The Capacity of Smartphone Peer-To-Peer Networks

Michael Dinitz
Johns Hopkins University, Baltimore, MD, United States
mdinitz@cs.jhu.edu

Magnús M. Halldórsson
Reykjavík University, Iceland
mmh@ru.is

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

Alex Weaver
Georgetown University, Washington, DC, United States
aweaver@cs.georgetown.edu

Abstract

We study three capacity problems in the mobile telephone model, a network abstraction that models the peer-to-peer communication capabilities implemented in most commodity smartphone operating systems. The *capacity* of a network expresses how much sustained throughput can be maintained for a set of communication demands, and is therefore a fundamental bound on the usefulness of a network. Because of this importance, wireless network capacity has been active area of research for the last two decades.

The three capacity problems that we study differ in the structure of the communication demands. The first problem is pairwise capacity, where the demands are (source, destination) pairs. Pairwise capacity is one of the most classical definitions, as it was analyzed in the seminal paper of Gupta and Kumar on wireless network capacity. The second problem we study is broadcast capacity, in which a single source must deliver packets to all other nodes in the network. Finally, we turn our attention to all-to-all capacity, in which all nodes must deliver packets to all other nodes. In all three of these problems we characterize the optimal achievable throughput for any given network, and design algorithms which asymptotically match this performance. We also study these problems in networks generated randomly by a process introduced by Gupta and Kumar, and fully characterize their achievable throughput.

Interestingly, the techniques that we develop for all-to-all capacity also allow us to design a one-shot gossip algorithm that runs within a polylogarithmic factor of optimal in every graph. This largely resolves an open question from previous work on the one-shot gossip problem in this model.

2012 ACM Subject Classification Computing methodologies → Distributed algorithms; Networks → Network algorithms

Keywords and phrases Capacity, Wireless, Mobile Telephone, Throughput

Digital Object Identifier 10.4230/LIPIcs.DISC.2019.14

Related Version A full version of this paper is available at `https://arxiv.org/abs/1908.01894`.

Funding *Michael Dinitz*: Supported in part by NSF award CCF-1535887.
Magnús M. Halldórsson: Supported in part by Icelandic Research Fund grant 174484.
Calvin Newport: Supported in part by NSF award CCF-1733842.
Alex Weaver: Supported in part by NSF award CCF-1733842.

1 Introduction

In this paper, we study the classical capacity problem in the mobile telephone model: an abstraction that models the peer-to-peer communication capabilities implemented in most commodity smartphone operating systems. The capacity of a network expresses how much sustained throughput can be maintained for a set of communication demands. We focus on three variations of the problem: *pairwise* capacity, in which nodes are divided into pairwise packet flows, *broadcast* capacity, in which a single source delivers packets to the whole network, and *all-to-all* capacity, in which all nodes deliver packets to the whole network.

For each variation we prove limits on the achievable throughput and analyze algorithms that match (or nearly match) these bounds. We study these results in both *arbitrary* networks and *random* networks generated with the process introduced by Gupta and Kumar in their seminal paper on wireless network capacity [19]. Finally, we deploy our new techniques to largely resolve an open question from [24] regarding optimal one-shot gossip in the mobile telephone model. Below we summarize the problems we study and the results we prove, interleaving the relevant related work.

The Mobile Telephone Model. The *mobile telephone model* (MTM), introduced by Ghaffari and Newport [14], modifies the well-studied *telephone model* of wired peer-to-peer networks (e.g., [10, 15, 4, 17, 9, 16]) to better capture the dynamics of standard smartphone peer-to-peer libraries. It is inspired, in particular, by the specific interfaces provided by Apple's Multipeer Connectivity Framework [2].

In this model, the network is modeled as an undirected graph $G = (V, E)$, where the nodes in V correspond to smartphones, and an edge $\{u, v\} \in E$ indicates the devices corresponding to u and v are close enough to enable a direct peer-to-peer radio link. Time proceeds in synchronous rounds. As in the original telephone model, in each round, each node can either attempt to initiate a connection (e.g., place a telephone call) with at most one of its neighbors, or wait to receive connection attempts. Unlike the original model, however, a waiting node can accept at most one incoming connection attempt. This difference is consequential, as many of the celebrated results of the original telephone model depend on the nodes' ability to accept an unbounded number of incoming connections (see [14, 6] for more discussion).[1] This restriction is motivated by the reality that standard smartphone peer-to-peer libraries limit the number of concurrent connections per device to a small constant (e.g., for Multipeer this limit is 8). Once connected, a pair of nodes can participate in a bounded amount of reliable communication (e.g., transfer a constant number of packets/rumors/tokens).

Finally, the mobile telephone model also allows each node to broadcast a small $O(\log n)$-bit advertisement to its neighbors at the start of each round before the connection decisions are made. Most existing smartphone peer-to-peer libraries implement this *scan-and-connect* architecture. Notice, the mobile telephone model is harder than the original telephone model due to its connection restrictions, but also easier due to the presence of advertisements. The results is that the two settings are formally incomparable: each requires its own strategies for solving key problems.

[1] This behavior is particularly evident in studying PUSH-PULL rumor spreading in the telephone model in a star network topology. This simple strategy performs well in this network due to the ability of the points of the star to simultaneously pull the rumor from the center. In the mobile telephone model, by contrast, any rumor spreading strategy would be fundamentally slower due to the necessity of the center to connect to the points one by one.

In recent years, several standard one-shot peer-to-peer problems have been studied in the MTM, including rumor spreading [14], load balancing [7], leader election [24], and gossip [24, 25]. This paper is the first to study ongoing communication in this setting.

The Capacity Problem. Capacity problems are parameterized with a network topology $G = (V, E)$, and a flow set F made up of pairs of the form (s, R) (each of which is a *flow*), where $s \in V$ indicates a source (sometimes called a sender), and $R \subset V$ indicates a set of destinations (receivers). For each flow $(s, R) \in F$, source s is tasked with routing an infinite sequence of packets to destinations in R. The throughput achieved by a given destination for a particular flow is the average number of packets it receives from that flow per round in the limit, and the overall throughput is the smallest throughput over all the destinations in all flows (see Section 2.2 for formal definitions). We study three different capacity problems, each defined by the different constraints they place on the flow set F.

Results: Pairwise Capacity. The pairwise capacity problem divides nodes into source and destination pairs in F, i.e., the given flows are between pairs of nodes rather than from a source to a general destination set. We begin with pairwise capacity as it was the primary focus of Gupta and Kumar's seminal paper on the capacity of the protocol and physical wireless network models [19]. They argued that it provides a useful assessment of a network's ability to handle concurrent communication.

We begin in Section 3.1 by tackling the following fundamental problem: given an arbitrary connected network topology graph $G = (V, E)$ and a flow set F that divides the nodes in V into sender and receiver pairs, is it possible to efficiently calculate a packet routing schedule that approximates the optimal achievable throughput? We answer this question in the affirmative by establishing a novel connection between pairwise capacity and the classical concurrent multi-commodity flow (MCF) problem. To do so, we first transform a given G and F into an instance of the MCF problem. We then apply an existing MCF approximation algorithm to generate a fractional flow that achieves a good approximation of the optimal flow in the network. Finally, we apply a novel rounding procedure to transform the fractional flow into a schedule. We prove that this resulting schedule provides a constant approximation of the optimal achievable throughput.

