# Approximation Algorithm for Directed Telephone Multicast Problem

Michael Elkin[1] ⋆ and Guy Kortsarz[2]

[1] School of Mathematics, Institute for Advanced Study, Princeton, NJ, USA, 08540.
elkin@ias.edu
[2] Computer Science department, Rutgers University, Camden, NY, USA.
guyk@crab.rutgers.edu

**Abstract.** Consider a network of processors modeled by an $n$-vertex *directed* graph $G = (V, E)$. Assume that the communication in the network is synchronous, i.e., occurs in discrete "rounds", and in every round every processor is allowed to pick one of its neighbors, and to send him a message. The *telephone $k$-multicast* problem requires to compute a schedule with minimal number of rounds that delivers a message from a given single processor, that generates the message, to all the processors of a given set $\mathcal{T} \subseteq V$, $|\mathcal{T}| = k$. The processors of $V \setminus \mathcal{T}$ may be left uninformed.
The telephone multicast is a basic primitive in distributed computing and computer communication theory. In this paper we devise an algorithm that constructs a schedule with $O(\max\{\log k, \frac{\log n}{\log k}\} \cdot br^* + k^{1/2})$ rounds for the *directed $k$-multicast* problem, where $br^*$ is the value of the optimum solution. This significantly improves the previously best-known approximation ratio of $O(k^{1/3} \cdot \log n \cdot br^* + k^{2/3})$ due to [EK03].
We show that our algorithm for the directed multicast problem can be used to derive an algorithm with a similar ratio for the *directed minimum poise Steiner arborescence* problem, that is, the problem of constructing an arborescence that spans a collection $\mathcal{T}$ of terminals, minimizing the sum of height of the arborescence plus maximum out-degree in the arborescence.

## 1 Introduction

Consider a network of processors modeled by an $n$-vertex graph $G = (V, E)$. Assume that the communication in the network is synchronous, i.e., occurs in discrete "rounds", and in every round every processor is allowed to pick one of its neighbors, and to send him a message. The *telephone $k$-multicast* problem requires to compute a schedule with minimal number of rounds that delivers a message from a given single processor, that generates the message, to all the

⋆ This material is based upon work supported by the National Science Foundation under agreement No. DMS-9729992. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

processors of a given set $\mathcal{T} \subseteq V$, $|\mathcal{T}| = k$. The processors of $V \setminus \mathcal{T}$ may be left uninformed. The case $\mathcal{T} = V$ is called *broadcast* problem.

The telephone multicast and broadcast are basic primitives in distributed computing and computer communication theory, and are used as building blocks for various more complicated tasks in these areas (cf. [HHL88]). The optimization variants of the multicast and broadcast primitives were intensively studied during the last decade. Most of the research focused on undirected graphs [BGNS98,KP95,R94,S00,F01]. Several approximation algorithms with a polylogarithmic ratio were suggested for the undirected minimum time multicast problem [BGNS98,R94,EK03].

A related notion is the *poise* of a directed graph [R94]. Let $G$ be a graph and $T$ a spanning arborescence in the graph. The poise of an arborescence is defined to be $maxdeg(T) + h(T)$ with $maxdeg$ being the maximum out-degree in the arborescence and $h(t)$ the height (maximum number of edges in a root to leaf path) of the arborescence. The poise of a graph $G$ is the minimal poise of some spanning tree $T$ of $G$.

### 1.1 Preliminaries

The input in the minimum time directed telephone multicast problem consists of a directed graph $G(V,E)$, a source vertex $s$ and a collection $\mathcal{T} \subseteq V$ of terminals. Throughout the run of the broadcast protocol, the vertices $V$ are split to the *informed* subset $I$ and and the *uninformed* vertices $U$. We denote $UT = U \cap \mathcal{T}$. At the start time of the the broadcast protocol $I = \{s\}$, $U \leftarrow V \setminus \{s\}$ and $UT \leftarrow \mathcal{T}$.

A *round* is a *directed matching* of a subset $X \subset I$ to a subset $Y \subseteq U$. Thus, the matching edges are directed from $X$ to $Y$. After the round, the $Y$ vertices become informed, namely, $I \leftarrow I \cup Y$ and $U \leftarrow U \setminus Y$. More importantly, $UT \leftarrow UT \setminus Y$. Thus, a schedule is an ordered tuple of rounds. The goal is to use the fewest possible number of rounds and inform all of $\mathcal{T}$. Namely, at the end of the protocol we must have $UT = \emptyset$.

