A combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem *

Michael Elkin †‡ Guy Kortsarz§

Abstract

Consider a synchronous network of processors, modeled by directed or undirected graph $G = (V, E)$, in which on each round every processor is allowed to choose one of its neighbors and to send a message to this neighbor. Given a processor $s \in V$, and a subset $T \subseteq V$ of processors, the *telephone multicast* problem requires to compute the shortest schedule (in terms of the number of rounds) that delivers a message from s to all the processors of T. The particular case $T = V$ is called *telephone broadcast* problem.

These problems have multiple applications in distributed computing. Several approximation algorithms with polylogarithmic ratio, including one with logarithmic ratio, for the *undirected* variants of these problems are known. However, all these algorithms involve solving large linear programs. Devising a polylogarithmic approximation algorithm for the directed variants of these problems is an open problem, posed by Ravi in [R94].

We devise a *combinatorial logarithmic* approximation algorithm for these problems, that applies also for the *directed broadcast* problem. Our algorithm has significantly smaller running time, and seems to reveal more information about the combinatorial structure of the solution than the previous algorithms that are based on linear programming.

We also improve the lower bounds on the approximation threshold of these problems. Both problems are known to be 3/2-inapproximable. For the undirected (resp., directed) broadcast problem we show that it is NP-hard (resp., impossible unless $NP \subseteq DTIME(n^{O(\log n)})$) to approximate it within a ratio of $3 - \epsilon$ for any $\epsilon > 0$ (resp., $\Omega(\sqrt{\log n})$).

*A preliminary version of this paper was published in STOC 2002, [EK02].

†Department of Computer Science, Yale University, New Haven, CT, USA, 06520-8285, elkin@cs.yale.edu.
This work was supported by the DoD University Research Initiative (URI) administered by the Office of Naval Research under Grant N00014-01-1-0795.

‡Part of this work was done in School of Mathematics, Institute for Advanced Study, Princeton, NJ, USA, 08540.

§Computer Science Department, Rutgers University, Camden, NY, USA. Email: guyk@crab.rutgers.edu

1 Introduction

Consider a network of processors modeled by a directed or undirected 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 a message to this neighbor. The *telephone broadcast* 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 remaining processors in the network. A more general, *telephone k-multicast* problem accepts as input also a set of terminals $T \subseteq V$ of size $|T| = k$, and requires to compute the shortest schedule that delivers the message to all the processors of T, whereas the processors of $V \setminus T$ may be left uninformed.

The telephone broadcast and multicast are basic primitives in distributed computing and computer communication theory, and are used as building blocks for various more complicated tasks in these areas (see, e.g., [HHL88]). The optimization variants of the broadcast and multicast primitives were intensively studied during the last decade [BGNS98, KP95, R94, S00, F01].

Kortsarz and Peleg [KP95] devised the first approximation algorithm for the *undirected broadcast* problem. Their algorithm constructs schedules that can be longer by $\Theta(\sqrt{n})$ than the optimal schedule for the instance at hand. In a breakthrough paper [R94], Ravi devised an algorithm that provides an $O(\frac{\log^2 k}{\log \log k})$-approximation for the *undirected multicast* problem. This result obviously implies an $O(\frac{\log^2 n}{\log \log n})$-approximation for the undirected broadcast problem. Bar-Noy et al. [BGNS98] improved this result and devised an algorithm that provides logarithmic approximation guarantees for the *undirected* multicast and broadcast problems.

Note that all these algorithms do not apply for the *directed* versions of these problems. The importance of designing algorithms for telephone broadcast on directed graphs was realized already some ten years ago by Ravi, who placed this problem in the first place in the list of open problems in the Conclusion section of [R94].

Also, both known approximation algorithms that provide polylogarithmic guarantees for these problems [R94, BGNS98] use large linear programming (henceforth, LP). To the best of our knowledge, the worst-case running time for solving the linear program of [BGNS98] is no smaller than $\Omega(n^6)$. The algorithm of [R94] is significantly faster (even though it is also an LP-based algorithm), and requires $\tilde{O}(|E||V|^2)$ expected time [1], or $\tilde{O}(|E||V|^3)$ deterministic time. However, its approximation guarantee is super-logarithmic.

In this paper we devise a deterministic *combinatorial logarithmic approximation algorithm* for the telephone broadcast and multicast problems, that applies to the *directed broadcast* problem as well. The worst-case running time of our algorithm is $\tilde{O}(|E||V|)$, which is a significant improvement over the worst-case running times of [BGNS98] and [R94].

We also study the more general *heterogeneous postal model* (see [BGNS98]). In this model each vertex v has a delay $0 \leq \rho(v) \leq 1$. The vertex that sends a message is "busy" at the first ρ time units starting from its sending time. In addition, every arc e has a delay $l(e)$ representing the time required to send the message over e. The broadcast problem in the heterogeneous postal model was studied in [BGNS98], where the authors have shown that their algorithm (based on linear programming) can be adapted to this model, and provides a logarithmic approximation guarantee for the corresponding broadcast problem. We show that our *combinatorial* algorithm also can

[1] We use the notation $\tilde{O}(f(n))$ to denote $O(f(n) \cdot \text{polylog}(f(n)))$.

be adapted to the heterogeneous postal model, and that it provides a logarithmic approximation guarantee for the *directed* and *undirected* versions of the broadcast problem in these models. The running time of the adapted version of our algorithm is $\tilde{O}(|E||V|)$. In addition, we generalize the heterogeneous postal model, and introduce the *arc-dependent heterogeneous postal model* in which the delay of a vertex v depends on the arc it uses to send the message. We believe that this model captures modern communication networks, in which the major components (links and processors) are not homogeneous, more truthfully than the heterogeneous postal model of [BGNS98], and than all the other models that were previously considered in the literature (such as the *postal model* of [BK94] and the logP model of [KSSS93]). We adapt our algorithm to the arc-dependent heterogeneous postal model, and show that it provides a logarithmic approximation guarantee for the *directed* and *undirected* versions of the corresponding broadcast problem. However, this most general version of our algorithm does use linear programming, and has slightly higher running time of $\tilde{O}(n^3)$. No approximation algorithm was previously known for this version of the problem.

From the point of view of the hardness of approximation, the best known lower bound on the approximation threshold for the (directed and undirected) telephone broadcast problems is 3/2 [S00]. In this paper we show that it is NP-hard to approximate the undirected broadcast problem within ratio of $3 - \epsilon$ for any $\epsilon > 0$, and that the directed telephone broadcast problem is $\Omega(\sqrt{\log n})$-inapproximable unless $NP \subseteq DTIME(n^{O(\log n)})$.

Implications for network design: A large body of research deals with network design problems in which the objective is to optimize more than one optimization criteria simultaneously (see, e.g., [MRR+93]). Such optimization problems are called *bicriteria optimization* problems. Consider the problem of constructing a spanning tree of the input graph that optimizes the maximum degree and the depth of the tree simultaneously. We call this problem the "*degree-depth* problem". Ravi [R94] has shown that this problem is closely related to the telephone broadcast problem, and designed the first bicriteria approximation algorithm for it that provides simultaneous polylogarithmic approximation to both the maximum degree and the depth. Defining *poise* of a tree to be the sum between its maximum degree and its depth, this result implies directly a polylogarithmic approximation guarantee for the problem of constructing a spanning tree with minimum poise (henceforth, the *poise* problem). The algorithm of [BGNS98] can be adapted to provide a logarithmic bicriteria approximation for the degree-depth problem, implying, consequently, a logarithmic approximation algorithm for the poise problem.

Our algorithm for the telephone broadcast problem can also be adapted to the degree-depth and poise problems, and it provides logarithmic bicriteria approximation for the former and logarithmic approximation for the latter. As in the case with the telephone broadcast problem, our algorithm is the first to apply to the *directed* versions of both problems, and is the first combinatorial algorithm for these problems. The discussion above about the worst-case running times of the three different approximation algorithms (those of [R94], [BGNS98], and ours) for the telephone broadcast problem is applicable for the degree-depth and poise problems as well.

Consequent developments: After a preliminary version of this paper was published in STOC 2002 [EK02], we published two more papers [EK03a, EK03b] on this subject. In [EK03a] we have shown that the techniques developed in the current paper can be employed to devise a *sublogarithmic* approximation algorithm for the *undirected* versions of the telephone broadcast and multicast problems, as well as for the undirected degree-depth and poise problems. Specifically, the approximation guarantee that is achieved in [EK03a] for the telephone multicast problem is

$O(\frac{\log k}{\log\log k})$. In [EK03b] we studied the *directed multicast* problem, and devised an approximation algorithm with multiplicative approximation guarantee of $O(\log n)$ and additive approximation guarantee of $O(\sqrt{k})$.