Inspired by Gupta and Kumar [19], in Section 3.2 we turn our attention to networks and flow pairings that are randomly generated using the process introduced in [19]. This process is parameterized with a network size $n \geq 2$ and communication radius $r > 0$. It randomly places the n nodes in a unit square and adds an edge between any pair of nodes within distance r. The source and destination pairs are also randomly generated.

For every given size n, we identify a *connectivity threshold* value $r_c(n) = \Theta(\sqrt{\log n/n})$, such that for any radius $r \leq r_c(n)$, with constant probability the network generated by the above process for n and r includes a source with no path to its destination – trivializing the optimal achievable throughput to 0. We then prove that for every radius r that is at least a sufficiently large constant factor larger than the threshold, there is a tight bound of $\Theta(r)$ on the optimal achievable throughput. These results fully characterize our algorithm from Section 3.1 in randomly generated networks.

Results: Broadcast Capacity. Broadcast capacity is another natural communication problem in which a single *source* node is provided an infinite sequence of packets to deliver to all other nodes in the network. Solutions to this problem would be useful, for example, in a scenario where a large file is being distributed in a peer-to-peer network of smartphone users in a setting without infrastructure. In Section 4.1 we study the optimal achievable throughput for this problem in arbitrary connected graphs. To do so, we connect the scheduling of broadcast packets to existing results on *graph toughness*, a metric that captures a graph's resilience to disconnection that was introduced by Chvátal [5] in the context of studying Hamiltonian paths.

In more detail, a graph G has a k-tree if there exists a spanning tree of G with maximum degree k. Let $d(G)$ be the smallest k such that G has a k-tree. This tree is also called a minimum degree spanning tree (MDST) of G. Building on a result of Win [29] that relates k-trees to toughness, we prove that for any given G with $d(G) > 3$, there exists a subset S of nodes such that removing S from G partitions the graph into at least $(d(G) - 2)|S|$ connected components.

As we formalize in Section 4.1, because each node in S can connect to at most one component per round (due to the connection restrictions of the mobile telephone model), $\Omega(d(G))$ rounds are required to spread each packet to all components, implying that no schedule achieves throughput better than $O(1/d(G))$.

In Section 4.2, we prove this bound tight by exhibiting a matching algorithm. The algorithm begins by constructing a k-tree T with $k \in \Theta(d(G))$ using existing techniques; e.g., [11, 8]. It then edge colors T and uses the colors as the foundation for a TDMA schedule of length $\Theta(k)$ that allows nodes to simulate the more powerful CONGEST model in which each node can connect with every neighbor in a round. In the CONGEST model, a basic pipelined broadcast provides constant throughput. When combined with the simulation cost the achieved throughput is an asymptotically optimal $\Omega(1/d(G))$.

It is straightforward for a centralized algorithm to calculate this schedule in polynomial time, but in some cases a pre-computation of this type might be impractical, or require too high of a setup cost.[2] With this in mind, we also provide a distributed version of this algorithm that converges to $\Omega(1/(d(G) + \log n))$ throughput in $\tilde{O}(D(T)d(G) + \sqrt{n})$ rounds, where $D(T)$ is the diameter of the spanning tree and $\tilde{O}$ hides polylog(n) factors. The algorithm further converges to an optimal $\Omega(1/d(G))$ throughput after no more than $O(n^2)$ total rounds – providing a trade-off between setup cost and eventual optimality.

Finally, in Section 4.3, we study the performance of our algorithm in networks generated randomly using the Gupta and Kumar process summarized above. We prove that for any communication radius sufficiently larger than the connectivity threshold, the network is likely to include an $O(1)$-tree, enabling our algorithms to converge to constant throughput. This result indicates that in evenly distributed network deployments the mobile telephone model is well-suited for high performance broadcast.

Results: All-to-All Capacity. All-to-all capacity generalizes broadcast capacity such that now *every* node is provided an infinite sequence of packets it must deliver to the entire network. Solutions to this problem would be useful, for example, in a local multiplayer gaming scenario in which each player needs to keep track of the evolving status of all other players connected in a peer-to-peer network.

Clearly, n separate instances of our broadcast algorithm from Section 4.2, one for each of the n nodes as the broadcast source, can be interleaved with a round robin schedule to produce $\Omega(1/(n \cdot d(G)))$ throughput. In Section 5, we draw on the same graph theory

[2] In the mobile telephone model, all nodes can learn the entire network topology in $O(n^2)$ rounds and then run a centralized algorithm locally to determine their routing behavior. Though this setup cost is averaged out when calculating throughput in the limit, it might be desirable to minimize it in practice.

connections as before to prove that this result is tight for all-to-all capacity. We then provide a less heavy-handed distributed algorithm for achieving this throughput. Instead of interleaving n different broadcast instances, it executes distinct instances of all-to-all gossip, one for each packet number, using a flood-based strategy on a low degree spanning tree. Finally, we apply the random graph analysis from Section 4.3 to establish that for sufficiently large communication radius, with high probability, the randomly generated graph supports $\Omega(1/n)$-throughput, which is trivially optimal in the sense that a receiver can receive at most one new packet per round in our model.

New Results on One-Shot Gossip. As we detail in Section 5.1, our results on all-to-all capacity imply new lower and upper bounds on one-shot gossip in the mobile telephone model. From the lower bound perspective, they imply that gossiping in graph G in the mobile telephone model requires $\Omega(n \cdot d(G))$ rounds. From the upper bound perspective, when we carefully account for the costs of our routing algorithm applied to spreading only a single packet from each source, we solve the one-shot problem with high probability in the following number of rounds:

$$O((D + \sqrt{n})\text{polylog}(n) + n(d(G) + \log n)) = \tilde{O}(d(G) \cdot n),$$

where D is the diameter of G. This algorithm is asymptotically optimal in any graph with $d(G) \in \Omega(\log n)$ and $D \in O(n/\log^x n)$ (where x is the constant from the polylog in the MDST construction time), which describes a large family of graphs. For all other graphs the solution is at most a polylog factor slower than optimal. This is the first known gossip solution to be optimal, or within log factors of optimal, in *all* graphs, largely answering a challenge presented by [24].

Motivation. Smartphone operating systems include increasingly robust support for opportunistic device-to-device communication through standards such as Apple's Multipeer Connectivity Framework [2], Bluetooth LE [18], and WiFi Direct [3]. Though the original motivation for these links was to support information transfer among a small number of nearby phones, researchers are beginning to explore their potential to enable large-scale peer-to-peer networks. Recent work, for example, uses smartphone peer-to-peer networking to provide disaster response [28, 26, 21], circumvent censorship [12], extend internet access [1, 13], support local multiplayer gaming [23] and improve classroom interaction [20].

