from other clusters can cause interference. One thus needs to obtain clusters with few neighboring clusters and somehow coordinate between clusters to avoid these collisions.

3) For low-diameter decompositions which cut an edge with probability β, the cluster diameter can be as high as $\frac{\log n}{\beta}$. As we demonstrate in Section 2.2 this leads to an overall logarithmic slowdown and thus an $O(d \log n + \log^{O(1)} n)$ running time - crushing the attempt to break this barrier. This logarithmic term in the cluster diameters is furthermore known to be necessary in certain settings.

Nonetheless, we overcome these difficulties! Our main technical contributions are:

• We adapt a new low-diameter decomposition of Miller et al. [15] to the radio network model, providing good bounds on the number of neighboring clusters which can interfere with transmissions from an entry point to the cluster center and back. Random backoff together with these properties allow us to solve problems 1 and 2 above.

• We present a refined analysis of distance properties of this low-diameter decomposition. Most importantly, for decompositions cutting edges with probability β, we prove a bound on the expected distance of a node v to its cluster center of roughly $\frac{\log k}{\beta}$, where k is the ratio of the size of balls of radius $\frac{\log n}{\beta}$ and $\frac{1}{\beta}$ around v. This is the first such guarantee for a low-diameter decomposition with strong diameter. This bound then implies that for a randomly-chosen scale a better clustering can be obtained on average, leading to $\frac{\log n \log\log n}{\log d}$ type bounds reflected in our result. We expect these properties of this low-width decomposition to be of further independent interest.

2. THE Divide and Chatter FRAMEWORK

In this section we outline our algorithm's framework, starting with the main radio network broadcast subroutines it relies on. We then give a simplified variant of our algorithm and discuss some of its shortcomings. Finally, we present our algorithm. As our obtained bounds improve prior bounds only when $d = \log^{\omega(1)} n$, we assume throughout our analysis that d is some sufficiently large $\log^{\Omega(1)} n$.

2.1 Radio Network Primitives

In addition to Decay, we rely on two other algorithms as primitives for our d-hop broadcast algorithm. We will rely on these algorithms in a black-box manner. The following two lemmas state our assumptions of these subroutines.

For our first primitive, we rely on the algorithm of [10], which on a network of diameter D and n, in $O(D \cdot \log^{O(1)} n)$ rounds, computes efficient, collision-free schedules for future transmissions. The algorithm of [10] assumes collision detection for its preprocessing stage, but we can simulate collision detection at an extra multiplicative $O(\log^2 n)$ blowup, by having each node which transmits during this stage transmit the same message for $O(\log^2 n)$ Decay rounds. A node transmit to by several neighbors will detect such a collision with high probability, since with high probability it will receive $\Omega(\log n)$ messages, and each of these transmitted message is equally likely to originate at any of its transmitting neighbors. This observation allows us to use the algorithm of [10] in a black box manner, given below.

Lemma 2.1 (**Broadcast with Preprocessing**, [10])**.** *A network of diameter D and n nodes can be preprocessed in $O(D \cdot \log^{O(1)} n)$ rounds, yielding a schedule which allows for one-to-all broadcast of k messages in $O(D + k \log n + \log^6 n)$ rounds with high probability. This schedule satisfies the following properties:*

- *For some prescribed node r, the schedule transmits messages to and from nodes at distance ℓ from r in $O(\ell + \log^6 n)$ rounds w.h.p.*
- *The schedule is periodic of period $O(\log n)$: it can be thought of as restarting every $O(\log n)$ steps.*

A further primitive we will need is low-diameter graph decompositions. In Section 3.1, we show how to implement the algorithm of Miller et al. [15] in a distributed setting, giving us the following lemma. In Section 3.2 we will prove some additional useful properties of this partitioning subroutine which will prove crucial for our algorithm and might be of independent interest.

Lemma 2.2. *For any $0 < \beta \leq 1$, a graph on n nodes can be partitioned into clusters each with strong diameter $O\big(\frac{\log n}{\beta}\big)$ with high probability, and every edge cut by this partition with probability $O(\beta)$. This algorithm can be implemented in the radio network setting in $O\big(\frac{\log^3 n}{\beta}\big)$ rounds.*

2.2 The Algorithm - General Approach

Given the above subroutines, we can outline a simplified variant of our algorithm which solves the d-hop broadcast problem in $O(d \cdot \log n + \log^{O(1)} n)$ rounds in expectation. This variant is presented below. For simplicity, we describe the algorithm as though d attempting to inform nodes at distance $\Theta(d)$. Doubling d, running the algorithm for $d = 2^i$ with $i = 1, 2, \ldots, \log_2 n$, yields the same guarantees as a function of d for all possible d simultaneously.

Our Algorithm – Take One.
(1) Compute the partition implied by Lemma 2.2 with parameter $\beta = \frac{1}{\log^{\Omega(1)} n}$.
(2) In each part of the decomposition, in tandem, run the preprocessing stage implied by Lemma 2.1.
(3) Alternate between steps of the precomputed schedules of step (2) and Decay.

An important point to note is the possibility of different clusters' schedules interfering with each other in steps (2) and (3). We ignore this issue for now and address it in Section 4.

Proposition 2.3. *Assuming no collision between different clusters' broadcasts, the above algorithm solves the d-hop broadcast problem in $O(d \cdot \log n + \log^{O(1)} n)$ rounds in expectation (though not with high probability).*

Proof. First, as each part has diameter $O(\frac{\log n}{\beta})$ with high probability, by Lemmas 2.1 and 2.2 the preprocessing of steps (1) and (2) can be done in $\log^{O(1)} n$ rounds. Now, consider a path p of length d from an informed source to some other node. By Lemma 2.2 the partitioning subroutine cuts an expected $O(d\beta)$ of the edges of this path. In addition each of the parts of the partition has diameter $O(\frac{\log n}{\beta})$ with high probability. Thus, the algorithm of Lemma 2.1 allows us to transmit a message within each part of the partition in $O(\frac{\log n}{\beta} + \log^6 n)$ rounds. Upon reaching the farthest node of p in some part, within $O(\log^2 n)$ Decay steps, this message will proceed to the next part which p traverses. Overall, the message will reach t after $O((d\beta + 2) \cdot (\frac{\log n}{\beta} + \log^6 n)) = O(d \cdot \log n + \log^{O(1)} n)$ rounds in expectation. ☐

As stated above the obtained bounds are weaker (and the algorithm more involved) than Decay. In the next section we present our full algorithm, together with intuition driving its design.

