The Telephone k-multicast problem*

Daniel Hathcock Carnegie Mellon University, USA dhathcoc@andrew.cmu.edu
Guy Kortsarz Rutgers University, Camden, USA guyk@camden.rutgers.edu
R. Ravi Carnegie Mellon University, USA ravi@andrew.cmu.edu

Abstract

We consider minimum time multicasting problems in directed and undirected graphs: given a root node and a subset of t terminal nodes, multicasting seeks to find a minimum number of rounds within which all terminals can be informed with a message originating at the root. In each round, the telephone model we study allows the information to move via a matching from the informed nodes to the uninformed nodes.

Since minimum time multicasting in digraphs is poorly understood compared to the undirected variant, we study an intermediate problem in undirected graphs that specifies a target $k < t$, and requires the only k of the terminal be informed in the minimum number of rounds. For this problem, we improve implications of prior results and obtain an $\tilde{O}(t^{1/3})$ multiplicative approximation. For the directed version, we obtain an *additive* $\tilde{O}(k^{1/2})$ approximation algorithm (with a poly-logarithmic multiplicative factor). Our algorithms are based on reductions to the related problems of finding k-trees of minimum poise (sum of maximum degree and diameter) and applying a combination of greedy network decomposition techniques and set covering under partition matroid constraints.

*This material is based upon work supported in part by the Air Force Office of Scientific Research under award number FA9550-23-1-0031 to RR.

1 Introduction

We study an information spreading problem that captures applications in distributed computing [15] and keeping distributed copies of databases synchronized [2]. A given graph models a synchronous network of processors that exchange information in rounds. There are several models describing how information may be exchanged between processors in the graph. In this work, we focus on the classic *Telephone Model* [7]: during a round, each vertex that knows the message can send the message to at most one of its neighbors.

In the **Minimum Time Telephone Multicast** (MTM) problem, we are given a network, modeled by a directed or undirected graph $G(V, E)$, a root vertex r that knows a message, and a set S of terminals. The message must be transmitted from r to S under the telephone model. In every round, there is a set of vertices $K \subseteq V$ that know the message (initially $K = \{r\}$), and the communication in a given round is described by a matching $\{(k_1, v_1), \ldots, (k_\ell, v_\ell)\}$ between some pairs of vertices $k_i \in K$ and $v_i \notin K$ for which $k_i v_i \in E$. In the directed setting, edge $k_i v_i$ must be directed from k_i to v_i. Following this round, all of the matched vertices $\{v_i\}$ are added to K. When $S = V$ this problem is called **The Minimum Time Broadcast** (MTB) problem.

The best-known approximation ratio for the MTM problem on an undirected graph is $O(\log t / \log \log t)$ [5], where $t = |S|$. In [3], it is shown that unless $P = NP$, the MTB problem admits no $3-\epsilon$ approximation for any constant ϵ. For directed graphs, the Minimum Time Broadcast problem admits an $O(\log n)$ approximation [3] in an n-node graph. The same paper shows that unless $P = Quasi(P)$ the problem admits no better than $\Omega(\sqrt{\log n})$ approximation.

However, for the directed case the multicast problem *seems* harder to approximate. The best-known approximation ratio for this problem is an additive $O(\sqrt{t})$ guarantee (with polylogarithmic multiplicative factor) [4]. This leaves a wide gap between the current best approximation algorithms for undirected versus directed multicast problems. In this work, we make progress toward closing that gap by studying an intermediate problem, the **Minimum Time Telephone k-Multicast** problem (k-MTM), defined below.

Input: A directed or undirected graph $G(V, E)$ with root r, a collection of terminals $S \subseteq V$ and a number $k \leq |S|$.
Required: Send the message originating at r to *any* k terminals of S in the telephone model in a minimum number of rounds.

In terms of approximability, the undirected k-MTM problem lies between the undirected and directed MTM problems (up to $\log k$ factors, see [10][1]), while the directed k-MTM problem generalizes all of the aforementioned problems.

Applications. Broadcast and multicast problems find numerous applications in distributed settings. For example, in the Network Aggregation problem, each user sends its data to a

[1] [10] deals with the degree-bounded versions of these problems, but their proof works as well for poise problems. See below for the connection between poise and k-MTM.

chosen central vertex r. This is equivalent to broadcasting in the local model for distributed computation (see [8]). Broadcasting is also crucial in Sensor Networks [12]. Another application is ensuring that the maximum information delay in vector clocks problems is minimized [13, 11].

One application of multicasting is to keep information across copies of replicated databases consistent, by broadcasting from the changed copy to the others [13, 6, 14]. If we are given a large set of t terminals of which we only want to keep replicated copies in some k of them, finding the best k to minimize the maximum synchronization time among these termiunals corresponds to the k-MTM problem.

Minimum Poise Trees. Any telephone multicast schedule defines a tree rooted at r, spanning all terminals. The parent of a vertex $u \neq r$ is defined to be the unique vertex that sends the message to u. Let T^* be the tree defined by the optimal schedule. The height of T^* (the largest distance in T^* from the root) is denoted by D^*. The largest out-degree[2] in T^* is denoted by B^*. The *poise* of T^* is defined as $p^* = B^* + D^*$ [16]. Denote by OPT the number of rounds used by the optimal schedule. Since at every round, each informed vertex can send the message to at most one neighbor, OPT $\geq B^*$ and OPT $\geq D^*$. Hence, in general we have OPT $\geq p^*/2$. A partial converse is shown in [16]. A ρ approximation for the **Minimum Poise Steiner Tree** implies an $O(\log t) \cdot \rho / \log\log t$ approximation for the MTM problem.

Following [16], approximating the k-MTM problem is equivalent (up to logarithmic factors in k) to approximating the following **Minimum Poise Steiner k-Tree** problem:

Input: A directed or undirected graph $G(V, E)$ with root r, a collection $S \subseteq V$ of terminals, and a number k.
Required: A k-tree rooted at r, namely a tree $T'(V, E)$ containing paths from r to k of the terminals, with minimum poise.