It remains largely an open question whether or not it will be possible to build large-scale network systems on top of smartphone peer-to-peer links. As originally argued by Gupta and Kumar [19], bounds for capacity problems can help resolve such questions for a given network model by establishing the limit to their ability to handle ongoing and concurrent communication. The results in this paper, as well as the novel technical tools developed to prove them, can therefore help resolve this critical question concerning this important emerging network setting.

2 Preliminaries

Here we define our model, the problem we study, and some useful mathematical tools and definitions. Due to space constraints, this version of the paper omits most proofs. All technical details can be found in the full version.

2.1 Model

The mobile telephone model describes a smartphone peer-to-peer network topology as an undirected graph $G = (V, E)$. The nodes in V correspond to the smartphone devices, and an edge $\{u, v\} \in E$ implies that the devices corresponding to u and v are within range to establish a direct peer-to-peer radio link. We use $n = |V|$ to indicate the network size.

Executions proceed in synchronous rounds labeled $1, 2, ...$, and we assume all nodes start during round 1. At the beginning of each round, each node $u \in V$ selects an *advertisement* of size at most $O(\log n)$ bits to broadcast to its neighbors $N(u)$ in G. After the advertisement broadcasts, each node u can either send a connection invitation to at most one neighbor, or wait to receive invitations. A node receiving invitations can accept at most one, forming a reliable pairwise connection. It follows from these constraints that the set of connections in a given round forms a matching.

Once connected, a pair of nodes can perform a bounded amount of reliable communication. For the capacity problems studied in this paper, we assume that a pair of connected nodes can transfer at most one packet over the connection in a given round. We treat these packets as black boxes that can only be delivered in this manner (e.g., you cannot break a packet into pieces, or attempt to deliver it using advertisement bits).

We assume when running a distributed algorithm in this model that each computational process (also called a *node*) is provided a unique ID that can fit into its advertisement and an estimate of the network size. It is provided no other *a priori* information about the network topology, though any such node can easily learn its local neighborhood in a single round if all nodes advertise their ID.

2.2 Problem

In this paper we measure capacity as the achievable throughput for various combinations of packet flow and network types. We begin by providing a general definition of *throughput* that applies to all settings we study. This definition makes use of an object we call a *flow set*, which is a set $F = \{(s_i, R_i) : 1 \leq i \leq k\}$ (for some $k \geq 1$) where each $s_i \in V$ and $R_i \subseteq V$ (for node set V). For a given flow set F, each $(s_i, R_i) \in F$ describes a packet flow of *type* i; i.e., source s_i is tasked with sending packets to all the destinations in set R_i. We refer to the packets from s_i as *i-packets.*

A *schedule* for a given G and F describes a movement of packets through the flows defined by F. Formally, a schedule is an infinite sequence of directed matchings, $M_1, M_2, ...$ on G, such that the edges in each M_t are labelled by packets, where we define a packet as a pair (i, j) with $i \in [|F|]$ and $j \in \mathbb{N}$ (i.e., (i, j) is the j'th packet of type i). We require that the packet labels for a schedule satisfy the property that if edge (u, v) in M_t is labelled with packet $p = (i, j)$, then there is a path in $\bigcup_{l<t} M_l$ from s_i to u where all edges on the path are labelled with p. (It is easy to see by induction that this corresponds precisely to the intuitive notion of packets moving through a mobile telephone network). We say that a packet p is *received* by a node u in round r if there is an edge $(v, u) \in M_r$ which is labelled p. A packet (i, j) is *delivered* by round r if every $x \in R_i$ receives it in some round t with $t \leq r$.

Given a schedule S for a graph G and flow set F, we can define the throughput achieved by the slowest rate, indicated in packets per round, at which any of the flows in F are satisfied in the limit. Formally:

▶ **Definition 2.1.** *Fix a schedule S defined with respect to network topology graph $G = (V, E)$ and flow set F. We say S* achieves throughput t *with respect to G and F, if there exists a* convergence round $r_0 \geq 1$*, such that for every $r \geq r_0$ and every packet type i:*

$$\left(\frac{del_i(r)}{r}\right) \geq t,$$

where $del_i(r)$ is the largest j such that for every $l \leq j$, packet (i, l) has been delivered by round r.

The above definition of throughput concerns performance in the limit, since r_0 can be arbitrarily large. In some cases, though, we might also be concerned with how quickly we achieve this limit. Our notion of convergence round allows us to quantify this, so we will provide bounds on the convergence round where relevant.

Many of the results in this paper concern algorithms that produce schedules. Our centralized algorithms take G and F as input and efficiently produce a compact description of an infinite schedule (i.e., an infinitely repeatable finite schedule). Our distributed algorithms assume a computational process running at each node in G, and for each $(s_i, R_i) \in F$, the source s_i is provided an infinite sequence of packets to deliver to R_i. An execution of such a distributed algorithm might contain communication other than the *flow packets* provided as input; e.g., the algorithm might distributedly (in the mobile telephone model) compute a routing structure to coordinate efficient packet communication. However, a unique schedule can be extracted from each such execution by considering only communication corresponding to the flow packets.

While our definition of throughput is for schedules and not algorithms, we will say that an algorithm *achieves* throughput α if it results in a schedule that achieves throughput α.

In the sections that follow, we consider three different types of capacity: *pairwise*, *broadcast*, and *all-to-all*. Each capacity type can be formalized as a set of constraints on the allowable flow sets. For each capacity type we study achievable throughput with respect to both *arbitrary* and *random* network topology graphs. In the arbitrary case, the only constraints on the graph is that it is connected. For the random case, we must describe a process for randomly generating the graph. To do so, we use the approach introduced for this purpose by Gupta and Kumar [19]: randomly place nodes in a unit square, and then add an edge between all pairs within some fixed radius. Formally:

▶ **Definition 2.2.** *For a given real value* radius r*, $0 < r \leq 1$, and network size $n \geq 1$, the* GK(n, r) network generation process *randomly generates a network topology $G = (V, E)$ as follows:*

1. *Let $V = \{u_1, u_2, ..., u_n\}$. Place each of the n nodes in V uniformly at random in a unit square in the Euclidean plane.*
2. *Let $E = \{(u_i, u_j) : d(u_i, u_j) \leq r\}$, where d is the Euclidean distance metric.*

We will use the notation $G \sim GK(n, r)$ to denote that G is a random graph generated by the $GK(n, r)$ process. When studying a specific definition of capacity with respect to a network randomly generated with the GK process, it is necessary to specify how the flow set is generated. Because these details differ for each of the three capacity definitions, we defer their discussion to their relevant sections.

2.3 Mathematical Preliminaries

We begin with some basic definitions. Fix some connected undirected graph $G = (V, E)$. We define $c(G)$ to be the number of components in G. In a slight abuse of notation, we define $G \setminus S$, for $S \subseteq V$, to be the graph defined when we remove from G the nodes in S and their adjacent edges. For a fixed integer $k > 1$, we say G *has a* k*-tree* if there exists a spanning tree in G with maximum degree k. Finally, let $d(G)$ be the smallest k such that G has a k-tree. That is, $d(G)$ describes the maximum degree of the *minimum degree spanning tree* (MDST) in G.

