The Energy Complexity of Broadcast*

Yi-Jun Chang
University of Michigan
Ann Arbor, MI

Varsha Dani
University of New Mexico
Albuquerque, NM

Thomas P. Hayes†
University of New Mexico
Albuquerque, NM

Qizheng He
IIIS, Tsinghua University
Beijing, China

Wenzheng Li
IIIS, Tsinghua University
Beijing, China

Seth Pettie‡
University of Michigan
Ann Arbor, MI

ABSTRACT

Energy is often the most constrained resource in networks of battery-powered devices, and as devices become smaller, they spend a larger fraction of their energy on *communication* (transceiver usage) not computation. As an imperfect proxy for true energy usage, we define *energy complexity* to be the number of time slots a device transmits/listens; idle time and computation are free.

In this paper we investigate the energy complexity of fundamental communication primitives such as Broadcast in *multi-hop* radio networks. We consider models with collision detection (CD) and without (No-CD), as well as both randomized and deterministic algorithms. Some take-away messages from this work are as follows.

Time lower bounds imply energy lower bounds. The *energy* complexity of Broadcast in a multi-hop network is connected to the *time* complexity of LeaderElection in a single-hop (clique) network. Many existing lower bounds on time complexity immediately transfer to energy complexity. For example, in the CD and No-CD models, Broadcast requires $\Omega(\log n)$ and $\Omega(\log^2 n)$ energy, respectively, w.h.p.

Energy- and time-efficient broadcasting. It requires $\Omega(D)$ time to solve Broadcast even allowing unlimited energy budget, where D is the diameter of the network. The complexity measures of energy and time are in conflict, and it is an open problem whether both can be minimized simultaneously. We show that it is possible to achieve near optimality in time complexity with only poly $\log n$ energy cost. For any constant $\epsilon > 0$, Broadcast can be solved in $O(D^{1+\epsilon} \log^{O(1/\epsilon)} n)$ time with $O(\log^{O(1/\epsilon)} n)$ energy.

CCS CONCEPTS

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

KEYWORDS

Broadcast; energy-aware computing; wireless network; distributed algorithm

ACM Reference Format:
Yi-Jun Chang, Varsha Dani, Thomas P. Hayes, Qizheng He, Wenzheng Li, and Seth Pettie. 2018. The Energy Complexity of Broadcast. In *PODC '18: ACM Symposium on Principles of Distributed Computing, July 23–27, 2018, Egham, United Kingdom,* Idit Keidar (Ed.). ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3212734.3212774

1 INTRODUCTION

In many networks of small wireless devices the scarcest resource is energy, and the majority of energy is often spent on *radio transceiver usage*—sending and receiving packets— not on computation. See, e.g., [35, Fig. 2], [6, Tab. 1], and [37, §3]. Rather than account for the energy profile of every mode of operation, we assume for simplicity that devices spend one unit of energy to send/listen and nothing for computation. It is not uncommon to use transceiver usage as a proxy for total energy [10, 19, 22].

1.1 The Model

The network is a connected undirected graph $G = (V, E)$ with devices associated with vertices. Each vertex knows nothing about the topology of G, except for some general parameters such as the number of vertices $n = |V|$, the maximum degree $\Delta = \max_v \deg(v)$, and the diameter $D = \max_{u,v} \text{dist}(u, v)$. Each of Δ and D can be upper bounded by n if it is unknown.

Time is partitioned into discrete slots, and all vertices agree on time slot zero, i.e., they simultaneously start at the same time. In each time slot, each device can choose to either (i) send a message, (ii) listen, or (iii) remain idle, where (i) and (ii) cost one unit of energy and (iii) is free. We allow unbounded message size and local computation power. A device is not allowed to simultaneously send and listen; and a sender does not know whether its message has been successfully delivered to its neighbors.

If a device chooses to send a message or remain idle, it gets no feedback from the environment. If a device chooses to listen and *exactly one* neighbor sends a message m, it receives m. The other cases depend on how the model handles collisions.

No-CD: If zero or at least two neighbors transmit, a listener hears a signal λ_S, indicating *silence*.

CD: If zero neighbors transmit, a listener hears λ_S; if at least two neighbors transmit, a listener hears λ_N, indicating *noise*.

*The full version of this paper is available at [9].
†Supported by NSF CAREER award CCF-1150281
‡Supported by NSF grants CCF-1514383 and CCF-1637546

PODC '18, July 23–27, 2018, Egham, United Kingdom

ACM ISBN 978-1-4503-5795-1/18/07.
https://doi.org/10.1145/3212734.3212774

LOCAL: Every listener hears *every* message transmitted by any neighbor. There are no collisions.[1]

All the models come in randomized and deterministic variants. In the deterministic setting, vertices are assigned distinct IDs in $\{1, \ldots, N\}$ and can use them to break symmetry. Unless otherwise stated, by default the maximum allowed failure probability for a randomized algorithm is $f = 1/\text{poly}(n)$. Randomized algorithms can generate private random bits to break symmetry, e.g., they can assign themselves $O(\log n)$-bit IDs, which are distinct w.h.p.

Energy Metrics. The energy cost E_v of a vertex v is defined as the number of time slots v transmits/listens; the energy complexity of an algorithm is $\max_{v \in V} E_v$. Thus, we aim to optimize the worst case energy cost per device, and not the total energy expenditure of all devices.

In this paper we assume that (i) transmitting and listening incur the same energy cost, and (ii) whether or not a message is received, the cost of listening is the same. While some works [3, 8, 17, 24] assume that only transmitting costs energy, a number of other papers consider the model where both transmitting and listening cost the same amount of energy [7, 10, 21, 25, 28].

The validity of these assumptions depends on the underlying wireless network technology. To cite a specific example, for a sender node called *Mica2*, the current consumption of transmitting ranges from 3.7mA (transmission power at -20dBm) to 21.5mA (at +10dBm); the current consumption for listening is always 7mA, regardless of whether a message is received (see [36]). That is, the costs of transmitting and listening are roughly of the same order. In fact, in networking and system research, "idle listening" (i.e., the device is in listening mode, but no message is received) has been identified as a major cause of energy loss [1, Section 9.1]. There are a number of papers on reducing idle listening [38, 40].

1.2 Our Contribution

In this paper we consider fundamental problems in arbitrary *multi-hop* network topologies, primarily Broadcast. At time zero there is a distinguished *source* device $s \in V$ holding a message m. By the end of the computation all vertices should know m. We establish lower and upper bounds on Broadcast in all collision-detection models, both randomized and deterministic. Some of the more interesting findings are as follows.

Time lower bounds on LeaderElection in single-hop networks extend to *energy* lower bounds on Broadcast in multihop networks. As a consequence, we get energy lower bounds on Broadcast of $\Omega(\log n)$ and $\Omega(\log \Delta \log n)$ in CD and No-CD, respectively. These lower bounds reflect the difficulty of local *contention resolution*, not on broadcasting *per se*. We give a more robust energy lower bound of $\Omega(\log D) = \Omega(\log n)$ that reflects the difficulty of getting a message across a long path. It applies to any collision-detection model, even LOCAL.

Even with an infinite energy budget we need $\Omega(D)$ time. We show that it is possible to achieve near optimality in both energy and time simultaneously. For any $\epsilon > 0$, there is a Broadcast algorithm taking $O(D^{1+\epsilon} \log^{O(1/\epsilon)} n)$ time and $O(\log^{O(1/\epsilon)} n)$ energy.

Given sufficient (slightly super-linear) time, regardless of the diameter D, the energy lower bounds can almost be achieved. For example, we give an algorithm for No-CD using time $O(n \log \Delta \log^2 n)$ and energy $O(\log \Delta \log^2 n)$.

1.3 Related Work

Single-hop Networks. In theory research, previous work on energy complexity has focused on fundamental problems in single-hop (clique) networks like LeaderElection (i.e., choose exactly one device as the leader) and ApproximateCounting (approximating the number of devices n to within a constant factor) [7, 10, 19–23, 31]. In the study of single-hop networks, it is typical to assume that n is unknown.

Nakano and Olariu [31] showed that in No-CD, n initially identical devices can assign themselves distinct IDs in $\{1, \ldots, n\}$ with $O(\log \log n)$ energy per device. Bender, Kopelowitz, Pettie, and Young [7] gave a randomized method for circuit-simulation in the CD model, which led to algorithms for LeaderElection and ApproximateCounting using $O(\log(\log^* n))$ energy and $n^{o(1)}$ time, w.h.p. An earlier algorithm of Kardas et al. [23] solves the problem in $O(\log^\epsilon n)$ time using $O(\log \log \log n)$ energy, *but only in expectation*. Chang et al. [10] proved that for these problems, $\Theta(\log(\log^* n))$ and $\Theta(\log^* n)$ energy are optimal in CD and No-CD, respectively, for poly(n)-time algorithms. They also give tradeoffs between time and energy, e.g., in No-CD, with $O\left(\log^{2+\epsilon} n\right)$ time we can use just $O\left(\epsilon^{-1} \log \log \log n\right)$ energy, w.h.p. For deterministic LeaderElection protocols, $\Theta(\log N)$ is optimal in CD and No-CD [10, 21], but if senders can also detect collisions, the energy complexity drops to $\Theta(\log \log N)$ [10].