2 Preliminaries

2.1 Graphs and Trees

Let $G = (V, E)$ be a directed graph (henceforth, digraph). Given a subset W of V, let $G(W) = (W, E(W))$, $E(W) = \{\langle u, w\rangle \in E \mid u, w \in W\}$, denote the subgraph *induced* by W.

The distance $dist(u, v)$ between u and v is the number of arcs in the shortest path between u and v. For a pair of subsets A, B of vertices, the distance from A to B is the minimum distance from a vertex $u \in A$ to a vertex $w \in B$.

The outdegree $outdeg(v)$ of a node v is the number of arcs leaving v.

For a vertex $u \in V$, let $N(u) = \{v \mid \langle u, v\rangle \in E\}$ denote the set of *neighbors* of u.

For a digraph $G = (V, E)$, let $G' = (V, E')$, $E' = \{(u, w) \mid \langle u, w\rangle \in E\}$ denote the undirected *underlying graph of* G.

Let T be a digraph whose underlying graph is a tree, i.e., an acyclic and connected graph.

Definition 2.1 *A digraph T as above is called an* arborescence*, if there is a vertex $r \in V$ (called* root*) such that for any other vertex $v \in V$ there exists a unique directed path from r to v. The vertex r is called the* root *of the arborescence.*

The depth of an arborescence T, denoted as $h(T)$, is defined by $h(T) = \max_{v \in V'} dist_T(r, v)$. The degree of an arborescence T, denoted $deg(T)$, is the maximum outdegree of a vertex in T, i.e., $\max_{v \in V'} outdeg(v)$. The set of leaves of T, denoted $L(T)$, is the set of the vertices with no outgoing arcs in the directed case, and with degree 1 in the undirected.

A collection of vertex-disjoint arborescences is called *forest*. The *height* (resp., *degree*) of the forest F, denoted $h(F)$ (resp., $deg(F)$), is defined by $h(F) = \max_{T \in F} h(T)$ (resp., $deg(F) = \max_{T \in F} deg(F)$).

2.2 Schedules

At any given moment, the vertices are split into the set of *informed* vertices I, and the set of *uninformed* vertices $U = V \setminus I$. At the beginning $I = \{s\}$. Further, U' denotes the subset of U that contains vertices that still need to be informed; in the broadcast case $U' = U$, while in the multicast case U' may be much smaller.

The message transmission is performed in *rounds* each requiring one time unit. A round is a matching between the subsets I and U with all the arcs of the matching oriented from a vertex in I to a vertex in U. In other words, on each round vertices that belong to a subset of the set of informed vertices send the message through this matching to a subset of vertices from the set U. Let I_i, U_i denote the subsets of informed and uninformed vertices, respectively, at the beginning of round i. The vertices of U that participate in the round (i.e., belong to the vertex set of the matching) become *informed*, and consequently, are removed from the subset U and are added to the subset I.

A *proper* schedule is a collection of matchings $M_1, M_2, M_3, \ldots$, such that M_i is a matching between the subsets I_i and U_i, and $I = \{s\}$ when the schedule starts, and $I = V$ when the schedule ends.

The *length* of a schedule is the number of rounds it uses.

The optimum value: Let *opt* be the minimum number of rounds required for broadcasting from s on the instance at hand. Since the size of I can at most double in each round, it follows that $opt \geq \lceil \log n \rceil$. In addition, $opt \leq n-1$ since at least one additional vertex becomes informed in each round (otherwise, the instance is infeasible). Our algorithm is provided with a guess opt' of the correct value of *opt*, and the algorithm distinguishes between the case that no schedule of length at most opt' exists, and the case when there exists a schedule of length at most $opt' \cdot \log n$. This enables to figure out the correct value of *opt* up to a logarithmic factor via binary search.

The optimum tree: Consider the optimum schedule. This schedule defines a directed tree denoted by T^*. The parent of u in the tree is the vertex that sent the message to the vertex u. Without loss of generality it can be assumed that such a vertex is unique.

The following observation is immediate.

Observation 2.2 *$h(T^*), deg(T^*) \leq opt$.*

Non-lazy schedules: A schedule is called *non-lazy* [BGNS98] if on each round a maximal matching M_i between I and U is used. A schedule of this type uses a simple greedy rule that ensures that the only idle informed vertices will be those that satisfy that each their neighbor is either in I or is informed in the current round by some other vertex from I.

2.3 Spiders

Spiders were shown to be useful for telephone broadcast in [BGNS98].

A set of directed paths $S = \{P_1, \ldots, P_q\}$ all starting at the same vertex v is called a *spider* if the paths are *vertex-disjoint* except for sharing v. The vertex v is called the *head* of the spider.

The *length* of the spider S, denoted $\ell(S)$, is $\max_{P \in S} |P|$.

The *degree* of the spider S, denoted $deg(S)$, is $|S| = q$, i.e., the number of paths in the spider.

The *value* of the spider S, denoted $val(S)$, is $deg(S) + \ell(S) - 1$.

The following observation is immediate.

Observation 2.3 *Using a non-lazy schedule, the head of a spider S can deliver the message to the other spider vertices in $val(S)$ rounds.* ∎

3 Overview of Algorithm

The algorithm is given for the more general problem of multicasting to a subset $\mathcal{T} \subseteq \mathcal{V}$ of *terminals*. The analysis of the upper bound is identical, but since performance is evaluated in terms of *opt*, the minimum number of rounds required for broadcast, instead of the minimum number of rounds for multicasting to $\mathcal{T}$, we obtain logarithmic upper bounds on the approximation thresholds of the directed and undirected broadcast problems, and the undirected multicast problem, but not on the threshold of the directed multicast problem. (In the case of the *undirected* multicast problem there is an easy way to modify the analysis so that the upper bound would be obtained in terms of the multicast optimum. Analogous modification, however, fails in the case of the *directed multicast* problem.)

An important combinatorial structure that will be used in this paper is called a *fork*. Informally, a fork is a pair of "short" vertex-disjoint paths starting with the same vertex v and ending at two uninformed vertices u, w. A path is short if its length is at most *opt*, the minimum number of rounds required for completing broadcast. The vertex v is called the *head* of the fork. As the paths are short and there are only two paths, it is easy to see that if v is informed then it can quickly inform both u and w.

Our algorithm iteratively finds as many vertex-disjoint forks as possible. It maintains a set U corresponding to the set of vertices that are uninformed and have not yet been covered by forks, as well as a set U' of uninformed terminals not covered by forks. The set U' is called a *packing*, if the graph contains no forks. A useful property that our algorithm utilizes is that vertices of a packing can be informed very efficiently.

We next provide a high-level description of the algorithm.

- **The extraction step:** Extract a maximal collection of vertex-disjoint forks from $G(U)$, i.e., keep extracting until $G(U)$ contains no more forks.
- **The recursion step:** Recursively inform the set of the fork heads.
- **The forking step:** Inform the vertices of the forks, now that the heads have been informed. Since the forks are vertex-disjoint and are short, this can be done in parallel.
- **The packing step:** It remains to inform the vertices of U' that remain in U, which now form a packing. This is done by a reduction to a minimal maximum-degree set-cover problem in a bipartite graph. The latter graph has the set of informed vertices on one side and the set U' on the other side. The cover is then found by a single computation of maximum flow.

4 Packing and Short Schedules

4.1 Packing

In this section, we describe one of the main tools of our algorithm. Let $U' \subseteq U$ be a subset of the uninformed vertices. We say that the vertices of U' are *dispersed*, or form a *packing* in $G(U)$, if for any two vertices $v, u \in U'$, every vertex $w \in U$ is of distance more than *opt* from either u or v in $G(U)$.

Formally:

Definition 4.1 *Let $G = (V, E)$ be a digraph, $U' \subseteq U$ be a subset of vertices, and opt be an integer. Then, a (U, U')-*fork triple *is a triple (u, t_1, t_2) with $t_1, t_2 \in U'$ and $u \in U$, such that $u \neq t_2$ and $dist_{G(U)}(u, t_1), dist_{G(U)}(u, t_2) \leq opt$. Also, U' is called a* packing with respect to the set U *if no (U, U')-fork exists.*

Observe that if no (U, U')-fork triple exists, then for each vertex w in U there may exist at most one descendent in the tree T^* that belongs to the subset U'. If we think of w as "covering" its descendents, then U' is a packing in the sense that each vertex of U' is covered by a different unique vertex of U.

A straightforward way for testing the existence of a (U, U')-fork triple in an n-vertex digraph $G = (V, E)$ is to examine all breadth-first-search (BFS) trees in $G(U)$. Such an examination requires $O(|E|n)$ time.