Some of our results will use the following simple corollary of a theorem of Win [29]. The proof, which utilizes the notion of *graph toughness* [29], can be found in the full version.

▶ **Theorem 2.3.** *Fix an undirected graph $G = (V, E)$ and degree $k \geq 3$. If $d(G) > k$, then there exists a non-empty subset of nodes $S \subset V$ such that there are more than $c(G \setminus S) > (k-2) \cdot |S|$.*

3 Pairwise Capacity

In their seminal paper [19], Gupta and Kumar approached the question of network capacity by considering the maximum throughput achievable for a collection of disjoint pairwise flows, each consisting of a single source and destination. They studied achievable capacity in both *arbitrary* networks as well as *random* networks. In this section, we apply this approach to the mobile telephone model.

To do so, we formalize the *pairwise capacity* problem as the following constraint on the allowable flow sets (see Section 2.2): for every pair $(s_i, R_i) \in F$, it must be the case that $R_i = \{x\}$ (i.e., $|R| = 1$), and neither s nor x shows up in any other pair in F.

3.1 Arbitrary Networks

We begin by designing algorithms that (approximate) the maximum achievable throughput in an arbitrary network. For now we will not focus on the convergence time, since our definition of capacity applies in the limit, so we describe the following as a centralized algorithm (the time required for each node to gather the full graph topology and run this algorithm locally to generate an optimal routing schedule is smoothed out over time). But as usual when considering centralized algorithms, we will care about the running time.

Formally, we define the Pairwise Capacity problem to be the optimization problem where we are given a graph $G = (V, E)$ and a pairwise flow set F, and are asked to output a description of an (infinite) schedule which maximizes the throughput. Our algorithm will in particular output a finite schedule which is infinitely repeated. Our approach is to establish a strong connection between multi-commodity flow and optimal schedules, and then apply existing flow solutions as a step toward generating a near optimal solution for the current network. In other words, we give an approximation algorithm for Pairwise Capacity via a reduction to a multi-commodity flow problem.

▶ **Theorem 3.1.** *There is a (centralized) algorithm for Pairwise Capacity that achieves throughput which is a $(3/2 + \epsilon)$-approximation of the optimal throughput, for any $\epsilon > 0$. The convergence time is $n^{O(1)}\epsilon^{-2}$ and the running time is $n^{O(1)}\epsilon^{-1}$.*

Multi-Commodity Flow. In the *maximum concurrent multi-commodity flow (MCMF)* problem, we are given a triple (D, M, cap), where $D = (V_D, E_D)$ is a digraph, M is collection $M \subseteq V_D \times V_D$ of node-pairs (each representing a *commodity*), and $cap : E_D \to \mathbb{R}_0^+$ are flow capacities on the edges. Let $K = |M|$ be the number of commodities. The output is a collection $f = (f_1, f_2, \ldots, f_K)$ of flows satisfying conservation and capacity constraints. Namely, for each flow f_i and for each vertex $v \in G$ where $v \notin \{s_i, t_i\}$, the flow into a node equals the flow going out: $\sum_{e=(u,v)\in E_D} f_i(e) = \sum_{e'=(v,w)\in E_D} f_i(e')$. Also, the flow through each edge is upper bounded by its capacity: $f(e) = \sum_{i=1}^{K} f_i(e) \leq cap(e)$. Let $v(f_i) = \sum_{w,e=(s_i,w)\in E_D} f_i(e)$ be the *value* of flow i, or the total flow of commodity i leaving its source. The value of the total flow f is $v(f) = \min_{i=1}^{K} v(f_i)$, and our goal is to maximize $v(f)$. We refer to f as an *MCMF flow* and the constituent commodity flows as *subflows*.

The MCMF problem can be solved in polynomial-time by linear programming. There are also combinatorial approximation schemes known, and our version of the problem can be approximated within a $(1+\epsilon)$-factor in time $\tilde{O}((m+K)n/\epsilon^2)$ [22].

We first show how to round an MCMF flow to use less precision while limiting the loss of value. We say that a MCMF flow is *ϕ-rounded* if the flow of each commodity on each edge is an integer multiple of $1/\phi$: $\lfloor f_i(e)\cdot\phi\rfloor = f_i(e)\cdot\phi$, for all i, and all edges e. We show how to produce a rounded flow of nearly the same value.

▶ **Lemma 3.2.** *Let f be a MCMF flow and ϕ be a number. There is a rounding of f to a ϕ-rounded flow f' with value at least $v(f') \geq v(f)(1 - Km/\phi)$, and it can be generated in polynomial time.*

Proof. We focus on each subflow f_i. By standard techniques, each subflow f_i can be decomposed into a collection of paths $P_1,\ldots,P_s$ and values $\alpha_1,\ldots,\alpha_s$, with $s \leq m = |E|$, such that $f_i(e) = \sum_{j,P_j\ni e}\alpha_j$ for each edge e. Let $\alpha'_j = \lfloor \alpha_j\cdot\phi\rfloor/\phi$, for each j, and observe that $\alpha'_j \geq \alpha_j - 1/\phi$. We form the ϕ-rounded flow f' by $f'_i(e) = \sum_{j,P_j\ni e}\alpha'_j$, for each edge e. It is easily verified that conservation and capacity constraints are satisfied. By the bound on α', it follows that the value of the rounded flow is bounded from below by $v(f'_i) \geq v(f_i) - s/\phi \geq v(f_i) - m/\phi$. The value of each flow is trivially bounded from below by $v(f_i) \geq 1/K$ (which is achieved by sending $1/K$ of each commodity flow along a single path). Thus, $v(f'_i) \geq (1 - Km/\phi)v(f_i)$. ◀

We now turn to the reduction of Pairwise Capacity to MCMF. Given $G = (V,E)$ and F, along with a parameter τ, we form the flow network $\mathcal{D}_\tau = (D, M, cap_\tau)$ as follows. The undirected graph $G = (V,E)$ is turned into a digraph $D = (V_D, E_D)$ with two copies v^{in}, v^{out} of each vertex: $V_D = \{v^{in}, v^{out} : v \in V\}$ and edges $E_D = \{(u^{out}, v^{in}) : uv \in E\} \cup \{(v^{in}, v^{out}) : v \in V\}$. The source/destination pairs carry over: $M = \{(s^{in}, t^{out}) : (s,\{t\}) \in F\}$. Finally, capacities of edges in E_D are $cap_\tau(u_{out}, v_{in}) = \infty$ and $cap_\tau(v_{in}, v_{out}) = 1 + t_v\cdot\tau/2$, where t_v is the number of source/destination pairs in F in which v occurs. Observe that there is a one-to-one correspondence between simple paths in G and in D (modulo the in/out version of the start/end node).