Throughout the paper we denote $|\mathcal{T}| = k$, and the optimum value for the instance at hand is denoted by $br^*$. As the number of informed vertices can at most double at every round, $br^* \geq \lceil \log_2 k \rceil$. In addition, since at every round we can inform at least one additional vertex, we may assume that $br^* \leq n - 1$. We assume that the value of $br^*$ is known and can be used by the algorithm. Indeed, the correct $br^*$ value can be found by binary search between the maximum possible value of $n - 1$ and the minimum possible value of $\lceil \log_2 k \rceil$.

Given a subgraph $G'$, the distance (minimum number of edges in a shortest path) between $u$ and $v$ is denoted by $dist(u, v, G')$. Given an arborescence $T$, its height (the largest distance from a root to a leaf) is denoted by $h(T)$. The set of neighbors of $u$ (vertices at distance 1 from $u$) is denoted by $N(u)$.

The graph induced by a set of nodes $U$ is denoted by $G(U)$.

We denote by $deg(v)$ the out-degree of a vertex $v$. The out-degree of $v$ in a subgraph $G'$ is denoted by $deg(v, G')$.

For a vertex $v$ and positive integer $\ell = 1, 2, \ldots$, the *$\ell$-out-neighborhood* in $G$ is the vertex set $\{u \mid d_G(v, u) \leq \ell\}$.

A leaf in a rooted arborescence is a vertex that has no children.

An $(r, t)$ approximation algorithm for a minimization problem $P$ is an algorithm that given an instance $\alpha$ of $P$ returns a solution of value at most $r \cdot opt(\alpha) + t$, where $opt(\alpha)$ is the value of the optimal solution of the problem $P$ on the instance $\alpha$.

*The busy schedule procedure.* One of the tools that are used in our multicast procedure is the well known busy schedule. In the busy schedule, an informed node $u$ considers its set of neighbors. If there exists an uninformed neighbor of $u$, then $u$ chooses an *arbitrary* uninformed neighbor and sends it the message. This schedule is called *non lazy* (the terminology is due to [BGNS98]).

Thus, the busy schedule is the simple greedy strategy that essentially makes sure that whenever possible no informed vertex is "idle".

Finally, we use the following lemma due to [EK02].

**Lemma 1.** [EK02] *Let $Q$ be an arborescence rooted at $s$ with leaf set $L$ and depth $h$. Assume that $s$ knows the message. Then the busy schedule is a multicast scheme to all the vertices in $Q$ in no more than $h + |L|$ rounds.*

### 1.2 Previous Research

The optimization variants of the broadcast and multicast primitives were intensively studied during the last decade. See, for example, [BGNS98,BK94,KP95,M93,R94,S00,F01,EK02,EK03]. Table 1 summarizes the previous research.

| | **Broadcast** | | **Multicast** | |
|---|---|---|---|---|
| | *u.b.* | *l.b.* | *u.b* | *l.b* |
| Directed graphs | $\log n$ [EK02] | $\sqrt{\log n}$ [EK02] | $(k^{1/3} \cdot \log n, k^{2/3})$ [EK03] | $\log k$ [F01] |
| Undirected graphs | $\log n / \log\log n$ [EK03] | 3 [EK02] | $O(\log k / \log\log k)$ [EK02] | 3 [EK02] |

**Table 1.** Known positive and negative approximability results for the telephone broadcast and multicast problems. The positive results appear in columns marked "u.b." (stands for "upper bounds"), and the negative results appear in columns marked "l.b." (stands for "lower bounds").

### 1.3 Our results

For the directed $k$-multicast problem, currently the best-known approximation is $(k^{1/3} \cdot \log n, k^{2/3})$ due to [EK03]. We improve upon this and devise an algorithm that given input with optimal schedule of length $br^*$ constructs a schedule of length $O(br^* \cdot \max\{\log k, \frac{\log n}{\log k}\} + k^{1/2})$ (that is, the approximation is

$(\max\{\log k, \frac{\log n}{\log k}\}, k^{1/2}))$. For the special case of $br^* = \Omega(\sqrt{k})$ (for example, graphs with high diameter) the ratio is $O(\log n)$. It is easy to show that for $k = \Omega(n^\epsilon)$ with $\epsilon > 0$ a (possibly small) constant and $br^* = \Omega(\sqrt{k})$, the achieved ratio (that is, $O(\log n)$) is the best possible up to a constant factor, unless $P = NP$. The running time of our algorithm is $\tilde{O}(n^3)$ (the notation $\tilde{O}$ ignores polylogarithmic factors).