2.3 The Divide and Chatter Algorithm

In order to improve upon the bound given in the previous section, for our final algorithm we will effectively rely on a new useful property of PARTITION(β). Informally, this property states that picking $\beta \in [d^{0.001}, d^{0.01}]$ at random, the expected distance of a node to its "cluster center" is $O(\frac{\log n}{\beta} \cdot \frac{\log \log n}{\log d})$. In each cluster, we take the cluster center to be the prescribed node r which the algorithm of Lemma 2.1 allows transmission to and from in time proportional to the distance to r. Consequently, following the above analysis of our algorithm's simplified variant, we would expect that the time to transmit a message to the end of a path of length d be

$$O\left((d\beta + 2) \cdot \left(\frac{\log n}{\beta} \cdot \frac{\log \log n}{\log d} + \log^{O(1)} n\right)\right)$$

$$= O\left(d \cdot \frac{\log n \log \log n}{\log d} + \log^{O(1)} n\right)$$

Of course, we cannot simply naïvely multiply the expected number of cut edges by the distance to the cluster center, and we are sweeping a lot of technical detail under the rug in this description, including the aforementioned issue of potential collisions, but this high-level idea will guide us in the design and analysis of our algorithm.

The above approach entails a few technical difficulties – most prominently how to decide in a centralized manner on a/several random choice(s) of β, as reaching consensus among the nodes of the graph seems tantamount to the all to one broadcast problem. The solution we suggest to this problem involves a two-tiered approach – we first partition the graph into coarse-grained clusters within which the multi-message properties of the algorithm of Lemma 2.1 allow us to transmit the random choices of β for later, finer-grained partitions, with these messages transmitted at a logarithmic rate which will prove to be a lower-order term for the overall algorithm. Our final algorithm is given below.

Algorithm Divide and Chatter.
(1) Compute a coarse-grained clustering by running PARTITION(β) with $\beta = d^{-0.5}$.
(2) Compute a schedule in each cluster of step (1).
(3) Using these schedules, transmit within each cluster $d^{0.3}$ randomly-chosen values $I = \{i_j \mid j \in [d^{0.3}]\}$ with $(2 \log n)^i \in [d^{0.001}, d^{0.01}]$ for all $i \in I$.
(4) Obtain $d^{0.3}$ finer clusterings with $\beta = (2 \log n)^{-i}$ for each $i \in I$ within the coarse clusterings.
(5) For each of the $d^{0.3}$ clusters in each finer-grained clustering of step (4), compute schedules.
(6) Repeat $\Theta(d^{0.9})$ times - use schedules of fine clusterings of the different i in round-robin order, running for $O(d^{0.1} \frac{\log n \log \log n}{\log n})$ rounds with each i.

Our main result, which we prove in Section 5, is this.

Theorem 2.4. *Running Divide and Chatter on an n-node radio network for $O(d \cdot \frac{\log n \log \log n}{\log d} + \log^{O(1)} n)$ rounds solves the d-hop broadcast problem with high probability.*

A corollary of Lemmas 2.1 and 2.2 is that steps (1)-(5) can be implemented in $O(d)$ rounds. (We elaborate on this in Section 4). Given the running time of step (6), the running time follows. The remainder of the paper will be dedicated to providing implementations of steps (1)-(6) overcoming collisions and allowing this algorithm to succeed with high probability. In Section 3 we discuss our radio network implementation of the low-diameter graph decomposition subroutine and prove some useful properties its output satisfies. In Section 4 we discuss the issue of possible collisions and how to overcome them. Finally, in Section 5 we prove our main result.

3. PARTITIONING THE GRAPH

In [15] Miller et al. gave a parallel algorithm for partitioning a graph's node set, such that for a chosen β each edge is cut (has its endpoints in distinct parts) with probability at most β and each cluster has strong diameter $O(\frac{\log n}{\beta})$ with high probability. This algorithm picks for each node v a random shift value δ_v distributed exponentially with parameter β, and every node u belongs to a cluster $\mathcal{C}_v$ of a node v (which we term the *cluster center* of the cluster $\mathcal{C}_v$) which minimizes the shifted distance to u. That is, u belongs to $\mathcal{C}_v$ for $v = \arg\min_v\{d(u, v) - \delta_v\}$. This is algorithm PARTITION(β), implied by Lemma 2.2 and presented below. An equivalent way to think of PARTITION(β) is as follows: each node v starts growing a cluster at "time" $-\delta_v$ (one hop per step), and every node joins the first cluster to reach it. The algorithm as stated is sequential, but can be readily parallelized, as we discuss in the next subsection.

Algorithm partition(β) [[**15**]]
For every node v, pick a value $\delta_v \sim Exp(\beta)$. Assign every node u to the cluster of the node v minimizing $d(u,v) - \delta_v$.

3.1 Radio Network Model Implementation

In order to parallelize the above algorithm, Miller et al. [15] presented an equivalent algorithm, wherein each node v is assigned a starting time $\max_u \delta_u - \delta_v$, followed by ball growing around nodes unclustered at their start time. However, this implementation requires reaching consensus among the nodes regarding the maximum shift value, which we cannot afford in our model. Nonetheless, as the maximum possible shift value is bounded by $\max_u \delta_u \leq \frac{2\log n}{\beta}$ with high probability, the following is an implementation of PARTITION(β) in the radio network model, as we shall see in Lemma 3.1.

Algorithm partition(β)
(radio network implementation).
For every node v, pick $\delta_v \sim Exp(\beta)$. Let v's start time be $start_v \leftarrow \frac{2\log n}{\beta} - \lfloor \delta_v \rfloor$. For $\frac{2\log n}{\beta}$ epochs of length $O(\log^2 n)$ each, do the following: Upon start of epoch number t, for every node v not yet in any cluster with starting time $start_v = t$, assign v to its own cluster, with v its cluster center. During an epoch, all nodes not in any cluster at epoch start, listen. All nodes in a cluster at epoch start broadcasting their cluster center name using algorithm Decay. Any listening node receiving a cluster center name joins the first such cluster center's cluster.

Lemma 3.1. *The above radio network implementation of* PARTITION(β) *outputs a partition of the nodes, with every cluster having strong diameter $O\big(\frac{\log n}{\beta}\big)$ with high probability. It terminates in $O\big(\frac{\log^3 n}{\beta}\big)$ rounds.*

Proof. First, as Decay solves the 1-hop broadcast problem in $O(\log^2 n)$ steps with high probability, any node neighboring a clustered node at the beginning of an epoch will join a cluster during this epoch with high probability and so clusters are connected and have diameter at most $\frac{2\log n}{\beta}$. Next, as all δ_u are distributed exponentially with parameter β, the probability of any δ_u being greater than $\frac{2\log n}{\beta}$ is $\exp\big(-\frac{2\log n}{\beta}\cdot\big) = \frac{1}{n^2}$, and so with high probability $\frac{2\log n}{\beta} \geq \max_u \delta_u$.[1] Consequently, with high probability every node either joins a cluster before its start time or becomes its own cluster center, as its start time lies in the range $[0, \frac{2\log n}{\beta}]$. Finally, the running time is immediate from the algorithm's description. $\square$

The sequential implementation of PARTITION(β) has every cluster $\mathcal{C}$ grow by one hop per round after $\mathcal{C}$'s inception. Our radio network implementation exhibits the same behavior, in that every cluster grows by one hop per epoch, as every epoch relies on Decay to solve 1-hop, and thus succeeds with high probability. Throughout our analysis, we condition on this high probability event occurring.