Multi-hop Networks. Energy efficiency of multi-hop networks have also been studied in theory research. Berenbrink et al. [8] studied broadcasting and gossiping algorithms, and they measured energy cost by the total number of messages transmitted. They presented a Broadcast algorithm that takes $O\left(D \log(n/D) + \log^2 n\right)$ time with expected $O\left(\log^2 n / \log(n/D)\right)$ transmissions per vertex. Gasieniec et al. [17] considered the following problem in a *known* graph topology. Given a graph with a distinguished source vertex, design a transmission schedule to minimize broadcasting time, subject to the requirement that each device only transmits at most k times. For $k = 1$, they showed that $D + \Omega(\sqrt{n - D})$ time is necessary and that $D + O(\sqrt{n} \log n)$ is sufficient. The lower bound extends to $D + \Omega((n - D)^{1/(2k)})$, and the upper bound to $O\left(n^{1/(k-2)} \log^2 n\right)$.

Some papers in the literature consider optimization problems related to energy efficiency. Kirousis et al. [24] studied the following problem. Given stations in d-dimensions, $d \in \{1, 2, 3\}$, pick transmission radii to satisfy some network properties (e.g., the network should be connected and have small diameter). The objective is to minimize the sum of the broadcasting energies. A related problem is the minimum energy broadcast routing problem (MEBR) [4]. Given coordinates of stations and a distinguished source, come up with transmission power and transmission schedule to broadcast a message. The goal is to minimize total power of all transmitters. A

[1] Lower bounds in the LOCAL model are robust since they capture the difficulty of synchronization, not on the subtleties of any particular collision-detection model. This model bears the same name as Linial's LOCAL model [29, 34] and is very similar to it. In the traditional LOCAL model vertices do not have to choose between transmitting and listening, and there is no cost associated with communication.

6-approximation algorithm via MST heuristic has been shown by Ambühl [3].

Broadcasting Protocols. Broadcast is a well-studied problem in multi-hop networks. The seminal *decay* algorithm of Bar-Yehuda et al. [5] solves Broadcast in No-CD in $O\left(D\log n+\log^2 n\right)$ time. This bound was later improved to $O\left(D\log\frac{n}{D}+\log^2 n\right)$ [12, 26]. The $\log^2 n$ term is known to be necessary [2], and the $D\log\frac{n}{D}$ term is known to be optimal [27] for a restricted class of algorithms that forbid "spontaneous transmission" (i.e., vertices that have yet to learn the message are forbidden from transmitting).

Haeupler and Wajc [18] recently gave a broadcast algorithm in the No-CD model that runs in $O\left(D\frac{\log n\log\log n}{\log D}+\log^{O(1)} n\right)$ time, demonstrating that spontaneous transmissions *are* useful. Czumaj and Davies [11] improved this to $O\left(D\frac{\log n}{\log D}+\log^{O(1)} n\right)$ and gave a LeaderElection algorithm of the same complexity, improving [15]. See [16] for an $O(D+\log^6 n)$-time Broadcast algorithm in the CD model.

1.4 Organization and Technical Overview

In Section 2 we show two simple lower bounds. We prove that even for a simple network topology—a path—and the strongest model—randomized LOCAL—the Broadcast problem still requires $\Omega(\log n)$ energy. We then present a generic reduction showing that the *energy* complexity of Broadcast in a multi-hop network is at least the *time* complexity of LeaderElection in a single-hop network, with the other aspects of the model being the same (CD or No-CD, deterministic or randomized). The take-away message from these lower bounds is that the cost of Broadcast arises from two causes: (i) the cost of synchronization, for propagating messages along long paths (when D is large), and (ii) the cost of contention-resolution in a vertex's 2-hop neighborhood (when Δ is large).

In Section 3 we introduce the basic tool SR-comm used by all our algorithms. In Section 4 we show a simple randomized algorithm in No-CD based on iterative clustering. For graphs of unbounded degree, our algorithm takes $O(n\log^3 n)$ time and $O(\log^3 n)$ energy in No-CD, which is actually the product of our two lower bounds. In Section 5, we present an algorithm in nearly diameter time.

2 LOWER BOUNDS

We prove two lower bounds on the energy-complexity of Broadcast.

Theorem 1. *Consider a path graph $P=(v_1,\ldots,v_n)$, where each vertex v_i does not know its position i. Suppose that v_1 attempts to broadcast a message m. For any randomized* LOCAL Broadcast *algorithm $\mathcal{A}$, with probability $1/2$, at least one vertex spends $\log_{13} n-1$ energy before receiving the message m.*

Proof. We may assume, without loss of generality, that the algorithm $\mathcal{A}$ works as follows. Every vertex begins in exactly the same state, except for v_1, which knows the message m. Each vertex v locally generates a string r_v of random bits, and afterward, behaves deterministically.[2] At any moment in time, each vertex v_l maintains an interval $[\alpha,\beta]$ such that v_l knows r_{v_j} if and only if $j\in[\alpha,\beta]$. Whenever v_l transmits a message, it transmits every useful piece of information it knows, namely the concatenation of $r_{v_\alpha},\ldots,r_{v_\beta}$. After each wakeup, a vertex decides the time of its next wakeup and mode (transmit or listen) based on all the information it has learned so far. It is easy to see that any algorithm in this model can be simulated with the same round- and energy-complexity if vertices only communicate the r_{v_j}-strings they know.

Let I be an interval of the path $(v_1,\ldots,v_n)$ not including v_1. Intuitively, the event $\mathcal{E}_i[I]$ holds if there is some device in I that, after its ith wakeup, knows of no information outside I. This definition has two undesirable properties. First, it necessarily depends on the behavior of (i.e., random bits generated by) vertices outside I. Second, even if we could make it independent of the random bits outside I, the event may still depend subtly on *where* the path I is embedded within $(v_2,\ldots,v_n)$. The actual event $\mathcal{E}_i[I]$ assumes "worst case" embedding of I and "best case" behavior of vertices outside I. Therefore, $\mathcal{E}_i[I]$ depends *only* on the strings of random bits generated by I-vertices. In particular:

$$\begin{aligned}\mathcal{E}_i[I]\ :\ &\exists(v^\star\in I)\ \forall(\text{embedding of } I \text{ in } (v_2,\ldots,v_n))\\ &\forall(r_{v_j}\mid v_j\notin I)\ \forall(v_k\notin I):\\ &\quad\text{After its } i\text{th wakeup, } v^\star \text{ does not know } r_{v_k}.\end{aligned}$$

Observe that because of the quantification over all random strings outside I and the embedding of I in $(v_2,\ldots,v_n)$, we are considering a large class of *potential* executions of $\mathcal{A}$, which necessarily includes the actual execution. If $\mathcal{E}_i[I]$ occurs, we write $v^\star[I]$ to denote the leftmost vertex $v^\star\in I$, satisfying the statement of $\mathcal{E}_i[I]$.

The lower bound is by induction, with an induction hypothesis that is probabilistic. In particular, we assume, for each interval I of length $(13)^i$, that

$$\Pr(\mathcal{E}_i[I])\geq 1/2.$$

The assumption is valid in the base case $i=0$, since $\Pr(\mathcal{E}_0[I])=1$ when I contains a single node that has yet to wakeup. Let I be an interval of length $(13)^{i+1}$ partitioned into 13 subintervals $I_1,\ldots,I_{13}$ of length $(13)^i$. We apply the inductive hypothesis to each subinterval and conclude that $\Pr(\mathcal{E}_i[I_j])\geq 1/2$. Moreover, since these events are independent (they depend on disjoint sets of random strings),

$$\Pr\left(\sum_{j=1}^{13}\mathbf{1}_{\{\mathcal{E}_i[I_j]\}}\geq 5\right)\geq\Pr(\text{Binom}(13,\tfrac{1}{2})\geq 5)>5/6.$$

Suppose that the number of subintervals I_j satisfying $\mathcal{E}_i[I_j]$ is, in fact, at least 5, and let $J_1,\ldots,J_5$ be the first 5 such subintervals. At this point in the proof $J_1,\ldots,J_5$ have no distinguishing characteristics. Because of this, the *times* of the ith wakeups for the vertices $v^\star[J_1],\ldots,v^\star[J_5]$ are independent and identically distributed random variables. Among these 5 vertices, suppose the one whose ith wakeup is earliest is some $v^\star\in\{v^\star[J_2],v^\star[J_3],v^\star[J_4]\}$. After this wakeup, $v^\star$ could not have learned any random string outside I. Every such string must have been communicated through either $v^\star[J_1]$ or $v^\star[J_5]$, and each of these vertices has only been awake at most $i-1$ times. Thus, assuming $J_1,\ldots,J_5$ exist, with probability $3/5$ there is a vertex $v^\star$ that wakes up i times and knows of no information outside I. Since $J_1,\ldots,J_5$ exist with probability $5/6$,

[2] For example, if the algorithm $\mathcal{A}$ assumes that vertices have distinct $O(\log n)$-bit IDs, these can be generated without communication, with probability $1-1/\text{poly}(n)$.

the probability that such a $v^\star$ exists is at least $(5/6)(3/5) = 1/2$, which concludes the induction. □

Next, we prove Broadcast lower bounds for the No-CD and CD models, which hold even in constant diameter graphs. There are some very subtle issues about the randomized No-CD lower bound, see the end of the section for discussion.