In the more general *postal model* (introduced in [BGNS98]) each vertex $v$ has a delay number $0 \leq \mu(v) \leq 1$. The vertex sending a message is "busy" only at the first $\mu$ time units starting from the initial sending time. After $\mu$ time units, $v$ is free to send another message. In addition, every edge $e$ has a delay number $\ell_e$ representing the time required to send the message over $e$. We extend our algorithm to this more general problem, and show that this problem admits the same approximation ratio.

We also derive the first non-trivial approximation algorithm for the minimum poise Steiner arborescence problem. The approximation ratio that we achieve for this problem is $(O(\log n), O(\sqrt{k}))$. This results can be considered a bicriteria approximation result [HRR+]; Our algorithm accepts as input a graph that has a spanning arborescence of height $h^*$ and maximum out-degree $d^*$, and produces a directed Steiner arborescence of height $O(h^*)$ and maximum out-degree $O(\log n \cdot d^*) + O(\sqrt{k})$.

Comparing our algorithm with the algorithm of [EK03] (the latter algorithm achieves approximation ratio of $(O(k^{1/3} \log n), k^{2/3}))$, our algorithm is significantly simpler. Specifically, the algorithm of [EK03] makes a heavy use of broadcasting through *jungles*. The jungle (introduced in [EK03]) is a collection of not necessarily vertex-disjoint trees, that, nevertheless, possess some other useful properties. Broadcast through jungles was shown in [EK03] to be a useful technique that enables to achieve the only known today *sublogarithmic* approximation ratio for *undirected* $k$-multicast problem. This technique was applied in [EK03] to the *directed $k$-multicast* problem, achieving the first (and the only known prior to the current paper) *sublinear* approximation ratio for this problem. In this paper we show that in the case of *directed $k$-multicast* problem, the jungles can be replaced by ordinary forests, achieving a significantly better approximation ratio, and simplifying the algorithm.

### 1.4 Overview of the Algorithm

The multicast schedule that we define can be roughly divided into two parts. In the first part of the schedule, the goal is to construct a partition of the vertex set $V$ into two disjoint subsets $I$ and $U$. The set $I$ will be referred as the set of *informed* vertices, and the set $U$ will be referred as the set of *uninformed* ones. This partition has to satisfy certain properties. First, there should be a polynomially-computable schedule for informing the vertices of the set $I$. Second, no uninformed vertex (that is, a vertex that belongs to $U$) is allowed to have in its vicinity "many" uninformed terminals (vertices of the set $UT = U \cap \mathcal{T}$). These vicinities are defined with respect to the graph $G(U)$ induced by the vertex set

$U$. These properties will be captured formally by the notion of $\sqrt{k}$ degree-height partition, that will be defined later on.

Roughly speaking, given a partition that satisfies these properties, the schedule that informs $UT$ can be efficiently constructed. In other words, the vertices of the set $UT$ of yet uninformed terminals can be informed within $O(\max\{\log k, \log_k n\}) \cdot br^*$ additional rounds.

Note that the number of rounds used in this part of the schedule is essentially the smallest possible (unless $P = NP$) as indicated by the lower bound of [F01]. However, to construct a complete schedule we still need to inform the vertices of the set $U$. In our schedule this task requires $2\sqrt{k}$ rounds. Moreover, after the first part of our schedule, the number of terminals at distance $br^*$ from $u \in U$ can be as high as $\sqrt{k}$. It follows that any improvement in constructing the subschedule for informing the vertices of the set $U$, would imply an amalogous improvement of the approximation guarantee of the entire algorithm.

In Section 2 we describe the algorithm for constructing the partition with the desired properties. This is done by a greedy partitionining technique that uses minimum distance arborescences.

We then explain how to complete the multicast given the partition $(I, U)$. For this end, we reduce this task to certain generalization of the set-cover problem, that we call *multiple set-cover* problem. We present a logarithmic approximation algorithm for this problem that uses linear programming. We believe that both the multiple set-cover problem and our approximation algorithm for it are of independent interest.

The second part of the algorithm uses the algorithm for the multiple set-cover problem and has two stages: in the first stage, it informs a subset $D \subseteq U$ of vertices, that satisfies that every vertex $u \in UT \setminus D$ has a nearby vertex $w(u) \in D$. In the second stage, the shortest path arborescences are used to complete the task of informing $UT \setminus D$.