[1]This is precisely the reason clusters output by the sequential version of PARTITION(β) have small diameter.

In the following section we prove that our radio network implementation of PARTITION(β) inherits the useful properties of its sequential and parallel counterparts stated in Lemma 2.2. Moreover, we prove several additional useful properties of this algorithm which will prove important in obtaining our main result.

3.2 Partitioning the Graph - Analysis

In this section we state and prove multiple properties of PARTITION(β) useful in the analysis of Divide and Chatter. We start with a definition which will lead our analysis.

Definition 3.1. *For a given node v denote by $N_i = N_i(v) = \mathcal{B}(v, (2\log n)^i)$ the set of nodes at distance $(2\log n)^i$ from v, and by $k_i(v) = \frac{|N_{i+1}| - |N_i|}{|N_i|}$ the ratio of number of nodes at distance in the range $((2\log n)^i, (2\log n)^{i+1}]$ to the number of nodes at distance $(2\log n)^i$ or less from v.*

The expected cost of our protocol will depend on $k_i(v)$. We start by showing these $k_i(v)$ are small on average.

Lemma 3.2. *If we pick an $i \in [s,t]$ uniformly at random for $t - s = \Theta(\log_{\log n} d)$, then $\mathbb{E}[\log k_i(v)] = O\left(\frac{\log n \cdot \log\log n}{\log d}\right)$.*

Proof. As $|N_{i+1}| \geq |N_i|$ for all $i \in [s,t]$ by definition and $|N_1| \geq 1$ and $|N_{t+1}| \leq n$ we have

$$\prod_{i=s}^{t} k_i(v) = \prod_{i=s}^{t} \frac{|N_{i+1}| - |N_i|}{|N_i|} \leq \prod_{i=s}^{t} \frac{|N_{i+1}|}{|N_i|} \leq \frac{|N_{t+1}|}{|N_1|} \leq n$$

Taking out logs we obtain $\sum_{i=s}^{t} \log k_i(v) \leq \log n$. So, if we denote by $\delta \triangleq t - s + 1$, then as $\delta = \Theta(\frac{\log d}{\log\log n})$, we have

$$\mathbb{E}[\log k_i(v)] = \frac{1}{\delta} \cdot \sum_{i=s}^{t} \log k_i(v) = O\left(\frac{\log n \cdot \log\log n}{\log d}\right) \square$$

The range of i considered in Lemma 3.2 will prove useful later, as we will require $(2\log n)^i \in [d^a, d^b]$ for $a, b = \Theta(1)$.

The following simple observation will allow us to show that rounding the δ_v doesn't change the probability space by much compared to the version of algorithm PARTITION(β) in which the δ_v are not rounded down.

Observation 3.3. *For any pair of distances $d(u,v)$ and $d(u,w)$ and shift values δ_v, δ_w, we have*

$$\Pr[d(u,v) - \lfloor \delta_v \rfloor - (d(u,w) - \lfloor \delta_w \rfloor) \leq d] \leq \Pr[d(u,v) - \delta_v - (d(u,w) - \delta_w) \leq d + 1]$$

Next, we bound the probability of a node to be another node's cluster center, which will prove crucial in bounding the expected distance of a node to its cluster center.

Lemma 3.4. *Let $\beta = (2\log n)^{-i}$, $i \geq 0$. Then, for any two nodes u and v at distance $d = d(u,v) > \frac{1}{\beta}$, the probability of u being v's cluster center after running* PARTITION(β) *is at most $\frac{\exp(-d\beta+2)}{|N_i|}$.*

Proof. For u to be v's cluster center, we must have in particular that u's cluster reached N_i before the start time of all nodes in N_i; i.e., u's start time must be at least $d - \frac{1}{\beta}$ smaller than the start time of all nodes $w \in N_i$, and so its shift value δ_u must be greater than the shift values of all nodes $w \in N_i$ by at least $d - \frac{1}{\beta} - 1$. (The minus one term is due to the rounding of the shift values, as discussed in Observation 3.3.)

That is, if we denote by $\Delta = \max_{w \in N_i} \delta_w$ the maximum shift value of any node in N_i, we find that the probability of u being v's cluster center is at most $\Pr\left[\delta_u \geq \Delta + d - \frac{1}{\beta} - 1\right]$. But as $d - \frac{1}{\beta} - 1 \geq 0$, this probability is equal to

$$\Pr\left[\delta_u \geq \Delta + d - \frac{1}{\beta} - 1 \,\middle|\, \delta_u \geq \Delta\right] \cdot \Pr\left[\delta_u \geq \Delta\right]$$

By memorylessness of the exponential distribution we have

$$\Pr\left[\delta_u \geq \Delta + d - \frac{1}{\beta} - 1 \,\middle|\, \delta_u \geq \Delta\right] = \Pr\left[\delta_u \geq d - \frac{1}{\beta} - 1\right] \leq \exp(-d\beta + 2)$$

On the other hand, as the random shifts are i.i.d we have that $\Pr\left[\delta_u \geq \Delta\right] = \Pr\left[\delta_u \geq \max_{w \in N_i} \delta_w\right] = \frac{1}{|N_i|+1} \leq \frac{1}{|N_i|}$. Combining both bounds, we find that the probability of u being v's cluster center is at most $\frac{\exp(-d\beta+2)}{|N_i|}$ □

The following corollary, while insufficient for our needs, hints at the usefulness of the random choice of $\beta = (2\log n)^{-i}$, given our prior bound on the expectation of $\log k_i(v)$.