Theorem 2. Broadcast *is subject to the following energy lower bounds, where the failure probability is fixed at* $f = 1/\text{poly}(n)$ *for randomized algorithms. (i) deterministic* No-CD*:* $\Omega(\Delta)$*; (ii) randomized* No-CD*:* $\Omega(\log \Delta \log n)$*; (iii) randomized* CD*:* $\Omega(\log n)$.

Proof. Consider the LeaderElection problem in a single-hop network, where the number of vertices is unknown, but is guaranteed to be upper bounded by n'. The goal of LeaderElection is to have a time slot τ where exactly one vertex transmits, and all remaining vertices listen. In the single-hop network model, we allow all vertices to send and listen simultaneously, i.e., we are in the *full duplex* model. We also make a distinction between randomized and deterministic models.

- **Randomized Model.** Each vertex is anonymous, i.e., they do not have IDs. We allow all vertices to have *shared randomness*, but each vertex still has its own private randomness. The maximum allowed failure probability is f'. Let $T_r(n', f')$ be the time complexity for LeaderElection this model. It is known that $T_r(n', f') = \Omega(\log \log n' + \log \frac{1}{f'})$ for CD [33, 39],[3] and also $T_r(n', f') = \Omega(\log n' \log \frac{1}{f'})$ for No-CD [14].
- **Deterministic Model.** Each vertex has a distinct ID in $[n']$. The algorithm is not allowed to fail for any possible assignment of distinct IDs. Let $T_d(n')$ be the deterministic time complexity. It is known that $T_d(n') = \Omega(n')$ [19, Theorem 1.6].

There is a very subtle issue regarding the $\Omega(\log \log n' + \log \frac{1}{f'})$ lower bound; see the end of this section for discussion.

Let G_k be the bipartite graph $K_{2,k}$ with two parts $\{s, t\}$ and $\{v_1, \ldots, v_k\}$, where the vertex s is attempting to broadcast a message. Let $\mathcal{A}$ be any Broadcast algorithm that applies to the graphs G_k, for all $1 \le k \le \Delta$. If $\mathcal{A}$ is randomized and has failure probability at most f, then we claim that $\mathcal{A}$ uses at least $T_r(\Delta, f)/2$ energy. If $\mathcal{A}$ is deterministic, then we claim that it $\mathcal{A}$ uses at least $T_d(\Delta)/2$ energy. These two claims imply the statement of the lemma (with $f = 1/\text{poly}(n)$).

A Generic Reduction. These claims are proved by the following generic reduction. Let $\mathcal{A}$ be any Broadcast algorithm on G_k, for all $1 \le k \le \Delta$, that takes E energy. We transform it into a LeaderElection algorithm $\mathcal{A}'$ in the aforementioned single-hop network model with $n' = \Delta$ that takes $2E$ time.

For the algorithm $\mathcal{A}$ to solve Broadcast, the vertex t has to receive a message from s, and so there must be one time slot $\tau^\star$ where exactly one vertex in $\{v_1, \ldots, v_k\}$ transmits and t listens.

We call the vertices in the corresponding single-hop network $\{v'_1, \ldots, v'_k\}$. Intuitively, each vertex v'_i in the single-hop network simulates v_i in the multi-hop network, and we treat $\{s, t\}$ as the communication channel. Each v'_i uses its private random bits to simulate the private random bits of v_i.

[3] The randomized CD $\Omega(\log \frac{1}{f'})$ lower bound, which applies even for $k = 2$, is folklore.

Assumptions about $\mathcal{A}$. Without loss of generality, we make the following assumptions about $\mathcal{A}$. If we are in the randomized model, we let the two vertices s and t announce all their local random bits in the first two rounds of $\mathcal{A}$.

If we are in the deterministic model, we assume that the IDs of the vertices in $\{v_1, \ldots, v_k\}$ are chosen from the range $\{1, \ldots, \Delta\}$, and we let the two vertices s and t announce their IDs in the first two rounds of $\mathcal{A}$.

Simulation of Time Slot τ *of* $\mathcal{A}$. We show how to simulate a time slot τ of $\mathcal{A}$ in the single-hop network model. The simulation of the first two special rounds are straightforward, as follows. In the randomized model, we use the shared randomness in the single-hop network to simulate the random bits of s and t. In the deterministic model, we simply set $\text{ID}(s) = n'+1 = \Delta+1$ and $\text{ID}(t) = n'+2 = \Delta+2$.

In what follows, we assume $\tau > 2$. If at least one of s and t listens at time slot τ in $\mathcal{A}$, the simulation costs 1 round; otherwise, the simulation skips this round. Thus, the total amount of time for the simulation is at most $2E$.

During the simulation, we maintain an inductive hypothesis that each vertex v'_i (in the single-hop network, right before they simulate the time slot τ of $\mathcal{A}$) already knows the entire history and information of the three vertices v_i, s, and t during all time slots $\{1, \ldots, \tau - 1\}$ in $\mathcal{A}$ in the multi-hop network.

The inductive hypothesis, together with the above assumption, implies that v'_i is able to perfectly predict the actions of the three vertices v_i, s, and t at time τ in $\mathcal{A}$ in the multi-hop network.

If both s and t do not listen at time τ in $\mathcal{A}$, then nothing needs to be done in the simulation. The reason is that at time τ in $\mathcal{A}$, the channel feedback must be silence for everyone.

If at least one of s and t listens at time τ in $\mathcal{A}$, we use one round in the single-hop network to simulate the time slot τ in $\mathcal{A}$. Specifically, for each vertex v'_i, all we need to do is to let v'_i calculate the channel feedback of v_i, s, and t at time τ in $\mathcal{A}$.

Note that v'_i already knows the channel feedback of v_i (at time τ in $\mathcal{A}$) since it knows the actions of s and t. To let v'_i learn the channel feedback of s and t (at time τ in $\mathcal{A}$), we simply let each $v'_i \in \{v'_1, \ldots, v'_k\}$ do what v_i does at time τ in $\mathcal{A}$, and then the channel feedback in the single-hop network that each v'_i receives is the same as the channel feedback received by s and t in the multi-hop network model. □

Theorem 2 complements Theorem 1 by showing another $\Omega(\log n)$ energy lower bound (by setting $f = 1/\text{poly}(n)$) in CD, even when $D = O(1)$. On graphs with unbounded degree, Theorem 2 implies $\Omega(\log^2 n)$ energy lower bounds in No-CD, and $\Omega(n)$ lower bounds in *deterministic* No-CD.

Remark. Our $\Omega(\log \Delta \log n)$ lower bound relies on the existence of an $\Omega(\log n' \log 1/f')$ lower bound for the LeaderElection problem in single-hop networks, where we set $n' = \Delta$ and $f' = 1/\text{poly}(n)$. The known lower bound [14] for this result only applies to uniform algorithms. There is another lower bound [33] that works for all algorithms but requires $f' = 1/n' = 1/\Delta$. If we use [14], then the $\Omega(\log \Delta \log n)$ lower bound only applies to uniform algorithms. If we use [33], then we get an $\Omega(\log^2 \Delta)$ lower bound that applies to all algorithms.

Next, we discuss an issue about applying the lower bound of [14] in the proof of Theorem 2. First of all, we are only interested in the regime of $f' = 1/\text{poly}(n)$, and so we assume $f' < 1/n^2 < 1/\Delta^2$. Recall that we consider a single-hop network with at most $n' = \Delta$ vertices.

The randomized No-CD $\Omega(\log n' \log \frac{1}{f'})$ lower bound of [14] only applies to *uniform* algorithms in the sense that for each round τ, there is a sending probability p_τ such that each vertex sends with probability p_τ using fresh randomness independently at round τ. Notice that in the No-CD model the channel feedback for all vertices must be silence all the time before the first successful transmission.

We show that this lower bound can still be applied in our setting. The only issue that we need to deal with is that we allow shared randomness. Let $\mathcal{A}$ be any algorithm (with shared randomness) that solves LeaderElection with failure probability f', and let $\mathcal{A}[r]$ be the algorithm with respect to a string r that serves as the shared randomness. Then $\mathcal{A}[r]$ must be uniform.

