

# Broadcast in Almost Mixing Time

Anton Paramonov  

ETH Zurich, Switzerland

Roger Wattenhofer  

ETH Zurich, Switzerland

---

## Abstract

We study the problem of broadcasting multiple messages in the CONGEST model. In this problem, a dedicated source node  $s$  possesses a set  $M$  of messages with every message of size  $O(\log n)$  where  $n$  is the total number of nodes. The objective is to ensure that every node in the network learns all messages in  $M$ . The execution of an algorithm progresses in rounds, and we focus on optimizing the round complexity of broadcasting multiple messages.

Our primary contribution is a randomized algorithm for networks with expander topology. The algorithm succeeds with high probability and achieves a round complexity that is optimal up to a factor of the network's mixing time and polylogarithmic terms. It leverages a multi-COBRA primitive, which uses multiple branching random walks running in parallel. A crucial aspect of our method is the use of these branching random walks to construct an optimal (up to a polylogarithmic factor) tree packing of a random graph, which is then used for efficient broadcasting.

We also prove the problem to be NP-hard in a centralized setting and provide insights into why lower bounds that can be matched in expanders, namely graph diameter and  $\frac{|M|}{\minCut}$ , cannot be tight in general graphs.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Random network models; Theory of computation  $\rightarrow$  Random walks and Markov chains; Theory of computation  $\rightarrow$  Distributed algorithms; Theory of computation  $\rightarrow$  Graph algorithms analysis

**Keywords and phrases** Distributed algorithms, Expander Graphs, Random graphs, Broadcast, Branching random walks, Tree packing, CONGEST model

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2026.71

**Related Version** *Full Version:* <https://arxiv.org/abs/2502.02165> [42]

**Acknowledgements** The authors thank Diana Ghinea and Bernhard Haeupler for their helpful feedback, which helped improve this paper.

## 1 Introduction

The CONGEST model [43] was originally introduced to study computer networks with limited communication channels. Though later, despite its simplicity, it has given rise to a wide range of theoretical challenges. Progress on those has not only allowed for faster distributed algorithms, but also led to a deeper understanding of fundamental graph abstractions (see, e.g., [23, 22, 15, 45, 6]).

The problem we address in this work fits precisely into this tradition. We study the broadcast of multiple messages, where a single source node holds a collection of messages that must be disseminated so that every node in the network receives them as quickly as possible. In real-world systems, this setting naturally models the distribution of data chunks in peer-to-peer file sharing or block chunks dissemination in blockchain protocols. Theoretically, however, the problem reduces to the tree packing in the underlying network graph – a concept still not fully understood.

We show that this problem admits a fast solution in expander networks. In particular, we present an algorithm that broadcasts multiple messages with near-optimal round complexity: the overhead depends only on the network's mixing time, up to a small polylogarithmic



© Anton Paramonov and Roger Wattenhofer;

licensed under Creative Commons License CC-BY 4.0

43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026).

Editors: Meena Mahajan, Florin Manea, Annabelle McIver, and Nguyễn Kim Thăng

Article No. 71; pp. 71:1–71:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



factor. The key technical contribution enabling this result is a nearly-optimal distributed tree packing procedure for a random graph, which serves as the structural backbone of our dissemination strategy and may be of independent interest beyond this application.

We now proceed to define the setting formally.

## 1.1 Model and Problem

The CONGEST model is defined as follows. The network is modeled as a graph with  $n$  nodes, where execution progresses in synchronous rounds. In each round, a node can send a message of size  $O(\log n)$  bits to each of its neighbors. Importantly, nodes do not have prior knowledge of the network topology.

Although the CONGEST model has been extensively studied over the past two decades, the fundamental problem of broadcasting multiple messages remains unsolved for general topologies.

► **Definition 1** (Multi-message broadcast). *A dedicated source node  $s$  possesses a set  $M$  of  $k$  messages, where each message  $m \in M$  has a size of  $O(\log n)$  bits. The objective is to ensure that every node in the network learns all messages in  $M$ .*

As pointed out by Ghaffari [21], the problem suggests an  $\Omega(D + k)$  round complexity lower bound, where  $D$  is the diameter of the graph. For example, consider a path graph with  $s$  as its first node. Any algorithm would require at least  $D + k - 1$  rounds to transmit all messages to the last node. However, this bound is only *existential*, meaning there exists a graph for which  $\Omega(D + k)$  rounds are needed. In contrast, consider a complete graph with  $k = n$ . Here, broadcasting can be completed in two rounds: in the first round,  $s$  sends the  $i$ -th message to the node  $i$ , and in the second round, each node broadcasts the message it received in round 1. This is significantly better than the  $\Omega(k) = \Omega(n)$  bound suggested by the path graph example. These contrasting cases highlight the importance of algorithms that adapt to the underlying topology. Our paper presents such an algorithm, achieving *universal optimality* [20] on expander graphs. Specifically, it completes the multi-message broadcast on every expander  $G$  in a number of rounds within a small overhead of the best possible for  $G$ . Before stating our results formally, we introduce some necessary terminology.

## 1.2 Preliminaries and Notation

Throughout the paper, for a graph  $G$ ,  $E(G)$  is the edge set,  $V(G)$  is the vertex set,  $D(G)$  is the diameter,  $d_G(v)$  is the degree of a node  $v$  in  $G$ ,  $\delta(G)$  is the smallest vertex degree, and  $\Delta(G)$  is the largest vertex degree. We do not explicitly specify  $G$  if it is obvious from the context, e.g., we can write  $\delta$  instead of  $\delta(G)$ . With high probability (w.h.p.) means with a probability of at least  $1 - O(\frac{1}{n^C})$  for some constant  $C > 0$ , with the probability being taken over both the randomness of the graph (when we assume random graphs) and the random bits of the algorithm. We assume that  $\tilde{O}$  and  $\tilde{\Omega}$  hide *polylog*( $k, n$ ) factors.

In multiple places in this paper, we are using classical Chernoff bounds.

► **Definition 2** (Chernoff bounds). *Let  $X = \sum_{i=1}^n X_i$  be the sum of independent Bernoulli random variables with  $\mathbb{E}[X] = \mu$ . Then, the following Chernoff bounds hold:*

$$\begin{aligned} \Pr(X \leq (1 - \delta)\mu) &\leq e^{-\delta^2\mu/2}, \quad 0 \leq \delta \\ \Pr(X \geq (1 + \delta)\mu) &\leq e^{-\delta^2\mu/(2+\delta)}, \quad 0 \leq \delta \\ \Pr(|X - \mu| \geq \delta\mu) &\leq 2e^{-\delta^2\mu/3}, \quad 0 \leq \delta \leq 1. \end{aligned}$$

One of the key graph-theoretical components in our approach is *tree packing*. A tree packing of a graph  $G$  is a collection of spanning subtrees of  $G$ . The tree packing is characterized by three parameters: (1) its size  $S$ , i.e., the number of trees, (2) its diameter  $H$ , i.e., the maximal diameter of a tree, and (3) its weight  $W$ , i.e., the maximal number of trees sharing a single edge.

The present work focuses specifically on two graph families, namely Erdős–Rényi graphs and expanders. An *Erdős–Rényi graph*  $G(n, p)$  is a graph on  $n$  vertices where each edge exists independently from others with probability  $p$  [16]. Throughout the paper, let  $C_p$  denote a sufficiently large constant<sup>1</sup> such that  $G(n, p)$  is connected w.h.p. for  $p \geq \frac{C_p \log n}{n}$ . The condition  $p = \Omega(\frac{\log n}{n})$  is necessary, since for  $p \leq \frac{\log n}{n}$ , there is a constant probability that the graph is disconnected [16].

We refer to a graph as an *expander* if it has small (polylogarithmic) mixing time<sup>2</sup>.

For our purposes, it will be convenient to define the mixing time of an undirected graph by reconsidering it as being bidirected, that is, with each undirected edge  $(u, v)$  replaced with  $(u, v)$  and  $(v, u)$ . The *mixing time*  $\tau_{mix}$  of a bidirected graph is defined as follows. Consider a lazy random process that starts at an arbitrary edge of the graph. At each step, with probability  $\frac{1}{2}$ , the process remains at the current edge, and with probability  $\frac{1}{2}$ , it transitions to a uniformly random adjacent edge (a directed edge  $e_2$  is adjacent to a directed edge  $e_1$  if they are of the form  $e_1 = (u, v)$  and  $e_2 = (v, w)$ ). It is known that this process admits a stationary distribution  $\pi$ , which is uniform over all edges. Moreover, regardless of the starting edge, the distribution  $D_t$  of the walk after  $t$  steps converges to  $\pi$ . We define  $\tau_{mix}$  as the smallest  $t$  for which  $D_t$  is inversely polynomial close to  $\pi$ . For a formal definition, please refer to the full version of the paper [42]. We are now ready to formally state our results.

### 1.3 Our Contribution

Our main result is an algorithm to solve the multi-message broadcast problem with only an overhead of  $\tilde{O}(\tau_{mix})$ . We highlight that our algorithm can be run on *any* graph, but it will only be efficient on graphs with small  $\tau_{mix}$ , i.e., expanders.