Corollary 3.5. *Denote by $CC(v)$ the cluster center of node v and by $DCC(v) = d(v, CC(v))$ its distance to v. Then, if we run* PARTITION*(β) with $\beta = (2\log n)^{-i}$,*

$$\mathbb{E}[DCC(v)] = O\left(\frac{\log k_i(v) + 1}{\beta}\right)$$

Proof. By definition, $\mathbb{E}[DCC(v)] = \sum_u d(u,v) \cdot \Pr[u = CC(v)]$. To show that $\sum_u d(u,v) \cdot \Pr[u = CC(v)] = O\big(\frac{\log k_i(v)+1}{\beta}\big)$, we consider three distance ranges. First, the nodes at distance at most $d(u,v) \leq \frac{\log k_i(v)+1}{\beta}$ from v clearly contribute at most $\frac{\log k_i(v)+1}{\beta}$ to this expectation. Next, we consider nodes at distance greater than $\frac{2\log n}{\beta}$ from v. For a node u at distance $d(u,v) \geq \frac{2\log n}{\beta}$ from v to be v's cluster center requires in particular that $\lfloor \delta_u \rfloor \geq \lfloor \delta_v \rfloor + d(u,v) \geq \frac{2\log n}{\beta}$. But as $\Pr[\max_u \delta_u \geq \frac{2\log n}{\beta} - 1] = O(\frac{1}{n})$, we find the contribution of all nodes at distance at least $\frac{2\log n}{\beta}$ from v to this expectation is at most $O(n \cdot \frac{1}{n}) = O(1)$.

Finally, we direct our attention to nodes u at distance between $\frac{\log k_i(v)+1}{\beta}$ and $\frac{2\log n}{\beta}$ from v. By N_i's definition, there are at most $|N_{i+1}| - |N_i|$ such nodes. By Lemma 3.4, for any $d \in [\frac{\log k_i(v)+1}{\beta}, \frac{2\log n}{\beta}]$ the probability of a given node at distance d from v being v's cluster center is at most $\frac{\exp(-d\beta+2)}{|N_i|}$, and so u's contribution to $\mathbb{E}[DCC(v)]$ is at most $d \cdot \frac{\exp(-d\beta+2)}{|N_i|}$. As this expression is monotone decreasing in d, we can upper bound it by assuming all nodes at distance $d \in [\frac{\log k_i(v)+1}{\beta}, \frac{2\log n}{\beta}]$ from v are at distance precisely $d' = \frac{\log k_i(v)+1}{\beta}$ from v, and so the total contribution of such nodes to this expectation is at most $\big(|N_{i+1}| - |N_i|\big) \cdot d' \cdot \frac{\exp(-d'\beta+2)}{|N_i|}$, which is precisely

$$= \frac{\log k_i(v) + 1}{\beta} \cdot k_i(v) \cdot \frac{e^2}{k_i(v) + 1}$$
$$= O\left(\frac{\log k_i(v) + 1}{\beta}\right)$$

Summing the above three contributions to the expectation, the lemma follows. □

As we shall see in Corollary 3.7, the probability of an edge to be cut by PARTITION(β) is at most $O(\beta)$. So, for an edge $e = (u,v)$, if we denote by $\mathbb{1}_e$ an indicator random variable for the event that the edge e is cut and denote by $DCC(v)$ the distance of v to its cluster center, given the above bound $\mathbb{E}[DCC(v)] = O\big(\frac{\log k_i(v)+1}{\beta}\big)$, we might expect $\mathbb{E}[\mathbb{1}_e \cdot DCC(v)] \leq O(\log k_i(v) + 1)$. As we shall see in Lemma 3.11, this is indeed the case. As our algorithm will cost $\mathbb{1}_{(u,v)} \cdot (DCC(u) + DCC(v))$ per edge (u,v) along a path from an informed node to an uninformed node (plus a global additive $O(d)$ term), this will imply our main result. But first, we proceed to prove our main technical lemma, which implies an equivalent of Lemma 2.2 for our radio network implementation of PARTITION(β), as well as several other useful properties of PARTITION(β) which we will rely on.

3.2.1 Key Lemma and Corollaries

The following lemma generalizes Lemma 4.4 in [15] for PARTITION(β). Observation 3.3 implies the same asymptotic properties hold for our partitioning with the random shifts rounded down. The lemma is implied by the proof of Lemma 2.2 in [14], though its statement is strictly stronger. For completeness we present a proof below.

Lemma 3.6. *Let $d_1, d_2, \ldots, d_n$ be n real values, and let δ_i be i.i.d exponential random variables with parameter β. For all $t \in [n]$ denote by $D^{(t)}$ the t-th smallest value in the multiset $\{d_i - \delta_i \mid i \in [n]\}$. (e.g., $D^{(1)}$ is the smallest such value). Then, for all $k \geq 1$,*

$$\Pr[D^{(k)} - D^{(1)} \leq d] \leq (1 - \exp(-d \cdot \beta))^{k-1}$$

The above holds even if we condition on the index set $I \subseteq [n]$ that defines the smallest k values of $\{d_i - \delta_i\}$ and index i_k that defines the k-th smallest value. I.e., $\{d_i - \delta_i \mid i \in I\} = \{D^{(j)} \mid j \in [k]\}$ and $d_{i_k} - \delta_{i_k} = D^{(k)}$.

Proof. For simplicity, suppose $I = [k]$ and $D^{(k)} = d_k - \delta_k$ (for $j \neq k$ we make no assumption regarding which $i \in I$ satisfies $D^{(j)} = d_i - \delta_i$). Consider some $i \in [k-1]$. By definition of $D^{(k)}$ we have $D^{(k)} \geq d_i - \delta_i$. Put otherwise, $\Pr[\delta_i \geq d_i - D^{(k)}] = 1$. By memorylessness of the exponential we thus have

$$\begin{aligned}\Pr[\delta_i \leq d + d_i - D^{(k)}] &= \Pr[\delta_i \leq d + d_i - D^{(k)} \mid \delta_i \geq d_i - D^{(k)}] \\ &\leq \Pr[\delta_i \leq d] \\ &= 1 - \exp(-d\beta)\end{aligned}$$

Where the inequality is strict only if $d_i - D^{(k)}$ is negative. Either way, we have $\Pr[\delta_i \leq d + d_i - D^{(k)}] \leq 1 - \exp(-d\beta)$. Now, by independence of the random variables δ_i we have

$$\begin{aligned}\Pr[D^{(k)} - D^{(1)} \leq d] &= \Pr[\bigwedge_{i=1}^{k-1} \delta_i \leq d + d_i - D^{(k)}] \\ &= \prod_{i=1}^{k-1} \Pr[\delta_i \leq d + d_i - D^{(k)}] \\ &\leq (1 - \exp(-d\beta))^{k-1}\end{aligned}$$ □

Lemma 3.6 together with Observation 3.3 implies many useful properties of algorithm PARTITION(β), given by the following corollaries. In the proofs of these corollaries the distances to some prescribed node plus $\frac{2\log n}{\beta}$ will play the roles of the d_i in Lemma 3.6's statement.

Corollary 3.7. *The probability of a fixed edge (u, v) being cut by* PARTITION(β) *is at most $O(\beta)$.*

Proof. For the edge (u, v) to be cut, u and v must belong to different clusters. This implies that the two smallest shifted and rounded distances from v are at most one apart. But by Lemma 3.6 and Observation 3.3 this happens with probability at most $\Pr[D^{(2)} - D^{(1)} \leq 2] \leq 1 - \exp(-2\beta) \leq 2\beta$. $\square$

The following useful property of PARTITION(β), proved to hold for the algorithm's sequential and parallel implementation in [14], follows from Lemma 3.6.