▶ **Lemma 3.3.** *The throughput of any schedule on (G,F) is at most $\tau^*/2$, where τ^* is the largest value such that $\mathcal{D}_{\tau^*}$ has MCMF flow of value τ^*.*

Proof. Let $\mathcal{A}$ be a mobile telephone schedule and let T be its throughput. We want to show that $\mathcal{D}_{2T}$ has MCMF flow of value $2T$; this is sufficient to imply the lemma. We assume that packets flow along simple paths, and we achieve that by eliminating loops from paths, if necessary. By the throughput definition, there is a round $r_0 = r^{\mathcal{A},T}$ such that for every round $r \geq r_0$ and every source/destination pair i, the number of i-packets delivered by round r is at least $T\cdot r$. Let X_i be the first Tr_0 i-packets delivered (necessarily by round r_0), for each type i, and let $X = \cup_i X_i$. For each edge $e = uv$ and pair i, let $q_i(u,v)$ be the number of packets in X_i that passed through e, from u to v. Also, for a vertex v, let $a_i(v)$ denote the number of i-packets originating at v, i.e., $a_i(v) = Tr_0$ if $v = s_i$ and $a_i(v) = 0$ otherwise. Similarly, let $b_i(v)$ be the number of i-packets with v as its destination. Finally, let $q_i(v)$ be the number of packets in X_i that flow through v, but did not originate or terminate at v, and observe that $q_i(v) = \sum_{w,vw\in E} q_i(v,w) - a_i(v) = \sum_{u,uv\in E} q_i(u,v) - b_i(v)$.

Define the collection $f = (f_1, f_2, \ldots, f_K)$ of functions where for each i, $f_i(u_{out}, v_{in}) = 2q_i(u,v)/r_0$, for each edge $e = uv \in E$, and $f_i(v_{in}, v_{out}) = 2(q_i(v) + a_i(v) + b_i(v))/r_0$, for each vertex in V. Observe that the flow $f_i(u_{out}, v_{in})$ corresponds to twice the number of i-packets going from u to v (scaled by factor $1/r_0$). The flow $f_i(v_{in}, v_{out})$ from v_{in} to v_{out} corresponds to the number of packets in X_i coming into v plus the number of those going out of v (scaled by factor $1/r_0$), counting those that go through v twice, but those originating or terminating at v only *once*. We claim that f is a valid MCMF flow in $\mathcal{D}_{2T}$ of value $2T$, which implies the lemma. Let $f_i^a(v) = 2a_i(v)/r_0$ ($f_i^b(v) = 2b_i(v)/r_0$) be the amount of type-i flow originating (terminating) at v, respectively.

First, to verify flow conservation at nodes, consider a type i, and observe first that all packets in X start at the source s_i and end at the destination t_i.

$$\begin{aligned} f_i(v_{in}, v_{out}) &= \frac{2(q_i(v) + a_i(v) + b_i(v))}{r_0} = \frac{2a_i(v)}{r_0} + \sum_{u, uv \in E} \frac{2q_i(u,v)}{r_0} \\ &= f_i^a(v) + \sum_{u, (u_{out}, v_{in}) \in E_D} f_i(u_{out}, v_{in}) \ . \end{aligned}$$

That is, the flow from each node v_{in} equals the flow coming in plus the flow generated at the node (noting also that no flow terminates at the node). Similarly, the flow into v_{out} equals the flow terminating at the node plus the node going out:

$$\begin{aligned} f_i(v_{in}, v_{out}) &= \frac{2(q_i(v) + a_i(v) + b_i(v))}{r_0} = \frac{2b_i(v)}{r_0} + \sum_{w, vw \in E} \frac{2q_i(v,w)}{r_0} \\ &= f_i^b(v) + \sum_{w, (v_{out}, w_{in}) \in E_D} f_i(v_{out}, w_{in}) \ . \end{aligned}$$

Second, to verify capacity constraints, observe that if $q(v) = \sum_i q_i(v)$ is the number of packets that flow through node v, then

$$2q(v) + \sum_i (a_i(v) + b_i(v)) \le r_0 \ ,$$

since v needs to handle flowing-through packets in two separate rounds and it can only process a single packet in a round. Thus, the flow through (v_{in}, v_{out}) is bounded by

$$f(v_{in}, v_{out}) = \frac{2}{r_0}(q(v) + \sum_i (a_i(v) + b_i(v))) = \frac{1}{r_0}(2q(v) + \sum_i (a_i(v) + b_i(v))) + t_v T \le 1 + t_v T \ ,$$

satisfying the capacity constraints.

Finally, it follows directly from the definition of f_i^a (or f_i^b) that the flow value is $2T$. ◀

To prove Theorem 3.1 we need to introduce edge multicoloring.

▶ **Definition 3.4.** *Given a graph $G = (V, E)$ and a* color requirement *$r(e) \in \mathbb{N}$ for each edge $e \in E$. An* edge multicoloring *of (G, r) is a function $\pi : E \to 2^{\mathbb{N}}$ that satisfies the following: a) if $e_1, e_2 \in E$ are adjacent then $\pi(e_1) \cap \pi(e_2) = \emptyset$, and b) $|\pi(e)| \ge r(e)$, for each edge $e \in E$. The number of colors used is $|\cup_e \pi(e)|$, the size of the support for π.*

We shall use the follow result on edge multicolorings.

▶ **Theorem 3.5** (Shannon [27]). *Given a graph $G = (V, E)$ and a* color requirement *$r(e) \in \mathbb{N}$ for each edge $e \in E$, there is a polynomial-time algorithm that edge multicolors (G, r) using at most $3\Delta_r(v)/2$ colors, where $\Delta_r(v) = \sum_{e \ni v} r(e)$.*

We can now prove Theorem 3.1.

Proof of Theorem 3.1. Let (G, F) be a given Pairwise Capacity instance and let $\epsilon > 0$. We perform binary search to find a value τ such that: a) An $1+\epsilon/4$-approximate MCMF algorithm produces flow f of value at least $\tau(1-\epsilon/4)$ on $\mathcal{D}_\tau$, and b) The same does not hold for $\tau(1+\epsilon/4)$. The resulting flow $f = (f_1, \ldots, f_K)$ is then of value at least $\tau^*(1-\epsilon/4)/(1+\epsilon/4) \geq \tau^*(1-\epsilon/4)^2 \geq \tau^*(1-2\epsilon/4)$. Recall that K is the number of commodities, and so $K = |F|$.

Let $N = 4\epsilon^{-1}Km$. We apply Lemma 3.2 to create from f an N-rounded flow $f' = (f'_1, \ldots, f'_K)$. By Lemma 3.2, this decreases the flow value by a factor of at most $1 - Km/N = 1-\epsilon/4$, i.e., $v(f') \geq (1-\epsilon/4)v(f) \geq (1-3\epsilon/4)\tau^*$.

We then form an edge multicoloring instance on G as follows. Each edge e requires $r(e)$ colors, where $r(e) = \sum_i r_i(e)$ and $r_i(e) = f'_i(e) \cdot N$. The weighted degree of each node v is then $d_r(v) = \sum_{e \ni v} r(e) = N \sum_{e \ni v} f'(e) \leq N \sum_{e \ni v} f(e) = \sum_{e=(v,u) \in E_D} f(e) + \sum_{e'=(w,v) \in E_D} f(e) \leq 2N$, by node capacity constraints. We apply the algorithmic version of Shannon's Theorem 3.5 to edge multicolor (G, r) with at most $3N$ colors. This induces an initial schedule of length $3N$, which is then repeated as needed. Within each $3N$ rounds, $\sum_{v, e=(s_i, v)} r_i(e) = N \cdot v(f'_i)$ i-packets depart from its source s_i.