► **Theorem 3.** *There exists a randomized distributed algorithm that for any graph  $G$  solves the multi-message broadcast problem in  $O(\log^3 n \cdot \tau_{mix} \cdot \text{OPT})$  rounds with high probability, where  $\tau_{mix}$  is the mixing time of  $G$  and  $\text{OPT}$  is the optimal round complexity for the given problem instance.*

In the analysis, we use  $k/\delta(G)$  as a natural lower bound for  $\text{OPT}$ . Indeed, if a node has degree  $\delta(G)$ , it requires at least  $k/\delta(G)$  rounds to receive all  $k$  messages. While this bound is meaningful on expander graphs, we show in Section 5.2 that it can be far from tight even on graphs with constant diameter, which highlights the necessity of exploiting expansion properties in our approach.

We design the algorithm for an arbitrary expander by building on the algorithm for random graphs, which achieves near-optimal performance in an important special case when the network is modeled as an Erdős–Rényi graph  $G(n, p)$ .

► **Theorem 4.** *For an Erdős–Rényi graph  $G(n, p)$  with  $p \geq \frac{C_p \log n}{n}$ , there exists a distributed randomized algorithm that completes the broadcast in  $O(\log^2 n + \log n \cdot \frac{k}{\delta(G)})$  rounds w.h.p.*

<sup>1</sup> It suffices to take  $C_p = 2700$ . It was not a concern for the present work to optimize this constant.

<sup>2</sup> An alternative way is to say that expander graphs are those that feature an inverse polylogarithmic *conductance*; these two notions are equivalent.

To obtain Theorem 4, we use the following result of independent interest, which contributes to a line of work [24, 11, 21, 7, 19] on low-diameter tree packing:

► **Theorem 5.** *For an Erdős–Rényi graph  $G(n, p)$  with  $p \geq \frac{C_p \log n}{n}$ , there exists a distributed randomized algorithm that produces a tree packing of size  $\delta(G)$ , diameter  $O(\log n)$ , and weight  $O(\log n)$  w.h.p.*

We construct the latter algorithm by utilizing multiple Coalescing Branching Random Walks [12] that run in parallel. To the best of our knowledge, this technique has never been used before in the context of distributed algorithms.

Finally, to map the terrain of the problem, we prove the hardness result in the centralized setting. Namely, we show that computing the exact number of rounds required for multi-message broadcast is NP-hard on general graphs.

## 2 Related Work

### Previous Work

The first work to address universal optimality for the multi-message broadcast problem in the CONGEST model was by Ghaffari [21]. In that paper, the algorithm consists of two phases: (1) constructing a *tree packing*, and (2) performing the broadcast using the constructed tree packing. With a tree packing of diameter  $H$ , size  $S$ , and weight  $W$ , one can complete a multi-message broadcast in  $O((H + \frac{k}{S}) \cdot W)$  rounds by splitting messages uniformly across the trees and propagating them sequentially within each tree. However, the limitation of [21] is that constructing the tree packing requires  $\tilde{\Omega}(D + k)$  rounds, preventing the approach from achieving universal optimality. Observe that the problem can be solved in  $O(D + k)$  rounds by a naive strategy: first construct a BFS tree rooted at  $s$ , and then downcast messages in it one by one. Thus, the result of [21] yields an improvement only when multiple consecutive instances of the problem are solved, allowing the precomputed tree packing to be reused.

A subsequent work by Ghaffari et al. [8] considered the tree packing approach on highly connected graphs, i.e., graphs with high edge connectivity  $\lambda$ . The primary result of this work is an algorithm that runs in  $\tilde{O}(\frac{n+k}{\lambda})$  rounds. This complexity is optimal when  $k = \Omega(n)$ , as  $\frac{k}{\lambda}$  represents an information-theoretic lower bound. However, the algorithm may incur a  $\Omega(n)$  factor overhead in cases where  $\lambda$  and  $k$  are small compared to  $n$ .

Notably, both [21] and [8] consider a slightly more general problem where initially  $M$  is not necessarily known to a single node, but different nodes can possess disjoint subsets of  $M$ . We adhere to our version, where  $M$  is initially held by a single node, as it simplifies the presentation. Importantly, when a tree packing is available, the multiple-source version can be reduced to the single-source version without increasing the round complexity (see Remark 16).

### Routing

One fundamental information-dissemination problem in distributed computing is *routing*, where the goal is to deliver a set of messages from source nodes to their respective destination nodes. Unlike *broadcast*, which involves sending the same message(s) to all nodes, routing requires sending individual message(s) for each source-destination pair.

Ghaffari et al. [25] approach this problem for expanders by constructing a hierarchy of recursively embedded Erdős–Rényi graphs, and achieve routing in  $\tau_{\text{mix}} \cdot 2^{O(\sqrt{\log n \log \log n})}$  rounds. Note that  $2^{O(\sqrt{\log n \log \log n})}$  dominates  $\log^c n$  for any constant  $c$ . This result was

further improved in [26], which reduces the round complexity to  $O(2\sqrt{\log n})$ . Subsequently, Chang and Thatchaphol [9] presented a deterministic version of expander routing, matching (up to polylogarithmic factors) the round complexity of the randomized algorithm by Ghaffari et al.

For general graphs, Haeupler et al. [29] provide a routing algorithm that runs in  $\text{poly}(D) \cdot n^{o(1)}$  rounds. Their approach leverages *expander decomposition* and *hop-constrained expanders* – subgraphs with small diameter and strong expansion properties. In fact, [29] obtain a stronger result: given that for every source-sink pair  $(s_i, t_i)$ , the source  $s_i$  is at most  $h$  hops from its destination  $t_i$ , routing can be completed in  $O(D + \text{poly}(h)) \cdot n^{o(1)}$  rounds.

Unlike the above approaches, our paper provides an algorithm with only a polylogarithmic overhead.

### Network Information Flow

The network information flow problem [1] is defined as follows. The network is a directed graph  $G(V, E)$  with edge capacities, a source node  $s \in V$ , and sink nodes  $T \subseteq V$ . The question is: at what maximal rate can the source send information so that all of the sinks receive that information at the same rate? In the case of a single sink  $t$ , the answer is given by the  $\text{max-flow}(s, t)$ . However, when there are multiple sinks  $T$ , the value  $\min_{t \in T} \text{max-flow}(s, t)$  may not be achievable if nodes are only allowed to relay information. In fact, the gap can be as large as a factor of  $\Omega(\log n)$  [33]. Nevertheless, if intermediate nodes are allowed to send (linear [37]) codes of the information they receive, then  $\min_{t \in T} \text{max-flow}(s, t)$  becomes achievable [1]. Notably, in the specific case where  $T = V \setminus \{s\}$  (the setting considered in the present paper), the rate of  $\min_{t \in T} \text{max-flow}(s, t)$  becomes achievable without coding [50]. The decentralized version of network information flow was studied in [31, 18, 30]. The most relevant work in this direction is [47] by Swamy et al., where the authors establish an optimal algorithm for the case of random graphs whose radius is almost surely bounded by 3. Our approach works for general expanders, and in the case of random graphs, allows an expected radius to grow infinitely with  $n$  (see [10] for analysis of the diameter of a random graph).

The key difference between the network information flow problem and the multi-message broadcast in CONGEST is that in our problem, the focus is on round complexity, whereas in the information flow problem, the solution is a “static” assignment of messages to edges, and the focus is on throughput.

### Tree Packing

The problem of tree packing has been extensively studied, as summarized in the survey by Palmer [41]. Foundational results in this area include those by Tutte [48] and Nash-Williams [40], who demonstrated that an undirected graph with edge connectivity  $\lambda$  contains a tree packing of size  $\lfloor \frac{\lambda}{2} \rfloor$ . Edmond [14] extended this result to directed graphs, showing that such graphs always contain  $\lambda$  pairwise edge-disjoint spanning trees rooted at a sender  $s \in V$ , where  $\lambda$  is the minimum number of edges that must be removed to make some node unreachable from  $s$ . However, these results do not address the diameter of the tree packing.

Chuzhoy et al. [11] tackled the challenge of finding tree packings with small diameter. They presented a randomized algorithm that, given an undirected graph with edge connectivity  $\lambda$  and diameter  $D$ , outputs with high probability a tree packing of size  $\lfloor \frac{\lambda}{2} \rfloor$ , weight 2, and diameter  $O((101k \log n)^D)$ .

Tree packing on random graphs was studied by Gao et al. [19], who showed that asymptotically almost surely, the size of a spanning tree packing of weight 1 for a Erdős–Rényi graph  $G(n, p)$  is  $\min \left\{ \delta(G), \frac{|E(G)|}{n-1} \right\}$ , which corresponds to two straightforward upper bounds.

In the CONGEST model, tree packing was investigated by Censor-Hillel et al. [7]. They proposed an algorithm to decompose an undirected graph with edge connectivity  $\lambda$  into fractionally edge-disjoint weighted spanning trees with total weight  $\lceil \frac{\lambda-1}{2} \rceil$  in  $\tilde{O}(D + \sqrt{n\lambda})$  rounds. Furthermore, they proved a lower bound of  $\tilde{\Omega}(D + \sqrt{\frac{n}{\lambda}})$  on the number of rounds required for such a decomposition.