4.2 Mentor vertices

Consider a packing U' with respect to the subset U. We assign to each vertex u of U' a *mentor* vertex $m(u)$, which is the *last* vertex along the directed path between s and u in T^* belonging to I. The definition is valid since such a path starts with $s \in I$ and ends at $u \in U$. Observe that the path in T^* from $m(u)$ to u, contains only vertices from the subset U (except $m(u)$.) Intuitively, the mentors aid in fast transmission of the message to the vertices of the subset U'.

Let $P^*(m(u), u)$ denote the arcs of the path from $m(u)$ to u in T^*. Then the following lemma holds.

Lemma 4.2 *The graph induced by the set $\bigcup_{u \in U'} P^*(m(u), u)$ is a collection of vertex-disjoint spiders. The spider heads belong to I, while other spider vertices belong to U. The value of the forest is at most $2 \cdot opt$.*

Proof: By the definition of $m(u)$, all the vertices that belong to the unique path in T^* from $m(u)$ to u except $m(u)$, belong to the set U. We show that no two paths $P^*(m(z), z)$ and $P^*(m(u), u)$, $z \neq u$, can intersect, unless $m(z) = m(u)$. In the latter case, $m(u) = m(z)$ is the only vertex common to the two paths.

For the sake of contradiction, suppose that some $v \in U$ is contained in paths $P^*(m(u), u)$ and $P^*(m(z), z)$. By the definition of a mentor, the path from v to u and the path from v to z both belong to $G(U)$. In addition, the paths $P^*(m(u), u)$, $P^*(m(z), z)$ are of length at most *opt* (see Observation 2.2). It follows that the distance to u from v in $G(U)$, and also the distance from v to z in $G(U)$, is at most *opt*. This contradicts the assumption that U' is a packing.

Finally, note that both the depth and the maximum degree of the tree T^* are at most *opt*. Hence, $|P^*(m(u), u)| \leq opt$ for every vertex $u \in U'$, and the degree of each spider (which is a subgraph of T^*) is, consequently, at most *opt*. Hence, the value of this forest of spiders is at most $2 \cdot opt$. ▮

It can now be seen that if U' is a packing, not too many vertices can have v as their mentor. Let $\nu(v) = \{u \mid v = m(u)\}$ denote the number of vertices with v as a mentor, for $v \in I$.

Lemma 4.3 *If U' is a packing, then $\nu(v) \leq opt$, for all $v \in I$.*

Proof: Observe that $\nu(v)$ is precisely the degree of the appropriate spider from the forest $\bigcup_{u \in U'} P^*(m(u), u)$. Hence, by Lemma 4.2, $\nu(v) \leq opt$. ▮

4.3 Informing packing

Set covers of low degree: Let $B = (V_1, V_2, A)$ be a bipartite graph. We say that $S \subseteq V_1$ *covers* V_2, if every vertex $v_2 \in V_2$ has a neighbor in S.

Definition 4.4 *We say that a cover S of V_2 has* degree d *if it is possible to assign a neighbor $\psi(v_2) \in S \cap N(v_2)$ to each $v_2 \in V_2$ so that each vertex in S is assigned at most d vertices of V_2. In other words, for each $v \in S$, $|\{v_2 \in V_2 \mid \psi(v_2) = v\}| \leq d$.*

We next present a reduction from the problem of determining whether there exists a cover for V_2 of degree at most L to the maximum flow problem. Orient the edges of the bipartite graph B from V_1 to V_2 and assign unit capacity to each of them. Add a source s with arcs of capacity L to each vertex of V_1 and a sink t with an arc of unit capacity from each vertex of V_2. There exists a cover for V_2 of degree at most L if and only if there exists a flow of capacity $|V_2|$. By the Integrality of the maximum flow, this flow directly translates to a cover. (See Figure 4.3.)

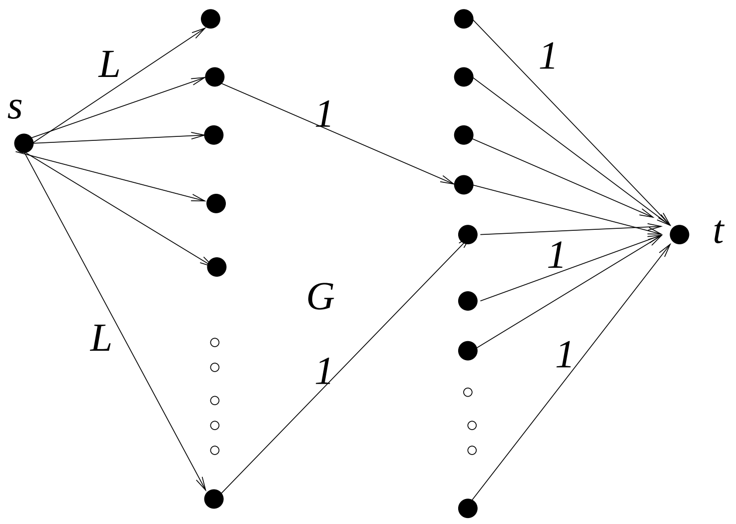

Figure 1: Computing a cover with minimal maximum degree via maximum flow.

To find a cover of minimum degree, we perform binary search for the value of L. Thus, a cover for V_2 with minimal maximum degree can be found in time $O(\log n)T(m, n)$, where $T(m, n) = O(m \cdot n \cdot \log n)$ is the time complexity of computing the maximum flow in an n-vertex graph with m edges [GT88].

The graph H: Consider the following graph $H(I, U', \hat{E})$ that describes all pairs $v \in I$ and $u \in U'$, with v being a potential mentor for u. Formally:

Definition 4.5 *The bipartite graph $H(I, U', \hat{E})$ has $I \cup U'$ as its vertex set. Two vertices $v \in I$ and $u \in U'$ are connected by an undirected edge in $\hat{E}$ if $dist_{G(U \cup \{v\})}(v, u) \leq opt$. In other words, this condition means that there exists a path of length at most opt from v to u that uses only vertices from the set U (except v).*

Low-degree covers for H: Our goal is to show that every packing U' has a low-degree cover in H. Consider the assignment ψ from U' to I defined by $\psi(u) = m(u)$, where $m(u)$ is the mentor of the vertex u. Note that the arc $\langle u, m(u)\rangle$ belongs to $\hat{E}$, and thus the assignment ψ defines a cover. In other words, $M = \{m(u) \mid u \in U'\}$ is a subset of I so that each $u \in U'$ has a neighbor in M (where the neighborhood is with respect to the graph H). By Lemma 4.3, $\nu(v) = |\{u \mid \psi(u) = v\}| \leq opt$. We have proven the following corollary.

Corollary 4.6 *There exists a cover of U' in $H(I, U', \hat{E})$ of maximum degree at most opt, and such a cover can be found in time $O(\log n \cdot T(m, n))$, where $T(m, n) = O(m \cdot n \cdot \log n)$ is the time required for computing a maximum flow in an n-vertex graph with m edges.*

Using a low-degree cover to inform U': Given a cover S of U' with maximum degree at most *opt*, consider the shortest path $P(\psi(u), u)$ from $\psi(u)$ to u in the graph $G(\{\psi(u)\} \cup U)$. The proof of the following lemma is similar to the one of Lemma 4.2.

Lemma 4.7 *The union of the paths $P(\psi(u), u)$ induces a collection of vertex-disjoint spiders, whose heads belong to the cover S and whose values are at most $2 \cdot opt$.*

Proof: The paths $P(\psi(u), u)$ can only intersect at their start vertices, as otherwise a fork triple can be extracted (see the proof of Lemma 4.2). The number of paths in each spider is at most the degree of the cover, which is at most *opt*. In addition, the length of each path in the spider is at most *opt*, by the definition of the graph $H(I, U', \hat{E})$. Hence, the value of each spider is at most $2 \cdot opt$. ∎

Since all the spiders have small values, the following procedure informs a packing U' using a small number of rounds.

Algorithm 4.8 *Inform-Packing(I, U, U')*

1. *Construct the graph $H(I, U', \hat{E})$.*
2. *Find an assignment $\psi : U' \mapsto I$ of degree at most opt that covers H, using flow computation.*
3. *For each $u \in U$, let $P(\psi(u), u)$ be the shortest path from $\psi(u)$ to u in $G(\{\psi(u)\} \cup U)$.*
4. *Each spider head informs all the vertices that belong to the spider using a non-lazy schedule.*

Lemma 4.9 *Let $\mathcal{P}$ be the spider collection induced by the union of the paths $P(\psi(u), u)$, $u \in U'$. Any non-lazy schedule broadcasts over $\mathcal{P}$ in at most $2 \cdot opt$ rounds, informing all the remaining vertices of the set U'.*

Proof: By Lemma 4.7, $\mathcal{P}$ is a union of vertex-disjoint spiders, each of value at most $2 \cdot opt$. Since the spiders are vertex-disjoint, the lemma now follows from Observation 2.3. ∎

4.4 Transforming U' into a packing: the extraction step

The following algorithm accepts as input a subset U of V and the subset U' and deletes vertices from U, so that for the resulting set U at the end of the extraction, no fork triple is left; U' is a packing. The collection of extracted forks is denoted by $\mathcal{F}$, and their heads by $\mathcal{R}$.