We focus on approximating these poise problems.

Definition 1.1. *A $O(f(k))$-additive approximation for the Minimum Poise k-tree problem returns a tree T with k terminals, with maximum degree*[3] *$\tilde{O}(B^*) + O(f(k))$ and height $O(D^*)$.*

1.1 Our results

We give an $O(\sqrt{k})$-additive approximation for the directed versions.

Theorem 1.2. *Minimum Poise Steiner k-tree problem on directed graphs admits a polynomial time $\tilde{O}(k^{1/2})$-additive approximation. This implies the same approximation for the Minimum Time Telephone k-multicast problem.*

[2] For simplicity, we say degree instead of out-degree for the rest of the paper when discussing directed graphs.

[3] The $\tilde{O}$ notation hides poly-logarithmic factors in k

The second part of the statement follows from [16].

In [9], a multiplicative $O(\sqrt{k})$-approximation is given for the directed Min-Max Degree k-Tree problem, which asks to find a tree spanning k terminals while minimizing the maximum degree. Their algorithm iteratively finds trees containing $\sqrt{k} \cdot B^*$ terminals, and uses flows to connect them to the root. Our directed result is more general than that of [9] in that it can handle both degree bounds and height bounds. Moreover, our approximation for degree is stronger, since we get an additive $O(\sqrt{k})$ approximation. Therefore, it may be better than the approximation of [9] in the case that B^* is large. Our approximation ratio for the diameter is constant.

Our result is also more general than the $O(\sqrt{t})$-additive approximation for directed MTM of [4], as it handles the k-tree version of the problem, and recovers the same $O(\sqrt{t})$-additive approximation in the case $k = t$ (up to logarithmic factors). In [4], the so-called *multiple set-cover* problem is used, a variant of set cover, while our result uses max coverage subject to a matroid constraint.

For undirected graphs, we give an $\tilde{O}(t^{1/3})$ approximation, which is a better ratio in the worst case if k is close to t. This represents progress toward closing the gap between the approximability of undirected and directed MTM, since in [10] it is shown that the undirected k-MTM problem lies between undirected and directed MTM in terms of approximability.

Theorem 1.3. *The Minimum Poise Steiner k-tree problem on undirected graphs admits a polynomial time $\tilde{O}(t^{1/3})$ approximation, and therefore the Minimum Time Telephone k-multicast problem admits the same approximation.*

The $O(\sqrt{k})$ additive ratio can be as bad as $\Omega(\sqrt{t})$ multiplicative ratio, if B^* is constant and $k = \Omega(t)$. Therefore, in the worst case, an $\tilde{O}(t^{1/3})$ approximation is a better ratio. In addition, if $B^* = o(t^{1/6})$ and $k = \Omega(t)$, the multiplicative ratio gives a better additive ratio.

1.2 Technical Overview

For the directed case, our techniques are based on [4]. However, our problem is harder since it is not clear which k terminals to choose. An important difference is that we use an approximation algorithm for maximizing set coverage (a submodular function) under matroid constraints [1]. The multiplicative approximation for the undirected case builds on this, and requires several graph decomposition techniques to be carefully combined.

For both results, we denote the maximum degree as B^* and height as D^* of an optimal minimum poise tree T^*. It can be assumed that D^* and B^* are known by trying all possibilities, as there are only polynomially many. Moreover, since D^* is known, all vertices of distance greater than D^* from the root may be removed.

Directed Min-Poise Steiner k-Tree. In order to get an $O(\sqrt{k})$ additive approximation for the directed min-poise Steiner k-tree problem, we employ a greedy strategy. We iteratively find a collection of vertex-disjoint trees, each covering (i.e., containing) exactly $\sqrt{k}$ terminals and of height at most D^*, until no more can be found. We call these *good trees.*

In the case that at least $\sqrt{k}$ many good trees are found, an additive $O(\sqrt{k})$-approximation follows by taking any $\sqrt{k}$ of the good trees along with shortest paths from the root r to the roots of each of these trees. This yields a subgraph (not necessarily a tree, since the shortest paths may not be disjoint from the good trees) with maximum out-degree at most $2\sqrt{k}$, and radius (maximum distance from r) at most $2 \cdot D^*$. Moreover, the subgraph contains k terminals. Now the non-disjointness may be overcome by returning a shortest path tree spanning this subgraph. This gives the desired approximation.

In the other case that fewer than $\sqrt{k}$ good trees are found, we may still connect them to the root via shortest paths. This gives a subgraph of low poise, but does not yet cover k terminals. If $k_1 < k$ terminals are covered, we must determine how to cover $k - k_1$ additional terminals without inducing high degree or height.

This is the main technical contribution of the directed result: we can recast the covering of $k - k_1$ additional terminals as a set cover instance, and the desired poise guarantees can be obtained by imposing a partition matroid constraint on the sets in the instance. Then, an algorithm for approximating submodular function maximization subject to a matroid constraint [1] is applied.

Partition Matroid Set Coverage Procedure. Suppose we are given a partition of the graph into $A \cup C = V$ with $r \in A$, such that all of A is reachable with low poise and contains k_1 terminals. We want to cover at least $k - k_1$ terminals in C with low poise, and we know that there exists a tree T^* rooted at r which does so.

Say that a node $c \in C$ *covers* all the terminals in C that it can reach within distance D^*. In this way, we define a set cover instance over the ground set of terminals in C in which each set is identified by an edge (a, c) between a node in $a \in A$ and a node in $c \in C$. The set (a, c) contains all terminals covered by c. Defining the sets this way allows us to enforce degree constraints in the nodes in A since the sets can be partitioned by their member in A. That is, we form a partition with the parts $X(a) = \{(a, c) : c \in C, ac \in E\}$ for each $a \in A$. We now impose the constraint that at most B^* sets may be chosen from any part $X(a)$, reflecting the desired degree constraint. This is a set cover instance with a partition matroid constraint with a requirement of $k - k_1$ coverage.