### Branching Random Walks in Networks

The cover time of a random walk [36] on a graph is the time needed for a walk to visit each node at least once. Unfortunately, the expected value of this quantity is  $\Omega(n \log n)$  even for a clique, making this primitive less useful in designing fast algorithms. Consequently, several attempts have been made to accelerate the cover time. Alon et al. [2] proposed initiating multiple random walks from a single source. Subsequent work by Elsässer and Sauerwald refined their bounds, demonstrating that  $r$  random walks can yield a speed-up of  $r$  times for many graph classes. Variations of multiple random walks have been applied in the CONGEST model to approximate the mixing time [39], perform leader election [35, 27], and evaluate network conductance [17, 4].

A branching random walk [46] (BRW) modifies the classical random walk by allowing nodes to emit multiple copies of a walk upon receipt, rather than simply relaying it. This branching behavior potentially leads to exponential growth in the number of walks traversing the graph, significantly reducing the cover time. Gerraoui et al. [28] demonstrated that BRWs can enhance privacy by obscuring the source of gossip within a network. Recently, Aradhya et al. [3] employed BRWs to address permutation routing problems on subnetworks in the CONGEST model.

Despite these applications, to the best of our knowledge, the branching random walk remains underexplored in distributed computing, and this work seeks to showcase its untapped potential.

## 3 Algorithm Overview

In this section, we provide a high-level overview of our algorithm, which consists of two major parts. First, we embed a virtual Erdős–Rényi graph  $G(n, p)$  atop the physical network  $H$ , and then we solve the problem on  $G$ . The reason we do this embedding is to transform an arbitrary expander into an almost-regular one. We explain the embedding procedure in Section 3.1, and from that point on, we focus solely on solving multi-message broadcast on an Erdős–Rényi graph. In Section 3.2, we overview COBRA – the main building block of our algorithm for random graphs, and in Section 3.3 we describe the algorithm itself.

### 3.1 Embedding a Random Graph

In this section, we describe how to embed a random graph atop a given expander. A related technique was used by Ghaffari et al. [25], where it suffices to embed a sparse Erdős–Rényi graph with expected degree  $O(\log n)$ . In our setting, however, it is crucial to preserve the *minimum degree* of the host graph. To this end, we reuse Lemma 7 from [25], but we also introduce additional ideas of node groups and rejection sampling to meet our stronger requirements.

Embedding a graph  $G$  atop a host graph  $H$  involves creating virtual nodes  $V(G)$  and establishing edges  $E(G)$  between them so that

- Every virtual node  $u \in V(G)$  is simulated by some physical node  $\text{host}(u) \in V(H)$ .
- If a virtual node  $u \in V(G)$  sends a message to  $v \in V(G)$  along the edge  $(u, v) \in E(G)$ , this should be simulated by  $\text{host}(u)$  sending the same message to  $\text{host}(v)$  via some path in  $H$ .

Our construction will guarantee that each round of communication in  $G$  can be simulated in  $O(\tau_{\text{mix}}(H) \cdot \log n)$  rounds in  $H$  and that  $\delta(G)$  will be close to  $\delta(H)$ . This gives us the following “lifting”: if there is an algorithm that solves a problem in  $f(k, \log(|V(G)|), \delta(G))$  rounds on  $G$  for some function  $f$ , then there is an algorithm that solves a problem in essentially  $f(k, \log n, \delta(H)) \cdot \tau_{\text{mix}}(H) \cdot \log n$  rounds on  $H$ . Formally,

► **Theorem 6.** *Assume there exists an algorithm that solves multi-message broadcast on an Erdős–Rényi graph  $G$  in  $O(\frac{k}{\delta(G)} \cdot \log(|V(G)|) + \log^2(|V(G)|))$  rounds w.h.p. Then there exists an algorithm that solves multi-message broadcast on any graph  $H$  on  $n$  vertices in*

$$O\left(\frac{k}{\delta(H)} \cdot \tau_{\text{mix}}(H) \cdot \log^2 n + \tau_{\text{mix}}(H) \cdot \log^3 n\right)$$

rounds w.h.p.

This round complexity consists of three terms:  $O((\frac{k}{\delta(H)} \cdot \log n + \log^2 n) \cdot \tau_{\text{mix}} \cdot \log n)$  for simulating an algorithm,  $O(\tau_{\text{mix}}(H) \cdot \log^2 n)$  rounds for constructing an embedding of an Erdős–Rényi graph knowing  $\tau_{\text{mix}}(H)$ , and  $O(\tau_{\text{mix}}(H) \cdot \log^2 n)$  for estimating the  $\tau_{\text{mix}}(H)$ . Below, we describe the embedding and estimation parts.

### Embedding Construction

We now outline how to construct the embedding of an Erdős–Rényi graph atop the expander  $H$  preserving the minimal degree, assuming nodes have an estimate of  $\tau_{\text{mix}}$ .

We need to define a set of nodes of  $G$ . As an auxiliary concept, we start by defining *sub-nodes*. First, rethink  $H$  as being bidirectional, that is, replace every undirected edge  $(u, v) \in E(H)$  with two directed edges  $(u, v)$  and  $(v, u)$ . Now, we associate a *sub-node* with each directed edge, and we say that a node  $u \in V(H)$  simulates all the sub-nodes of its outgoing edges. Next, for each  $u \in V(H)$ , we arbitrarily group sub-nodes  $u$  is simulating into *groups* of size exactly  $\delta(H)$ . This might leave some residual sub-nodes that will not be assigned to any group; we call those *inactive*, and others are called *active*. Note that at most half of all sub-nodes can be inactive. Finally, we define nodes in  $G$  to be the aforementioned groups, and we make a node  $u \in V(H)$  simulate a node  $v \in V(G)$  if it simulates the sub-nodes in the respective group.

To establish edges in  $G$ , we launch lazy random walks of length  $\tau_{\text{mix}}(H)$  from each active sub-node. The following lemma guarantees that those can be executed in parallel in only  $O(\tau_{\text{mix}}(H) \cdot \log n)$  rounds w.h.p.

► **Lemma 7** (Lemma 2.5 [25]). *Let  $H = (V, E)$  be an  $n$ -node graph. Suppose we wish to perform  $T = n^{O(1)}$  steps of a collection of independent lazy random walks in parallel. If each node  $v \in V$  initiates at most  $d_H(v)$  walks, then w.h.p., the  $T$  steps of all walks can be performed in  $O(T \cdot \log n)$  rounds in a distributed setting.*

If a random walk that started from a sub-node  $a$ , terminates at an active sub-node  $b$ , we call such a walk *successful* and we propagate the “success” message back to  $a$  by simply executing the walk in reverse. This establishes the edge between  $\text{group}(a) \in V(G)$  and

group( $b$ )  $\in V(G)$ . If, on the other hand, the walk from  $a$  terminates at an inactive node, we call such a walk *failed* and we propagate the “failure” message back to  $a$ . Sub-nodes that received a “failure” message retry the same process again. The following lemma shows that only a few retries are needed.

► **Lemma 8.** *After at most  $O(\log n)$  retries, all sub-nodes will execute a successful random walk w.h.p.*

**Proof.** By the definition of a mixing time, a random walk from a given sub-node is distributed (almost) uniformly among all the sub-nodes. Hence, given that at least half of all sub-nodes are active, a probability of success in one round is at least  $1/2$  (minus a negligible term due to the “almost” uniformity). Therefore, the probability of not succeeding once after  $100 \log n$  trials is no more than  $\frac{1}{n^{10}}$ . Applying the union bound over all sub-nodes completes the proof. ◀

Once paths are established, communication along  $(u, v) \in E(G)$  is simulated by routing a message along the corresponding walk path in  $H$ , which, by Lemma 7, can be done in  $O(\tau_{mix}(H) \cdot \log n)$  rounds.

We remark that the process described above is equivalent to sampling with replacement  $\delta(H)$  neighbors for each node  $v \in V(G)$  independently (almost) uniformly at random. While this is not a canonical definition of an Erdős–Rényi graph, the resulting distribution is equivalent, modulo a negligible inversely polynomial small probability.

### On Distributed Estimation of Mixing Time

Note that in order to implement this simulation, nodes do not need prior knowledge of  $\tau_{mix}$ . Instead, they can estimate the mixing time of the network using the decentralized algorithm of Kempe and McSherry [34], which runs in  $O(\tau_{mix} \cdot \log^2 n)$  rounds (without prior knowledge of  $\tau_{mix}$ ). Their algorithm can be used to estimate the second principal eigenvalue  $\lambda$  of the transition matrix, which relates to the mixing time via the following inequality:

► **Theorem 9** ([44]). *Given a graph on  $n$  nodes with mixing time  $\tau_{mix}$  and second principal eigenvalue  $\lambda$ , it holds that*

$$\left( \frac{1}{1-\lambda} - 1 \right) \log n \leq \tau_{mix} \leq 2 \log n \cdot \frac{1}{1-\lambda}.$$

Combining all together, we can prove Theorem 6.

**Proof of Theorem 6.** First, estimate the mixing time  $\tau_{mix}(H)$  of the graph  $H$  using the approach from [34]. This takes  $O(\tau_{mix}(H) \cdot \log^2 n)$  rounds.

Let the nodes of  $H$  discover  $\delta(H)$ . This can be done through building a BFS tree in  $H$  within  $O(D(H)) = O(\tau_{mix}(H))$  rounds.