Corollary 3.8. *After running* PARTITION(β) *the probability of a fixed node u having nodes from t distinct clusters at distance d or less from u is at most $(1-\exp(-(2d+1)\cdot\beta))^{t-1}$.*

Proof. Let w be u's cluster center. Let v be some node at distance $d(u, v) \leq d$ from u, belonging to a different cluster whose cluster center is w'. Denote by $d_{-\delta}(v, w) \triangleq \frac{2\log n}{\beta} + d(v, w) - \lfloor\delta_w\rfloor$. Then $d_{-\delta}(v, w') \leq d_{-\delta}(v, w)$. Alternatingly applying triangle inequality and $d(u, v) \leq d$, we have

$$\begin{aligned} d_{-\delta}(u, w') &\leq d(u, v) + d_{-\delta}(v, w') \\ &\leq d + d_{-\delta}(v, w) \\ &\leq d + d(u, v) + d_{-\delta}(u, w) \\ &\leq 2d + d_{-\delta}(u, w) \end{aligned}$$

So, if u is at distance at most d from t different clusters, the t smallest values among $d_{-\delta}(u, w) = \frac{2\log n}{\beta} + d(u, w) - \lfloor\delta_w\rfloor$ differ by at most $2d$ from each other, so by Observation 3.3 the non-rounded shifted distances differ by at most $2d + 1$. Plugging these values into Lemma 3.6 yields the claimed bound. $\square$

The following is a useful special case of the above corollary, given that we always run PARTITION(β) with $\beta = d^{-\Theta(1)}$.

Corollary 3.9. *For $\beta = d^{-\Theta(1)}$, with high probability, every node neighbors $O(\frac{\log n}{\log d})$ clusters.*

Lemma 3.10. *Let $d_1, d_2, \ldots, d_n$ be n real values, and let δ_i be i.i.d exponential random variables with parameter β. For all $t \in [n]$ denote by $D^{(t)}$ the t-th smallest value in the multiset $\{d_i - \delta_i \mid i \in [n]\}$. Furthermore, for any d, let A_d be the event that $D^{(1)} = d_i - \delta_i$ for some i with $d_i = d$. Then,*

$$\Pr[\Big(D^{(2)} - D^{(1)} = O(1)\Big) \wedge A_d] = O(\beta) \cdot \Pr[A_d]$$

Proof. Condition on the pair of indices $i_1, i_2 \in [n]$ such that $D^{(1)} = d_{i_1} - \delta_{i_1}$, $D^{(2)} = d_{i_2} - \delta_{i_2}$ and $d_{i_1} = d$. By Lemma 3.6, taking total expectation over all possible i_1 with $d_{i_1} = d$, we obtain $\Pr[D^{(2)} - D^{(1)} = O(1) \mid A_d] = O(\beta)$. $\square$

The following lemma is key to our approach's effectiveness.

Lemma 3.11. *Let v be a node. For a run of* PARTITION(β) *with $\beta = (2\log n)^{-i}$, let $\mathbb{1}_v$ be an indicator random variable for the event that the lowest two values of $d(u, v) - \lfloor\delta_u\rfloor$ differ by some $O(1)$, and let $DCC(v)$ be the distance of v to its cluster center. Then, $\mathbb{E}[\mathbb{1}_v \cdot DCC(v)] = O(1 + \log k_i(v))$.*

Proof. For any integer d, let B_d be the event that v's distance to its cluster center is $DCC(v) = d$, and let E be the event that $\mathbb{1}_v = 1$. Then, by definition

$$\mathbb{E}[\mathbb{1}_v \cdot DCC(v)] = \sum_d d \cdot \Pr[E \wedge B_d]$$

For E to hold, the two smallest (non-rounded) shifted distances $d(u, v) - \delta_u$ must differ by $O(1)$, by Observation 3.3. On the other hand, for v's distance to cluster center to be d requires the smallest value $d(u, v) - \lfloor\delta_u\rfloor$ to be determined by a node u at distance $d(u, v) = d$. So, by Lemma 3.11 we have $\Pr[E \wedge B_d] = O(\beta) \cdot \Pr[B_d]$. Using this bound, we find

$$\begin{aligned} \mathbb{E}[\mathbb{1}_v \cdot DCC(v)] &= \sum_d d \cdot \Pr[E \wedge B_d] \\ &= \sum_d d \cdot O(\beta) \cdot \Pr[B_d] \\ &= O(\beta) \cdot \mathbb{E}[DCC(v)] \end{aligned}$$

which, by Corollary 3.5 is $O(\log k_i(v) + 1)$. $\square$

In particular, by Lemma 3.2, we have the following.

Corollary 3.12. *Let $\mathbb{1}_v$ and $DCC(v)$ be as in Lemma 3.11. Then if $\beta = (2\log n)^{-i}$, with i chosen uniformly in $[s, t]$ with $t - s = \Theta(\log_{\log n} d)$, then $\mathbb{E}[\mathbb{1}_v \cdot DCC(v)] = O(\frac{\log n \cdot \log\log n}{\log d})$.*

As we transmit messages to and from the cluster center within clusters using Lemma 2.1's schedules and transmit across clusters using Decay in $O(\log^2 n)$ rounds (and spend at most $O(\log n)$ rounds until starting to use the cluster's schedule, by Lemma 2.1) the expected cost of informing a node at the end of a path p starting at an informed node is $\sum_{v\in p} \mathbb{E}[\mathbb{1}_v \cdot (DCC(v) + \log^2 n)]$, which together with Corollary 3.12 yields the following.

Lemma 3.13. *Let p be a path from an informed node to some node t. Assuming no collisions, t is informed after an expected $O(|p| \cdot \frac{\log n \cdot \log\log n}{\log d} + \log^{O(1)} n)$ rounds of step (6).*

The above approach as presented does not address the issue of potential collisions (indeed, Lemma 3.13 states as much explicitly) and therefore needs to be refined. In the next section we show how to obtain the same asymptotic behavior, while addressing potential collisions.

4. OVERCOMING COLLISIONS

In this section we show how to address potential collisions. The potential collisions can be divided into three kinds.

(1) Collisions during preprocessing – precomputing clusterings and schedules and transmitting I.

(2) Collisions when broadcasting using fine-grained clusters' schedules, due to other fine-grained clusters from different coarse clusters.

(3) Collisions when broadcasting using fine-grained clusters' schedules, due to other fine-grained clusters from the same coarse cluster.