In the following algorithm, it is assumed for simplicity that when a fork is found from $u \in U$ to t_1, t_2 in U', the two paths used from u to t_1 and from u to t_2 are found by the BFS procedure (are shortest paths). These two paths may intersect.

Algorithm 4.10 *Extract-forks* (G, U, U')

1. $\mathcal{F} \leftarrow \emptyset$.
2. **While** *there exists a* (U, U')*-fork triple* F *do:*
   (a) *Add* F *into* $\mathcal{F}$.
   (b) $U \leftarrow U \setminus V(F)$, $U' \leftarrow U' \setminus V(F)$, *where* $V(F)$ *is the set of vertices in* F.
3. *return* $(U, U', \mathcal{F})$.

It is easy to see that the resulting forks in $\mathcal{F}$ are vertex-disjoint, and that the returned set U' is a packing with respect to the returned set U.

Observe that if a BFS exploration in $G(U)$ that started from a vertex u discovers that there are no two close to u vertices of U', then the vertex u cannot serve as a head of a fork triple. Therefore, the number of BFS explorations is at most $|V|$, and hence, extracting the fork triples can be accomplished in time $\tilde{O}(|E| \cdot |V|)$.

5 The algorithm: Combining the pieces

The input to the algorithm is the directed graph G, source vertex s, and a set U' of terminals. The algorithm accepts a guess of *opt* as part of its input.

5.1 The algorithm

The algorithm is initially invoked as *ApproximateBroadcast(G,s,V)*, and then it invokes itself recursively with different subsets serving as third parameter. The sequence of these subsets is a monotone decreasing one with respect to containment.

Algorithm 5.1 *ApproximateBroadcast*$(G(V, E), s, U')$

1. $U \leftarrow V \setminus s$, $I \leftarrow \{s\}$.
2. /* *Transforming* U' *into a packing* */
   If U' *is not a* (U, U')*-packing* **then**
   (a) /* *The extraction step* */
   Let $(U, U', \mathcal{F}) \leftarrow$ *Extract-forks*(G, U).
   Let $\mathcal{R}$ *be the set of heads of* $\mathcal{F}$.

(b) /* *The recursive step* */
Recursively invoke ApproximateBroadcast$(G, s, \mathcal{R})$.

(c) /* *The forking step* */
Use a non-lazy schedule to inform the forks in F.
Let $I \leftarrow I \cup V(\mathcal{F})$.

3. /* *The packing step* */
Invoke Inform-Packing(I, U, U').

5.2 Analysis of the number of rounds

The following observation follows from the fact that each fork triple contains at most $2 \cdot opt + 1$ vertices.

Observation 5.2 *Each invocation of the forking step (Line 2c in Algorithm ApproximateBroadcast) requires at most* $2 \cdot opt$ *rounds.*

Lemma 5.3 *The number of rounds used by algorithm ApproximateBroadcast for broadcasting to an arbitrary set* U' *is at most* $4 \cdot (\log_2(|U'|) + 1) \cdot opt$.

Proof: We first show that each time Lines 2c, 3 of Algorithm ApproximateBroadcast are executed, the number of rounds required is at most $4opt$. The two steps that perform the actual broadcast are the fork-extracting and packing steps. The bound of $4 \cdot opt$ per iteration follows from Lemma 4.9 and Observation 5.2.

We now analyze the depth of the recursion. Let U'_i (resp., $\mathcal{R}_i$) be the set of terminals (resp., roots of the forks) on iteration i. Since the forks are vertex-disjoint, and since $|U'_{i+1}| \leq \mathcal{R}_i$, the depth of the recursion is at most $\log_2 |U'| + 1$. Hence, overall, the number of rounds of the constructed broadcast is at most $4 \cdot (\log_2(|U'|) + 1) \cdot opt$. ∎

To summarize,

Theorem 5.4 *Algorithm ApproximateBroadcast is an* $O(\log n)$*-approximation algorithm for the directed telephone broadcast problem.*

Note that the main obstacle to generalizing Algorithm ApproximateBroadcast to the multicast problem is the fact that the set $\mathcal{R}$ is not necessarily contained in the set U'. In the undirected case this obstacle can be overcome by using *fork pairs* instead of fork triples. Intuitively, a fork pair is a pair of nearby terminals of U', that are still not used for other fork pairs. From each fork pair, precisely one terminal is inserted to $\mathcal{R}$, guaranteeing both $|\mathcal{R}| \leq |U'|/2$ and $\mathcal{R} \subseteq U'$. This results in an $O(\log k)$-approximation for the *undirected* k*-multicast problem.*

5.3 The running time of the algorithm

The most time-consuming computational tasks in each iteration are extractions of forks and computations of maximum flow. Recall that the extraction step can be completed with at most $|V|$ BFS computations, in time $O(|V||E|)$, while the time for finding a low-degree cover is $O(|V||E| \log n)$ [GT88]. Since each iteration reduces the number of target terminals by a factor of at least 2, the depth of the recursion is $O(\log n)$. Hence, the time complexity of the algorithm is $O(|V||E| \log^2 n)$. Finding the optimal value of *opt* via binary search adds one more logarithmic factor.

5.4 The postal and arc-dependent postal models

In this section we consider some generalized versions of telephone communication.

The heterogeneous postal model: We start with the *heterogeneous postal* model, introduced in [BGNS98], in which each vertex v has a delay $0 \leq \rho(v) \leq 1$. The vertex that sends a message is "busy" at the first ρ time units starting from its sending time. In addition, every arc e has a delay $l(e)$ representing the time required to send the message over e. (Note that in the context of these generalized models one has to consider a continuous time scale instead of a discrete one, and hence, the notion of "round" becomes obsolete.)

To adapt Algorithm ApproximateBroadcast to the heterogeneous postal model, the paths in forks should be weighted (with weight function determined by the delay function l). In addition, when computing a low-degree cover of U', the delay $\rho(v)$ for $v \in I$ has to be taken into account. A vertex v can have up to $\lfloor opt/\rho(v) \rfloor$ children in the tree T^* (that is, possibly more than opt). Hence, the capacities of the arcs that are adjacent to the source s_1 in the flow graph $H(I, U', \hat{E})$ (see Section 4.3) should be set as $opt/\rho(v)$. Finally, when constructing the arc set $\hat{E}$, the distances between $v \in I$ and $u \in U'$ in the graph $G(\{v\} \cup U)$ should take into account the length of the arcs (are weighted distances).

The rest of the proof can be carried through in an analogous way. Particularly, the cover with bounded degree gives rise to a collection of "narrow" spiders. A vertex v that heads one of these spiders and has degree at most $opt/\rho(v)$ in the spider can deliver the message to all its children in the spider in at most opt time units.

The arc-dependent postal model: We next generalize our algorithm even further. Consider the *arc-dependent heterogeneous postal* model in which the delay of a vertex v depends on the arc through which it chooses to send the message.

Consider again the construction of the flow graph $H(I, U', \hat{E})$. Observe that despite the assumption that the set U' is a packing, even in the standard telephone model for a given pair of vertices $v \in I$, $u \in U'$, there might be more than one shortest path from v to u in the graph $G(U \cup \{v\})$. Let $N(v)$ denote the set of neighbors of v. For each vertex $z \in N(v)$, let P_z denote the shortest path from v to u that passes through the vertex z. Let $\omega(P_z) = \sum_{e \in P_z} l(e)$ denote the *weight* of the path P_z. The edge (v, u) is added to the flow graph $H(I, U', \hat{E})$ if *for some* $z \in N(v)$, the weight of the path P_z plus the delay $\rho(\langle v, z \rangle)$ incurred by the vertex v while sending a message through the arc $\langle v, z \rangle$ is at most opt. (Observe that this is a direct generalization of the construction for the heterogeneous postal model.)