In what follows, we show that there exists a string $r^\star$ such that the failure probability of $\mathcal{A}[r^\star]$ is at most $n'f' < \sqrt{f'}$, and so the $\Omega(\log n' \log \frac{1}{f'})$ lower bound of [14] also applies to $\mathcal{A}$. We say that a string r is *good* for a number $k \in [n']$ if the failure probability of $\mathcal{A}[r]$ is at most $n'f'$ when we run $\mathcal{A}[r]$ on a single-hop network with k vertices. It suffices to show that there is a string $r^\star$ that is good for all $k \in [n']$. Suppose that such $r^\star$ does not exist. Then there exists a number $k^\star \in [n']$ such that with probability at least $1/n'$ a uniformly random string r is not good for $k^\star$. This implies that for an execution of $\mathcal{A}$ on a single-hop network of $k^\star$ vertices, the failure probability is higher than $(1/n')(n'f') = f'$, which is a contradiction.

3 BASIC BUILDING BLOCKS

Given two disjoint vertex sets S and R, the task SR-comm is defined as follows. Each vertex $u \in S$ attempts to transmit a message m_u, and each vertex in R attempts to receive one message. An SR-comm algorithm guarantees that for every $v \in R$ with $N(v) \cap S \neq \emptyset$, with probability $1 - f$, v receives a message m_u from at least one vertex $u \in N(v) \cap S$.

Lemma 3. *In the randomized* No-CD *model,* SR-comm *can be solved with high probability, i.e., $f = 1/\text{poly}(n)$, in time $O(\log \Delta \log n)$ and energy $O(\log \Delta \log n)$.*

Proof. Use the $O(\log \Delta \log 1/f)$-time algorithm of [5], which is also known as *decay*. □

For the randomized CD model, we present a generic transformation which turns an algorithm $\mathcal{A}$ for LeaderElection in single-hop networks (satisfying some additional requirements) to an algorithm $\mathcal{A}'$ that solves SR-comm.

Requirements for $\mathcal{A}$. We require that in an execution of $\mathcal{A}$, within time $T(n', f)$ there is a successful communication with probability $1 - f$. The algorithm $\mathcal{A}$ is executed on a single-hop network, where the number of vertices is unknown, but is guaranteed to be at most n'. The vertices in the single-hop network are allowed to simultaneously send and listen. Since we do not measure the energy of $\mathcal{A}$, we simply assume that all vertices (including senders) always listen to the channel.

We assume that algorithm $\mathcal{A}$ is *uniform* in the following sense. For each time slot t, there is an integer $k_t \in \{0, 1, \ldots, \lceil \log n' \rceil\}$ such that each vertex transmits with the same probability $p = 2^{-k_t}$ independently at the time slot t. The number k_t depends solely on the history (i.e., channel feedback) of the algorithm execution before time t. Since all vertices always listen to the channel, they have the same history.

The Generic Transformation. We show how to obtain a randomized algorithm $\mathcal{A}'$ that solves SR-comm in time $T(\Delta, f) \cdot \lceil \log \Delta \rceil$ with energy cost $2 \cdot T(\Delta, f)$, but in a multi-hop network, where vertices cannot simultaneously send and listen.

The algorithm $\mathcal{A}'$ consists of $T(\Delta, f)$ epochs, each of which consists of $\lceil \log \Delta \rceil$ time slots.

The protocol for vertices in S is as follows. In each epoch, each vertex $v \in S$ transmits at the ith time slot of this epoch with probability 2^{-i} in such a way that the total number of transmissions of v during an epoch is at most 2. This can be achieved since $1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots = 2$.

Each vertex $u \in R$, in parallel, simulates the algorithm $\mathcal{A}$ executed on a single-hop network of size $|N(u) \cap S|$, as follows. We simulate the ith time slot of $\mathcal{A}$ during the ith epoch. In the simulation, u serves as the communication channel. By inductive hypothesis, we assume that before the ith epoch, u already knows the history of the execution up to time $i - 1$, and so u has enough information to calculate the number k_i. During the ith epoch, u only listens at the k_ith slot. Notice that the channel feedback that u receives simulates the ith time slot of the execution of $\mathcal{A}$ on a single-hop network of size $|N(u) \cap S|$, as each vertex in $N(u) \cap S$ transmits with probability 2^{-k_i} at the k_ith slot of an epoch.

Recall that in an execution of $\mathcal{A}$, with probability $1 - f$, there is some successful communication by time $T(|N(u) \cap S|, f) \leq T(\Delta, f)$. Thus, for each $u \in R$, with probability $1 - f$, there must be one epoch where exactly one vertex in $N(u) \cap S$ is transmitting, and so the vertex $u \in R$ receives a message. Based on this generic transformation, we obtain Lemma 4.

Lemma 4. *In the randomized* CD *model,* SR-comm *can be solved with energy $O(\log \log \Delta + \log 1/f)$ and runtime $O(\log \Delta(\log \log \Delta + \log 1/f))$. For the special case where each $v \in S$ is adjacent to at most one vertex in R, the energy cost is $O(\log \log \Delta) + X$, where X is a random variable drawn from an exponential distribution* Exponential(λ)*, for some $\lambda = O(1)$.*

Proof. Apply the generic transformation to the uniform leader election protocol of [32]. The runtime of the algorithm of [32] is actually $O(\log \log n' + \log 1/f)$, *for every* f. That is, the runtime is can be written as $O(\log \log n') + X$, where X is a random variable drawn from an exponential distribution Exponential(λ), for some $\lambda = O(1)$. This already implies the energy cost of each $u \in R$ is $O(\log \log n') + X$, since each $u \in R$ can simply terminates after its simulation is done. However, in general, we cannot obtain this improvement for vertices in S, since a vertex $v \in S$ can be adjacent to many R-vertices.

For the special case where each $v \in S$ is adjacent to at most one vertex $u \in R$, the vertex v is only involved in the simulation associated with u. In this case, we are able to show that the energy cost of $O(\log \log n') + X$ can be achieved for all vertices. Consider

the following modifications to the above generic transformation. In the first round, all vertices in R speak, and all vertices in S listen. This allows each vertex in S to check whether it is adjacent to a vertex in R. Those vertices in S that are not adjacent to a vertex in R terminate after the first round. For each epoch i, we allocate an additional time slot at the end of the ith epoch to let each vertex $u \in R$ inform all its neighbors in S whether or not the simulation associated with u has finished (i.e., u has already received a message from a vertex in $u \in N(v) \cap S$). If it is done, then all vertices in $S \cap N^+(v)$ terminate. □

Remark 5. *In Lemma 4, if a vertex v satisfies either (i) $v \in S$ and $N(v) \cap R = \emptyset$, or (ii) $v \in R$ and $N(v) \cap S = \emptyset$, then the energy cost of v can be lowered to $O(1)$ in the* CD *model. Due to the ability of a vertex to distinguish between noise and silence, in $O(1)$ time, each $v \in S$ can check whether $N(v) \cap R = \emptyset$, and similarly each $v \in R$ can check whether $N(v) \cap S = \emptyset$ in $O(1)$ time. We will make use of this observation to reduce the energy cost of algorithms in the* CD *model.*

4 BASIC ENERGY-EFFICIENT RANDOMIZED ALGORITHMS

In this section we show that Broadcast can be solved using $O(\log^3 n)$ energy in randomized No-CD.

Layers of Vertices. A labeling $\mathcal{L} : V(G) \mapsto \{0, \ldots, n-1\}$ is said to be *good* if it has the following property. Each vertex v with $\mathcal{L}(v) > 0$ must have a neighbor u such that $\mathcal{L}(u) = \mathcal{L}(v) - 1$. A vertex v is called a *layer-i* vertex if $\mathcal{L}(v) = i$. The intuition underlying the definition of a good labeling is that it represents a clustering of vertices. If we let each layer-i vertex select a layer-$(i-1)$ neighbor as its parent, then we obtain a partition of $V(G)$ into $|\mathcal{L}^{-1}(0)|$ clusters. Each cluster C is a rooted tree T, where the root r is the unique layer-0 vertex in the cluster C. However, it is possible that a vertex has multiple choices of its parent, so the clustering resulting from a good labeling is, in general, not unique.

We say that two layer-0 vertices u and v are *$\mathcal{L}$-adjacent* if there exists a path $P = (u, u_1, \ldots, u_a, v_b, \ldots, v_1, v)$ such that $\mathcal{L}(u_i) = i$ for all $i \in [a]$ and $\mathcal{L}(v_j) = j$ for all $j \in [b]$. The graph $G_{\mathcal{L}}$ is on vertex set $\mathcal{L}^{-1}(0)$ and edge set $\{\{u, v\} \mid u \text{ and } v \text{ are } \mathcal{L}\text{-adjacent}\}$.

In the following lemma we show that Broadcast can be solved energy-efficiently if we already have a good labeling $\mathcal{L}^\star$ with small number of layer-0 vertices.