In this extended abstract we outline the proof of a slightly weaker approximation guarantee, specifically, $(O(\log n), O(\sqrt{k}))$. Most proofs are omitted.

## 2 Constructing the partition and informing the vertices of $U$

### 2.1 Computing a $\sqrt{k}$ partition

We start with the definition of the $\sqrt{k}$ degree-height partition.

**Definition 1.** *A* $\sqrt{k}$ degree-height partition *(henceforth,* $\sqrt{k}$ partition*) is a partition of $V$ into two disjoint sets $V = I \cup U$, $I \cap U = \emptyset$, $S \in I$, such that in the graph $G(U)$ induced by $U$ no vertex $u \in U$ can reach via a directed path of length at most $br^*$ more than $\sqrt{k}$ terminals in $U \cap \mathcal{T} = UT$.*

Again, intuitively, the vertices of the set $I$ are informed. It remains to inform the terminals of $UT$.

**Intuition:** To understand why a $\sqrt{k}$ partition is useful, assume that we are given a $\sqrt{k}$ partition and that the remaining goal is to inform $UT$.

In Section 3.2 it will be shown that a subset $D$ of $U$ that has the property that each terminal $u \in UT$ has a nearby vertex $v \in D$ (the distance is measured in $G(U)$) can be efficiently informed.

Then, we define a collection of arborescences and use it to send the message to the uninformed terminals of $UT$ via this collection. For each vertex $u \in D$, let $T_u$ denote the arborescence rooted in $u$. The sets of leaves of each multicast arborescence $T_u$ are contained in the set $UT \setminus D$. These properties are used to guarantee that the trees $T_u$ are relatively shallow, and that each of them has at most $\sqrt{k}$ leaves.

*The algorithm:* The following procedure starts with $U = V \setminus \{s\}$ and $I = \{s\}$ and informs more vertices (changing $U$ and $I$) so as to reach a $\sqrt{k}$ degree-height partition $(U, I)$ with all $I$ informed.

We say that $u \in U$ is $\sqrt{k}$-good if its $br^*$-out-neighborhood in $G(U)$ contains at most $\sqrt{k}$ terminals.

**Input:** A graph $G = (V, E)$, a set $\mathcal{T}$ of terminals, and a source $s \in V$.

**Output:** A $\sqrt{k}$ partition $(I, U)$ of $V$.

Procedure $Comp - Par$

1. $I \leftarrow \{s\}$; $U \leftarrow V \setminus \{s\}$;
2. $Roots \leftarrow \emptyset$; /* The roots are later used to inform the new vertices of $I$. */
3. **While** $G(U)$ admits a $\sqrt{k}-$good vertex $u$ **Do:**
   (a) Let $Cl(u)$ be the $\lfloor\sqrt{k}\rfloor$ uninformed terminals closest to $u$, in $G(U)$.
   (b) Let $T_u$ be the shortest path arborescence leading from $u$ to $Cl(u)$ in $G(u)$.
   (c) $I \leftarrow I \cup V(T_u)$, $U \leftarrow U \setminus V(T_u)$, $Roots \leftarrow Roots \cup \{u\}$.
4. Output $(I, U, Roots)$

*Properties of the algorithm:* The following claim is derived directly from the algorithm.

*Claim.* (1) The pair $(I, U)$ output by Procedure $Comp - Par$ is a $\sqrt{k}$ partition. (2) The set $Roots$ has cardinality $|Roots| \leq \sqrt{k} + 2$. (3) It is possible to use $2br^* + 2\sqrt{k} + 2$ rounds or less in order to transform $U = V \setminus \{s\}$, $I = \{s\}$ into a new partition $(I, U)$ so that $I$ is the set of informed vertices and $(U, I)$ is a $\sqrt{k}$ degree-height partition.

# 3 From a $\sqrt{k}$ degree-height partition to a complete schedule

## 3.1 An algorithmic tool: the multiple set-cover problem

Consider the following problem called the multiple set-cover problem. Let $G(V_1, V_2, E)$ be a bipartite graph. Recall that a set $S \subseteq V_1$ is called a *set-cover* of $V_2$ if $N(S,G) = V_2$, namely, every $v_2 \in V_2$ has at least one neighbor in $S$. The minimum set-cover problem is to find a set-cover $S \subset V_1$ of $V_2$ with minimum cardinality.