Let $C_u(v) \subseteq N(v)$ be the subset of $N(v)$ that contains only those neighbors z of $N(v)$ for which $\omega(P_z) + \rho(v, z) \leq opt$. Suppose that for every vertex $u \in U'$ that is connected to v in $\hat{E}$ we would pick an arbitrary vertex $z_u \in C_u(v)$ for serving as a relay station for broadcasting the message from v to u (this is exactly what is implicitly done in the telephone and heterogeneous postal models). The weighted degree of the vertex v in the spider $\bigcup\{P_{z_u} \mid u \in U', (v, u) \in \hat{E}\}$ would be $deg(v) = \sum\{\rho(\langle v, z_u \rangle) \mid u \in U', (v, u) \in \hat{E}\}$. Note that unlike the telephone and heterogeneous postal models, in the arc-dependent model $deg(v)$ depends on the choice of the vertices z_u. To tackle this difficulty, we assign to each edge $(v, u) \in \hat{E}$ capacity $\rho(\langle v, z_u \rangle)$, where z_u is the vertex of $C_u(v)$ that minimizes $\rho(\langle v, z_u \rangle)$. We need to choose a cover $S \subseteq I$ and an assignment $\psi : U' \to I$ that minimizes $\max_{v \in S}\{deg_\psi(v)\}$, where $deg_\psi(v)$ is given by $\sum\{\rho(\langle v, z_u \rangle) \mid u \text{ s.t. } \psi(u) = v\}$.

We claim that the set $S = \{\psi(u) \mid u \in U'\}$, for $\psi(u) = m(u)$, is a cover of U' of weighted degree at most *opt*. Indeed, let $v = m(u)$. Consider the path $P^*(v, u)$ from v to u in T^*. Let w be the neighbor of v in this path.

Note that $w \in C_u(v)$. Hence the delay in sending the message from v to w is $\rho(v, w) \geq \rho(v, z_u)$. Therefore, the weighted degree of the assignment defined by T^* is a lower bound on the optimum broadcast time. Thus, this weighted degree cannot be larger than *opt*. It follows that the assignment $\psi(u) = m(u)$ yields a cover assignment for U' with maximum weighted degree at most *opt*.

We want to find a cover S and an assignment ψ that minimizes (or almost minimizes) $\max_{v \in S}\{deg_\psi(v)\}$. The problem of finding the best weighted degree assignment is known in the literature as the problem of *scheduling of independent parallel machines*. This problem is NP-hard and, in fact, is 3/2-inapproximable ([LST90]).

For our purposes, any constant approximation for this problem is sufficient. Unfortunately, we are not aware of a combinatorial algorithm that provides a constant approximation ratio for this problem. We use the 2-approximation algorithm of [LST90] that formulates the problem as an integer linear program, relaxes it to allow fractional solutions, finds a basic feasible solution, and uses a standard rounding technique.

Hence, this way we find a (weighted) cover for the set U' of degree at most $2 \cdot opt$. It follows that the value of the resulting spiders is at most $3 \cdot opt$. The rest of the analysis can be carried through in a straightforward way. To summarize, our algorithm can be adapted to provide a logarithmic approximation guarantee for the directed and undirected versions of broadcast problem in the arc-dependent heterogeneous postal model. The approximation guarantee of the generalized algorithm is only by a constant factor greater than the approximation guarantee of our algorithm for the telephone broadcast problem, i.e., it is $O(\log n)$. Note, however, that for this most general version of our algorithm we do use linear programming. The particular linear programs that are used in the algorithm are solvable via Lagrangian relaxation in time $\tilde{O}(n^3)$ [LST90], and so the overall running time of the generalized algorithm is at most $\tilde{O}(n^3)$ as well.

5.5 Implications for network design

A bicriteria approximation for depth and out-degree: Our algorithm provides a bicriteria approximation for the depth-degree problem. In other words, given a digraph for which there exists a spanning arborescence of degree h and maximum degree d, our algorithm constructs a spanning arborescence of maximum depth $O(\log n) \cdot h$ and maximum degree $O(\log n) \cdot d$.

The tree is built "backwards" (from the leaves to the root). Throughout the algorithm we maintain a forest. The number of trees in the forest gradually decreases until they merge into a single spanning arborescence.

We show how to build this arborescence recursively. The collection of forks $\mathcal{F}$ that is computed by our algorithm on the first level of the recursion forms the initial forest. The heads $\mathcal{R}$ need now to be recursively connected to the root s. Let T' be the arborescence connecting s to $\mathcal{R}$ that was computed by the recursion. By the inductive hypothesis, the set T' is indeed an arborescence, and, in particular, each arc appears in T' at most once. However, observe that T' may not be arc or vertex-disjoint with the forest $\mathcal{F}$. We need to unite T' and $\mathcal{F}$ in order get a single spanning arborescence. This is done by taking the graph induced by $\mathcal{F} \cup T'$, and computing the shortest path arborescence from s to all the vertices $v \neq s$ in $G(\mathcal{F} \cup T')$.

Clearly, the resulting shortest path arborescence T'' has depth $O(\log n) \cdot h$, because each recursive iteration can add at most h to the depth of the arborescence. Similarly, on each recursive invocation, for every vertex v at most d arcs that are adjacent to v are added to the arborescence. Hence, the upper bound of $O(\log n) \cdot d$ on the maximum degree follows.

Hence, our algorithm provides an $(O(\log n), O(\log n))$-bicriteria approximation for both the directed and undirected versions of the degree-depth problem. The algorithm is combinatorial, and its running time is $\tilde{O}(|E| \cdot |V|)$. Note that the same algorithm provides a logarithmic approximation for the directed and undirected poise problems.

6 Hardness Results

In this section we present some lower bounds on the approximation thresholds of the (undirected and directed) telephone broadcast problems.

Definition 6.1 *Given an undirected bipartite graph $G = (V_1, V_2, E)$, a subset of vertices $S \subseteq V_1$ is a* set-cover *for G if for any $v_2 \in V_2$ there exists $v_1 \in S$ such that $(v_1, v_2) \in E$.*

Given a constant $0 < c \leq 1$, and an integer-valued function $t = t(n)$, the YES-instance of the set-cover $(t(n), c)$-promise problem *is an undirected bipartite graph G for which* there exists a set-cover *S of size $|S| \leq t(n)$. The* NO-instance of the set-cover $(t(n), c)$-promise problem *is an undirected bipartite graph G for which* any set-cover *S has size $|S| \geq c \log n \cdot t(n)$.*

The set-cover $(t(n), c)$-promise problem *is given either a YES-instance or a NO-instance of the problem, and the goal is to determine whether it is a YES-instance or a NO-instance.*

It is known [LY95] that there exists a constant c such that the set-cover $(t(n), c)$-promise problem is NP-hard, for $t(n) = \lceil \sqrt{n} \rceil$.

Given an instance of the set-cover $(t(n), c)$-promise problem, an undirected bipartite n-vertex graph $G = (V_1, V_2, E)$, $|V_1| = |V_2| = n/2$, we construct a graph $\bar{G}$ in the following way. Insert into $\bar{G}$ an isomorphic copy of G. For $j = 1, 2$, let $\bar{V}_j$ be the image of V_j under the isomorphism, and, more generally, for a subset $X \subseteq V$ of vertices, let $\bar{X}$ be the image of X under the isomorphism. Add a new vertex s and connect it with all the vertices of $\bar{V}_1$ via outgoing arcs. In addition, add a directed path of length $\lceil (c/2) t(n) \log n \rceil$ that starts in s and contains $\lceil (c/2) t(n) \log n \rceil$ new vertices. Denote by s' the tail of the path. Construct a complete binary arborescence T of depth $\lceil \log n \rceil$, rooted at s', with the set of leaves that contains $\bar{V}_1$ and (maybe) some new vertices, and such that all the remaining vertices of T are new vertices. Finally, replace each star $(\bar{v}_1, \bar{v}_2^1), (\bar{v}_1, \bar{v}_2^2), \ldots, (\bar{v}_1, \bar{v}_2^r)$, with $\bar{v}_1 \in \bar{V}_1$, $\bar{v}_2^1, \bar{v}_2^2, \ldots, \bar{v}_2^r \in \bar{V}_2$, by a complete binary arborescence of depth $\lceil \log r \rceil$ rooted at $\bar{v}_1$ and whose set of leaves $L(T)$ contains the set $\{\bar{v}_2^1, \bar{v}_2^2, \ldots, \bar{v}_2^r\}$ and (maybe) some new vertices. All the other vertices of the tree T are new vertices.

This completes the description of $\bar{G}$ (see Figure 2). Its construction is very similar to the reduction of [S00], that proves the hardness of 3/2 for the problem. We next use $\bar{G}$ as a building block in our reduction that shows the hardness of $\Omega(\sqrt{\log n})$ for the directed broadcast problem, and then, via a similar reduction, a hardness of $3 - \epsilon$ for any $\epsilon > 0$ for the undirected broadcast problem.