Let $r_0 = \frac{4n}{\epsilon}(3N)$. Consider the situation after round $r \geq r_0$. Observe that each packet is forwarded at least once during each $3N$ rounds, and thus it is delivered within $n(3N)$ rounds after it is transmitted from its source, since each path used is simple. Thus, the total number of type-i packets that remain in the system in the end is at most a $\epsilon/4$-fraction of the delivered packets. Averaged over the r rounds gives throughput of

$$\frac{N \cdot v(f'_i)}{3N} \cdot (1-\epsilon/4) = \frac{1}{3} v(f'_i) \cdot (1-\epsilon/4) \ .$$

Hence, the throughput achieved is at least

$$T \geq \frac{1}{3} v(f')(1-\epsilon/4) \geq \frac{1}{3} v(f)(1-\epsilon/4) \geq \frac{1}{3}\tau^*(1-\epsilon) \ . \tag{1}$$

By Lemma 3.3, the throughput is then $3/2+\epsilon$-approximation of optimal.

The computation performed is dominated by the application of Shannon's algorithm, which runs in time $O((\Delta_r + n)\hat{m})$, where $\hat{m}$ is the number of multiedges and $\Delta_r \leq 2N$ is the maximum weighted degree. Here, $\hat{m} = \sum_e q(e) = N \sum_e \sum_i f_i(e) \leq N \cdot m$. Hence, the number of computational steps is at most $O(mN^2) = O(m^3K^2\epsilon^{-2})$. The convergence time is $r_0 = \frac{4n}{\epsilon}(3N) = O(nmK\epsilon^{-2})$. ◀

We note that the factor $3/2$ cannot be avoided in a reduction to flow. Consider the graph G on six vertices $V = \{s_i, t_i : i = 0, 1, 2\}$ and edges $\{s_i t_{i'}, t_i t_{i'} : i = 0, 1, 2, i' = i - 1 \bmod 3\}$. The optimal throughput is $1/3$, with respect to $F = \{(s_i, t_i) : i = 0, 1, 2\}$. This corresponds to the directed graph D on nine nodes: $\{s_i, t_i^{in}, t_i^{out} : i = 0, 1, 2\}$ and edges $\{(s_i, t_{i'}^{in}), (t_{i'}^{out}, t_i^{in}), (t_i^{in}, t_i^{out}) : i = 0, 1, 2, i' = i - 1 \bmod 3\}$, and three subflows: $M = \{s_i, t_i^{out} : i = 0, 1, 2\}$. Then, $\mathcal{D}_1 = (D, M, cap_1)$, where $cap_1(t_i^{in}, t_i^{out}) = 2$, has flow of value 1.

3.2 Random Networks

We now consider achievable throughput for the pairwise capacity problem in networks randomly generated with the GK process defined in Section 2.2. Following the lead of the original Gupta and Kumar capacity paper [19], we assume the flow sets are also randomly generated with uniform randomness and contain all the nodes (i.e., every node shows up as a source or destination). A minor technical consequence of this definition is that it requires us to constrain our attention to even network sizes.

We begin in Section 3.2.1 by identifying a threshold value for the radius r below which the randomly generated network is likely to be disconnected, trivializing the achievable throughput to 0. In Sections 3.2.2 and 3.2.3, we then prove that for any radius value r that is at least a sufficiently large constant factor greater than the threshold, with high probability in n, the optimal achievable throughput is in $\Theta(r)$.

3.2.1 Connectivity Threshold

When analyzing networks and flows generated by the $GK(n,r)$ network generation process, we must consider the radius parameter r. If r is too small, then we expect a network in which some sources are disconnected from their corresponding destinations, making the best achievable throughput trivially 0. Here we study a *connectivity threshold* value $r_c(n) = \sqrt{\frac{\alpha \log n}{n}}$, defined with respect to a network size n and a constant fraction α. We prove that for any $r \leq r_c(n)$, with probability at least $1/2$, given a network generated by $G(n,k)$ and a random pairwise flow set F, there exists at least one pair in F that is disconnected.

► **Theorem 3.6.** *There is some constant $\alpha > 0$ so that for every sufficiently large even network size n and radius $r \leq r_c(n) = \sqrt{\frac{\alpha \log n}{n}}$, if $G \sim GK(n,r)$ and F is a random pairwise flow set, then with probability at least $1/2$ there exists $(s, \{x\}) \in F$ such that s is disconnected from x in G.*

At a high level, to prove this theorem we divide the unit square into a grid consisting of *boxes* of side length r, and then group these boxes into *regions* made up of 3×3 collections of boxes. If a given region has a node u in the center box, and all its other boxes are empty, then u is disconnected from any node not in its own box. Our proof calculates that for a sufficiently small constant fraction α used in the definition of the connectivity threshold, with probability at least $1/2$, there will be a node u such that u is isolated as described above, and u is part of a source/destination pair with another node v located in a different box.

Given this setup, the main technical complexity in the proof is carefully navigating the various probabilistic dependencies. One place where this occurs is in proving the likelihood of empty regions. For sufficiently small α values, the expected number of non-empty regions is non-zero, but we cannot directly concentrate on this expectation due to the dependencies between emptiness events. These dependencies, however, are dispatched by leveraging the negative association between the indicator variables describing a region's emptiness (e.g., if region i is not empty, this *increases* the chance that region $j \neq i$ is empty).

3.2.2 Bound on Achievable Throughput

In the previous section, we identified a radius threshold $r_c(n)$ below which a randomly generated network is likely to disconnect a source and destination, reducing the achievable throughput to a trivial 0. Here we study the properties of the networks generated with radius values on the other side of this threshold. In particular, we show that for any radius $r \geq r_c(n)$, with high probability, the randomly generated network and flow set will allow an optimal throughput bounded by $O(r)$. The intuition for this argument is that if nodes are evenly distributed in the unit square, a constant fraction of senders will have to deliver packets from one half of the square to the other, necessarily requiring many packets to flow through a small column in the center of the square, bounding the achievable throughput.

▶ **Theorem 3.7.** *For every sufficiently large even network size n and radius $r \geq r_c(n)$, given a network $G \sim GK(n,r)$ and a random pairwise flow set F, the throughput of every schedule (w.r.t. G and F) is $O(r)$ with high probability.*

3.2.3 Tightness of the Throughput Bound

In Section 3.2.2, we proved an upper bound of $O(r)$ on the achievable throughput in a network generated by $GK(n,r)$, for $r \geq r_c(n)$, and random pairwise flows. Here we show this result is tight by showing how to produce a schedule that achieves throughput in $\Omega(r)$ with respect to a random G and F. Formally:

▶ **Theorem 3.8.** *There exists a constant $\beta > 1$ such that, for any sufficiently large network size $n \geq 2$ and radius $r \geq \beta r_c(n)$, if $G \sim GK(n,r)$ and F is a random pairwise flow set, then with high probability in n there exists a schedule that achieves throughput in $\Omega(r)$ with respect to G and F.*

At a high level, our argument divides the unit square into box of side length $\approx r$. We prove that with high probability, both nodes and pairwise demands are evenly distributed among the boxes. This allows a schedule that efficiently moves many packets in parallel up and down columns to the row of their destination, and then moves these packets left and right along the rows to reach their destination. The time required for a given packet to make it to its destination is bounded by the column and row length of $\approx 1/r$, yielding an average throughput in $\Theta(r)$. The core technical complexity of this argument is the careful manner in which packets are moved onto and off a set of parallel paths while avoiding more than a small amount of congestion at any point in their routing.