In Sections 4.1, 4.2 and 4.3 we show how to deal with collisions of types 1, 2 and 3, respectively. For the first type of collisions we show that at a multiplicative polylogarithmic cost these collisions can be sidestepped. As the cost of the preprocessing steps will be $O(d/\log^{O(1)} n)$, this blowup will prove inconsequential. For the latter two kinds of collisions, this approach will prove too costly, as these collisions occur during the step whose running time dominates the algorithm's running time. Instead, we will consider a partition of every path p of length $|p| \in [d/2, d]$ from an informed node to an uninformed node t into disjoint subpaths of length $d^{0.1}$. By Lemma 3.13, the expected time to inform the last node of any such sub-path after the first node of this sub-path is informed (assuming no collisions) is $O(d^{0.1} \cdot \frac{\log n \log\log n}{\log n})$, so the expected time to inform the last node of p is $O(d \cdot \frac{\log n \log\log n}{\log n})$. In Sections 4.2 and 4.3 we show how to overcome collisions while guaranteeing the same expectation for the time to inform a node at the end of p.

An important tool we will rely on is correlated randomness within clusters. It may seem surprising that this could be achievable, but as we shall see, it is readily obtained by extending algorithm PARTITION(β).

Lemma 4.1. *For any b, at an $O\big(\frac{b}{\log n}\big)$ multiplicative cost to PARTITION(β)'s running time, all nodes of each cluster can agree on b bits of randomness with high probability.*

Proof. The extra b bits can be transmitted by replacing every step of PARTITION(β) by $O\big(\frac{b}{\log n}\big)$ steps, in each step broadcasting $\log n$ bits, originating at the cluster center. □

This implies Step (3) of our algorithm, in which we transmit $d^{0.3}$ values, blows up the coarse-grained clustering's running time by a factor of $O(d^{0.3})$.

4.1 Collisions During Precomputation

Lemma 2.1's and 2.2's respective algorithms' preprocessing take $O(D \cdot \log^{O(1)} n)$ rounds for a clustering of maximum diameter D. By our choices of β for the clusterings and Lemma 3.1, the coarse-grained clustering has maximum diameter $\sqrt{d}\log n$ and the $d^{0.3}$ fine-grained clusterings have maximum diameter $d^{0.01}\log n$. For the coarse-grained clusters, together with the blowup for step (3) mentioned above, this clustering's preprocessing takes $d^{0.8}\log n$ rounds. The preprocessing stages for the $d^{0.3}$ fine-grained clusterings take $O(d^{0.31}\log^{O(1)})$ rounds. Both bounds are $O(d/\log^{O(1)} n)$, so we can afford to blow up these algorithms' running time by a polylogarithmic factor in order to overcome potential collisions. As the next lemma asserts, this can be done.

Lemma 4.2. *Both the algorithms of Lemmas 2.1 and 2.2 can be implemented to succeed with high probability despite potential collisions, at a multiplicative cost of $\log^{O(1)} n$ to their preprocessing stages' running times.*

Proof. Collisions while running PARTITION(β) are addressed in the proof of Lemma 3.1. As for the Algorithm of Lemma 2.1, we replace every step of the algorithm of Lemma 2.1 by $O(\log^2 n)$-round meta-steps. In every constituent step of a meta-step, in any cluster C, with probability $\frac{1}{\log n}$, all nodes of the cluster C that broadcast in the original algorithm, do indeed broadcast. This is done by reusing $O(\log^2 n)$ random bits per cluster in round-robin order. (By Lemma 4.1, we may assume each cluster has $O(\log^2 n)$ bits corresponding to outcomes of such biased coin flips, by blowing up the running time of PARTITION(β) by a further $O(\log n)$ factor.) By Corollary 3.9, with high probability each node v neighbors at most $O(\frac{\log n}{\log d}) = O(\log n)$ different clusters. Consequently, the probability of a node v receiving a message it receives in the algorithm of Lemma 2.1 is $\Omega(1/\log n)$ and so with high probability this node will receive this message $\Omega(\log n)$ times, and in particular will receive it at least once, in the corresponding meta-step. Taking union bound over all nodes and steps of the algorithm, we find that with high probability this collision-overcoming strategy simulates the original algorithm despite potential collisions between clusters. □

4.2 Collisions Between Coarse Clusters

The following definition will prove constructive in addressing collisions of type 2.

Definition 4.1. *We say a node is* bad *if it has two different coarse-grained clusters within distance $d^{0.01} \cdot \log n$ of it. We say a path is* bad *if one of its nodes is bad.*

Consider a path p' of length $d^{0.1}$. By Corollary 3.8 and union bound, the probability of p' being bad is at most $\sum_{v\in p'} \Pr[v \text{ bad}] = O(d^{0.1} \cdot d^{0.01} \cdot \log n \cdot d^{-0.5}) = O(d^{-0.3})$. So, by linearity of expectation, the expected number of bad length-$d^{0.1}$ sub-paths of p in any partition of p into length-$d^{0.1}$ sub-paths is $O\big(\frac{d^{0.9}}{d^{0.3}}\big) = O(d^{0.6})$. Unfortunately, this expectation cannot be ascribed to the sum of independent random variables, so we cannot directly apply Chernoff bounds to obtain high concentration. Nonetheless, a more delicate analysis will yield the desired concentration, by writing the number of bad sub-paths as the sum of random variables which are themselves sums of independent random variables with high probability.

Lemma 4.3. *For any partition of a path p of total length $|p| \in [d/2, d]$ into length-$d^{0.1}$ sub-paths $p_1, p_2, \ldots, p_{d^{0.9}/|p|}$, the number of bad p_i is $O\big(d^{0.6}\big)$ with high probability.*

Proof. By the above discussion $\Pr[p_i \text{ bad}] = O(d^{-0.3})$. Since all random shift values used to create the coarse-grained clustering by PARTITION(β) with $\beta = d^{-0.5}$ are at most $\frac{2\log n}{\beta}$ with high probability, conditioning on $\max_u \delta_u \le \frac{2\log n}{\beta}$ does not change the probability space much; in particular, the probability of a sub-path p_i being bad remains $O\big(d^{-0.3}\big)$. But, if $\max_u \delta_u \le \frac{2\log n}{\beta}$, then the events that any two sub-paths p_i, p_j of p at distance at least $\frac{5\log n}{\beta} \ge \frac{4\log n}{\beta} + 2d^{0.1}$ apart depend on a disjoint set of shift values. That is, the events that such distant sub-paths are bad are independent.