Lemma 6. *Let $\mathcal{L}^\star$ be a good labeling of G. Each vertex knows its $\mathcal{L}^\star$-label and two integers $d, L \geq 1$ such that (i) d is an upper bound on the diameter of $G_{\mathcal{L}^\star}$, and (ii) L is an upper bound on the number of layers. Then* Broadcast *can be solved by a randomized algorithm with high probability in time $T = T_{n,d,L}$ using energy $E = E_{n,d,L}$.*

LOCAL :	$T = O(Ld)$	$E = O(d)$
CD :	$T = O(Ld \log n \log \Delta)$	$E = O(d + \log n)$
No-CD :	$T = O(Ld \log n \log \Delta)$	$E = O(d \log n \log \Delta)$

Proof. Let v be the vertex that attempts to broadcast some message m. The goal of the Broadcast problem is to relay the message m to all vertices in the graph. This can be solved by first (1) do Up-cast to relay the message from v to some layer-0 vertex; (2) repeat (Down-cast, All-cast, Up-cast) for d times to let all layer-0 vertices receive the message m; and then (3) do Down-cast to ensure that all vertices in the graph have the message m.

- Down-cast. For $i = 0, \ldots, L-2$, do SR-comm with S being the set of layer-i vertices that hold the message m, and R being the set of all layer-$(i+1)$ vertices that have not received the message m. Each vertex in S attempts to broadcast the message m.
- All-cast. Do SR-comm with S being the set of all vertices that hold the message m, and R being the set of all vertices that have not received the message m. Each vertex in S attempts to broadcast the message m.
- Up-cast. For $i = L-1, \ldots, 1$, do SR-comm with S being the set of layer-i vertices that hold the message m, and R being the set of all layer-$(i-1)$ vertices that have not received the message m. Each vertex in S attempts to broadcast the message m.

We use SR-comm with $f = 1/\text{poly}(n)$. Thus, the Broadcast problem can be solved in $O(Ld) \cdot T'(n, \Delta)$ time and $O(d) \cdot E'(n, \Delta)$ energy, where $T'(n, \Delta)$ and $E'(n, \Delta)$ are the runtime and the energy cost of SR-comm; see Lemmas 3 and 4. By the observations made in Remark 5, the energy cost can be further reduced to $O(d + E'(n, \Delta)) = O(d + \log n)$. In the above algorithm, each vertex v is involved in $O(d)$ invocations of SR-comm, and all but $O(1)$ of them satisfy either (i) $v \in S$ and $N(v) \cap R = \emptyset$, or (ii) $v \in R$ and $N(v) \cap S = \emptyset$. □

In what follows, we show that a good labeling $\mathcal{L}^\star$ with small number of layer-0 vertices can be computed efficiently. Our strategy is to begin with the trivial all-0 good labeling, and then in each iteration use the current good labeling $\mathcal{L}$ to obtain a new good labeling $\mathcal{L}'$ such that (i) each layer-0 vertex remains layer-0 with some probability (to be determined), and (ii) no new layer-0 vertex is created.

Computing a New Labeling $\mathcal{L}'$ from $\mathcal{L}$. Let $p \in (0, 1)$ and $s \geq 1$ be two parameters to be chosen later. The algorithm for computing $\mathcal{L}'$ is as follows: (1) initially, set $\mathcal{L}'(v) = \perp$ for all vertices, but each layer-0 vertex v sets $\mathcal{L}'(v) = 0$ independently with probability p; (2) repeat (Down-cast, All-cast, Up-cast) s times, and then do Down-cast; (3) any vertex v that has yet to obtain a new $\mathcal{L}'$ label (i.e., $\mathcal{L}'(v) = \perp$) retains its old label, setting $\mathcal{L}'(v) = \mathcal{L}(v)$.

- Down-cast. For $i = 0, \ldots, n-2$, do SR-comm with S being the set of layer-i vertices of $\mathcal{L}$ such that its $\mathcal{L}'$ label is not $\perp$, and R being the set of all layer-$(i+1)$ vertices of $\mathcal{L}$ such that its $\mathcal{L}'$ label is $\perp$. Each vertex in S attempts to broadcast its $\mathcal{L}'$ label. Each vertex in R that receives the message m sets its $\mathcal{L}'$ label to be $m + 1$.
- All-cast. Do SR-comm with S being the set of all vertices such that its $\mathcal{L}'$ label is not $\perp$, and R being the set of all vertices such that its $\mathcal{L}'$ label is $\perp$. Each vertex in S attempts to broadcast its $\mathcal{L}'$ label. Each vertex in R that receives the message m sets its $\mathcal{L}'$ label to be $m + 1$.
- Up-cast. For $i = n-1, \ldots, 1$, do SR-comm with S being the set of layer-i vertices of $\mathcal{L}$ such that its $\mathcal{L}'$ label is not $\perp$, and R being the set of all layer-$(i-1)$ vertices of $\mathcal{L}$ such that its $\mathcal{L}'$ label is $\perp$. Each vertex in S attempts to broadcast its

$\mathcal{L}'$ label. Each vertex in R that receives the message m sets its $\mathcal{L}'$ label to be $m+1$.

We use SR-comm with $f = 1/\text{poly}(n)$. It is straightforward to verify that the algorithm indeed computes a good labeling $\mathcal{L}'$, w.h.p. The algorithm takes $O(ns) \cdot T'(n,\Delta)$ time and $O(s) \cdot E'(n,\Delta)$ energy, where $T'(n,\Delta)$ and $E'(n,\Delta)$ are the runtime and the energy cost of SR-comm; see Lemmas 3 and 4. In the CD model, the energy cost is $O(s + \log n)$; see Remark 5.

We show that each layer-0 vertex in $\mathcal{L}$ remains layer-0 in $\mathcal{L}'$ with probability at most $p + (1-p)^{\min\{s+1,w\}} + 1/\text{poly}(n)$, where $w = |\mathcal{L}^{-1}(0)|$. Assuming all invocations of SR-comm succeed, which happens with probability $1 - 1/\text{poly}(n)$, there are two ways for a layer-0 vertex v in $\mathcal{L}$ to remain layer-0 in $\mathcal{L}'$.

- The vertex v sets $\mathcal{L}'(v) = 0$ at Step (1), and this occurs with probability p.
- All vertices u within distance s to v (in $G_{\mathcal{L}}$) have $\mathcal{L}'(u) = \perp$ at Step (1), and this occurs with probability at most $(1-p)^{\min\{s+1,w\}}$.

We are in a position to prove the main theorems of this section.

Theorem 7. *The* Broadcast *problem can be solved by a randomized algorithm with high probability in the following runtime $T = T(n,\Delta)$ and energy cost $E = E(n,\Delta)$.*

LOCAL :	$T = O(n \log n)$	$E = O(\log n)$
CD :	$T = O(n \log \Delta \log^2 n)$	$E = O(\log^2 n)$
No-CD :	$T = O(n \log \Delta \log^2 n)$	$E = O(\log \Delta \log^2 n)$

Proof. Set $p = 1/2$ and $s = 1$. As long as the number of layer-0 vertices in $\mathcal{L}$ is greater than 1, each layer-0 vertex in $\mathcal{L}$ remains layer-0 in $\mathcal{L}'$ with probability at most $p + (1-p)^{\min\{s+1,w\}} + 1/\text{poly}(n) \le 1/2 + 1/4 + 1/\text{poly}(n) = 3/4 + 1/\text{poly}(n)$. Thus, after $O(\log n)$ iterations of computing a new labeling from an old labeling, we obtain a good labeling $\mathcal{L}^\star$ such that the number of layer-0 vertices is exactly 1, with high probability. Applying Lemma 6 (with $L = n$ and $d = 0$) gives the theorem. □

Recall that the energy cost for computing $\mathcal{L}'$ from $\mathcal{L}$ is $O(s + \log n)$ (instead of $O(s \log n)$) in the CD model. Using this fact, the energy cost can be improved in the CD model without affecting the time too much. This result is omitted from this extended abstract. Refer to the full version of the paper [9].

5 ENERGY-EFFICIENT BROADCAST WITH NEARLY OPTIMAL TIME

In this section, we show that it is possible to achieve near diameter time $O(D^{1+\epsilon}\text{poly}(\log n))$ while keeping relatively low energy complexity $O(\text{poly}(\log n))$. Throughout this section we are working in the No-CD model for simplicity. A couple log factors can be saved by adapting our algorithm to the CD model.

Our algorithm is based on a subroutine Partition(β), described by Miller, Peng, and Xu [30] and further analyzed by Haeupler and Wajc [18]. The goal of Partition(β) is to produce the following random clustering. Each vertex v picks $\delta_v \sim$ Exponential(β), $\beta \in (0,1)$, and assigns v to the cluster of u that minimizes $\text{dist}(u,v) - \delta_u$. This algorithm is as follows [18].

Partition(β). Every vertex v picks a value $\delta_v \sim$ Exponential(β). Let v's start time be $\text{start}_v \leftarrow \max\{1, \frac{2\log n}{\beta} - \lceil \delta_v \rceil\}$. There are $\frac{2\log n}{\beta}$ epochs numbered 1 through $\frac{2\log n}{\beta}$. At the beginning of epoch t, if v is not yet in any cluster and $\text{start}_v = t$, v becomes the cluster center of its own cluster. During the epoch, we execute SR-comm with failure probability $f = 1/\text{poly}(n)$, where S is the set of all clustered vertices and R the set of all as-yet unclustered vertices. Any vertex $v \in R$ receiving a message (we call this "merging request") from $u \in S$ joins the cluster of u.