4 Broadcast Capacity

The broadcast capacity problem assumes a designated *source* node has an infinite sequence of packets to spread to the entire network, implementing a one-to-all packet stream. Formally, this version of the capacity problem constrains the flow set to only contain a single pair of the form $\{s, V \setminus \{s\}\}$, for some source $s \in V$. As we will show, the achievable throughput for this problem in a given network graph G is strongly related to $d(G)$, the maximum degree of the minimum degree spanning tree (MDST) for G (see Section 2.3).

4.1 A Bound on Achievable Throughput for Arbitrary Networks

We establish that the maximum degree of an MDST in G – that is, $d(G)$ – bounds the achievable throughput, with larger values of $d(G)$ leading to lower throughput. The bound is primarily graph theoretic: arguing a fundamental limit on the rate at which packets can spread through a given topology.

▶ **Theorem 4.1.** *Fix a connected network graph $G = (V,E)$ and broadcast flow set F with source s. Then every schedule achieves throughput at most $O(1/d(G))$.*

Proof. Fix some $G = (V,E)$, $s \in V$, and $\mathcal{A}$, as specified by the theorem statement. If $d(G) \leq 4$ then the theorem is trivially true as all throughput values are in $O(1)$. Assume therefore that $d(G) > 4$. This allows us to apply Theorem 2.3 for $k = d(G) - 1$, which establishes that there exists a non-empty subset $S \subset V$ such that $c(G \setminus S) > q \cdot |S|$, for $q = k - 2 = d(G) - 3 > 1$ (where, as defined in Section 2.3, $c(G \setminus S)$ is the number of connected components after removing nodes in S from graph G).

Let C be the set of components in $G \setminus S$ that do not include the source s. Fix a packet t spread by s. We say t *arrives* at $C_i \in C$ in round $r \geq 1$, if this is the first round in which a node in C_i receives packet t. In this case, t must have been previously received by some bridge node in S that is adjacent to C_i. This holds because if t can make it from s's component to C_i without passing through a node in S, then removing S would not disconnect C_i.

Fix any packet count $i \geq 1$. Each packet requires $|C| = c(G \setminus S) - 1 \geq q|S|$ arrival events before it completes spreading. As we established above, each arrival event requires a given node in S to receive the given packet. Because each node in S can receive at most one packet per round, there are at most $|S|$ arrival events per round in the network.

Putting together these pieces, let T_i be the number of rounds required to spread i packets. We can lower bound this value as:

$$T_i \geq \frac{i \cdot |C|}{|S|} = \frac{i(q|S|)}{|S|} = iq \ .$$

It follows that for every schedule, and every i, at least T_i rounds are required to spread i packets – yielding a throughput upper bounded by $\frac{i}{T_i} \leq \frac{i}{i \cdot q} = 1/q = 1/(d(G) - 3)$, which yields the theorem. ◀

4.2 An Optimal Routing Algorithm for Arbitrary Networks

Here we describe a routing algorithm that achieves broadcast capacity throughput in $\Omega(1/d(G))$, when executed in a connected graph G. The high-level idea is to first construct an MDST T in the graph G. We then edge color T using $O(d(G))$ colors, and use this coloring to simulate the standard CONGEST model, parameterized so that a constant number of packets can fit within its bandwidth limit. We analyze a straightforward pipelining flooding algorithm for the CONGEST model that converges to constant throughput. When combined with our simulator, which requires $O(d(G))$ real rounds to simulate each CONGEST round, the result is a solution that achieves an average latency of $O(d(G))$ rounds per packet, providing the claimed $\Omega(1/d(G))$ throughput.

As in the pairwise setting, we can do this in a centralized fashion at the cost of a large convergence time (in particular, it takes up to $O(n^2)$ rounds to gather the graph topology locally before we can run a centralized algorithm). In order to decrease the convergence time, we describe in the full version a distributed version of this strategy that still converges to an optimal $\Omega(1/d(G))$ throughput in $O(n^2)$ rounds, but guarantees to converge to at least $\Omega\left(\frac{1}{d(G)+\log n}\right)$ throughput in $\tilde{O}(D(T) \cdot d(G) + \sqrt{n})$ rounds, where $D(T) \leq n$ is the diameter of a spanning tree T built by the algorithm and $\tilde{O}(\cdot)$ suppresses polylog(n) factors.

Formally, we prove the following theorem:

▶ **Theorem 4.2.** *There exists a (distributed) algorithm which, when executed in a connected network topology $G = (V, E)$ of size $n = |V|$, with a broadcast capacity flow set with source $s \in V$, achieves throughput in $\Omega(1/(d(G) + \log n))$ with convergence round $\tilde{O}(n \cdot d(G))$ and achieves throughput in $\Omega(1/d(G))$ with convergence round $O(n^2)$.*

4.3 Random Networks

The preceding broadcast capacity results hold for any connected network graph. Here we study the problem in networks randomly generated by the GK process with a communication radius sufficiently larger than the threshold $r_c(n)$.

Leveraging techniques from Section 3.2.3, we prove that such random networks are likely to contain a constant degree MDST, which, as established in Theorem 4.2, support constant throughput.

▶ **Theorem 4.3.** *There exists a (distributed) algorithm, such that for any sufficiently large network size $n > 1$ and constant $\beta \geq 1$, and radius $r \geq \beta r_c(n)$, if $G \sim GK(n, r)$ then with high probability the algorithm achieves constant throughput (for any s).*

5 All-to-All Capacity

We now consider the all-to-all capacity problem, which assumes all nodes begin with an infinite sequence of packets to spread to all other nodes. Formally, this variation of the capacity problem considers only the following canonical flow set: $F_{all} = \{(s, V \setminus \{s\}) : s \in V\}$.

In Section 4, we described and analyzed an algorithm that achieved a throughput in $\Omega(1/d(G))$ for delivering packets from a single source to the whole network. To solve all-to-all capacity, we could run n instances of this algorithm: one for each source, rotating through the different instances in a round robin fashion. This approach provides a baseline throughput result of $\Omega(1/(n \cdot d(G)))$. The key questions are whether or not this bound is tight, and whether there are simpler or more natural strategies than deploying round robin interleaving of single-source broadcast.