Denote by X the number of bad sub-paths in $p_1, \ldots, p_{d^{0.9}}$ and by X_i the bad sub-paths p_j indexed $j \equiv i \mod \frac{5\log n}{\beta}$. By the above, if we condition on $\max_u \delta_u \le \frac{2\log n}{\beta}$, these X_i are sums of independent Bernoulli random variables with success probability $O(d^{-0.3})$, and are thus upper-bounded by binomials $Y_i \sim Bin(n', p')$, with parameters $n' = \frac{d^{0.9}\cdot\beta}{5\log n}$ and $p' = \Theta(d^{-0.3})$. Plugging in $\beta = d^{-0.5}$, we find that $\mathbb{E}[Y_i] = n'p' = \Omega(d^{0.9} \cdot d^{-0.5}/\log n \cdot d^{-0.3}) = \Omega(\log n)$. By Chernoff bound each i satisfies $X_i \le Y_i = O(\mathbb{E}[Y_i]) = O(n'/d^{0.3})$ with high probability. So, the number of bad sub-paths satisfies $X = O\big(d^{0.6}\big)$ with high probability. □

The above lemma yields the following useful corollary.

Corollary 4.4. *On any path p of length $|p| \in [d/2, d]$ from an informed node to an uninformed node, the time spent by Divide and Chatter to transmit messages along bad sub-paths of p is $O(d)$ with high probability.*

Proof. By Lemma 4.3 p's bad sub-paths have overall length $O(d^{0.6} \cdot d^{0.1}) = O(\frac{d}{\log^2 n})$ with high probability. As our algorithm runs a step of Decay every two steps and Decay solves the 1-hop broadcast problem in $O(\log^2 n)$ rounds with high probability, transmitting along all bad sub-paths takes $O(d)$ rounds with high probability. □

4.3 Cutting out Collision-Prone Nodes

In this section we address collisions between fine-grained clusters within the same coarse-grained cluster when using their schedules; i.e., collisions of type 3.

The natural approach to overcome potential collisions between different fine-grained clusters in any coarse-grained cluster when running the schedules of Lemma 2.1 is simple: make nodes that border a cluster (other than their own) not participate in their clusters' fast schedule. That is, rather than taking entire clusters to be the induced subgraphs for which we run the schedules of Lemma 2.1, take the clusters (set)minus their nodes bordering other clusters, hoping that these new subgraphs have the same useful properties as the entire clusters. This approach, however, does not quite work, as these new subgraphs could have much higher diameter, and indeed need not even be connected. The following definition will prove useful in refining this approach.

Definition 4.2. *Consider a run of* PARTITION*(β), and let δ_v be the exponential variables associated with each node. We say a node v is* risky *if the lowest two values in the (multi)set $\{d(u,v) - \lfloor \delta_v \rfloor \mid u \in V\}$ differ by at most one. Otherwise, we say the node v is* safe.

Recall that we condition on the (high probability) event that PARTITION(β) grows each cluster by one hop per epoch after the cluster's inception; that is, one hop every $O(\log^2 n)$ rounds, corresponding to one step of the sequential implementation. Given this conditioning, the set of safe nodes contains no nodes bordering other clusters, so running the fast schedules in their induced subgraphs avoids potential collisions between clusters. Crucially for our use, for any cluster $\mathcal{C}$, all safe nodes of $\mathcal{C}$ induce a connected component of diameter no greater than that of $\mathcal{C}$, as any safe node's path to its cluster center must be comprised solely of safe nodes. Therefore, the preprocessing stage of Lemma 2.1 can be run on subgraphs induced by the safe nodes in a cluster in the same time bounds as required to do the same for the entire cluster, namely $O(d)$ rounds. It remains to show how to notify risky nodes that they are risky without changing PARTITION(β)'s running time by too much.

Lemma 4.5. PARTITION*(β) can be run in $O(\log^{O(1)} n/\beta)$ rounds so that all risky nodes are informed that they are risky with high probability.*

Proof. We extend the radio network implementation discussed in Section 3.1. By Lemma 4.1 we may assume each cluster center sends some $O(\log^2 n)$ random bits to its cluster's nodes by blowing up the running time by $O(\log n)$.

Recall that each epoch of PARTITION(β) is implemented using $O(\log n)$ Decay phases in which clustered nodes inform their one-hop neighborhood of the fact. By a phase of Decay we mean broadcast with probability 2^{-i} for $i = 0, 1, 2, \dots, \log_2 n - 1$. We modify the above in the following ways, using the random bits of each cluster in round-robin order: (1) during each Decay phase, for each cluster, all nodes of the cluster remain silent with probability $\frac{1}{2}$. (2) after each epoch, we run for a further $O(\log^2 n)$ phases of Decay during which for each cluster $\mathcal{C}$, with probability $\frac{1}{\log n}$, all risky nodes of $\mathcal{C}$ which have been notified of being risky broadcast one epoch after joining their cluster using a phase of Decay. As in our proof of Lemma 4.2, this guarantees that a node v neighboring such a risky node in its cluster will receive a message from one such neighbor with high probability within these $O(\log^2 n)$ Decay phases.

Consider a node v which borders a cluster other than its own, and is therefore risky. As each cluster is silent independently and with constant probability, the following hold with high probability: v receives $\Omega(\log n)$ message during any epoch neighbors of v may broadcast. Moreover, v receives a message from at least two distinct clusters either during the epoch in which v joins its cluster, or in the following epoch (depending on whether the two lowest values in $\{d(u,v) - \lfloor \delta_u \rfloor \mid u \in V\}$ differ by zero or one.) Either way, if v neighbors a cluster other than its own, v is notified that it is risky.

Now, consider a node w which does not border a cluster other than its own. Then, w is risky if and only if some node(s) v preceding w in a shortest path from w's cluster center u to w is (are) risky. As these risky nodes all have the same distance from the cluster center, they all broadcast at the end of the same epoch (the epoch following w's joining its cluster), and so w is notified of being risky one epoch after joining its cluster. Repeating the above, every risky node is notified of the fact by PARTITION(β)'s termination and no safe node is incorrectly notified of being risky. $\square$

Lemma 4.6. *Let p be a path from an informed node to some node t. Then, if schedules are computed and run only for safe nodes of fine-grained clusters, t is informed after an expected $O(|p| \cdot \frac{\log n \cdot \log\log n}{\log d})$ rounds of step (6).*

Proof. Once a risky node is informed of the message the next node along p is informed in at most $O(\log^2 n)$ rounds, by virtue of the Decay steps. By Lemma 3.6 a node is risky with probability $O(\beta)$ and so the cost of informing risky nodes is $O(|p| \cdot \beta \cdot \log^2 n)$ rounds in expectation. For the largest possible β value we use in step (6), namely $\beta = O(1/d^{0.001})$, this is $O(|p|)$. We proceed to bound the cost incurred by safe nodes.