We first analyze the properties of the graph $\bar{G}$.

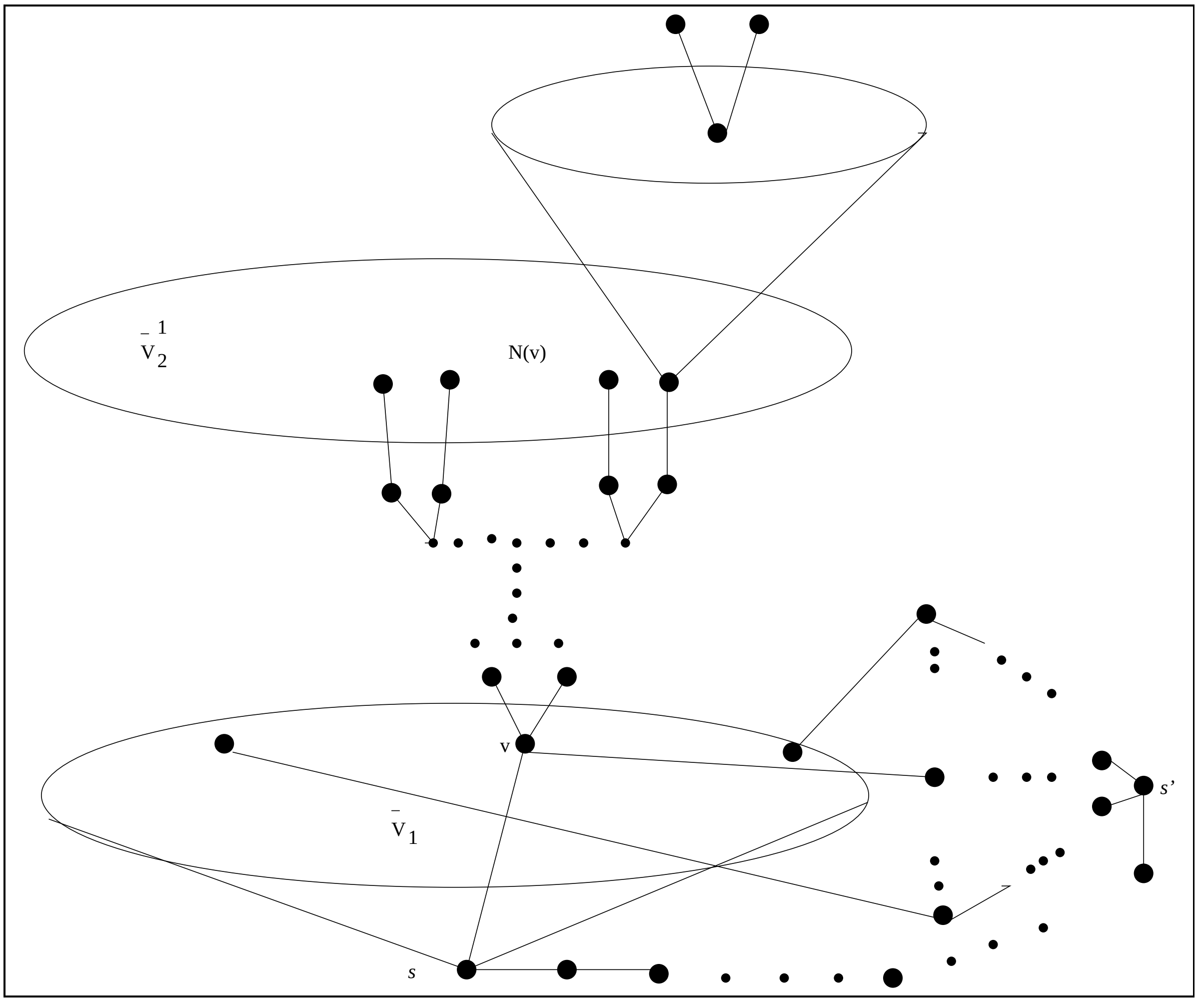

Figure 2: The first level of the graph $\bar{G}^i$. The path from s to s' is of length $\Theta(t(n) \cdot \log n)$. The vertex s' is connected via a complete binary arborescence to the vertex set $\bar{V}_1$. The vertices of $\bar{V}_1$ are connected via complete binary arborescences to their neighbors in $\bar{V}_2$. In turn, the vertices of the set $\bar{V}_2$ serve as sources of the next level of the recursive construction.

Lemma 6.2 *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then there exists an $(\{s\}, \bar{V}_2)$-schedule Π of length $|\Pi| = O(t(n))$ for the instance $(\bar{G}, s)$ of the directed telephone broadcast problem.*

Proof: We describe the schedule Π that satisfies the the assertion of the lemma. Let $S \subseteq V_1$ be an optimal set cover of the instance G. The schedule starts with delivering the message from the vertex s to all the vertices of the set $\bar{S}$. Note that as s is connected via outgoing arcs to all the vertices of $\bar{V}_1$, it follows that all the vertices of the set $\bar{S}$ will be informed within $|S|$ rounds. By assumption, G is a YES-instance of the set cover $(t(n), c)$-promise problem, and hence $|S| \leq t(n)$. Hence, this step requires at most $t(n)$ rounds.

Next, the schedule delivers the message from $\bar{S}$ to all the vertices of the set $\bar{V}_2$. Observe that since the set S is a set cover for the set V_2, only $O(\log n)$ rounds will be required to deliver the message to all the vertices of $\bar{V}_2$ through the auxiliary arborescences that connect the sets $\bar{V}_1$ and $\bar{V}_2$. Hence, altogether, the length of this schedule is at most $t(n) + \log n = O(t(n))$ (since $t(n) = \lceil \sqrt{n} \rceil$), and it informs all the vertices of the set $\bar{V}_2$. ∎

Lemma 6.3 *If G is a NO-instance of the set-cover $(t(n), c)$-promise problem, then any $(\{s\}, \bar{V}_2)$-schedule Π for the instance $(\bar{G}, s)$ of the directed telephone broadcast problem has length $|\Pi| = \Omega(t(n) \log n)$.*

Proof: By construction, the only way to inform the vertices of the set $\bar{V}_2$ is to inform all the vertices of a subset $\bar{S} \subseteq \bar{V}_1$ for some set cover S of V_1.

There are two possible ways to inform such a subset $\bar{S}$: either using the arcs that connect s to the vertices of $\bar{V}_1$, or through the path between s and s', and the complete binary arborescence T (or by combining these two ways).

Note that as G is a NO-instance, it follows that $|S| = |\bar{S}| = \Omega(t(n) \cdot \log n)$. Therefore, the first way requires $\Omega(t(n) \cdot \log n)$ rounds of telephone broadcast. Also, recall that the length of the path from s to s' is $\Omega(t(n) \cdot \log n)$, and hence informing even a single vertex $v \in \bar{V}_1$ using the second way would require $\Omega(t(n) \cdot \log n)$ rounds. Hence, any schedule that informs a subset $\bar{S} \subseteq \bar{V}_1$ of size $\Omega(t(n) \cdot \log n)$ requires $\Omega(t(n) \cdot \log n)$ rounds of telephone broadcast, proving the claim. ∎

Note that Lemmata 6.2 and 6.3 imply $\Omega(\log n)$-inapproximability of the *directed multicast problem*, that was known [F01].

However, since both for the YES and NO-instance of the set-cover, informing $\bar{V}_1$ requires $\Omega(t(n) \log n)$ rounds, this reduction by itself may provide only a constant hardness of approximation for the *broadcast* problems. Indeed, by a careful choice of parameters and some modifications to $\bar{G}$, it is shown in [S00] that this reduction yields a hardness of 3/2 for the undirected broadcast problem.

Consider the following recursive construction of a triple $(\bar{G}^i = (\bar{V}^i, \bar{E}^i), s^i, \bar{V}_2^i)$, where $(\bar{G}^i, s^i)$ is an instance of the directed broadcast problem, and $\bar{V}_2^i \subseteq \bar{V}^i$. Let $(\bar{G}^1, s^1, \bar{V}_2^1) = (\bar{G}, s, \bar{V}_2)$. Let also $Z(\bar{G}^1)$ denote the set $\bar{V}_2^1$. Given a construction of $(\bar{G}^i, s^i, \bar{V}_2^i)$, the triple $(\bar{G}^{i+1}, s^{i+1}, \bar{V}_2^{i+1})$ is constructed in the following way.