The algorithm Partition(β) can be implemented in No-CD, and it takes $O(\frac{\log^3 n}{\beta})$ time and $O(\frac{\log^3 n}{\beta})$ energy.

The cluster graph is defined as the graph resulting from contracting each cluster to a vertex. Our strategy for solving Broadcast is to iteratively apply the clustering algorithm Partition(β) to the cluster graph until it has diameter $\text{poly}(\log n)$.

Lemma 8 presents some useful properties of Partition(β). In Lemma 9 we prove that the diameter of the cluster graph shrinks by a factor of $O(\beta)$ with high probability.

Lemma 8 ([18, 30]). *The algorithm* Partition(β) *partitions the vertices into clusters with the following properties.*

(1) *The probability of any edge $\{u,v\}$ having its endpoints in different clusters is at most 2β.*
(2) *Let u be any vertex. The probability that vertices in $N^d(u) \cup \{u\}$ are in at least t distinct clusters is at most $\left(1 - e^{-(2d+1)\beta}\right)^{t-1}$. As a special case, for $d = 1$ (i.e., if we only care about u and its neighbors) this probability is at most $\left(1 - e^{-3\beta}\right)^{t-1}$.*

Proof. The two properties are due to [18, Corollary 3.7] and [18, Corollary 3.8], respectively. □

Lemma 9 (Concentration bound on diameter). *Suppose that the diameter of the graph G is $D = \frac{\alpha \log^2 n}{\beta^4}$, for some number α. Then the diameter of the cluster graph resulting from* Partition(β) *is at most $3\beta D$, with probability $1 - n^{-\Omega(\alpha)}$.*

Proof. Let $k = 2 \cdot \frac{2\log n}{\beta}$ be twice the number of epochs, and so the maximum diameter of any cluster is at most k. Consider any two vertices u and v such that $\text{dist}(u,v) > 3\beta D = 3\beta \cdot \frac{\alpha \log^2 n}{\beta^4} = \frac{3\alpha \log^2 n}{\beta^3}$. Let $P = (w_1, w_2, \ldots, w_\ell, w_{\ell+1})$ be a shortest path from $u = w_1$ to $v = w_{\ell+1}$ of length ℓ. Define X_i to be the indicator random variable that w_i and w_{i+1} are contained in different clusters. Then $X = \sum_{i=1}^{\ell} X_i$ is an upper bound on the distance between the cluster of u and the cluster of v in the cluster graph.

If $|i-j| > k = \frac{4\log n}{\beta}$, then X_i and X_j are independent. Thus, we can color $\{X_i\}_{i=1,\ldots,\ell}$ by $\chi = \frac{4\log n}{\beta}$ colors in such a way that variables of the same color are independent. By [13, Theorem 3.2], we have the following inequality: $\Pr[X \ge \mathrm{E}[X] + t] \le \exp(-2t^2/(\chi \cdot \ell))$. By linearity of expectation and Lemma 8(1), $\mathrm{E}[X] \le 2\beta\ell$. Thus, by setting $t = \beta\ell$, we have

$$\Pr[X \ge 3\beta\ell] \le \exp(-\Omega(\beta^3 \ell / \log n)) = n^{-\Omega(\alpha)}.$$

The lemma follows by a union bound over all $O(n^2)$ possible pairs $\{u,v\}$. Notice that if $\text{dist}(u,v) \le 3\beta D$, then the distance between

the cluster of u and the cluster of v in the cluster graph is already at most $3\beta D$. □

5.1 Outline of the Algorithm

We fix the parameter $\beta = \frac{1}{\log^{1/\epsilon} n}$. Our randomized No-CD algorithm for Broadcast consists of two phases. The first phase is to iteratively run Partition(β) on the current cluster graph $\log_{1/(3\beta)} D$ times. The second phase is to apply Lemma 6 to the last clustering to solve Broadcast.

Throughout the procedure, the cluster graph is implemented as a good labeling, and each layer-0 vertex is responsible for simulating a cluster. The communication between clusters can be done using Down-cast, All-cast, Up-cast in a way similar to that in Section 4.

Details of the First Phase. After performing Partition(β) to get a new clustering, we will later see that the maximum number of layers in any cluster is multiplied by at most $\frac{4\log n}{\beta} + 1 < \frac{5\log n}{\beta} = 5\log^{1+\frac{1}{\epsilon}} n$. Recall that there are in total $\frac{2\log n}{\beta}$ epochs. In each epoch, when we merge a cluster C into another cluster C', the number of layers in C is increased by at most two times the number of layers in C'. See Section 5.4 for implementation details of merging clusters.

Using the above number, throughout the first phase, the maximum number of layers of the underlying good labeling is at most

$$\mathcal{D} = \left(\frac{5\log n}{\beta}\right)^{\log_{1/(3\beta)} D} = D^{\left(\frac{\log \frac{5\log n}{\beta}}{\log \frac{1}{3\beta}}\right)} = D^{1+\epsilon(1+O(1/\log\log n))}.$$

By Property 2 of Lemma 8, with high probability, for each vertex u, the number of distinct clusters that vertices in $N^+(u) = N(u) \cup \{u\}$ belong to is at most

$$\mathcal{C} = O\left(\log_{1/3\beta} n\right) = O\left(\log_{\log^{1/\epsilon} n} n\right) = O\left(\frac{\epsilon \log n}{\log\log n}\right).$$

We will later see that, based on the implementation of the cluster structure in Section 5.2, we can simulate one round of Partition(β) on the cluster graph using $O(\mathcal{D}\mathcal{C}\log^3 n)$ rounds and $O(\mathcal{C}\log^3 n)$ energy in the underlying graph G. The simulation of Partition(β) on the cluster graph is given in Section 5.3. The maintenance of good labeling underlying the clustering is described in Section 5.4. To summarize, the performance for the first phase is:

Time: $\log_{1/(3\beta)} D \cdot O(\log^{3+1/\epsilon} n) \cdot O(\mathcal{D}\mathcal{C}\log^3 n)$,

Energy: $\log_{1/(3\beta)} D \cdot O(\log^{3+1/\epsilon} n) \cdot O(\mathcal{C}\log^3 n)$.

Details of the Second Phase. Consider the cluster graph resulting from the first phase, and let $\mathcal{L}^\star$ be the underlying good labeling. Recall that $\mathcal{D}$ is the maximum number of layers of $\mathcal{L}^\star$. In view of Lemma 9, after the first phase, the diameter of the cluster graph is less than $O(\frac{\log^2 n}{\beta^4}) = O(\log^{2+4/\epsilon} n)$. Notice that the diameter of the cluster graph must be greater than or equal to the diameter of $G_{\mathcal{L}^\star}$. We apply Lemma 6 with $d = O(\log^{2+4/\epsilon} n)$ and $L = \mathcal{D} = D^{1+\epsilon(1+O(1/\log\log n))}$, to solve Broadcast with the following cost:

Time: $O(D^{1+\epsilon(1+O(1/\log\log n))}\log^{4+4/\epsilon} n)$,

Energy: $O(\log^{4+4/\epsilon} n)$.

By doing a variable change $\epsilon' = \epsilon(1 + O(1/\log\log n))$, we have the following theorem.

Theorem 10. *For any $\epsilon \in (0,1)$, there is a randomized* No-CD *algorithm that, w.h.p., solves* Broadcast *in time $O(D^{1+\epsilon}\log^{O(\frac{1}{\epsilon})} n)$ using energy $O(\log^{O(\frac{1}{\epsilon})} n)$.*

5.2 Cluster Structure

We assume that each vertex v has a unique number ID(v), and has a good labeling $\mathcal{L}(v)$. Recall that a good labeling, in general, does not give rise to a unique clustering. To fix a specific clustering, consider the following modifications. We define the cluster id of a cluster C by ID(r), where r is the unique layer-0 vertex in C. We assume that each vertex $v \in C$ knows the cluster id CID(v) = ID(r). We assume the cluster center r has generated a sufficiently long random string $R(r)$, and each vertex $v \in C$ knows $R(v) \stackrel{\text{def}}{=} R(r)$. We call this the *shared random string* of the cluster C.

Suppose that all vertices agree on the two parameters $\mathcal{C}$ and $\mathcal{D}$ meeting the following conditions. For each vertex u, the vertices in $N^+(u)$ belong to at most $\mathcal{C}$ distinct clusters. The number $\mathcal{D}$ is an upper bound on the number of layers of the good labeling. We claim that the following two tasks can be done with $O(\mathcal{C}\log^3 n)$ time and $O(\mathcal{C}\log^3 n)$ energy.

- Downward transmission. Let $i \geq 0$ and V' be a subset of layer-i vertices that have some messages to send. The goal is to have each layer-$(i+1)$ vertex with at least one V'-neighbor *in the same cluster* receive a message from any such neighbor, with high probability.
- Upward transmission. Let $i > 0$ and V' be a subset of layer-i vertices that have some messages to send. The goal is to have each layer-$(i-1)$ vertex with at least one V'-neighbor *in the same cluster* receive a message from any such neighbor, with high probability.