In the full version, we answer both questions in the affirmative by generalizing our argument from Theorem 4.1 to prove that no schedule achieves better than $O\left(\frac{1}{d(G) \cdot n}\right)$ throughput, and then exhibiting a matching distributed algorithm *SG* that uses a more natural strategy than round robin broadcast. Formally:

▶ **Theorem 5.1.** *When executed in a connected network topology $G = (V, E)$ of size $n = |V|$, with high probability in n: the* SG *algorithm achieves throughput in $\Omega\left(\frac{1}{d(G) \cdot n}\right)$ with respect to G and F_{all}. Furthermore, every schedule achieves throughput at most $O\left(\frac{1}{n \cdot d(G)}\right)$ with respect to G and F_{all}.*

Finally, notice that a direct corollary of our argument from Section 4.3, which establishes that a random graph contains a constant degree MDST (for sufficiently large radius) with high probability, is that with this same probability *SG* achieves $\Omega(1/n)$ throughput (which is best possible for all-to-all capacity).

5.1 Implications for One-Shot Gossip

Existing results for one-shot gossip in the mobile telephone model are expressed with respect to the vertex expansion (denoted α) of the graph topology [25, 24]. The best known results requires $O((n/\alpha)\text{polylog}(n))$ rounds, which is not tight in all graphs as vertex expansion does not necessarily characterize optimal gossip.[3] A key open question from [24] is whether it is possible to produce a gossip algorithm that is optimal (or within log factors of optimal) in *all* network topology graphs. The techniques used in the above capacity bounds help us prove the following, which largely resolves this open question:

[3] Consider, for example, a path of length n, which has $\alpha = 2/n$. It is possible to pipeline n messages through this network in $\Theta(n)$ rounds, which is much faster than $\tilde{O}(n/\alpha) = \tilde{O}(n^2)$.

▶ **Theorem 5.2.** *Fix a connected network topology $G = (V, E)$ with diameter D, size $n = |V|$, and MDST degree $d(G)$. Every solution to the one-shot gossip in G requires $\Omega(d(G)\cdot n)$ rounds. There exists an algorithm solves the problem in $O((D + \sqrt{n})\text{polylog}(n) + n(d(G) + \log n)) = \tilde{O}(d(G) \cdot n)$ rounds, with high probability in n.*

References

1 Gianluca Aloi, Marco Di Felice, Valeria Loscrì, Pasquale Pace, and Giuseppe Ruggeri. Spontaneous smartphone networks as a user-centric solution for the future internet. *IEEE Communications Magazine*, 52(12):26–33, 2014.

2 Apple. MultipeerConnectivity | Apple Developer Documentation. *Internet:* `https://developer.apple.com/documentation/multipeerconnectivity` *[Accessed : 07/20/2018]*, 2018.

3 Daniel Camps-Mur, Andres Garcia-Saavedra, and Pablo Serrano. Device-to-device communications with Wi-Fi Direct: Overview and experimentation. *IEEE Wireless Communications*, 20(3):96–104, 2013.

4 Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. Rumour Spreading and Graph Conductance. In *Proceedings of the ACM-SIAM symposium on Discrete Algorithms (SODA)*, 2010.

5 Vasek Chvátal. Tough graphs and Hamiltonian circuits. *Discrete Mathematics*, 5(3):215–228, 1973.

6 Sebastian Daum, Fabian Kuhn, and Yannic Maus. Rumor spreading with bounded in-degree. In *Proceedings of the International Colloquium on Structural Information and Communication Complexity*, 2016.

7 Michael Dinitz, Jeremy Fineman, Seth Gilbert, and Calvin Newport. Load balancing with bounded convergence in dynamic networks. In *INFOCOM*, pages 1–9, 2017.

8 Michael Dinitz, Magnús M. Halldórsson, Taisuke Izumi, and Calvin Newport. Distributed Minimum Degree Spanning Trees. In *Proceedings of the ACM Symposium on the Principles of Distributed Computing (PODC)*, 2019 (to appear.).

9 Nikolaos Fountoulakis and Konstantinos Panagiotou. Rumor spreading on random regular graphs and expanders. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 560–573. Springer, 2010.

10 Alan M Frieze and Geoffrey R Grimmett. The shortest-path problem for graphs with random arc-lengths. *Discrete Applied Mathematics*, 10(1):57–77, 1985.

11 Martin Fürer and Balaji Raghavachari. Approximating the Minimum-Degree Steiner Tree to within One of Optimal. *Journal of Algorithms*, 17(3):409–423, 1994.

12 Open Garden. FireChat. *Internet:* `https://www.opengarden.com/` *[Accessed: 07/20/2018]*, 2018.

13 Open Garden. The Open Garden Hotspot. *Internet:* `https://www.opengarden.com/` *[Accessed: 07/20/2018]*, 2018.

14 Mohsen Ghaffari and Calvin Newport. How to Discreetly Spread a Rumor in a Crowd. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2016.

15 George Giakkoupis. Tight bounds for rumor spreading in graphs of a given conductance. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS)*, 2011.

16 George Giakkoupis. Tight bounds for rumor spreading with vertex expansion. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014.

17 George Giakkoupis and Thomas Sauerwald. Rumor spreading and vertex expansion. In *Proceedings of the ACM-SIAM symposium on Discrete Algorithms (SODA)*, pages 1623–1641. SIAM, 2012.

18 Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of Bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753, 2012.

19 Piyush Gupta and Panganmala R Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

20 Adrian Holzer, Sven Reber, Jonny Quarta, Jorge Mazuze, and Denis Gillet. Padoc: Enabling social networking in proximity. *Computer Networks*, 111:82–92, 2016.

21 Zongqing Lu, Guohong Cao, and Thomas La Porta. Networking smartphones for disaster recovery. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE, 2016.

22 Aleksander Madry. Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing (STOC)*, pages 121–130. ACM, 2010.

23 David Mark, Jayant Varma, Jeff LaMarche, Alex Horovitz, and Kevin Kim. Peer-to-Peer Using Multipeer Connectivity. In *More iPhone Development with Swift*, pages 239–280. Springer, 2015.

24 Calvin Newport. Leader Election in a Smartphone Peer-to-Peer Network. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2017. Full version available online at: `http://people.cs.georgetown.edu/~cnewport/pubs/le-IPDPS2017.pdf`.

25 Calvin Newport and Alex Weaver. Random Gossip Processes in Smartphone Peer-to-Peer Networks. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2019.

26 DG Reina, Mohamed Askalani, SL Toral, Federico Barrero, Eleana Asimakopoulou, and Nik Bessis. A survey on multihop ad hoc networks for disaster response scenarios. *International Journal of Distributed Sensor Networks*, 11(10):647037, 2015.

27 Claude E Shannon. A theorem on coloring the lines of a network. *Journal of Mathematics and Physics*, 28(1-4):148–152, 1949.

28 Noriyuki Suzuki, Jane Louie Fresco Zamora, Shigeru Kashihara, and Suguru Yamaguchi. Soscast: Location estimation of immobilized persons through sos message propagation. In *Proceedings of the International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, pages 428–435. IEEE, 2012.

29 Sein Win. On a connection between the existence of k-trees and the toughness of a graph. *Graphs and Combinatorics*, 5(1):201–205, 1989.