The problem of selecting sets to maximize the number of terminals covered subject to the matroid constraint is a special case of submodular function maximization subject to a matroid constraint. Moreover, T^* provides a certificate that there exists a collection of sets satisfying the matroid constraint and covering at least $k - k_1$ terminals in C. Hence, we may apply the $(1 - \frac{1}{e})$-approximation for this problem [1] to find a collection of sets satisfying the matroid constraint and covering at least $(1 - \frac{1}{e}) \cdot (k - k_1)$ terminals in C.

Given the choice of sets (a, c) by the algorithm, we identify a set of edges that may be added to extend our subgraph to cover these terminals. These newly covered terminals are then removed, and the process repeated. In each round, we can cover a constant fraction of the desired number of terminals, so we need only $O(\log k)$ rounds. Moreover, any given round induces additional degree of only B^* on nodes in A. The degree induced on nodes in C depends on the size of the parts $X(a)$, and this can be bounded in our applications (e.g., by $\sqrt{k}$ in the directed setting described above). Finally, the distance from the root of any node added is $O(D^*)$, so in total the poise of the subgraph remains low. In the end, we again output a shortest path tree spanning this subgraph.

Improvement in Undirected Graphs. In the undirected setting, the result can be improved by taking advantage of the fact that if a good (low-poise) tree covering many terminals is found, then we need only cover *any* node in that tree in order to cover all of those terminals with low poise (as opposed to the directed case where we would have to cover the *root* of that tree). Essentially, we may contract the tree and treat the contracted node as containing many terminals.

Specifically, we will maintain a set R of nodes that we have covered so far with low poise (by contracting, we can think of this simply as the root r). We first group the terminals in the remaining graph $C = V \setminus R$ as before by greedily finding disjoint trees of low poise, now each containing $t^{1/3}$ terminals, called *small trees*. Note that some terminals may not lie in any small tree. If the algorithm finds fewer than $t^{1/3}$ small trees, then the same matroid-constrained covering procedure from above can be applied to immediately get an *additive* $O(t^{1/3})$-approximation.

On the other hand, if there are many small trees, we show that progress can be made by either covering or discarding a large number of terminals at once. If we are able to aggregate $t^{1/3}$ small trees within a distance D^*, we have covered $t^{2/3}$ terminals and hence made sufficient progress in coverage: we can repeat this at most $t^{1/3}$ times to finish. However, we may have the additional complexity of the optimal tree containing terminals that are not in one of these small trees we computed in C. We handle this case by using the matroid procedure to extract as many terminals as any optimal solution might cover from the small trees, and then discarding all the terminals from *all* of the unused small trees. Since the number of small trees (each with $t^{1/3}$ terminals) is $\Omega(t^{1/3})$, this allows us to bound the number of such discarding iterations by $O(t^{1/3})$. In summary, we employ $O(t^{1/3})$ iterations of either covering or discarding $t^{2/3}$ terminals in the algorithm leading to the claimed $O(t^{1/3})$ multiplicative guarantee. Over the course of these iterations, the total degree accumulated by any node will be at most $\tilde{O}(t^{1/3}) \cdot B^*$ (Note this guarantee is now multiplicative, since a node can gain $\tilde{O}(B^*)$ degree in each of the $t^{1/3}$ covering iterations).

Finally, we remark that the improved guarantee in this setting is in terms of t, the total number of terminals, rather than k. This is because our algorithm relies on removing a large number of terminals from the entire set of t terminals, without necessarily covering all of them.

2 Preliminaries

Let $\mathrm{dist}(u, v)$ denote the number of edges in the shortest path from u to v in G. We denote by $G(U)$ the graph induced by U, and by $\mathrm{dist}_{G(U)}(u, v)$ the distance from u to v in the graph $G(U)$. Recall that we denote the minimum poise tree by T^*, its maximum degree by B^*, and its height by D^*.

Assumption 2.1. *Removing vertices of distance more than D^* from the root r in G does not change the optimal solution. Hence, we will assume for the rest of the paper that G only contains vertices of distance at most D^* from r.*

Remark 2.2. *For the rest of the paper, we assume that quantities such as $\sqrt{k}$ are integral. Making the algorithm precise requires using $\lceil\sqrt{k}\rceil$. However, the changes are minimal and elementary.*

For simplicity, we assume that every terminal has in-degree 1 and out-degree 0, by attaching new terminal vertices to every terminal (this only increases the poise by at most an additive constant). For undirected graphs, we assume that terminals have degree 1. Therefore, removing terminals can't turn a connected graph into a disconnected graph.

The input for the Set Cover problem is a universe $\mathcal{U}$ and a collection $\mathcal{S}$ of sets $S_i \subseteq \mathcal{U}$. We say that a set S_i *covers* all the elements that belong to this set. The goal is to find a a sub-collection of sets $\mathcal{S}' \subseteq \mathcal{S}$ of minimum size that covers all elements, namely, $\bigcup_{S_i \in \mathcal{S}'} S_i = \mathcal{U}$. The Set Coverage problem under matroid constraints has the input of Set Cover, and in addition, a matroid $\mathcal{M}$ defined over the sets $\mathcal{S}$. The goal is to select an independent set $\mathcal{I}$ in the Matroid so that $|\bigcup_{S_i \in \mathcal{I}} S_i|$ is maximum. A partition matroid instance divides $\mathcal{S}$ into pairwise disjoint collections of sets $\mathcal{S}_i$, whose union is all of $\mathcal{S}$. For every collection $\mathcal{S}_i$, there is a bound p_i on the number of sets that can be selected from $\mathcal{S}_i$. A collection of sets containing at most p_i sets from each $\mathcal{S}_i$ is precisely an independent set in the partition matroid. The goal is to find an independent set in the partition matroid that covers the largest number of elements. This problem is a special case of maximizing a submodular function under matroid constraints and admits a polynomial time $1-1/e$-approximation [1]. The procedure of [1] is one of the main tools in our algorithm. We called this procedure the `Matroid` procedure.