Knowing  $\tau_{mix}(H)$  and  $\delta(H)$ , compute an embedding of an Erdős–Rényi graph  $G(n', \frac{\delta(H)}{n'})$  as described above, where  $n'$  is the number of sub-node groups in  $H$ . Make the source node for a multi-message broadcast in  $G$  any node in  $G$  that is simulated by the source of an original problem instance in  $H$ . Solve the problem in  $G$  in  $O(\frac{k}{\delta(G)} \cdot \log n + \log^2 n) = O(\frac{k}{\delta(H)} \cdot \log n + \log^2 n)$  rounds, simulating each round of  $G$  in  $O(\tau_{mix}(H) \cdot \log n)$  rounds in  $H$ . ◀

### 3.2 Coalescing-Branching Random Walk

We now overview the main primitive for our algorithm for an Erdős–Rényi graph – the COalescing-BRAnching Random Walk (COBRA walk). COBRA walk was first introduced by Dutta et al. [13] in their work “Coalescing-Branching Random Walks on Graph” [13], with subsequent refinements presented in [12, 38, 5]. The COBRA walk is a generalization of the classical random walk, defined as follows: At round 0, a source node  $s \in V$  possesses a token. At round  $r$ , each node possessing a token selects  $\kappa$  of its neighbors uniformly at random, sends a token copy to each of them, and these neighbors are said to possess a token at round  $r + 1$ . Here,  $\kappa$ , referred to as the *branching factor*, can be generalized to any positive real number [12]. When  $\kappa = 1$ , the COBRA walk reduces to the classical random walk. From now on, we consider  $\kappa$  to always be 2. It is important to note that if a node receives multiple token copies in a round, it behaves as if it has received only one token; it will still choose  $\kappa$  neighbors uniformly at random. This property, where received token copies coalesce at a node, gives the primitive its name.

Cooper et al. [12] studied the cover time of the COBRA walk on regular expanders and obtained the following theorem, which we use in our result

► **Theorem 10** (Cooper et al. [12]). *Let  $G$  be a connected  $n$ -vertex regular graph. Let  $\lambda_2$  be the second largest eigenvalue (in the absolute value) of the normalized adjacency matrix of  $G$ . Then after  $O\left(\frac{\log n}{(1-\lambda_2)^3}\right)$  steps the COBRA walk covers  $G$  with probability  $1 - O\left(\frac{1}{n^2}\right)$ .*

We point out that in [12], the bound on probability is  $1 - O\left(\frac{1}{n}\right)$ , though the analysis, which is based on Chernoff bounds, can be adapted so that the probability is  $1 - O(n^C)$  for any constant  $C$  and the cover time is only multiplied by a constant.

In the upcoming analysis of our result, we will make sure that  $\lambda_2$  is no more than  $\frac{13}{14}$  w.h.p. Let  $C_T$  be a sufficiently large constant so that a COBRA walk covers a regular graph with  $\lambda_2 \leq \frac{13}{14}$  with probability at least  $1 - O\left(\frac{1}{n^2}\right)$  in  $C_T \cdot \log n$  rounds. From now on, we define  $T$  to be  $C_T \log n$ .

### 3.3 Random Graph Algorithm Description

The algorithm to solve the multi-message broadcast on an Erdős–Rényi graph  $G(n, p)$  proceeds through the following steps:

1. **Building a BFS Tree and Gathering Information:** Nodes construct a BFS tree rooted at the source node  $s$ . Using the tree, every node learns the total number of nodes  $|V|$ , the minimum degree  $\delta$ , and the maximum degree  $\Delta$ . This step takes  $O(D)$  rounds and does not require any prior knowledge of the graph topology.
2. **Regularizing the Graph:** Each node  $v$  adds  $\Delta - \deg(v)$  self-loops to its adjacency list to make the graph regular. In our analysis, we will show that each node adds a relatively small number of self-loops. This operation is purely local and requires no communication between nodes.
3. **Constructing Spanning Subgraphs through Multiple COBRA Walks:** The source node  $s$  initiates  $\delta$  COBRA walks by creating  $\delta$  tokens labeled 1 through  $\delta$ . When a token from the  $i$ -th COBRA walk is sent along an edge  $e : (u, v)$ , nodes  $u$  and  $v$  mark  $e$  as part of the  $i$ -th subgraph. Note that a single edge may belong to multiple subgraphs. When running multiple COBRA walks simultaneously, congestion can occur if a node attempts to send multiple tokens from different COBRA walks along the same edge in a single round. Since only one token can traverse an edge per round, this creates a bottleneck that needs to be managed. To address this, we organize the process into

*phases*, where each phase consists of 2 rounds. In each phase, spanning rounds  $\{2r, 2r + 1\}$ , every node  $u$  distributes two tokens for each COBRA walk whose token(s) it received during the previous phase. Since there are  $\delta$  COBRA walks, node  $u$  could have received at most  $p \leq \delta$  distinct tokens. Let these tokens be denoted by  $t_1, \dots, t_p$ . Node  $u$  then distributes each token twice, as follows: it enumerates its neighbors as  $v_1, \dots, v_\ell$ , with  $\ell \geq \delta$ , generates two independent random permutations  $\sigma_1, \sigma_2 \in S_\ell$ , and sends token  $t_i$  to neighbor  $v_{\sigma_1(i)}$  in round  $2r$  and to neighbor  $v_{\sigma_2(i)}$  in round  $2r + 1$ . Consequently, for any fixed COBRA walk  $i$  and any node  $u$ , the token belonging to the  $i$ -th COBRA walk is sent to a neighbor of  $u$  chosen uniformly at random, independently of the other tokens in that same COBRA walk. (Different cobra walks, however, need not be independent.) This step completes in  $T$  phases.

**Parallel execution.** For the rest of the algorithm, we run protocols on all the constructed subgraphs in parallel. To achieve this despite potential congestion (recall that each edge may belong to multiple subgraphs), we again organize the execution into phases. Now, each phase spans  $2T$  rounds, ensuring that messages sent along any shared edge are distributed across the protocols without conflict. Specifically, if a protocol would send a message along an edge in a particular round when executed independently, all such messages from different subgraphs are scheduled within the same phase. This phased execution allows all protocols to proceed in parallel while respecting the edge capacity.

4. **Constructing Tree Packings:** The source  $s$  initiates a BFS on each subgraph to transform it into a tree. By the end of this step, the algorithm constructs a tree packing  $\{T_i\}_{i \in [\delta]}$ . This step takes the number of phases that is at most the maximal eccentricity of  $s$  among all the subgraphs, that is at most  $O(T)$  phases, and hence  $2T \cdot O(T) = O(T^2)$  rounds.
5. **Distributing Messages:** The source node  $s$  evenly divides the set of messages  $M$  across the  $\delta$  trees, ensuring that each tree receives  $\frac{k}{\delta}$  messages. These messages are then downcasted along the trees one by one. To broadcast  $\frac{k}{\delta}$  messages in a single tree of diameter  $O(T)$  one needs  $O(\frac{k}{\delta} + T)$  rounds. Hence, doing it in parallel in all trees takes  $O(T \cdot (\frac{k}{\delta} + T))$  rounds.

## 4 Proof

In this section, we formally state our results and their auxiliaries for the Erdős–Rényi graph. We start by providing a high-level overview.

### 4.1 Proof Outline

The proof proceeds in three main steps. First, we argue that making a random graph regular by adding self-loops does not significantly affect its expansion properties. To this end, we rely on a result of Hoffman et al. [32], which shows that a random graph is a good expander with high probability. We also use standard Chernoff bound arguments to establish that random graphs are nearly regular, meaning the ratio between maximum and minimum degrees is close to one. Finally, we invoke a classical consequence of Weyl’s inequality, which ensures that small perturbations to the diagonal of a matrix only slightly affect its eigenvalues. Together, these ingredients imply that regularizing the graph preserves its spectral expansion up to a small error. We discuss the details at the end of this section, namely in subsection 4.4.

Assuming the expansion properties remain, the second step is to prove that each individual COBRA walk successfully covers the entire network. The permutation trick allows us to claim that, though COBRA walks are not independent, their marginal distributions stay as if they were. Using this, Theorem 10 by Cooper et al., together with a union bound, guarantees that all COBRA walks cover the whole graph within  $O(\log n)$  phases w.h.p.

In the final step, we argue that the algorithm produces a tree packing with size  $\delta(G)$ , diameter  $O(\log n)$ , and weight  $O(\log n)$ . This tree packing allows for broadcasting all messages in  $O(\log^2 n + \log n \cdot \frac{k}{\delta(G)})$  rounds. This is optimal up to additive  $\log^2 n$  and multiplicative  $\log n$ , as  $\frac{k}{\delta(G)}$  provides a natural lower bound for the optimal broadcast time: if there is a node with degree  $\delta$ , it needs at least  $k/\delta$  rounds to receive  $k$  messages.

We now give the detailed proof starting from step 2, assuming step 1.

## 4.2 Success of Multiple COBRAs

In this section, we demonstrate that multiple COBRA walks cover the graph fast, provided that it retains its expansion properties after adding self-loops. Formally, we assume the following lemma, which we prove in section 4.4.