Lemma 11. *In the* No-CD *model, both* Downward transmission *and* Upward transmission *can be solved by a randomized algorithm that takes $O(\mathcal{C}\log^3 n)$ time and $O(\mathcal{C}\log^3 n)$ energy.*

Proof. We only present the proof for Downward transmission, since Upward transmission can be solved analogously. The algorithm is as follows. Repeat the following procedure for $O(\mathcal{C}\log n)$ iterations. Each layer-i vertex $v \in V'$ joins the set S with probability $\frac{1}{\mathcal{C}}$, using the shared random string $R(v)$. Thus, for any two layer-i vertices $u, v \in V'$ in a cluster C, we must have either $u, v \in S$ or $u, v \notin S$. Run SR-comm with S being the above set, and R being the set of all layer-$(i+1)$ vertices. This algorithm takes $O(\mathcal{C}\log^3 n)$ time and $O(\mathcal{C}\log^3 n)$ energy.

Now we prove the correctness of this algorithm. Consider any layer-$(i+1)$ vertex v in cluster C, let $u_1, \ldots, u_x$ be all layer-i neighbors of v in C that are transmitting, and let $u_{x+1}, \ldots, u_k$ be all layer-i neighbors of v not in C that are transmitting. The vertices $u_1, \ldots, u_k$ are contained in at most $\mathcal{C}$ distinct clusters. Within $O(\mathcal{C}\log n)$ iterations, with high probability, there is an iteration where (i) $u_1, \ldots, u_x \in S$, and (ii) $u_{x+1}, \ldots, u_k \notin S$. Thus, v is able to receive a message from a neighbor in C in this iteration. We assume any message contains the cluster id, so that v can check whether a message it receives comes from a neighbor in C. □

5.3 Simulating Algorithms on the Cluster Graph

In view of the definition of SR-comm, we define the CD* model as follows. This model is basically the same as CD, but for the case where at least two neighbors are transmitting, the listener receives any one of these messages instead of receiving noise. The choice of the message that the listener receives can be arbitrary. Observe that Partition(β) works in CD*.

Consider one round of CD* on the *cluster graph* (the graph resulting from contracting each cluster into a vertex). Let $\mathcal{S}$ be the set of all clusters that are transmitting, and let $\mathcal{R}$ be the set of all clusters that are listening. This round can be simulated in the underlying graph G by the following three operations: (i) Down-cast allows the layer-0 vertex representing each cluster $C \in \mathcal{S}$ to broadcast a message to the entire cluster; (ii) All-cast allows messages to be transmitted between the clusters; (iii) Up-cast allows the layer-0 center of each cluster $C \in \mathcal{R}$ to obtain one message sent to the cluster, if any. Recall that the number of layers is at most $\mathcal{D}$.

- Down-cast. For each $C \in \mathcal{S}$, the center r of C generates some message m, and the goal is to let all vertices in C know m. This can be done by transmitting the message layer by layer. The algorithm is as follows. For $i = 0, \ldots, \mathcal{D}-2$, suppose all layer-$i$ vertices have received the message, and then execute Downward transmission to let all layer-$(i+1)$ vertices to receive the message. This operation takes $O(\mathcal{D}C \log^3 n)$ time and $O(C \log^3 n)$ energy.
- All-cast. Let S be the set of all vertices that belong to a cluster in $\mathcal{S}$, and let R be the set of all vertices that belong to a cluster in $\mathcal{R}$. Each $v \in S$ has a message to transmit, and the goal is to let each $u \in R$ such that $N(u) \cap S \neq \emptyset$ to receive some message. This can be solved in a way similar to Lemma 11, and takes $O(C \log^3 n)$ time and $O(C \log^3 n)$ energy.
- Up-cast. For each $C \in \mathcal{R}$, some vertices in a cluster C hold a message, and the goal is to let the center know any one of them, if at least one exists. The algorithm is similar to Down-cast. For $i = \mathcal{D}-1, \ldots, 1$, run Upward transmission to let layer-$(i-1)$ vertices receive messages from layer-i vertices. This takes $O(\mathcal{D}C \log^3 n)$ time and $O(C \log^3 n)$ energy.

Lemma 12. *In the* No-CD *model, we can simulate any* CD* *algorithm on the cluster graph, where each round of the* CD* *algorithm is simulated in $O(\mathcal{D}C \log^3 n)$ time using $O(C \log^3 n)$ energy.*

5.4 Maintaining a Good Labeling

In this section, we show the details of maintaining the good labeling $\mathcal{L}$ as well as other information, such as the cluster id CID(v) and shared random string $R(v)$, while some clusters are being merged.

Let W denote the set of all vertices that successfully received "merging requests" at some time during an execution of Partition(β) (more precisely, at an All-cast operation in Section 5.3). We assume that the merging request sent from a vertex v in a cluster C' contains the following information: ID(v), CID(v), $R(v)$, and $\mathcal{L}(v)$. For each $u \in W$, let $\phi(u)$ be the vertex in a neighboring cluster that successfully sent the merging request to u.

If a cluster C satisfies $C \cap W \neq \emptyset$, then it needs to accept one merging request, and merges itself to a neighboring cluster. More specifically, this is done as follows. First, within the cluster C, we select one leader vertex $v^\star \in C \cap W$, and then we re-root the cluster C at $v^\star$, and assign a new good labeling $\mathcal{L}'$ to all C-nodes.

For example, suppose that the accepted merging request is sent from the vertex $\phi(v^\star) = u^\star \in C'$. Then we need to merge C into C'. If the vertex label of $u^\star$ is 18, then we need to set the new label of $v^\star$ as 19. The rest of the vertices in C will receive labels 20, 21, ... This can be done via Up-cast and Down-cast in Section 5.3 on the old labeling $\mathcal{L}$. That is, this task can be accomplished in $O(\mathcal{D}C \log^3 n)$ time using $O(C \log^3 n)$ energy. The algorithm is as follows.

Step 1: Electing $v^\star$. Perform an Up-cast to let the cluster center of C elect a vertex $v^\star \in C \cap W$, and then perform a Down-cast to let all vertices in C know the decision.

Step 2: Update Labeling $\mathcal{L}'$. Initially, all vertices $v \in C$ have $\mathcal{L}'(v) = \perp$, except that $\mathcal{L}'(v^\star)$ is initialized as the layer number of $\phi(v^\star)$ plus 1. The $\mathcal{L}'$-label of all remaining vertices in C can be assigned using Up-cast and Down-cast as follows.

- Perform an Up-cast. The message of $v^\star$ is its $\mathcal{L}'$-label. Each vertex v that receives a message m sets $\mathcal{L}'(v) = m+1$, and it will transmit the message $m+1$ during the next Upward transmission.
- Perform a Down-cast. For each vertex v that has obtained a $\mathcal{L}'$-label, its message is its $\mathcal{L}'$-label (and it will not reset its $\mathcal{L}'$-label). Each vertex v that has not obtained a $\mathcal{L}'$-label sets $\mathcal{L}'(v) = m+1$, where m is the message it receives.

After the Down-cast in Step 2, everyone in C is guaranteed to receive a vertex label. Information about cluster id and shared random string can also be transmitted through this procedure.

Suppose that in this epoch of Partition(β), the clusters $C_1, \ldots, C_k$ are merged into a cluster C'. Before merging, we let L be maximum number of layers in a cluster $C_1, \ldots, C_k$, and let L' be the number of layers of C'. Then, the number of layers in the new cluster resulting from merging $C_1, \ldots, C_k$ into C' is at most $L' + (2L-1)$.

Since Partition(β) has $(2 \log n)/\beta$ epochs, the maximum number of layers in any cluster is multiplied by at most $1 + (4 \log n)/\beta$ after Partition(β).

6 CONCLUSION AND OMITTED RESULTS

Energy complexity is a natural and attractive concept in wireless radio networks. In this work we presented (to our best knowledge) the first algorithm that solved Broadcast in poly log n energy. Several results were omitted from this extended abstract. We briefly summarize these below. Refer to the full version [9] for details.

Simulation of LOCAL *Algorithms.* With a preprocessing step, one can simulate any LOCAL algorithm in No-CD by scheduling all transmissions to avoid collisions. Specifically, we show that any algorithm $\mathcal{A}$ for the LOCAL model taking time T and energy E can be simulated in randomized No-CD using $O(\Delta^2 T + \Delta \log \Delta \log n)$ time and $O(\Delta(E + \log \Delta \log n))$ energy. In particular, when $\Delta = O(1)$, the Broadcast problem can be solved by a randomized No-CD algorithm in $O(n \log n)$ time with energy cost $O(\log n)$, which shows that the LOCAL lower bound on path graphs of Theorem 1 is matched by a No-CD algorithm on bounded-degree graphs.

Improved Randomized Algorithms for the CD *Model.* The energy complexity for the randomized CD model in Section 4 can be further improved to nearly match the lower bound, up to a small $O(\log\log\Delta/\log\log\log\Delta)$ factor. Specifically, we demonstrate a CD algorithm that takes $O\left(\frac{\log n(\log\log\Delta+(1/\xi))}{\log\log\log\Delta}\right)$ energy and $O\left(\Delta n^{1+\xi}\right)$ time, for any $\xi=\omega(\log\log n/\log n)$.