3 The Partition Matroid Cover Algorithm

In the next two sections, our algorithms for both the directed and undirected cases define a disjoint partition of the graph vertices into $A \cup C = V$. The root r always belongs to A, and we will ensure that all of A can be covered by a low poise tree rooted at r. In this section, we discuss how to cover sufficiently many terminals from C with low poise by connecting them to the root through A. We do this by defining an instance of the Set Coverage problem under a partition matroid constraint[4].

Definition 3.1. *Define a Set Coverage instance as follows.*

- *The items are $S \cap C$ (the terminals in C).*

[4]Note that the parameter k represents the *remaining* number of terminals we need to cover. Given a partition A, C we will assume that all terminals in A have been spanned, and thus we need to cover k terminals in C. That is, if A has k_1 terminals for some $k_1 < k$, we will set $k \leftarrow k - k_1$. Note that we are guaranteed that $C \cap T^*$ contains at least $k - k_1$ terminals supplying a feasible solution.

- *The sets (also called* pairs*) are* $\mathcal{S} = \{(a,c) \mid a \in A, c \in C, \text{ and } ac \in E\}$ *where* (a,c) covers *a terminal* $t \in S \cap C$ *if* $\text{dist}_{G(C)}(c,t) \leq D^*$*.*

The partition matroid is defined as follows.

Definition 3.2. *$\mathcal{S}$ is partitioned into collections*

$$X(a) = \{(a,c) \mid c \in C \text{ and } ac \in E\}$$

for every $a \in A$. The bound on the number of sets to be chosen from $X(a)$ is B^*.

By definition, the partition is disjoint and therefore, we have a valid partition matroid. Recall that $r \in A$. We use the following procedure.

Procedure `PMCover`(A, C).

1. $\mathcal{E}' \leftarrow \emptyset$, $S' \leftarrow S \cap C$.
2. While $k > 0$ do:
   (a) Define the partition matroid Set Coverage instance from A, C, S' as above with sets $\mathcal{S}'$ and apply Procedure `Matroid` of [1] to find an independent set of approximately maximum coverage. Let $\mathcal{I}$ be the independent set it returns.
   (b) $\mathcal{E}' \leftarrow \mathcal{E}' \cup \mathcal{I}$
   (c) Decrease k by the number of terminals covered by $\mathcal{I}$
   (d) Remove the terminals covered by $\mathcal{I}$ from S'.
3. Returns $\mathcal{E}'$

Analysis

We will show that for every $a \in A$, $|X(a) \cap \mathcal{E}'| \leq O(\log k) \cdot B^*$. This will be used to argue that if $(a,c) \in \mathcal{E}'$, we later may make a the parent of c in the tree we build without incurring high degree.

Definition 3.3. *Define a mapping from terminals in $T^* \cap C$ to $\mathcal{S}'$ as follows. For a terminal t, let $a = a_t$ be the vertex $a \in A$ that is an ancestor of t in T^* and among them $\text{dist}_{T^*}(a,t)$ is minimum. This vertex is well defined since $r \in A$ is the root of T^*. Let $c = c_t$ be the child of a in T^* that is an ancestor of t. Define $f(t) = (a,c)$.*

Claim 3.4. *There exists an independent set $\mathcal{I}^*$ in the partition matroid that covers at least k terminals in $C \cap S$.*

Proof. We show that every terminal in $t \in T^* \cap C$ is covered by some set. Let $a = a_t$ and let $c = c_t$. Since a has minimum distance to t from all vertices in A, the path from c to t belongs to $G(C)$. The number of edges in the path between c and t is at most $D^* - 1$. This implies that the set (a, c) covers t. Create a set $\mathcal{I}^* = \{f(t) \mid t \in T^* \cap S \cap C\}$. We note that $f(t) = f(t') = (a, c)$ may hold for two different terminals, but $\mathcal{I}^*$ includes every such pair (a, c) once (namely, $\mathcal{I}^*$ is a set and not a multiset). For any $a \in A$, the number of different pairs of the form $(a, c_1), (a, c_2), \ldots$ in $\mathcal{I}^*$ can't be more than B^*, because every such pair increases a's degree in T^* by 1. Thus, $\mathcal{I}^*$ is independent in the partition matroid. Since all terminals in $T^* \cap C$ are covered, k terminals are covered. $\square$

Claim 3.5. *Procedure* `PMCover` *returns a collection of pairs $\mathcal{E}'$ so that for every $a \in A$, $X(a) \cap \mathcal{E}' = O(\log k) \cdot B^*$ and $\mathcal{E}'$ covers k terminals. Thus if in some tree, vertex $a \in A$ is made the parent of all c for which $(a, c) \in \mathcal{E}'$, the degree of a will be bounded by $O(\log k) \cdot B^*$.*

Proof. Since Procedure `Matroid` returns an independent set in the partition matroid, at every iteration we have $|X(a) \cap \mathcal{I}| \leq B^*$. Claim 3.4 and the guarantee of Procedure `Matroid` by [1] imply that $(1 - 1/e)k$ terminals are covered. Let $k_{\text{or}} \leq k$ be the original number of terminals to be covered and k_{new} the number of terminals to be covered in a given iteration. Then in the next iteration,

$$k_{\text{new}} \leftarrow k_{\text{new}} - (1 - 1/e)k_{\text{new}} = \frac{k_{\text{new}}}{e}.$$

Therefore, after i iterations, k_{or}/e^i terminals remain to be covered. Hence, the number of iterations is $O(\log k)$. The claim follows. $\square$

4 Approximating the poise for directed graphs

Our algorithm maintains a set A (initialized with the root r) containing the terminals covered with low poise so far, and $C = V \setminus A$. Consider a set C and the graph $G(C)$ induced by C.

Definition 4.1. *A vertex $c \in C$ is ρ-good (with respect to C) if there are at least ρ terminals in C of distance at most D^* from c in $G(C)$. A ρ-good tree is a tree rooted at some c with* exactly *ρ terminals and height at most D^*.*

By assumption, the out-degree of terminals is 0. Therefore all terminals are leaves. Since we may discard non-terminal leaves, a ρ-good trees contains exactly $\sqrt{k}$ leaf terminals.