► **Lemma 11.** *With probability at least  $1 - O(\frac{1}{n^2})$ , for  $p \geq \frac{C_p \log n}{n-1}$ , an Erdős-Rényi graph  $G(n, p)$  can be transformed into  $G'$  by adding weighted self-loops to the nodes so that (1)  $G'$  is regular, (2)  $1 - \lambda_2(G') \geq \frac{1}{14}$ .*

Now, to see why multiple COBRAs do not congest, let us recall the permutation trick used in the algorithm. To be able to run multiple COBRA walks in parallel, we partition the execution into *phases*, each lasting 2 rounds. In each phase, spanning rounds  $2r, 2r + 1$ , every node  $u$  distributes two tokens for each COBRA walk whose token(s) it received during the previous phase. Since there are  $\delta$  COBRA walks, node  $u$  could have received at most  $p \leq \delta$  distinct tokens. Let these tokens be denoted by  $t_1, \dots, t_p$ . Node  $u$  then distributes each token twice, as follows: it enumerates its neighbors as  $v_1, \dots, v_\ell$ , with  $\ell \geq \delta$ , generates two independent random permutations  $\sigma_1, \sigma_2 \in S_\ell$ , and sends token  $t_i$  to neighbor  $v_{\sigma_1(i)}$  in round  $2r$  and to neighbor  $v_{\sigma_2(i)}$  in round  $2r + 1$ .

The claim below summarizes the properties resulting from the procedure described:

▷ **Claim 12.** For each COBRA walk, a phase corresponds exactly to a round in the execution where this COBRA walk runs in isolation. In particular, for a given COBRA walk, every token is sent independently of other tokens of this walk and to a uniformly chosen neighbor.

To conclude the analysis of multi-COBRA's performance on a random graph, we combine Lemma 11 and Claim 12 and get the following lemma.

► **Lemma 13.** *If the initial network graph is an Erdős-Rényi graph  $G(n, p)$  with  $p \geq \frac{C_p \log n}{n}$ , all COBRA walks cover the graph in  $O(T)$  rounds with probability at least  $1 - O(\frac{1}{n})$ .*

**Proof.** By Lemma 11 we know that  $G'$  – the graph we obtain from  $G$  after adding self-loops – has  $1 - \lambda_2(G') \geq \frac{1}{14}$  with probability at least  $1 - O(\frac{1}{n^2})$ . Therefore, according to Theorem 10, a COBRA walk succeeds to cover  $G'$  in  $O\left(\frac{\log n}{(1 - \lambda_2(G'))^3}\right) = O(T) = O(\log n)$  rounds with probability at least  $1 - O(\frac{1}{n^2})$ . And Claim 12 tells us that while the COBRA walks are not independent, the marginal distribution of each individual process matches the distribution it would have under independent execution. Hence, we can apply a union bound and conclude that  $\delta(G) \leq n$  COBRA walks cover  $G'$  in  $O(\log n)$  phases with probability at least  $1 - O(\frac{1}{n})$ . ◀

We remark that the analysis in [12] is done for simple regular graphs (i.e., regular graphs not featuring self-loops). However, the arguments apply verbatim if self-loops are allowed, with symbols reinterpreted to mean the number of outgoing edges of a node instead of the number of neighbors.

### 4.3 Tree Packing and Broadcast

In this section, we show how to obtain a tree packing from multi-COBRA's edge assignment and describe how we use this tree packing to broadcast the messages.

The following two lemmas speak about the properties of the spanning graphs obtained via multi-COBRA.

► **Lemma 14.** *After multi-COBRA completes all  $T$  phases, every edge of the graph belongs to at most  $O(\log n)$  subgraphs.*

**Proof.** At each phase, each edge  $(u, v)$  is assigned to at most 4 subgraphs (two via tokens from  $u$  to  $v$ , and two from  $v$  to  $u$ ), and there are  $T = O(\log n)$  phases. ◀

► **Lemma 15.** *After multi-COBRA completes all  $T$  phases, each subgraph has a diameter of  $O(\log n)$ .*

**Proof.** The multi-COBRA runs for  $T = O(\log n)$  phases, and in each phase, we add to each subgraph only those nodes that are neighbors of the nodes already included. Consequently, the diameter of the subgraph increases by at most two per phase. ◀

In the rest of this section, we will analyze protocols that run on all the subgraphs in parallel. To achieve this parallelism despite potential congestion (recall that each edge may belong to multiple subgraphs), the execution is again organized into phases. Now, each phase spans  $O(\log n)$  rounds, ensuring that messages sent along any shared edge are distributed across the protocols without conflict. Specifically, messages that would be sent on an edge in the same round by independently executed protocols are all scheduled within a single phase across subgraphs. This phased execution allows all protocols to proceed in parallel while respecting the edge capacity. As a result, the combined round complexity of the protocols increases by at most a factor of  $O(\log n)$  compared to running an individual protocol.

We are now ready to prove Theorem 5.

► **Theorem 5.** *For an Erdős–Rényi graph  $G(n, p)$  with  $p \geq \frac{C_p \log n}{n}$ , there exists a distributed randomized algorithm that produces a tree packing of size  $\delta(G)$ , diameter  $O(\log n)$ , and weight  $O(\log n)$  w.h.p.*

**Proof of Theorem 5.** The algorithm goes as follows. First, let nodes share the information of  $n$ ,  $\delta$  and  $\Delta$ . This can be done in  $O(D)$  rounds by constructing a BFS tree. Next, every node  $v$  adds  $\Delta - \deg(v)$  self-loops. Then, parties run multi-COBRA for  $T$  rounds that by Lemma 13 result with probability at least  $1 - O(\frac{1}{n})$  in  $\delta$  spanning subgraphs  $\{S_i\}_{i \in [\delta]}$ . By Lemma 15, those have diameter  $O(\log n)$ . Moreover, by Lemma 14, every edge of the graph belongs to at most  $O(\log n)$  subgraphs.

Now, we launch BFSs on all subgraphs in parallel to turn them into spanning trees. As discussed earlier in this section, this can be done in  $O(\log n \cdot \max_{i \in [\delta]} D(S_i)) = O(\log^2 n)$  rounds. As a result of doing so, the weight of every edge could only have decreased, and the diameter of each subgraph at most doubled. ◀

Having a tree packing with the properties described, we can prove Theorem 4 by adding a final piece.

► **Theorem 4.** *For an Erdős–Rényi graph  $G(n, p)$  with  $p \geq \frac{C_p \log n}{n}$ , there exists a distributed randomized algorithm that completes the broadcast in  $O(\log^2 n + \log n \cdot \frac{k}{\delta(G)})$  rounds w.h.p.*

**Proof of Theorem 4.** First, build a tree packing from Theorem 5. Then,  $s$  evenly distributes messages among the obtained trees, so that each tree receives  $\frac{k}{\delta}$  messages. After that, in each tree, nodes downcast corresponding messages. The algorithm for downcasting messages  $\{m_1, \dots, m_k\}$  from the root of a single tree works as follows. In the first round, the root sends the first message ( $m_1$ ) to all its immediate children. In the second round, the children forward  $m_1$  to their respective children (the root’s grandchildren), while the root simultaneously sends the second message ( $m_2$ ) to its immediate children. This process continues iteratively: in each subsequent round, the root sends the next message ( $m_i$ ) to its children, and all other nodes forward the message they received in the previous round to their respective children. This way, for  $k'$  messages and a tree of diameter  $H$  it takes  $H + k' - 1$  rounds for every node to discover every message.

Multiplying by a congestion factor of  $O(\log n)$ , we get that the round complexity of broadcasting  $k$  messages in  $\delta(G)$  spanning trees of diameter  $O(\log n)$  is

$$O\left(\log^2 n + \log n \cdot \frac{k}{\delta(G)}\right). \quad \blacktriangleleft$$

► **Remark 16.** In [21, 8], authors consider a problem where initially messages are spread over the network, that is every node possesses a subset of  $M$ . This seems like a more general version, however, when using a tree packing approach, these two problems are equivalent. The intuition is, nodes can first agree on the distribution of messages among trees, then upcast the messages to the root in their corresponding trees, and finally perform a downcast as described in our paper, all that in  $\tilde{O}(D(G) + \frac{k}{\delta(G)})$ . For the full proof, a reader is invited to see the proof of Theorem 1 in [8].

#### 4.4 Introducing Regularity while Maintaining Expansion

In this Section, we prove Lemma 11 that states that graph’s expansion properties are preserved after adding self-loops. We start by providing relevant concepts from spectral theory.

► **Definition 17.** *Let  $A$  be an  $n \times n$  matrix with entries from  $\mathbb{R}_{\geq 0}$  and let  $D$  be a diagonal matrix such that  $D_{ii} = \sum_{j \in [n]} A_{ij}$ . Assuming  $D_{ii} > 0$  for all  $i \in [n]$ , let  $\bar{A}$  denote a normalized version of  $A$ , i.e.  $\bar{A} = D^{-1/2} A D^{-1/2}$ .*

► **Definition 18.** *Let  $A$  be an  $n \times n$  matrix. Define  $\lambda_2(A)$  to be the second largest (in absolute value) eigenvalue of  $A$ . Let  $G$  be an undirected multi-graph and  $A$  be its weighted adjacency matrix. Define  $\lambda_2(G)$  as  $\lambda_2(\bar{A})$ .*

► **Theorem 19** (Hoffman et al. [32]). *For a positive constant  $C$  and  $p \geq \frac{C \log n}{n}$ , consider an Erdős–Rényi graph  $G(n, p)$ . Then, with probability at least  $1 - O(\frac{1}{n^{C-1}})$ , we have  $\lambda_2(G) = O(\frac{1}{\sqrt{pn}})$ .*

The following Lemma shows that with high probability, an Erdős–Rényi graph  $G(n, p)$  is almost regular.