The multiple set-cover problem is defined as follows.
**Input:** A bipartite graph $G(V_1, V_2, E)$ with $|V_1| + |V_2| = n$. The set $V_1$ is partitioned into a disjoint union of sets $V_1 = \bigcup_{j=1}^{d} A_j$.
**Output:** A set cover $S \subset V_1$ of $V_2$.

**Definition 2.** *For a cover $S$, let the* value *of $S$ be $val(S) = \max\{|S \cap A_i|\}_{i=1}^{d}$.*

**Optimization goal:** Minimize $val(S)$.

Observe that the usual set-cover problem is a special case of the multiple set-cover problem with $d = 1$. The case of general $d$ appears to be somewhat more complex. In particular, it is not clear to the authors whether a natural greedy algorithm achieves a logarithmic approximation ratio. However, it can be shown that randomized rounding [RT87] can be employed to achieve this approximation ratio. Note that this is tight up to a constant factor in view of the logarithmic lower bound of [RS97] on the approximation threshold of the set-cover problem.

**Theorem 1.** *There exists a deterministic polynomial-time $O(\log n)$-approximation algorithm for the multiple set-cover problem.*

### 3.2 An application of the multiple set cover problem for a multicasting task

Next, we present a reduction from the directed multicast problem to the multiple set-cover problem. This reduction, along with our approximation algorithm for the multiple set-cover problem, described in Section 3.1, is used to obtain the approximation algorithm for the directed multicast problem.

*The problem of informing a distance$-br^*$ dominating set $D$:* A *dominating set* $D$ in a directed graph $G$ is a set $D$ so that $D \cup N(D) = V$, that is, a vertex that is not in $D$ must have an incoming edge from a vertex of $D$. A *distance$-br^*$ dominating set* in $G$ is a set $D$ that satisfies that for every $v \in V$ there is a vertex $u \in D$ such that $dist(u,v) \leq br^*$. Finally, $D$ is a distance$-br^*$ dominating set with respect to a subset $W$ if the above condition holds for every vertex in $W$ (that is, the vertices of $W$ are of distance at most $br^*$ from $D$, but the vertices from $V \setminus W$ may be at larger than $br^*$ distance from $D$).

Let $I, U$ be a disjoint partition of $V$ such that the source $s$ belongs to the set $I$. Suppose that all the vertices of the set $I$ are already informed. It remains, therefore, to inform the vertices of $UT = U \cap \mathcal{T}$. We relax this problem as follows. Let $P_{I,U}$ be the problem of informing some distance$-br^*$ dominating set $D$ of $UT$ using minimum number of rounds. This is the first phase in the second part of the algorithm. Informing of the vertices of $UT \setminus D$ once the vertices of $D$ are informed is the second phase.

We show how to use the solution for the multiple set-cover problem to approximate the $P_{I,U}$ problem.

Consider the following bipartite graph $\mathcal{B}_{I,U}(\mathcal{V}_1, \mathcal{V}_2, \mathcal{E})$. Let

$$\mathcal{V}_1 = \{x_{(v,u)} \mid \quad v \in I,\ u \in U,\ (v,u) \in E\} \ .$$

Intuitively, the set $\mathcal{V}_1$ is the set of edges $(v,u)$ of the original graph $G$, with $v$ in $I$, and $u$ in $U$. Observe that, in a sense, each vertex $u \in U$ is "duplicated" many times. That is, if the vertex $u$ has $d$ incoming edges, it will appear in $d$ different vertices $x_{v,u}$ .

Let $A_v = \left\{x_{(v,u)} \mid \quad u \in N_G(v)\right\}$. The disjoint partition of $\mathcal{V}_1$ that is provided as input for the multiple set-cover problem is $\mathcal{V}_1 = \bigcup_{v \in V} A_v$.

The set $\mathcal{V}_2$ is set to $\mathcal{V}_2 \leftarrow UT$ (the set of terminals that are still uninformed).

We now define the edge set $\mathcal{E}$ of the graph $\mathcal{B}_{I,U}(\mathcal{V}_1, \mathcal{V}_2, \mathcal{E})$. There is an edge between a vertex $x_{(v,u)}$ to a vertex $w \in UT$ if there is a directed path of length at most $br^*$ from $u$ to $w$ in $G(U)$. See Figure 1 for an illustration.