Definition 4.2. *A set C of vertices, is a ρ-packing if there is no ρ-good vertex in C.*

Definition 4.3. *Let $\{T_i\}$ be a collection of vertex disjoint trees and let A be the set of vertices in $\bigcup_i T_i$. Let $C = V - A$. Then A, C is a ρ-additive partition if:*

1. *The trees T_i are ρ-good with respect to V, and are all vertex-disjoint.*

2. *There are at most ρ trees T_i.*

3. *C is a ρ packing.*

Let q_i be the root of T_i. Intuitively, since there are at most ρ trees T_i, we can add a shortest path P_i from the root r to each q_i, giving a tree rooted at r with low poise covering terminals in A. In addition, since C is a ρ-packing, at least k (meaning the number of *remaining* terminals to cover after covering those in A) of C's terminals can be covered with some collection of low poise trees. In particular, for each $c \in C$, we will denote by $T(c)$ the tree in $G(C)$ formed by taking a shortest path from c to every terminal within distance D^*. Since every $c \in C$ is not ρ-good, all such trees have max degree at most ρ.

The algorithm attempts to find a ρ-additive partition. It greedily finds ρ-good trees, and removes them until the set C that remains is a ρ-packing. Then the procedure `PMCover` can be used to connect the low poise trees covering A and C. However, there may be too many ρ-good trees in A for (A, C) to be a ρ-additive partition. In this case, it simply connects the root to any ρ of the trees T_i. By choosing $\rho = \sqrt{k}$, this ensures enough terminals are covered. The algorithm is as follows.

Procedure `Directed`(G, k):

1. Set $\rho = \sqrt{k}$.
2. (*Greedy Packing*) Let $A = \{r\}$, and $C = V - \{r\}$. While C is not a ρ-packing:
   - Find a ρ-good tree T in $G(C)$.
   - Remove the vertices of T from C and add them to A.

   Let $\{T_i\}$ denote the set of ρ-good trees found.
3. (*Many Trees*) If the number of ρ-good trees found is at least ρ, then:
   - Choose any ρ of the trees $\{T_i\}$ in A, and form the subgraph $H \subseteq G$ by including the root r, the chosen trees, and a shortest path from r to the root q_i of each chosen tree T_i.
   - Return a shortest path tree of D rooted at r.
4. (*Few Trees*) Otherwise, the number of ρ-good trees found is at most ρ. So (A, C) forms a ρ-additive partition. Apply the Procedure `Complete` on (A, C) and return the resulting tree.

In the case that a ρ-additive partition (A, C) is found, we use the following algorithm. See Figure 1 for a depiction of the algorithm at this step.

Procedure `Complete`(A, C):

1. Apply the procedure `PMCover` on (A, C) to get $\mathcal{E}'$.
2. Let $\mathcal{E} = \{ac : (a, c) \in \mathcal{E}'\}$, the set of arcs corresponding to sets chosen by *Cover*.

3. Form the graph H_C on vertex set $C \cup \{r'\}$, where r' is a new node. For each $c \in C$ appearing in some $ac \in \mathcal{E}$, include in H_C the arc (r', c) and the tree $T(c)$ (this is the tree in $G(C)$ formed by taking a shortest path from c to every terminal within distance D^*). Take a shortest path tree on H_C rooted at r', and let T_C be all of the edges from this tree in $G(C)$.

4. Form the subgraph $H \subseteq G$ by including the root r, each ρ-good tree T_i from A and a shortest path from r to its root q_i, the edges from $\mathcal{E}$, and the edges from T_C.

5. Return a shortest path tree of H rooted at r.

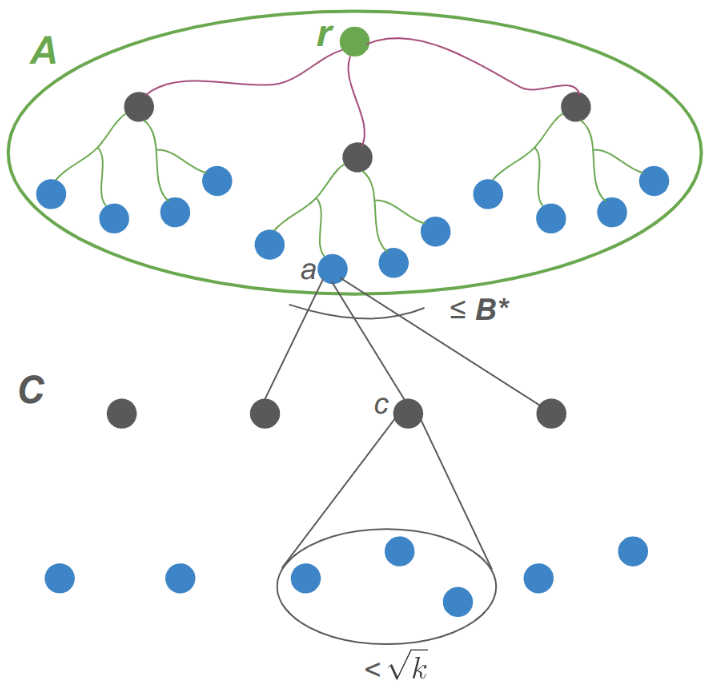

Figure 1: A depiction of the algorithm in the case that a ρ-additive partition is found. The set A includes the root r and all $\sqrt{k}$-good trees found, while C contains the remaining vertices. Terminals are depicted in blue. Short paths from r to the roots of the good trees are added (in red). Since C is a $\sqrt{k}$-packing, each vertex $c \in C$ can reach less than $\sqrt{k}$ terminals within distance D^*. Hence, we can run the `PMCover` procedure, with each iteration enforcing a degree constraint of B^* on each node in A, as shown.