Insert an isomorphic copy of $\bar{G}^1$ into $\bar{G}^{i+1}$. Set $s^{i+1} = s^1$. For $j = 1, 2$, let $\bar{V}_j^1$ be an image of $\bar{V}_j$ under the isomorphism. Insert the set $\bar{V}_2^1$ into $\bar{V}_2^{i+1}$. For every vertex $\bar{v}_2^1 \in \bar{V}_2^1$, construct a triple $(\bar{G}^i(\bar{v}_2^1), \bar{v}_2^1, \bar{V}_2^i(\bar{v}_2^1))$, where $\bar{G}^i(\bar{v}_2^1)$ is an isomorphic copy of $\bar{G}^i$ such that the isomorphism takes s^i to $\bar{v}_2^1$. All the other vertices of $\bar{G}^i(\bar{v}_2^1)$ are new vertices. Insert the set $\bar{V}_2^i(\bar{v}_2^1)$ into $\bar{V}_2^{i+1}$.

Form the set $Z(\bar{G}^{i+1})$ to be the union of the isomorphic copies of the sets $Z(\bar{G}^i(\bar{v}_2^1))$ for all the different vertices $\bar{v}_2^1 \in \bar{V}_2$.

This completes the construction of $(\bar{G}^{i+1} = (\bar{V}^{i+1}, \bar{E}^{i+1}), s^{i+1}, \bar{V}_2^{i+1})$.

First, we provide an estimate of the number of vertices of the graph $\bar{G}^i$.

Lemma 6.4 $|\bar{V}^i| = n^{O(i)}$, $i = 1, 2, \ldots$.

Proof: The proof follows by a straightforward induction on the number of levels i. ▮

Next, we analyze the graph $\bar{G}^i$ that is obtained by the reduction that was described above from a YES-instance G of the set-cover $(t(n), c)$-promise problem.

Lemma 6.5 *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then for any $i = 1, 2, \ldots$ there exists an $(\{s^i\}, \bar{V}_2^i)$-schedule Π for the instance $(\bar{G}^i, s^i)$ of the directed broadcast problem of length $|\Pi| = O(i \cdot t(n))$.*

Proof: The proof is by induction on i. The induction base follows from Lemma 6.2.

For the induction step consider the instance $(\bar{G}^{i+1}, s^{i+1})$. Note that by Lemma 6.2, all the vertices of the set $\bar{V}_2^1$ can be informed within $O(t(n))$ rounds. After that point every vertex $\bar{v}_2^1 \in \bar{V}_2^1$ needs to relay the message to all the vertices of the set $\bar{V}_2^i(\bar{v}_2^1)$ through an isomorphic copy of the graph $\bar{G}^i$. Since the different copies share no vertices, these broadcasts can be conducted in parallel, and, by the induction hypothesis, can be completed within $O(i \cdot t(n))$ rounds. ▮

Lemma 6.6 *For any $i = 1, 2, \ldots$, there exists a $(\{s^i\} \cup \bar{V}_2^i, \bar{V}^i \setminus \bar{V}_2^i)$-schedule Π of length $O(t(n) \cdot \log n)$.*

Proof: Consider the graph $\bar{G}^i$, and suppose that for every isomorphic copy of the graph $\bar{G}$ that is contained in $\bar{G}^i$, the vertex that corresponds to s is informed. Note that this assumption is weaker than the assumption that all the vertices of the set $\{s^i\} \cup \bar{V}_2^i$ are informed, as the latter assumption assumes that all the vertices of the set $\bigcup_{\bar{v}_2^1 \in \bar{V}_2^1} Z(\bar{G}^i(\bar{v}_2^1))$ are informed as well.

Consider some single copy of the graph $\bar{G}$. Observe that delivering the message from s to all the vertices of the set $\bar{V}_1$ through the path from s to s', and from s' through the complete binary arborescence to all the vertices of the set $\bar{V}_1$ requires only $O(t(n) \cdot \log n)$ rounds. Furthermore, only $O(\log n)$ additional rounds are required to relay the message from the vertices of the set $\bar{V}_1$ to all the vertices of the binary arborescences that connect between $\bar{V}_1$ and $\bar{V}_2$. Hence, altogether $O(t(n) \cdot \log n)$ rounds are required to deliver the message from s to all the other vertices of the copy $\bar{G}$, and, furthermore, these deliveries can be conducted in parallel in different copies. ▮

Corollary 6.7 *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then for any $i = 1, 2, \ldots$, there exists an $(\{s^i\}, \bar{V}^i)$-schedule of length $O(t(n) \cdot (i + \log n))$.*

Next, we turn to analyzing the graph $\bar{G}^i$ that is obtained by the reduction that was described above from a NO-instance G of the set-cover $(t(n), c)$-promise problem.

Lemma 6.8 *If G is a NO-instance of the set-cover $(t(n), c)$-promise problem, then for any $i = 1, 2, \ldots$, any $(\{s^i\}, \bar{V}^i)$-schedule Π for the instance $(\bar{G}^i, s^i)$ of the directed broadcast problem is of length $|\Pi| = \Omega(t(n) \cdot \log n \cdot i)$.*

Proof: The proof is by induction on i. The induction base follows directly from Lemma 6.3.

For the induction step, consider the graph $\bar{G}^{i+1}$, and consider the isomorphic copy of $\bar{G}^1$ that has the vertex s^{i+1} as the image of the vertex s^1 under the isomorphism. For every schedule Π, by Lemma 6.3, there exists a vertex $\bar{v}_2^1 \in \bar{V}_2^1$ in this copy that is informed only after $\Omega(t(n) \cdot \log n)$ rounds. Consider the copy $\bar{G}^i(\bar{v}_2^1)$ of the graph $\bar{G}^i$. Recall that by construction, this copy of $\bar{G}^i$ is a subgraph of the graph $\bar{G}^{i+1}$. Also, no vertex of this copy $\bar{G}^i(\bar{v}_2^1)$ can be informed before the vertex $\bar{v}_2^1$ is informed. By the induction hypothesis, delivering the message from the vertex $\bar{v}_2^1$ to all the other vertices of the copy $\bar{G}^i(\bar{v}_2^1)$ requires $\Omega(i \cdot t(n) \cdot \log n)$ rounds. The assertion of the lemma follows. ▮

Finally, by substituting $i = \log n$ it follows that

Theorem 6.9 *The directed broadcast problem is $\Omega(\sqrt{\log n})$-inapproximable, unless $NP \subseteq DTIME(n^{O(\log n)})$.*

Next, we establish inapproximability of $3-\epsilon$ for any $\epsilon > 0$ for the undirected broadcast problem. This is done via a similar reduction.

Specifically, the graph $\bar{G}$ is modified in the following way. First, insert into $\bar{\bar{G}}$ an isomorphic copy of $\bar{G}$, with all the directed arcs replaced by undirected edges. For $j = 1, 2$ let $\bar{\bar{V}}_j$ be the image of $\bar{V}_j$ under the isomorphism, and more generally, for a subset $X \subseteq V$, let $\bar{\bar{X}}$ denote its image under the isomorphism. Second, for every star $(v_1, v_2^1), (v_1, v_2^2), \ldots, (v_1, v_2^r)$ in G with $v_1 \in V_1$, $v_2^1, v_2^2, \ldots, v_2^r \in V_2$ (or, in other words, for every complete binary tree rooted at $\bar{v}_1$ and with the set of leaves that contains the set $\{\bar{v}_2^1, \ldots, \bar{v}_2^r\}$ in $\bar{G}$), insert a path of length $\lceil \log n \cdot c/2 \cdot t(n) \rceil$, that connects $\bar{v}_1$ with a vertex $\bar{v}_1'$, where all the vertices of this path except of $\bar{v}_1$ are new. Now, construct a complete binary tree rooted in $\bar{v}_1'$ and with the set of leaves that contains $\{\bar{v}_2^1, \ldots, \bar{v}_2^r\}$, and all the other vertices of the tree are new. This completes the construction of $\bar{\bar{G}}$.

Now, the construction of the graph $\bar{\bar{G}}^i$ for $i = 1, 2, \ldots$ given the graph $\bar{\bar{G}}$ is identical to the construction of $\bar{G}^i$ out of $\bar{G}$.