## 71:14 Broadcast in Almost Mixing Time

► **Lemma 20.** Let  $p \geq \frac{C_p \log n}{n-1}$ . Then for an Erdős–Rényi graph  $G(n, p)$ ,  $\frac{\Delta(G)}{\delta(G)} \leq 1 + \frac{1}{7}$  with probability at least  $1 - O(\frac{1}{n^2})$ .

**Proof.** Let us fix a vertex  $v$  and consider the number of its incident edges. For each potential edge  $e_i$ ,  $i \in [n-1]$  let us introduce an indicator variable  $\chi_i$  which is equal to 1 if the edge exists and to 0 if it does not. The number of edges  $v$  has is then  $\sum_{i \in [n-1]} \chi_i$ . The expectation of that is  $p(n-1)$ , and applying Chernoff bounds, we get the following bounds on the degree of  $v$

$$\Pr \left[ \deg(v) \geq \left(1 + \frac{1}{15}\right) C_p \log n \right] \leq \exp \left( -\frac{C_p \log n}{675} \right) \leq \frac{1}{2n^3},$$

and

$$\Pr \left[ \deg(v) \leq \left(1 - \frac{1}{15}\right) C_p \log n \right] \leq \exp \left( -\frac{C_p \log n}{675} \right) \leq \frac{1}{2n^3}.$$

Now, taking union bound over all vertices, we conclude that for every vertex  $v$  it holds that  $\left(1 - \frac{1}{15}\right) C_p \log n \leq \deg(v) \leq \left(1 + \frac{1}{15}\right) C_p \log n$  with probability at least  $1 - \frac{1}{n^2}$ . Thus, with probability  $1 - \frac{1}{n^2}$ , we have that  $\frac{\Delta(G)}{\delta(G)} \leq \frac{1 + \frac{1}{15}}{1 - \frac{1}{15}} = 1 + \frac{1}{7}$  ◀

The next ingredient is to show that the slight perturbation of the diagonal elements of the matrix induces only a little change in its eigenvalues.

► **Lemma 21.** Let  $A$  be an  $n \times n$  adjacency matrix of a connected graph and let  $D$  be a diagonal matrix such that  $D_{ii} = \sum_{j \in [n]} A_{ij}$ . Let  $E$  be a  $n \times n$  diagonal matrix such that  $0 \leq E_{ii} \leq \varepsilon$  for some  $0 < \varepsilon < 1$  and all  $i \in [n]$ . Then  $\lambda_2(\overline{A + DE}) \leq \lambda_2(\overline{A}) + 6\varepsilon$ .

**Proof sketch.** The idea of the proof is to express  $\overline{A + DE}$  as a sum of  $\overline{A}$  and matrices with the small spectral norms, and then apply a corollary of Weyl's Theorem [49], that is, for  $n \times n$  matrices  $M_1$  and  $M_2$ , it holds that

$$|\lambda_2(M_1 + M_2) - \lambda_2(M_1)| \leq \|M_2\|_2$$

The full proof goes as follows:

Let  $A' = A + DE$  and let  $D'$  be a diagonal matrix such that  $D'_{ii} = \sum_{j \in [n]} A'_{ij}$ . Note that  $D' = D + DE$  and hence  $(D')^{-1/2} = (D)^{-1/2}(I + E)^{-1/2}$ . Entries of the  $(I + E)^{-1/2}$  are of the form  $\frac{1}{\sqrt{1+E_{ii}}} \geq \frac{1}{\sqrt{1+\varepsilon}} \geq \sqrt{1-\varepsilon} \geq 1 - \varepsilon$ . Therefore, we can denote  $(I + E)^{-1/2}$  with  $I - E'$  where  $E'$  is a diagonal matrix with entries  $0 \leq E'_{ii} \leq \varepsilon$ . Now

$$\begin{aligned} \overline{A'} &= (D')^{-1/2} A' (D')^{-1/2} \\ &= (D^{-1/2} - D^{-1/2} E') (A + DE) (D^{-1/2} - D^{-1/2} E') \\ &= D^{-1/2} A D^{-1/2} - D^{-1/2} A D^{-1/2} E' + E - E E' - E' D^{-1/2} A D^{-1/2} + \\ &\quad E' D^{-1/2} A D^{-1/2} E' - E' E + E' E E' \\ &= \overline{A} - \overline{A E'} - E' \overline{A} + E' \overline{A E'} + E - 2E E' + E' E E' \end{aligned}$$

where to obtain \* we used the fact that diagonal matrices commute.

From Weyl's theorem, we conclude that

$$\begin{aligned} \lambda_2(\overline{A}') - \lambda_2(\overline{A}) &\leq \| -\overline{A}E' - E'\overline{A} + E'\overline{A}E' + E - 2EE' + E'EE' \|_2 \\ &\leq \|\overline{A}E'\|_2 + \|E'\overline{A}\|_2 + \|E'\overline{A}E'\|_2 + 2\|EE'\|_2 + \|E'EE'\|_2 \\ &\leq \|\overline{A}\|_2\|E'\|_2 + \|E'\|_2\|\overline{A}\|_2 + \|E'\|_2\|\overline{A}\|_2\|E'\|_2 + \\ &\quad 2\|E\|_2\|E'\|_2 + \|E'\|_2\|E\|_2\|E'\|_2 \end{aligned}$$

Now recall that the spectral norm for a real-valued symmetric matrix is the biggest absolute value of its eigenvalues, hence  $\|\overline{A}\|_2 = 1$  and  $\|E\|_2 \leq \varepsilon$ ,  $\|E'\|_2 \leq \varepsilon$ . Thus

$$\lambda_2(\overline{A}') - \lambda_2(\overline{A}) \leq \varepsilon + \varepsilon + \varepsilon^2 + 2\varepsilon^2 + \varepsilon^3 \leq 6\varepsilon \quad \blacktriangleleft$$

Finally, using Lemmas 20 and 21 alongside Theorem 19, we show that w.h.p., regularizing an Erdős–Rényi graph  $G(n, p)$  by adding self-loops for every node to reach  $\Delta(G)$  does not ruin its expansion properties.

► **Lemma 11.** *With probability at least  $1 - O(\frac{1}{n^2})$ , for  $p \geq \frac{C_p \log n}{n-1}$ , an Erdős–Rényi graph  $G(n, p)$  can be transformed into  $G'$  by adding weighted self-loops to the nodes so that (1)  $G'$  is regular, (2)  $1 - \lambda_2(G') \geq \frac{1}{14}$ .*

**Proof.** Let  $A$  be the adjacency matrix of  $G$ ,  $D$  be the degree matrix of  $G$  and  $E$  be the  $n \times n$  diagonal matrix with entries  $E_{ii} = \frac{\Delta(G)}{\deg(v_i)} - 1$ . By the Lemma 20, with probability  $1 - O(\frac{1}{n^2})$ ,  $G$  has  $\frac{\Delta}{\delta} \leq 1 + \frac{1}{7}$  and therefore,  $E_{ii} \leq \frac{1}{7}$  for all  $i \in [n]$  with probability  $1 - O(\frac{1}{n^2})$ .

Now, to each vertex  $v$  in  $G$ , add  $\Delta - \deg(v)$  self-loops to obtain a graph  $G'$ . Clearly,  $G'$  is  $\Delta(G)$ -regular. The adjacency matrix of  $G'$  will then be  $A + DE$  and hence, applying Lemma 21 we deduce that  $\lambda_2(G') \leq \lambda_2(G) + \frac{\delta}{9}$ .

By the Theorem 19, we know that with probability  $1 - O(\frac{1}{n^2})$ ,  $\lambda_2(G) \leq \frac{C}{\sqrt{p(n-1)}}$  for some constant  $C$ , which for large enough  $n$  is less than  $\frac{1}{14}$ , thus  $\lambda_2(G') \leq \frac{13}{14}$ . ◀

## 5 Sketching the terrain

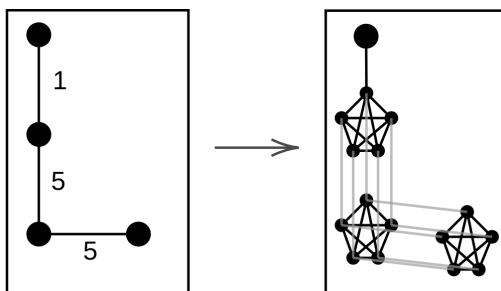
In this section, we share insights on the multi-message broadcast problem and why it is difficult to solve optimally. First, in Section 5.1, we sketch the proof of its NP-hardness in the centralized setting. Then, in Section 5.2, we argue that one can not hope to design an algorithm for a general graph that completes in  $\tilde{O}(D(G) + \frac{k}{\delta(G)})$  rounds, which implies a need for some fundamentally new techniques.