Analysis

For a directed tree, T, let $\deg_T(v)$ be the (out-)degree of the vertex in T. Now say that we run step *Greedy Packing* of `Directed` with $\rho = \sqrt{k}$.

Claim 4.4. *If Procedure* ***Directed*** *finds at least ρ ρ-good trees, then step Many Trees of Procedure* ***Directed*** *returns a tree with at least k terminals, maximum degree $O(\sqrt{k})$, and height $O(D^*)$*

Proof. Since each tree T_i is $\sqrt{k}$-good, it contains $\sqrt{k}$ terminals. Hence, the graph H contains at least k terminals, each of which can be reached by a path from the root. So the returned shortest path tree of H has at least k terminals, as desired

To bound the degrees in the returned tree, we just bound the degrees in H. The good trees T_i are disjoint, and each have maximum degree at most $\sqrt{k}$. Moreover, there are $\sqrt{k}$ of them, so there are only $\sqrt{k}$ shortest paths to their roots. Therefore, the degree contributed to any node $v \in H$ is at most $\sqrt{k}$ from the T_i, and at most 1 for each shortest path, for a total of $\deg_H(v) \leq 2\sqrt{k}$. Finally, each tree T_i in H has height at most D^*, while each shortest path from the root to some q_i has length at most D^* (by Assumption 2.1), so the returned shortest path tree has height at most $2 \cdot D^*$. □

Claim 4.5. *If Procedure* ***Directed*** *finds less than ρ ρ-good trees, then Procedure* ***Complete*** *finds a tree rooted at r with maximum degree $O(\log k) \cdot B^* + O(\sqrt{k})$, and height $O(D^*)$ that that contains at least k terminals of $C \cap S$.*

Proof. First, observe that H contains all terminals in A, as well as those terminals in C covered by procedure `PMCover`. In particular, by Claim 3.5, H contains at least k terminals, so the returned shortest path tree does as well.

Now we bound the degrees of nodes in the returned tree. The T_i making up A are disjoint $\sqrt{k}$-good trees each having maximum degree at most $\sqrt{k}$. And there are less than $\sqrt{k}$ of them, so we add at most $\sqrt{k}$ shortest paths to their roots q_i. Hence, for each node $v \in A$, the contribution to the degree $\deg_H(v)$ is at most $\sqrt{k}$ from the T_i, at most 1 for each shortest path, plus the contribution from $\mathcal{E}$. By Claim 3.5, the edges of $\mathcal{E}$ increase the degree of vertices in A by $O(\log k) \cdot B^*$, so in total $\deg_H(v) \leq O(\log k) \cdot B^* + 2\sqrt{k}$ for each $v \in A$.

All other vertices in H lie in C, and so their degree comes only from the $\sqrt{k}$ shortest paths (contributing at most 1 each), and the edges from T_C. Every tree $T(c)$ has depth at most D^* by definition. In particular, for any vertex $c \in C$, we must have $\deg_{T_C}(c) \leq \sqrt{k}$, since otherwise the subtree of T_C rooted at c has more than $\sqrt{k}$ leaves, which can all be assumed to be terminals. But this means that c has more than $\rho = \sqrt{k}$ terminals in C of distance at most D^*, contradicting that c is not ρ-good. Hence, $\deg_H(c) \leq 2\sqrt{k}$ for every $c \in C$.

Finally, the height of the output tree is at most $3 \cdot D^* + 1$, because we get height D^*, from the trees T_i, height D^* from the shortest paths, height D^* from T_C, and an additional edge from $\mathcal{E}$. □

Therefore, in either case we return a tree with at least k terminals with maximum degree $O(\log k) \cdot B^* + O(\sqrt{k})$ and height $O(D^*)$. This implies Theorem 1.2.

The following corollary is useful as it applies in case that the *Greedy Packing* step of Procedure `Directed` finds a ρ-additive partition (i.e., step *Few Trees* is executed) with some ρ that may be smaller than $\sqrt{k}$.

Corollary 4.6. *If Procedure* `Directed` *finds a ρ-additive partition A, C, then there exists polynomial time ρ-additive approximation for the corresponding min poise k-tree problem.*

5 The undirected case

In this section, we provide our $\tilde{O}(t^{1/3})$-approximation algorithm for the Minimum Time Telephone k-multicast problem on undirected graphs with t terminals, proving Theorem 1.3.

Preliminaries. We assume (for convenience) that the root r is a non-leaf node in T^*. Recall that we assume that all terminals have degree 1. We can now assume that after rooting T^* at r, the set of leaves in T^* and the set of terminals in T^* is the same set. Also recall that the height of the tree T^* rooted at r is at most D^*, since the diameter of T^* is at most D^*.

Algorithm outline. The idea in the undirected case is that if a low-poise tree covering many terminals is found, then we need only cover *any* node in that tree in order to cover all of those terminals with low poise (as opposed to the directed case where we would have to cover the *root* of that tree). Essentially, we may contract the tree and treat the contracted node as containing many terminals.

Specifically, we will maintain a set R of nodes we have covered with low poise (by contracting, we can think of this simply as the root r). We first partition the remaining graph $C = V \setminus R$ as before by greedily finding small trees.

Definition 5.1. *We say that a tree is* small size *if it contains exactly $t^{1/3}$ terminals. We say that a tree is* large *if it contains exactly $t^{2/3}$ terminals*

If this procedure succeeds in finding a $t^{1/3}$-additive partition, then we are done by Corollary 4.6. On the other hand, if we fail, we contract these small trees and show how to cover a sufficiently large number of them by either finding a single large tree reaching $t^{1/3}$ of these small trees, or by applying the procedure `PMCover`. In either case, we may then remove *all* of the terminals from these small trees, contract the newly covered nodes into R, and iterate the entire process to cover the remaining terminals. In each iteration, we show the total number of terminals discarded is large, so there cannot be too many iterations, and hence not too much additional degree is incurred.

We first give a simple algorithm that finds trees $\{T_i\}$ each with exactly $t^{1/3}$ terminals (leaves).