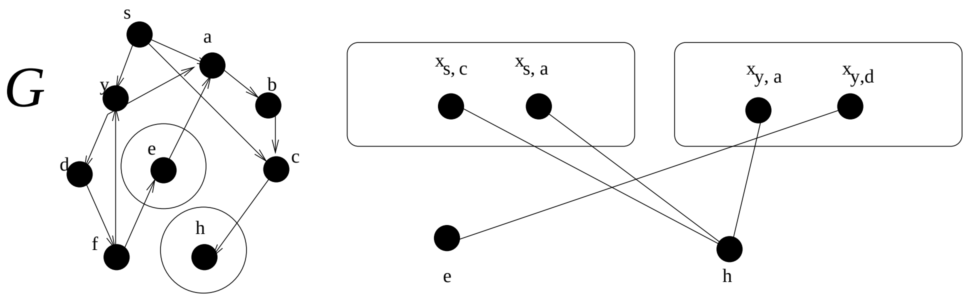


**Fig. 1.** An example of the application of the multiple set-cover problem. The terminals $e$ and $h$ are depicted by cycles around the vertices. It is easy to see that for the graph $G$, $br^* = 4$. The disjoint partition of the vertex set $V$ into $I \cup U$ that is obtained after one round is $I = \{s, y\}$ and $U = V \setminus I$. This partition induces the bipartite graph $B_{I,U}$ that is depicted on the right side. For example, the vertex $x_{s,a}$ is connected to $h$ as the distance between $a$ and $h$ in the graph $G(U)$ induced by $U$ is $3 \leq br^*$. In contrast, the respective distance of $h$ from $d$ is greater than $br^*$. Consequently, there is no edge between $x_{y,d}$ and $h$ in the bipartite graph $B_{I,U}$.

**Lemma 2.** *The multiple set-cover instance $\mathcal{B}$ admits a solution $S^* \subseteq \mathcal{V}_1$ such that $val(S^*) \leq br^*$.*

Now, assume that we compute the instance $\mathcal{B}_{I,U}$ of the multiple set-cover instance of $I$ and $U$ as described above. By Theorem 1 there exists a polynomial-time $O(\log n)$-approximation algorithm for the multiple set-cover problem. Hence, the algorithm returns a solution $D$ for the multiple set-cover problem with value $O(br^* \log n)$.

If $x_{v,u}$ is chosen into the cover, we say that the vertex $u$ is assigned to the vertex $v$. Observe that a vertex $u \in U$ can be assigned many times in the solution. In other words, it may happen that several variables $x_{v_1,u}, x_{v_2,u}, \ldots$ are rounded to 1. In this case it is enough to choose an arbitrary variable $x_{v_i,u}$, and set it to 1, while setting to zero all the others. Thus, each vertex $u$ is assigned to a unique vertex $v_i$.

**Lemma 3.** $\max_{v \in V_1} |D \cap A_v| = O(\log n) \cdot br^*$.

**Lemma 4.** *There is a polynomially computable schedule using $O(\log n \cdot br^*)$ rounds for $P_{I,U}$.*

### 3.3 Informing the set $UT$

We start with a $\sqrt{k}$ degree-height partition $(I, U)$ and elaborate on the following task: how to send the message from $I$ to $UT$ using as small number of rounds as possible. The algorithm has two phases. The first phase applies the solution for $P_{I,U}$. The second phase defines a collection of arborescences in the graph $(G(U)$ induced by $U$ and uses this collection in order to inform the terminals of $UT$.

For simplicity, we are first going to describe a solution for phase 2 that uses a collection of not necessarily disjoint arborescences. Later, we explain a fix that allows to maintain the useful properties of the defined arborescences, while reducing the collection of arborescences into a vertex-disjoint collection.

1. Compute the $P_{I,U}$ instance of $U$ and $I$ and compute its approximate solution $D$.
2. **Phase 1:** Every $v$ such that $D \cap A_v \neq \emptyset$ sends the message in arbitrary order to all the vertices in $\{u \mid \; x_{v,u} \in D \cap A_v\}$.
3. **Assignment Phase:** The assignment is performed by the following procedure called Procedure *Assign*:
   Procedure *Assign*
   (a) Initialize all the vertices of $UT \setminus D$ as "unassigned".
   (b) **While** there is an unassigned vertex in $UT \setminus D$ **do:**
      i. Let $u$ be a vertex in $D$ so that there exists an unassigned vertex $w,\ w \in UT \setminus D$ so that $dist(u, w, G(U)) \leq br^*$.
      ii. **For all** $z \in UT \setminus D$ so that $dist(u, z, G(U)) \leq br^*$, set $\psi(z) = u$ and declare that $z$ is assigned.
4. **Phase 2:** Let $u$ be a vertex that has been assigned to at least one vertex of $UT \setminus D$. Let $T_u$ be the shortest path arborescence leading from $u \in D$ to $W_u = \{w \mid \; u = \psi(w)\}$ in the graph induced by $U$. Let $\mathcal{J}$ be the collection of all arborescences for all the different vertices $u$.
   /* Observe that we are using a non-disjoint collection of arborescences; The arborescences $T_u$ are all computed over the same graph $G(U)$ and are not edge or vertex disjoint. */
   Use the busy schedule to broadcast over $\mathcal{J}$.