Lemma 6.10 *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then there exists an $(\{s\}, \bar{\bar{V}}_2)$-schedule Π of length $|\Pi| \le (c/2 + o(1))t(n)\log n$ for the instance $(\bar{\bar{G}}, s)$ of the undirected telephone broadcast problem.*

Proof: As G is a YES-instance of the set-cover problem, there exists a set cover $S \subseteq V_1$ for V_2 of size at most $t(n)$. Using the edges of the star that connect between the copy of the vertex s in $\bar{\bar{G}}$ and the copies of the vertices of S, it is possible to inform the latter vertices using $|S| \le t(n)$ rounds. Since S is a set cover for V_2, and since the distance between the sets $\bar{\bar{S}}$ and $\bar{\bar{V}}_2$ in the graph $\bar{\bar{G}}$ is $\frac{c}{2} \cdot \log n \cdot t(n) + O(\log n)$, it follows that it is possible to relay the message from the vertices of the set $\bar{\bar{S}}$ to all the vertices of the set $\bar{\bar{V}}_2$ in $(\frac{c}{2} + o(1)) \cdot \log n \cdot t(n)$ rounds. ▮

Lemma 6.11 *If G is a NO-instance of the set-cover $(t(n), c)$-promise problem, then any $(\{s\}, \bar{\bar{V}}_2)$-schedule Π for the instance $(\bar{\bar{G}}, s)$ of the undirected telephone broadcast problem has length $|\Pi| \ge 3/2 \cdot c \cdot t(n) \log n$.*

Proof: The proof of this lemma is analogous to that of Lemma 6.3. Like in that proof, it is easy to see that for informing all the vertices of the set $\bar{\bar{V}}_2$ it is necessary to inform a subset $\bar{\bar{S}} \subseteq \bar{\bar{V}}_1$ that satisfies that the subset S is a set cover for V_2. Informing the set $\bar{\bar{S}}$ via the edges that connect the vertex s to the vertices of $\bar{\bar{V}}_1$ requires at least $|\bar{\bar{S}}| \geq c \cdot t(n) \cdot \log n$ rounds (because G is a NO-instance). Informing a single vertex from the set $\bar{\bar{V}}_2$ using the path that connects s to s', and the complete binary tree T rooted in s' requires at least $c \cdot t(n) \cdot \log n$ rounds, because the length of this path between s and s' is $c \cdot t(n) \cdot \log n$. Hence, at least $c \cdot t(n) \cdot \log n$ rounds are required to inform a set $\bar{\bar{S}}$, where S is a set cover for V_2. Finally, since the distance between the sets $\bar{\bar{V}}_1$ and $\bar{\bar{V}}_2$ in the graph $\bar{\bar{G}}$ is at least $\frac{c}{2} \cdot t(n) \cdot \log n$, it follows that informing the vertices of the set $\bar{\bar{V}}_2$ once all the vertices of the subset $\bar{\bar{S}}$ are informed would require at least $\frac{c}{2} \cdot t(n) \cdot \log n$ additional rounds. Hence, altogether at least $\frac{3}{2} c \cdot t(n) \cdot \log n$ rounds are required to relay the message from s to all the vertices of the set $\bar{\bar{V}}_2$. ∎

Note that Lemmata 6.10 and 6.11 imply $(3 - o(1))$-inapproximability of the *undirected multicast* problem. However, since for both the YES and NO-instance of the set-cover problem, informing the subset $\bar{\bar{V}}_1$ in the instance $(\bar{\bar{G}}, s)$ of the undirected telephone broadcast problem requires $t(n) \cdot \log n(c/2 + o(1))$ rounds, the reduction that uses only $\bar{G}$ yields only $(3/2 - o(1))$-inapproximability for the *undirected broadcast* problem. This is, essentially, the reduction of [S00].

In the following lemmata we show that the construction of $\bar{\bar{G}}^i$ yields the desired $(3 - \epsilon)$-inapproximability.

Lemma 6.12 $|\bar{\bar{V}}^i| = n^{O(i)}$, $i = 1, 2, \ldots$.

Proof: Similarly to Lemma 6.4, the proof can be derived by a straightforward induction on i. ∎

The proof of the following lemma is analogous to the proofs of Lemmata 6.5, 6.6.

Lemma 6.13

1. *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then for any $i = 1, 2, \ldots$ there exists an $(\{s^i\}, \bar{\bar{V}}^i_2)$-schedule Π for the instance $(\bar{\bar{G}}^i, s^i)$ of the undirected broadcast problem of length $|\Pi| \leq i \cdot t(n)(c/2 + o(1)) \log n$.*

2. *For any $i = 1, 2, \ldots$ there exists a $(\{s^i\} \cup \bar{\bar{V}}^i_2, \bar{\bar{V}} \setminus \bar{\bar{V}}^i_2)$-schedule Π of length $|\Pi| \leq t(n) \log n(c/2 + o(1))$.*

The following corollary is immediate given Lemma 6.13.

Corollary 6.14 *If G is a YES-instance of the set-cover $(t(n), c)$-promise problem, then for any $i = 1, 2, \ldots$, there exists an $(\{s^i\}, \bar{\bar{V}}^i)$-schedule Π for the instance $(\bar{\bar{G}}^i, s^i)$ of the undirected broadcast problem of length $|\Pi| \leq (i + 1)t(n) \log n(c/2 + o(1))$.*

We next turn to the analysis of the NO-instance.

Lemma 6.15 *If G is a NO-instance of the set-cover $(t(n), c)$-promise problem, for any $i = 1, 2, \ldots$, any $(\{s^i\}, \bar{\bar{V}}^i)$-schedule for the instance $(\bar{\bar{G}}^i, s^i)$ of the undirected broadcast problem is of length $|\Pi| \geq 3 \cdot t(n) \log n(c/2 + o(1)) \cdot i$.*

Proof: The proof of this lemma is analogous to that of Lemma 6.8. The only difference is that Lemma 6.11 is used instead of Lemma 6.3. ∎

It follows that the undirected broadcast problem is $(3 - O(1/i))$-inapproximable unless $NP \subseteq DTIME(n^{O(i)})$.

Theorem 6.16 *For any $\epsilon > 0$, it is NP-hard to approximate the undirected broadcast problem within a ratio of $3 - \epsilon$.*

7 Discussion

We demonstrated that the approximation threshold of the *directed telephone broadcast* problem is between $O(\log n)$ and $\Omega(\sqrt{\log n})$. Bridging this gap is a challenging open problem. Determining the approximation threshold of the directed multicast problem is another important open problem. Here the gap is between an additive approximation of $O(\sqrt{k})$ (that comes together with a logarithmic multiplicative factor) [EK03b], and a lower bound of $\Omega(\log n)$ [F01].

Acknowledgments

We are grateful to Magnus M. Halldorsson for his help in writing this manuscript, and helpful discussions and comments. We are also grateful to Zvika Lotker for helpful and motivating discussions, particularly about the reduction of [S00].

References

[BGNS98] A. Bar-Noy, S. Guha, J. Naor, B. Schieber. Multicasting in Heterogeneous Networks. In *Proc. ACM Symp. on Theory of Computing*, STOC'98, pp. 448-453, 1998.

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

[EK02] M. Elkin and G. Kortsarz. A combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem. In *Proc. ACM Symp. on Theory of Computing*, STOC'02, pp. 438-447, Canada, Montreal, May 2002.

[EK03a] M. Elkin and G. Kortsarz. A sublogarithmic approximation algorithm for the undirected telephone broadcast problem: A path out of a jungle. In *Proc. Symp. on Discr. Algorithms*, SODA'03, pp. 76-85, Baltimore, MA, Jan. 2003.

[EK03b] M. Elkin and G. Kortsarz. An approximation algorithm for the directed telephone multicast problem. In *the Proc. International Colloquium on Automata, Languages and Programming* (ICALP) 2003, pp. 212-223.

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

[GT88] A. V. Goldberg, R. E. Tarjan. A new approach to the maximum flow problem. *Journal of ACM*, 35:921-940, 1988.

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

[KP95] G. Kortsarz and D. Peleg. Approximation algorithms for minimum time broadcast, *SIAM Journal on Discrete Mathematics*, vol. 8, pages 401-427, 1995.

[KSSS93] R. Karp, A. Sahay, E. Santos, and K. E. Schauser. Optimal broadcast and summation in the LogP model. In *Proc. 5th Ann. ACM Symp. on Parallel Algorithms and Architectures*, pp. 141-153, June 1993.

[KPST94] P. Klein, S. Plotkin, C. Stein and E. Tardos. Faster approximation algorithms for the unit capacity concurrent flow problem with applications to routing and finding sparse cuts. In *SIAM Journal of Computing*, 23/3, 1994, pp. 466-487.

[LST90] L. K. Lenstra and D. Shmoys and E. Tardos. "Approximation algorithms for scheduling unrelated parallel machines". *Math. Programming*, 46, pp. 259-271, 1990.

[LY95] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *J. Assoc. Comput. Mach.*, 41(5):960–981, 1994.

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

[MRR+93] M. V. Marathe, R. Ravi, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt. Many birds with one stone: Multi-objective approximation algorithms. In *Proc. ACM Symp. on the Theory of Computing (STOC '93)*, pp. 438-447, 1993.

[R94] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *Proc. 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. ACM Symp. on Theory of Computing*, STOC'94, pp. 575-584.

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