Procedure `Small`(G, k)

1. Apply step *Partition* from Procedure `Directed` on G with $\rho = t^{1/3}$. Denote the resulting trees as $\{T_i\}$.
2. If the procedure succeeds in finding a $t^{1/3}$-additive partition, apply Procedure `Complete` on A, C.

3. Else, return $\{T_i\}$

In case that Procedure *Partition* finds a $t^{1/3}$-additive partition A, C, we are guaranteed a $t^{1/3}$-additive ratio from Corollary 4.6. Hence, from now on we assume that Procedure *Partition* gives more than $t^{1/3}$ small trees T_i.

We will proceed to contract each of these small trees into super-terminals. The trees T_i that we compute, are built by step *Greedy Packing* from Procedure `Directed` with $\rho = t^{1/3}$. Hence, they have exactly $t^{1/3}$ terminals/leaves. We contract the terminals of every T_i into a single super-terminal q_i. Denote by $S(T_i)$ the terminals contained in T_i (i.e., those corresponding to q_i). As mentioned in the outline, we have the possibility that the terminals of an optimal tree may only intersect with a few of these super-terminals. We capture this in the following definitions.

Definition 5.2. *We say that q_i is a* true terminal *if $S(T_i) \cap T^* \neq \emptyset$.*

Definition 5.3. *Denote by k' the number of terminals in $(\bigcup_i T_i) \cap T^*$. Let $\mu = \lceil k'/t^{1/3} \rceil$.*

From the definitions, we can see that T^* overlaps with at least μ true terminals.

In the graph where the small trees have been contracted to super-terminals, we will attempt to find a $t^{1/3}$-packing of these super-terminals. For this, we generalize the definition of a $t^{1/3}$-packing in the set C with respect to the super-terminals.

Definition 5.4. *We say that $c \in C$ is a $t^{1/3}$-good vertex with respect to the super-terminals $\{q_i\}$ if there are at least $t^{1/3}$ terminals q_i of distance at most D^* from c, in $G(C)$. If there are no $t^{1/3}$-good vertices in C, C is called a $t^{1/3}$-packing with respect to $\{q_i\}$. If C is a $t^{1/3}$-packing, then R, C is called a $t^{1/3}$-additive partition with respect to $\{q_i\}$.*

We can now describe the details of the rest of the undirected algorithm. Specifically, if Procedure `Small` fails to find a $t^{1/3}$-additive partition, then there are two possibilities. Either $C = V \setminus R$ is a $t^{1/3}$ packing with respect to the q_i, or otherwise there is a $t^{1/3}$-good vertex in C.

If C is a $t^{1/3}$-packing we apply Procedure `PMCover` on R, C with terminals $\{q_i\}$ since R, C is a $t^{1/3}$-additive partition. The goal is covering μ super-terminals. We know that T^* covers at least μ true terminals q_i, so these can be reached with height D^* and maximum degree B^*. Therefore, our Procedure `PMCover` covers at least μ super-terminals. Note that the number of original terminals we actually cover is $\mu \cdot t^{1/3} \geq k'$. This follows because each q_i represents a tree T_i that contains $t^{1/3}$ terminals. We now discard all the terminals of $\bigcup_i T_i$. Since the number of T_i is at least $t^{1/3}$, the total number of discarded terminals is $t^{2/3}$.

The other case is that C is not a $t^{1/3}$-packing with respect to $\{q_i\}$. Let $v \in C$ be a $t^{1/3}$-good vertex and let Q_v be the corresponding tree. Note that Q_v is a large tree since it spans $t^{1/3}$ of the q_i, each representing $t^{1/3}$ terminals. We connect r to Q_v via a shortest path P from r to Q_v, and contract $r \cup P \cup Q_v$ into r. Then we discard the terminals of Q_v. Since Q_v is a large tree, the number of terminals discarded is $t^{2/3}$.

In summary, in both cases $t^{2/3}$ terminals are discarded. Therefore the number of iterations in our algorithm is at most $t^{1/3}$.

The degree of vertices in R increases by $O(\log k) \cdot B^*$ every time `PMCover` is applied. Alternatively, a large tree Q_v is created and we only need a path P from r to Q_v. This increases the degree of some vertices in R by exactly 2. This gives a total degree of $2 \cdot t^{1/3}$ because of the bound on the number of iterations.

The main procedure

`Undirected`(G, k)

1. $R \leftarrow \{r\}$, $S' \leftarrow S$.
2. While $k > 0$ do:
   (a) Apply Procedure `Small` with $\rho = t^{1/3}$ on $C = V \setminus R$. If it succeeds, return the resulting tree.
   (b) If `Small` fails, contract the terminals from each T_i in the resulting packing into a corresponding super-terminal q_i.
   (c) If $C = V \setminus R$ is not a $t^{1/3}$-packing with respect to $\{q_i\}$ do:
      i. Find a large tree Q_v inside $G(C)$.
      ii. Compute a shortest path P from r to Q_v.
      iii. $R \leftarrow R \cup P \cup Q_v$.
      iv. Remove from S' all the terminals of Q_v and update k.
   (d) Else, $C = V \setminus R$ is a $t^{1/3}$-packing.
      i. Apply Procedure `PMCover` with $A = R$ and $C = V - R$ with the goal of covering super-terminals.
      ii. Let $\mathcal{E}$ be the edges corresponding to the returned sets $(a, c) \in \mathcal{E}'$, and T_C the shortest path tree on the corresponding trees $T(c)$ (as in line 3 of `Complete`). Write $Q = \mathcal{E} \cup T_C$.
      iii. $R \leftarrow R \cup Q$.
      iv. Remove from S' all the terminals $\bigcup_i T_i$ and update k.
3. Return the tree induced by R.