### 5.1 NP-Hardness

We prove that determining the optimal number of rounds for the multi-message broadcast problem in CONGEST is NP-hard in the centralized setting. To do that, we reduce the Set splitting problem to the multi-message broadcast.

► **Definition 22 (Set splitting problem).** *Given a family  $F$  of subsets of a finite set  $S$ , decide whether there exists a partition of  $S$  into two subsets  $S_1, S_2$  such that all elements of  $F$  are split by this partition, i.e., none of the elements of  $F$  is completely in  $S_1$  or  $S_2$ .*

In our reduction, for simplicity of presentation, we allow edges to have arbitrary bandwidth instead of  $\tilde{O}(1)$ , since, as we show, this can be simulated in CONGEST. In the reduction, the initial set  $S$  corresponds to the set  $M$  of messages, and  $s$  has two dedicated children to



■ **Figure 1** An example of mapping a graph with arbitrary bandwidths to a graph suitable for CONGEST.

which it can send  $n_1$  and  $n_2$  messages, respectively with  $n_1 + n_2 = k$ , simulating the splitting. Deciding how to split messages between these two children is the only “smart” choice an algorithm should make; all other nodes are just forwarding messages they receive. For the full version of the proof, please see the full version of the paper [42].

## 5.2 Straightforward Lower Bounds are not Enough

It is tempting to argue for an approximation factor of an algorithm by comparing its round complexity to two straightforward lower bounds:  $D(G)$  and  $\frac{k}{\minCut(G)}$ . Unfortunately, those are not sufficient as there is an instance (see Figure 2) where  $D(G) = O(1)$  and  $\frac{k}{\minCut(G)} = O(1)$ , but the optimal answer is  $\Omega(\sqrt{k})$ . As for NP-hardness, we consider a more general model where edges might have arbitrary bandwidth, but we show that this can be simulated in CONGEST.

To transform a graph with arbitrary bandwidths to a graph with all bandwidths equal to 1, we do the following. The source  $s$  corresponds to a single node in the new graph. For a node  $v \neq s$  in the original graph, let  $B$  denote the maximal bandwidth of its adjacent edges. In the new graph, node  $v$  then corresponds to a clique of  $B$  nodes. We call this clique a  $v$ -clique. If in the original graph nodes  $v \neq s$  and  $u \neq s$  were connected by an edge of bandwidth  $b$ , we pick (arbitrary)  $b$  nodes in  $v$ -clique,  $b$  nodes in  $u$ -clique, and draw  $b$  edges between picked nodes to establish a perfect matching. For every edge  $(s, u)$  of bandwidth  $b$ , we connect the new source with  $b$  arbitrary nodes of the  $u$ -clique. We call the resulting graph *the corresponding CONGEST graph*.

▷ **Claim 23.** Consider the original graph  $G$  and its corresponding CONGEST graph  $G'$ . Then

$$D(G) \leq D(G') \leq 2D(G) + 1.$$

*Proof idea.* The first inequality is straightforward. We prove the second inequality by induction on the length of the path, that is if there is a path in  $G$  from  $u$  to  $v$  of length  $l$ , then for any nodes  $u'$  and  $v'$  in  $u$ -clique and  $v$ -clique respectively, there is a path between  $u'$  and  $v'$  in  $G'$  of length  $2l + 1$ . ◁

▷ **Claim 24.** Consider the original graph  $G$  and its corresponding CONGEST graph  $G'$ . Then

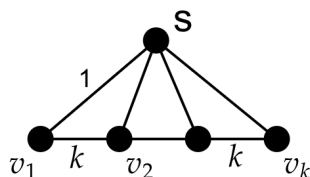
$$\min\{\minCut(G), \min_{v \in V(G) \setminus \{s\}} \text{size of the } v\text{-clique} - 1\} \leq \minCut(G') \leq \minCut(G).$$

Proof. The second inequality is straightforward. For the first inequality, note that each cut of  $G'$  either cuts some clique or does not. In case it does not, it corresponds to a cut in  $G$  and has the same size. In case it does, it is at least the size of the induced cut for that clique, which is at least  $\min_{v \in V(G) \setminus \{s\}} \text{size of the } v\text{-clique} - 1$ .  $\triangleleft$

$\triangleright$  **Claim 25.** Consider the original graph  $G$  and its corresponding CONGEST graph  $G'$ . Together with a set  $M$  of messages, they define a multi-message broadcast problem in generalized CONGEST and CONGEST, respectively. Let  $\text{OPT}(G)$  and  $\text{OPT}(G')$  denote the optimal round complexities for  $G$  and  $G'$  respectively. Then  $\text{OPT}(G) \leq \text{OPT}(G')$ .

Proof. Consider an execution  $E'$  for  $G'$  which achieves  $\text{OPT}$ . We claim that we can build an execution  $E$  for  $G$ , such that for every round  $r$  of  $E'$  and for every  $v \in V(G)$ , after round  $r$  in  $E$ ,  $v$  knows all the messages that the nodes of  $v$ -clique know after round  $r$  in  $E'$ . To do so, consider a round  $r$  and some  $v \in V(G)$ . Let us say that nodes in  $v$ -clique in  $E'$  in round  $r$  receive messages  $M_1$  from the  $u_1$ -clique, messages  $M_2$  from the  $u_2$ -clique, and so forth for all neighboring cliques. Then, in  $E$   $u_i$  sends  $M_i$  to  $v$  satisfying the invariant. Note that  $u_i$  can do this in terms of the bandwidth by construction of the corresponding CONGEST graph, and in terms of knowing  $M_i$  by the invariant.  $\triangleleft$

We now give an example of an instance of a problem where  $D(G) = O(1)$  as well as  $\frac{k}{\text{minCut}(G)} = O(1)$ , but the optimal round complexity is  $\Omega(\sqrt{k})$ . The graph we consider is the corresponding CONGEST graph  $G'$  to the graph  $G$  depicted in Figure 2.



**Figure 2** An example graph  $G$  where diameter and minimum cut are not telling. Here, edge labels denote bandwidth.

First, note that  $D(G) = 2$ , hence by Claim 23,  $D(G') = O(1)$ . Second, notice that  $\text{minCut}(G) = k$  and the minimal maximal bandwidth of an edge adjacent to some node in  $V(G) \setminus \{s\}$  is equal to  $k$ , therefore, by Claim 24,  $k - 1 \leq \text{minCut}(G') \leq k$ . Finally, by Claim 25,  $\text{OPT}(G') \geq \text{OPT}(G)$ , where  $\text{OPT}$  is the optimal round complexity. Therefore, it is sufficient to show that  $\text{OPT}(G) = \Omega(\sqrt{k})$ .

We claim that  $\Omega(\sqrt{k})$  rounds are needed for  $v_1$  only to get to know  $M$  (become saturated). For the sake of contradiction, assume that we can saturate  $v_1$  in  $\leq \sqrt{k} - 1$  rounds. That means that it can be saturated without using the edges  $(s, v_{\sqrt{k}+1})$  and  $(v_{\sqrt{k}+1}, v_{\sqrt{k}+2})$ . But if we remove those edges,  $\text{minCut}(s, v_1) \leq \sqrt{k}$ , implying that the number of rounds needed is at least  $\frac{k}{\sqrt{k}} = \sqrt{k}$ , a contradiction.

## 6 Conclusion

In this work, we explored the multi-message broadcast problem within the CONGEST model. We presented an algorithm that achieves universal optimality, up to polylogarithmic factors, for networks modeled as random graphs and extended our results through a lifting technique to general expander graphs, paying an additional factor of mixing time.

Our study introduces several promising avenues for future investigation. One intriguing direction involves developing distributed processes capable of rapidly covering graphs that also maintain composability, that is, efficiently supporting multiple simultaneous executions under congestion constraints. Another important direction is identifying graph properties beyond traditional metrics such as diameter and minimum cut, which would facilitate establishing tight lower bounds. Subsequently, algorithms matching these lower bounds could be devised, closing the open problem of multi-message broadcast.