Time and Energy Optimal Algorithm for Path Graphs. We design a Broadcast algorithm for path graphs that has $O(n)$ worst case runtime with $O(\log n)$ expected energy cost. This shows that optimality in time and energy can be simultaneously achieved.

Deterministic Algorithms. We have a deterministic algorithm for the CD model, also based on iterative clustering. Its energy complexity is $O(\log^3 N\log n)$ but the runtime is $O(nN^2\log n\log N)$.

Open Problems. Assuming energy usage is paramount, can one design Broadcast algorithms meeting our lower bounds: $\Omega(\log n)$ in CD and $\Omega(\log\Delta\log n)$ in No-CD? Can we get the best of both worlds: near optimality in time and energy? Specifically, is there a small constant c for which $O(D\log^c n)$ time and $O(\log^c n)$ energy suffice to solve Broadcast? Can we simultaneously achieve $O(n)$ time and $O(\log n)$ per-vertex energy on *arbitrary* graphs?

REFERENCES

[1] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. 2007. A survey on wireless multimedia sensor networks. *Computer Networks* 51, 4 (2007), 921–960.

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

[3] C. Ambühl. 2005. An Optimal Bound for the MST Algorithm to Compute Energy Efficient Broadcast Trees in Wireless Networks. In *Automata, Languages and Programming*, L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1139–1150.

[4] C Ambühl. 2008. *Minimum Energy Broadcasting in Wireless Geometric Networks.* Springer US, Boston, MA, 1–99.

[5] R. Bar-Yehuda, O. Goldreich, and A. Itai. 1992. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *J. Comput. System Sci.* 45, 1 (1992), 104–126.

[6] M. Barnes, C. Conway, J. Mathews, and D. K. Arvind. 2010. ENS: An Energy Harvesting Wireless Sensor Network Platform. In *Proceedings of the 5th International Conference on Systems and Networks Communications (ICSNC)*. 83–87.

[7] M. A. Bender, T. Kopelowitz, S. Pettie, and M. Young. 2016. Contention Resolution with Log-logstar Channel Accesses. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*. 499–508.

[8] P. Berenbrink, C. Cooper, and Z. Hu. 2009. Energy efficient randomised communication in unknown AdHoc networks. *Theoretical Computer Science* 410, 27 (2009), 2549 – 2561.

[9] Yi-Jun Chang, Varsha Dani, Thomas P. Hayes, Qizheng He, Wenzheng Li, and Seth Pettie. 2017. The Energy Complexity of Broadcast. *CoRR* abs/1710.01800 (2017). arXiv:1710.01800 http://arxiv.org/abs/1710.01800

[10] Y.-J. Chang, T. Kopelowitz, S. Pettie, R. Wang, and W. Zhan. 2017. Exponential separations in the energy complexity of leader election. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. 771–783.

[11] A. Czumaj and P. Davies. 2017. Exploiting Spontaneous Transmissions for Broadcasting and Leader Election in Radio Networks. In *Proceedings of the 2017 ACM Symposium on Principles of Distributed Computing (PODC)*.

[12] A. Czumaj and W. Rytter. 2003. Broadcasting algorithms in radio networks with unknown topology. In *Proceedings of 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 492–501.

[13] D. P. Dubhashi and A. Panconesi. 2009. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press.

[14] M. Farach-Colton, R. J. Fernandes, and M. A. Mosteiro. 2006. Lower Bounds for Clear Transmissions in Radio Networks. In *Proceedings of the 7th Latin American Symposium on Theoretical Informatics (LATIN)*. 447–454.

[15] M. Ghaffari and B. Haeupler. 2013. Near Optimal Leader Election in Multi-Hop Radio Networks. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 748–766.

[16] M. Ghaffari, B. Haeupler, and M. Khabbazian. 2015. Randomized broadcast in radio networks with collision detection. *Distributed Computing* 28, 6 (2015), 407–422.

[17] L. Gsieniec, E. Kantor, D. R. Kowalski, D. Peleg, and C. Su. 2007. Energy and Time Efficient Broadcasting in Known Topology Radio Networks. In *Distributed Computing*, A. Pelc (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 253–267.

[18] B. Haeupler and D. Wajc. 2016. A faster distributed radio broadcast primitive. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 361–370.

[19] T. Jurdzinski, M. Kutylowski, and J. Zatopianski. 2002. Efficient algorithms for leader election in radio networks. In *Proceedings of the 21st Annual ACM Symposium on Principles of Distributed Computing (PODC)*. 51–57.

[20] T. Jurdzinski, M. Kutylowski, and J. Zatopianski. 2002. Energy-Efficient Size Approximation of Radio Networks with No Collision Detection. In *Proceedings of the 8th Annual International Conference on Computing and Combinatorics (COCOON)*. 279–289.

[21] T. Jurdzinski, M. Kutylowski, and J. Zatopianski. 2002. Weak Communication in Radio Networks. In *Proceedings of the 8th International European Conference on Parallel Computing (Euro-Par)*. 965–972.

[22] T. Jurdzinski, M. Kutylowski, and J. Zatopianski. 2003. Weak communication in single-hop radio networks: adjusting algorithms to industrial standards. *Concurrency and Computation: Practice and Experience* 15, 11–12 (2003), 1117–1131.

[23] M. Kardas, M. Klonowski, and D. Pajak. 2013. Energy-efficient leader election protocols for single-hop radio networks. In *Proceedings of the 42nd International Conference on Parallel Processing (ICPP)*. 399–408.

[24] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. 2000. Power consumption in packet radio networks. *Theoretical Computer Science* 243, 1 (2000), 289 – 305.

[25] M. Klonowski, M. Kutylowski, and J. Zatopianski. 2012. Energy efficient alert in single-hop networks of extremely weak devices. *Theoretical Computer Science* 453 (2012), 65 – 74. http://www.sciencedirect.com/science/article/pii/S0304397512001016 Algorithmic Aspects of Wireless Sensor Networks.

[26] D. R. Kowalski and A. Pelc. 2005. Broadcasting in undirected ad hoc radio networks. *Distributed Computing* 18, 1 (2005), 43–57.

[27] E. Kushilevitz and Y. Mansour. 1998. An $\Omega(D\log(N/D))$ Lower Bound for Broadcast in Radio Networks. *SIAM J. Comput.* 27, 3 (1998), 702–712.

[28] M. Kutyłowski and W. Rutkowski. 2003. Adversary Immune Leader Election in ad hoc Radio Networks. In *Algorithms - ESA 2003*, Giuseppe Di Battista and Uri Zwick (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 397–408.

[29] N. Linial. 1992. Locality in Distributed Graph Algorithms. *SIAM J. Comput.* 21, 1 (1992), 193–201.

[30] G. L. Miller, R. Peng, and S. C. Xu. 2013. Parallel graph decompositions using random shifts. In *Proceedings of the twenty-fifth annual ACM symposium on Parallelism in algorithms and architectures (SPAA)*. ACM, 196–203.

[31] K. Nakano and S. Olariu. 2000. Energy-Efficient Initialization Protocols for Single-Hop Radio Networks with No Collision Detection. *IEEE Trans. Parallel Distrib. Syst.* 11, 8 (2000), 851–863.

[32] K. Nakano and S. Olariu. 2002. Uniform leader election protocols for radio networks. *IEEE Transactions on Parallel and Distributed Systems* 13, 5 (2002), 516–526.

[33] C. Newport. 2014. Radio Network Lower Bounds Made Easy. In *Proceedings of the 28th International Symposium on Distributed Computing (DISC)*. 258–272.

[34] D. Peleg. 2000. *Distributed Computing: A Locality-Sensitive Approach.* SIAM.

[35] J. Polastre, R. Szewczyk, and D. Culler. 2005. Telos: enabling ultra-low power wireless research. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*. 364–369.

[36] V. Shnayder, M. Hempstead, B.-r. Chen, G. W. Allen, and M. Welsh. 2004. Simulating the Power Consumption of Large-scale Sensor Network Applications. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, New York, NY, USA, 188–200.

[37] K. M. Sivalingam, M. B. Srivastava, and P. Agrawal. 1997. Low power link and access protocols for wireless multimedia networks. In *Proceedings of the IEEE 47th Vehicular Technology Conference (VTC)*, Vol. 3. 1331–1335.

[38] Y. Wang, F. Nunez, and F. J Doyle. 2012. Energy-efficient pulse-coupled synchronization strategy design for wireless sensor networks through reduced idle listening. *IEEE transactions on signal processing* 60, 10 (2012), 5293–5306.

[39] D. E Willard. 1986. Log-logarithmic selection resolution protocols in a multiple access channel. *SIAM J. Comput.* 15, 2 (1986), 468–477.

[40] W. Ye, J. Heidemann, and D. Estrin. 2002. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 3. 1567–1576 vol.3.