Analysis

Claim 5.5. *T^* contains at least μ true terminals.*

Proof. If the number of true terminals is at most $\mu - 1$, the number of terminals in $\bigcup_i T_i$ is at most $(\mu - 1) \cdot t^{1/3} < k'$ and this is a contradiction. □

Claim 5.6. *The number of iterations in Procedure `Undirected` is at most $t^{1/3}$.*

Proof. If a tree Q_v is found, then it is a large tree hence it contains at least $t^{2/3}$ terminals. These terminals are discarded in the iteration. Else, the terminals of $S \cap \bigcup_i T_i$ are discarded and this, again, this removes $t^{2/3}$ terminals, since we have at least $t^{1/3}$ different small T_i's (because procedure `Small` failed). Since in either case $t^{2/3}$ terminals are discarded and the total number of terminals is t, the number of iterations is at most $t^{1/3}$. $\square$

Claim 5.7. *Let v be a vertex so that $v \notin R$. A single iteration of Procedure* `Undirected` *increases v's degree by at most $2 \cdot t^{1/3} + 2$. Moreover, if v's degree increases, v is contracted into r in that iteration.*

Proof. The degree of a vertex increases only if it belongs to a large tree Q_v (or its path P from r), or it belongs to the subgraph Q computed by Procedure `PMCover`. In the first case, the degree increases by at most $t^{1/3}$ from any one of the T_i's in Q_v, at most $t^{1/3}$ more for the paths from v to these T_i's, and at most 2 more for the path P from r to Q_v, for a total of at most $2 \cdot t^{1/3} + 2$. In the second case, the degree increases by at most $t^{1/3}$ from T_C and 1 from $\mathcal{E}$, by an identical argument to Claim 4.5 (the proof of correctness of Procedure `Complete`).

In both cases, the vertex is immediately contracted into r. $\square$

Claim 5.8. *At every iteration, the degree of vertices in R is increased by at most $O(\log k) \cdot B^*$.*

Proof. If a large tree Q_v is found, a shortest path from r to Q_v is computed. This increases the degree of any vertex by at most 2. Otherwise, Procedure `PMCover` is applied. This increases the degree of vertices of R by $O(\log k) \cdot B^*$. The claim follows. $\square$

Claim 5.9. *The returned tree contains k terminals, has maximum degree $\tilde{O}(t^{1/3}) \cdot B^*$ and diameter $O(D^*)$*

Proof. By Claim 5.7, an iteration of Procedure `Undirected` increases the degree of a vertex $v \notin R$ by at most $O(t^{1/3})$, any v whose degree increases immediately joins R. Now we bound the degree added to a vertex in R. By Claim 5.8 at every iteration the degree of $v \in R$ can increase by $O(\log k) \cdot B^*$. By Claim 5.6, the number of iterations of Procedure `Undirected` is is bounded by $t^{1/3}$. Therefore the total degree of a vertex is at most

$$O(t^{1/3}) + O(\log k) \cdot B^* \cdot t^{1/3} = O(\log k) \cdot t^{1/3} \cdot B^*.$$

In addition, the diameter of every Q_v or Q found, is $O(D^*)$. The distance of r to any Q or Q_v is at most D^* as well. This assures that the diameter is $O(D^*)$.

Finally, we argue that k terminals are covered. Fix a particular iteration of the algorithm. If Procedure `Small` succeeds in finding a $t^{1/3}$-additive partition, then we immediately cover the remaining number of terminals necessary by applying the Procedure `Complete`. Otherwise, we argue that among the terminals discarded in this iteration, the algorithm covers at least as many as T^* covers. Indeed, if C is not a $t^{1/3}$-packing with respect to $\{q_i\}$, then all terminals discarded are covered. On the other hand, if C is a $t^{1/3}$-packing, then by applying Procedure `PMCover`, Claim 5.5 ensures that at least μ super-terminals are covered. Hence at least $\mu \cdot t^{1/3} \geq k'$ terminals are covered, which is precisely the number of terminals covered by T^* among those discarded. $\square$

Using [16] we get the following corollary that proves Theorem 1.3.

Corollary 5.10. *The Minimum Time Telephone k-Multicast problem on undirected graphs admits a polynomial time, $\tilde{O}(t^{1/3})$-approximation algorithm.*

References

[1] G. Călinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011. 3, 4, 6, 7, 8

[2] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, pages 1–12, 1987. 1

[3] M. Elkin and G. Kortsarz. Combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem. *SIAM journal on Computing*, 35(3):672–689, 2002. 1

[4] M. Elkin and G. Kortsarz. An approximation algorithm for the directed telephone multicast problem. *Algorithmica*, 45(4):569–583, 2006. 1, 3

[5] M. Elkin and G. Kortsarz. A sublogarithmic approximation algorithm for undirected telephone multicast. *Journal of Computing and System Sciences*, pages 648–659, 2006. 1

[6] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. *Technical report, UCLA/CSD-TR 02*, 2002. 2

[7] S. M. Hedetniemi, S. T. Hedetniemi, and A. L. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, pages 319–349, 1988. 1

[8] R. Impagliazzo and R. Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. 2

[9] R. Khandekar, G. Kortsarz, and Z. Nutov. On some network design problems with degree constraints. *J. Comput. Syst. Sci.*, 79(5):725–736, 2013. 3

[10] G. Kortsarz and Z. Nutov. The minimum degree group steiner problem. *Discret. Appl. Math.*, 309:229–239, 2022. 1, 3

[11] G. Kossinets, J. Kleinberg, and D. Watts. The structure of information pathways in a social communication network. In *SIGKD*, pages 435–443, 2008. 2

[12] D. R. Kowalski and A. Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Comput.*, 19(3):185–195, 2007. 2

[13] A. Nikzad and R. Ravi. Sending secrets swiftly: Approximation algorithms for generalized multicast problems. In *ICALP*, pages 568–607, 2014. 2

[14] M. Onus and A. W. Richa. Minimum maximum-degree publish-subscribe overlay network design. *IEEE/ACM Trans. Netw.*, 19(5):1331–1343, 2011. 2

[15] D. Peleg. *Distributed computing: a locality-sensitive approach.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000. 1

[16] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *FOCS*, pages 202–213, 1994. 2, 3, 16