---

## References

- 1 Rudolf Ahlswede, Ning Cai, S-YR Li, and Raymond W Yeung. Network information flow. *IEEE Transactions on information theory*, 46(4):1204–1216, 2000. doi:10.1109/18.850663.
- 2 Noga Alon, Chen Avin, Michal Koucky, Gady Kozma, Zvi Lotker, and Mark R Tuttle. Many random walks are faster than one. In *Proceedings of the twentieth annual symposium on parallelism in algorithms and architectures*, pages 119–128, 2008. doi:10.1145/1378533.1378557.
- 3 Vijeth Aradhya, Seth Gilbert, and Thorsten Götte. Distributed Branching Random Walks and Their Applications. In Silvia Bonomi, Letterio Galletta, Etienne Rivière, and Valerio Schiavoni, editors, *28th International Conference on Principles of Distributed Systems (OPODIS 2024)*, volume 324 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:20, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.OPODIS.2024.36.
- 4 Tuğkan Batu, Amitabh Trehan, and Chhaya Trehan. All you need are random walks: Fast and simple distributed conductance testing. In *International Colloquium on Structural Information and Communication Complexity*, pages 64–82. Springer, 2024.
- 5 Petra Berenbrink, George Giakkoupis, and Peter Kling. Tight bounds for coalescing-branching random walks on regular graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1715–1733. SIAM, 2018. doi:10.1137/1.9781611975031.112.
- 6 Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed lovász local lemma. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 479–488, 2016. doi:10.1145/2897518.2897570.
- 7 Keren Censor-Hillel, Mohsen Ghaffari, and Fabian Kuhn. Distributed connectivity decomposition. In *Proceedings of the 2014 ACM symposium on Principles of distributed computing*, pages 156–165, 2014. doi:10.1145/2611462.2611491.
- 8 Shashwat Chandra, Yi-Jun Chang, Michal Dory, Mohsen Ghaffari, and Dean Leitersdorf. Fast broadcast in highly connected networks. In *Proceedings of the 36th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '24, pages 331–343, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3626183.3659959.
- 9 Yi-Jun Chang and Thatchaphol Saranurak. Deterministic distributed expander decomposition and routing with applications in distributed derandomization. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 377–388. IEEE, 2020. doi:10.1109/FOCS46700.2020.00043.
- 10 Fan Chung and Linyuan Lu. The diameter of sparse random graphs. *Advances in Applied Mathematics*, 26(4):257–279, 2001. doi:10.1006/AAMA.2001.0720.
- 11 Julia Chuzhoy, Merav Parter, and Zihan Tan. On packing low-diameter spanning trees. *arXiv preprint arXiv:2006.07486*, 2020. arXiv:2006.07486.
- 12 Colin Cooper, Tomasz Radzik, and Nicolas Rivera. The coalescing-branching random walk on expanders and the dual epidemic process. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pages 461–467, 2016. doi:10.1145/2933057.2933119.

- 13 Chinmoy Dutta, Gopal Pandurangan, Rajmohan Rajaraman, and Scott Roche. Coalescing-branching random walks on graphs. *ACM Transactions on Parallel Computing (TOPC)*, 2(3):1–29, 2015. doi:10.1145/2817830.
- 14 J. Edmonds. Edge-disjoint branchings. In R. Rustin, editor, *Combinatorial Algorithms*, pages 91–96. Algorithmics Press, New York, New York, 1972.
- 15 Michael Elkin and Shay Solomon. Distributed approximate maximum matching. *ACM Transactions on Algorithms (TALG)*, 13(1):1–27, 2017.
- 16 Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.*, 5:17–61, 1960.
- 17 Hendrik Fichtenberger and Yadu Vasudev. A two-sided error distributed property tester for conductance. In *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.19.
- 18 Christina Fragouli and Emina Soljanin. Decentralized network coding. In *Information Theory Workshop*, pages 310–314. IEEE, 2004. doi:10.1109/ITW.2004.1405320.
- 19 Pu Gao, Xavier Pérez-Giménez, and Cristiane M Sato. Arboricity and spanning-tree packing in random graphs with an application to load balancing. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 317–326. SIAM, 2014. doi:10.1137/1.9781611973402.23.
- 20 Juan A Garay, Shay Kutten, and David Peleg. A sublinear time distributed algorithm for minimum-weight spanning trees. *SIAM Journal on Computing*, 27(1):302–316, 1998. doi:10.1137/S0097539794261118.
- 21 Mohsen Ghaffari. Distributed broadcast revisited: Towards universal optimality. In *International Colloquium on Automata, Languages, and Programming*, pages 638–649. Springer, 2015. doi:10.1007/978-3-662-47666-6\_51.
- 22 Mohsen Ghaffari. Distributed mis via all-to-all communication. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC)*, pages 113–122, 2016.
- 23 Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks ii: low-congestion shortcuts, mst, and min-cut. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 202–219. SIAM, 2016. doi:10.1137/1.9781611974331.CH16.
- 24 Mohsen Ghaffari and Fabian Kuhn. Distributed minimum cut approximation. In *International Symposium on Distributed Computing*, pages 1–15. Springer, 2013. doi:10.1007/978-3-642-41527-2\_1.
- 25 Mohsen Ghaffari, Fabian Kuhn, and Hsin-Hao Su. Distributed mst and routing in almost mixing time. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 131–140, 2017. doi:10.1145/3087801.3087827.
- 26 Mohsen Ghaffari and Jason Li. New distributed algorithms in almost mixing time via transformations from parallel algorithms. *arXiv preprint arXiv:1805.04764*, 2018. arXiv:1805.04764.
- 27 Seth Gilbert, Peter Robinson, and Suman Sourav. Leader election in well-connected graphs. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 227–236, 2018. doi:10.1145/3212734.3212754.
- 28 Rachid Guerraoui, Anne-Marie Kermarrec, Anastasiia Kucherenko, Rafael Pinot, and Sasha Voitovych. On the inherent anonymity of gossiping. *arXiv preprint arXiv:2308.02477*, 2023. doi:10.48550/arXiv.2308.02477.
- 29 Bernhard Haeupler, Harald Räcke, and Mohsen Ghaffari. Hop-constrained expander decompositions, oblivious routing, and distributed universal optimality. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1325–1338, 2022. doi:10.1145/3519935.3520026.

- 30 Tracey Ho, Sidharth Jaggi, Svitlana Vyetrenko, and Lingxiao Xia. Universal and robust distributed network codes. In *2011 Proceedings IEEE INFOCOM*, pages 766–774. IEEE, 2011. doi:10.1109/INFOCOM.2011.5935297.
- 31 Tracey Ho, Ralf Koetter, Muriel Medard, David R Karger, and Michelle Effros. The benefits of coding over routing in a randomized setting. In *IEEE international symposium on information theory*, pages 442–442, 2003.
- 32 Christopher Hoffman, Matthew Kahle, and Elliot Paquette. Spectral gaps of random graphs and applications. *International Mathematics Research Notices*, 2021(11):8353–8404, 2021.
- 33 Sidharth Jaggi, Peter Sanders, Philip A Chou, Michelle Effros, Sebastian Egner, Kamal Jain, and Ludo MGM Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory*, 51(6):1973–1982, 2005. doi:10.1109/TIT.2005.847712.
- 34 David Kempe and Frank McSherry. A decentralized algorithm for spectral analysis. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 561–568, 2004. doi:10.1145/1007352.1007438.
- 35 Shay Kutten, Gopal Pandurangan, David Peleg, Peter Robinson, and Amitabh Trehan. Sublinear bounds for randomized leader election. *Theoretical Computer Science*, 561:134–143, 2015. doi:10.1016/J.TCS.2014.02.009.
- 36 Gregory F Lawler and Vlada Limic. *Random walk: a modern introduction*, volume 123. Cambridge University Press, 2010.
- 37 S-YR Li, Raymond W Yeung, and Ning Cai. Linear network coding. *IEEE transactions on information theory*, 49(2):371–381, 2003. doi:10.1109/TIT.2002.807285.
- 38 Michael Mitzenmacher, Rajmohan Rajaraman, and Scott Roche. Better bounds for coalescing-branching random walks. *ACM Transactions on Parallel Computing (TOPC)*, 5(1):1–23, 2018. doi:10.1145/3209688.
- 39 Anisur Rahaman Molla and Gopal Pandurangan. Distributed computation of mixing time. In *Proceedings of the 18th International Conference on Distributed Computing and Networking*, pages 1–4, 2017.
- 40 C St JA Nash-Williams. Edge-disjoint spanning trees of finite graphs. *Journal of the London Mathematical Society*, 1(1):445–450, 1961.
- 41 Edgar M Palmer. On the spanning tree packing number of a graph: a survey. *Discrete Mathematics*, 230(1-3):13–21, 2001. doi:10.1016/S0012-365X(00)00066-2.
- 42 Anton Paramonov and Roger Wattenhofer. Broadcast in almost mixing time. *CoRR*, abs/2502.02165, 2025. arXiv:2502.02165.
- 43 David Peleg. *Distributed computing: a locality-sensitive approach*. SIAM, 2000.
- 44 Nicolás Rivera, John Sylvester, Luca Zanetti, and Thomas Sauerwald. Lecture 10: Random walks on graphs. <https://www.cl.cam.ac.uk/teaching/1920/Probability/materials/Lecture10.pdf>, 2020. Lecture notes, University of Cambridge.
- 45 Václav Rozhoň and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 350–363, 2020.
- 46 Zhan Shi et al. *Branching random walks*, volume 2151. Springer, 2015.
- 47 Vasuki Narasimha Swamy, Srikrishna Bhashyam, Rajesh Sundaresan, and Pramod Viswanath. An asymptotically optimal push–pull method for multicasting over a random network. *IEEE transactions on information theory*, 59(8):5075–5087, 2013. doi:10.1109/TIT.2013.2253543.
- 48 William Thomas Tutte. On the problem of decomposing a graph into n connected factors. *Journal of the London Mathematical Society*, 1(1):221–230, 1961.
- 49 Wikipedia contributors. Weyl’s inequality. [https://en.wikipedia.org/wiki/Weyl%27s\\_inequality](https://en.wikipedia.org/wiki/Weyl%27s_inequality), 2025. [Accessed: 17-May-2025].
- 50 Yunnan Wu, Philip A Chou, and Kamal Jain. A comparison of network coding and tree packing. In *International Symposium on Information Theory, 2004. ISIT 2004. Proceedings.*, page 143. IEEE, 2004. doi:10.1109/ISIT.2004.1365182.