The following claim applies.

*Claim.* Phase 1 requires $O(\log n) \cdot br^*$ rounds.

We now consider some of the properties of the *jungle* $\mathcal{J}$

*Claim.* The maximum height of a tree in $\mathcal{J}$ is at most $br^*$.

In addition, by the properties of the $\sqrt{k}$ partition, the number of leaves in $T_u$ is at most $\sqrt{k}$ (as otherwise more than $\sqrt{k}$ of the $UT$ vertices are reachable from $u$ via a path of length at most $br^*$).

As the arborescences are not vertex disjoint, the broadcasting task on one arborescence may conflict with the broadcasting task on other arborescences (since a single vertex is not allowed to send the message to two neighbors in the same round). It, therefore, follows that in the way Phase 2 is described now, the busy schedule may use too many rounds to inform the vertices of $\bigcup T_u$. The following change is used to form a vertex-disjoint collection of arborescences out of the jungle $J$.

Define a level on every vertex in every arborescence in $\mathcal{J}$. The root $u$ of the arborescence $T_u$ is at level 0 and the level of a non-root in $T_u$ is one larger than the level of its parent. For simplicity, we say that every root $u$ has a parent at level $-1$. For every $v$ appearing in several arborescences, let $par^*(v)$ be the parent of $v$ in one of the arborescences, so that the level of $par^*(v)$ is minimum among the level of all the other parents of $v$ in all other arborescences. Define a collection $\mathcal{F}$ of arborescences by taking the directed graph induced by the edges $(par^*(v), v)$ (the edges going from $par^*(v)$ into $v$). Note that some vertices of $U \setminus \mathcal{T}$ may become leaves (namely, have no children). Such vertices are discarded. Moreover, we discard every vertex that has no terminals in its sub-arborescence in $\mathcal{F}$.

*Claim.* The above definition changes the collection of arborescences $\mathcal{J}$ into a forest $\mathcal{F} = \{T'_w\}$, namely, into a collection of vertex disjoint arborescences.

*Claim.* (1) For every arborescence $T'$ in $\mathcal{F}$, there exists a tree $\tau \in \mathcal{J}$ such that $root(\tau) = root(T')$. (2) The height of every arborescence in $\mathcal{F}$ is at most $br^*$. (3) The number of leaves in every $T'_w$ in $\mathcal{F}$ is at most $br^*$.

*Claim.* There exists a procedure that accepts as input a $\sqrt{k}$ degree-height partition $(U, I)$ and outputs a schedule with $O(\log n \cdot br^*) + 2\sqrt{k}$ rounds that informs $UT$. Furthermore, this procedure requires polynomial time.

## 4 The multicast algorithm and its analysis

In this section the pieces are put together and a schedule with $O(\log n \cdot br^*) + 2\sqrt{k}$ rounds is derived.

### 4.1 The algorithm

We start with a formal description of Procedure *Multicast*.

**Input:** A graph $G = (V, E)$, a set $\mathcal{T}$ of terminals and a source $s \in V$.

**Output:** A schedule with $O(\log n \cdot br^*) + 2\sqrt{k}$ rounds.
Procedure *Multicast*

1. Invoke Procedure $Comp - Par$ to derive $(I, U, Roots)$.
2. Use a shortest path arborescence $T$ leading from $s$ to $Roots$. Invoke the busy schedule to send the message from $s$ to $Roots$.
3. For $u \in Roots$, let $T_u$ be the arborescence as defined in Line 3b in Procedure $Comp - Par$. Every $u \in Roots$ broadcasts over $T_u$ (in parallel) using the busy schedule.
4. Use the method described in Section 3.2 to inform a distance$-br^*$ dominating set $D$ of $UT$.
5. Use the forest $\mathcal{F}$ described in Section 3.3 to inform $UT$ using the busy schedule over $\mathcal{F}$.