Consider some cluster of PARTITION(β). For a node v to be the first (resp., last) safe node along p in its cluster requires the next (resp., previous) node along p to be risky, hence the two lowest values of $\{d(u,v) - \lfloor \delta_v \rfloor \mid u \in V\}$ must differ by at most $O(1)$. By Lemma 3.11, if we denote by $\mathbb{1}_v$ an indicator for the event that v is a first or last safe node in its cluster along p, then for a fixed choice of i we have $\mathbb{E}[\mathbb{1}_v \cdot DCC(v)] = O(1 + \log k_i(v))$. As in our proof of Lemma 3.13, the time to transmit along p is commensurate to this expression, summed over all $v \in p$. By our bound on $\mathbb{E}_i[\log k_i(v)]$ of 3.2, the lemma follows. $\square$

5. FINAL ANALYSIS

In this section we prove the main result of this paper. For succinctness, we say a path is *crossed* after r rounds if r rounds after the first node on this path is informed of a message, so is the last node. We will give a bound on the time to cross a path from an informed to an uniformed node.

Theorem 5.1. *The algorithm Divide and Chatter solves the d-hop broadcast problem in the radio network model in $O(d \cdot \frac{\log n \log\log n}{\log d}) + \log^{O(1)} n$ rounds with high probability.*

Proof. As discussed in Section 4, the preprocessing steps (1)-(5) all take $O(d)$ rounds despite potential collisions. It remains to prove $O(d \cdot \frac{\log n \log\log n}{\log d} + \log^{O(1)} n)$ rounds of step (6) suffice to solve the d-hop broadcast problem.

Consider a path p of length $|p| \in [d/2, d]$ from an informed node to an uninformed node, and let $p_1, p_2, \dots, p_{|p|/d^{0.1}}$ be a partition of p into $\Theta(d^{0.9})$ sub-paths of length $d^{0.1}$. By Corollary 4.4, the time to cross all bad sub-paths is $O(d)$ with high probability. We proceed to bound the time spent on good sub-paths.

Now, consider X_j, the time to cross a good sub-path p_j. By Lemma 4.6, we know that $\mathbb{E}[X_j] = O(d^{0.1}\frac{\log n \log\log n}{\log d})$. We let the number of rounds spent using each random choice of $\beta = (2\log n)^{-i}$ in step (6) be twice this expectation. Therefore, by Markov's inequality the probability of crossing a sub-path within this many rounds is at least one half. As we ignore partial progress of the message along a sub-path when attempting to cross the same sub-path, these attempts are independent assuming independently-chosen i. Now, while the random choices of i are not strictly speaking independent, note that since we choose $d^{0.3}$ random i values and a sub-path of length $d^{0.1}$ is crossed with high probability within $O(\log n)$ attempts, the choices of i are independent w.h.p. for "close" sub-paths p_k, p_ℓ with $|k-\ell| \le d^{0.2}/\log n$. Thus, by Chernoff bound, running step (6) for $2\mathbb{E}[X]$ rounds a total of $\Theta(d^{0.2}/\log n) = \Omega(\log n)$ times, we find that in any set of $d^{0.2}/\log n$ consecutive sub-paths of p we cross good sub-paths at least $d^{0.2}/\log n$ times with high probability; in particular, we cross *all* good sub-paths in this set. Summing over all such consecutive sets of sub-paths of p, we find that after $\Theta(d^{0.9})$ attempts to cross a path during $O(d^{0.1}\frac{\log n \log\log n}{\log d})$ rounds, we cross all good sub-paths of p. We conclude that for $d = \log^{\Omega(1)} n$, Divide and Chatter solves the d-hop broadcast problem in the radio network mode in $O(d\frac{\log n \log\log n}{\log d})$ rounds with high probability. □

6. ACKNOWLEDGMENTS

This research supported in part by NSF grant CCF-1527110 "Distributed Algorithms for Near Planar Networks" and NSF-BSF grant "Coding for Distributed Computing".

7. REFERENCES

[1] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, *A lower bound for radio broadcast*, Journal of Computer and System Sciences **43** (1991), no. 2, 290–298.

[2] R. Bar-Yehuda, O. Goldreich, and A. Itai, *On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization*, Journal of Computer and System Sciences **45** (1992), no. 1, 104–126.

[3] I. Chlamtac and S. Kutten., *On broadcasting in radio networks: Problem analysis and protocol design.*, IEEE Transactions on Communications **33** (1985), no. 12, 1240–1246.

[4] Ferdinando Cicalese, Fredrik Manne, and Qin Xin, *Faster centralized communication in radio networks*, Algorithms and Computation, Springer, 2006, pp. 339–348.

[5] A. Czumaj and W. Rytter, *Broadcasting algorithms in radio networks with unknown topology*, In Proc. IEEE Symp. on Foundations of Computer Science (2003), 492–501.

[6] Michael Elkin and Guy Kortsarz, *Improved schedule for radio broadcast*, Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, 2005, pp. 222–231.

[7] Iris Gaber and Yishay Mansour, *Centralized broadcast in multihop radio networks*, Journal of Algorithms **46** (2003), no. 1, 1–20.

[8] Leszek Gasieniec, David Peleg, and Qin Xin, *Faster communication in known topology radio networks*, In Proc. ACM Symp. on Principles of Distributed Computing (2005), 129–137.

[9] M. Ghaffari and B. Haeupler, *Near-optimal leader election in multi-hop radio networks*, In Proc. ACM-SIAM Symp. on Discrete Algorithms (2013), 748–766.

[10] Mohsen Ghaffari, Bernhard Haeupler, and Majid Khabbazian, *Randomized broadcast in radio networks with collision detection*, Proceeding of the ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC) (2013), 325–334.

[11] D. Kowalski and A. Pelc, *Broadcasting in undirected ad hoc radio networks*, In Proc. ACM Symp. on Principles of Distributed Computing (2003), 73–82.

[12] Dariusz R Kowalski and Andrzej Pelc, *Optimal deterministic broadcasting in known topology radio networks*, Distributed Computing **19** (2007), no. 3, 185–195.

[13] E. Kushilevitz and Y. Mansour, *An $\Omega(d\log(n/d))$ lower bound for broadcast in radio networks*, SIAM Journal on Computing **27** (1998), no. 3, 702–712.

[14] Gary L Miller, Richard Peng, Adrian Vladu, and Shen Chen Xu, *Improved parallel algorithms for spanners and hopsets*, Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, ACM, 2015, pp. 192–201.

[15] Gary L Miller, Richard Peng, and Shen Chen Xu, *Parallel graph decompositions using random shifts*, Proceedings of the twenty-fifth annual ACM symposium on Parallelism in algorithms and architectures, ACM, 2013, pp. 196–203.

[16] David Peleg, *Time-efficient broadcasting in radio networks: a review*, Distributed Computing and Internet Technology, Springer, 2007, pp. 1–18.