Combining Claim 2.1, Claim 3.3 and Claim 3.3, we get:

**Theorem 2.** *The directed telephone multicast problem admits an approximation ratio of* $(O(\log n), O(\sqrt{k}))$.

## References


[AS92] N. Alon and J. H. Spencer, The Probabilistic Method, Wiley, 1992.

[BGNS98] A. Bar-noy, S. Guha, J. Naor and B. Schieber. Multicasting in Heterogeneous Networks, In *Proc. of 30th ACM Annual Symp. on Theory of Computing* 1998.

[BK94] A. Bar-Noy and S. Kipnis, "Designing Broadcasting Algorithms in the Postal Model for Message-Passing Systems," *Mathematical System Theory*, Vol. 27, pp. 431–452, 1994.

[C52] H. Chernoff, A measure of asymptotic efficiency for tests of an hypothesis based on the sum of observations, *Annals of Mathematical Statistics*, 23:493-509, 1952.

[EK02] M. Elkin and G. Kortsarz, A combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem, In *Proc. of 34th ACM Annual Symp. on Theory of Computing,* , pp. 438-447, 2002.

[EK03] M. Elkin and G. Kortsarz, A sublogarithmic approximation algorithm for the undirected telephone broadcast problem: a path out of a jungle. In *Proc. of 14th Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 76-85, 2003.

[F01] P. Fraigniaud Approximation Algorithms for Minimum-Time Broadcast under the Vertex-Disjoint Paths Mode, In *9th Annual European Symposium on Algorithms (ESA '01)*, LNCS Vol. 2161, pp. 440-451, 2001.

[FR90] M. Furer and B. Raghavachari. An *NC* approximation algorithm for the minimum degree spanning tree problem. *In Proc. of the 28th Annual Allerton Conf. on Communication, Control and Computing*, pp. 274–281, 1990.

[HHL88] S. Hedetniemi, S. Hedetniemi, and A. Liestman. A survey of broadcasting and gossiping in communication networks. *Networks*, 18: 319-349, 1988.

[HRR+] H. B. Hunt, M. V. Marathe, R. Sundaram, R. Ravi, S. S. Ravi and D. J. Rosenkrantz. Bicriteria network design problems, *Journal of Algorithms*, Vol. 28, No. 1, 142-171 (1998).

[K-79] L. G. Khachiyan. A Polynomial Algorithm in Linear Programming, *Doklady Akademii Nauk SSSR 244*, pp. 1093-1096, 1979.

[K-84] N. Karmarkar, A new polynomial-time algorithm for linear programming, *Combinatorica*, vol. 4 (1984), pp. 373–396.

[KP95] G. Kortsarz and D. Peleg. Approximation algorithms for minimum time broadcast, *SIAM journal on discrete methods*, vol. 8, pp. 401-427, 1995.

[LST-90] L. K. Lenstra and D. Shmoys and E. Tardos, "Approximation algorithms for scheduling unrelated parallel machines", *Math. Programming*, 46, 259-271. (1990).

[KR01] R. Krishnan and B. Raghavachari. The Directed Minimum-Degree Spanning Tree Problem. *FSTTCS 2001*, 232-243.

[M93] M. Middendorf. Minimum Broadcast Time is NP-complete for 3-regular planar graphs and deadline 2. *Inf. Process. Lett.* 46 (1993) 281-287.

[MR95] R. Motwani and P. Raghavan, Randomized Algorithms, *Cambridge University Press*, 1995.

[PST95] S. A. Plotkin and D. B. Shmoys and E. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research* 20, 1995, 257-301.

[R88] P. Raghavan, probabilistic construction of deterministic algorithms: approximating packing integer programs. *Journal of computer and system sciences*, 37:130-143,1988.

[R94] R. Ravi, Rapid rumor ramification: Approximating the minimum broadcast time. In *Proc. of the IEEE Symp. on Foundations of Computer Science (FOCS '94)*, pp. 202-213, 1994.

[RS97] R. Raz, S. Safra. A Sub-Constant Error-Probability Low-Degree Test, and a Sub-Constant Error-Probability PCP Characterization of NP, in *Proc. of the 29th Symp. on Theory of Comp.*, pp. 475-484, 1997.

[RT87] P. Raghavan and C. Thompson, Randomized Rounding, *Combinatorica*, vol. 7, pp. 365-374, 1987.

[S00] C. Schindelhauer, On the Inapproximability of Broadcasting Time, In *The 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'00)*, 2000.