

Approximating the Degree-Bounded Minimum Diameter Spanning Tree Problem¹

Jochen Könemann,² Asaf Levin,³ and Amitabh Sinha⁴

Abstract. We consider the problem of finding a minimum diameter spanning tree with maximum node degree B in a complete undirected edge-weighted graph. We provide an $O(\sqrt{\log_B n})$ -approximation algorithm for the problem. Our algorithm is purely combinatorial, and relies on a combination of *filtering* and *divide and conquer*.

Key Words. Approximation algorithms, Spanning trees, Bicriteria approximation, Degree-bounded spanning trees.

1. Introduction. The importance of algorithms for designing efficient networks in today's interconnected world can hardly be overstated. The operative word here is “efficient,” and, indeed, there are many (often conflicting) ways to measure the efficiency of a network. Suppose a telecommunication company is building a communication network. While budgeting constraints may require the company to minimize total cost, there are also quality of service and technological constraints which may require the network to have low diameter and low degree.

Low diameter is essential to ensure that any pair of nodes can communicate fast. It is also useful to force reliability constraints, as explained in the following (see also [13] and [19]): Assume that an edge e fails with probability $1 - p_e$, and that all failures occur independently. Then the probability that a path e_1, e_2, \dots, e_k is operational is $p_{e_1} \times p_{e_2} \times \dots \times p_{e_k}$. Given a certain threshold value for the desired reliability, there is a corresponding parameter D such that the diameter of the network defined by edge length $(\log p_e)_{e \in E}$ is required to be at most D . Therefore, the reliability constraint is transformed into a diameter constraint.

Degree constraints appear naturally in graph-theoretic abstractions of communication network design problems. As an example, consider the so-called *IP multicast* [8], [9] problem where we would like to disseminate centrally stored information from a server node to a set of client hosts. The standard solution is to compute a tree in the given

¹ This material by the first author is based upon work supported by the National Science Foundation under Grant No. 0105548. This material by the third author is based upon work supported by the National Science Foundation under Grant No. 0105548 and a Carnegie Bosch Institute Fellowship, done while the author was a graduate student at the Tepper School of Business, Carnegie Mellon University.

² Department of Combinatorics and Optimization, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1. jochen@math.uwaterloo.ca.

³ Faculty of Industrial Engineering and Management, The Technion, Haifa 32000, Israel. levinas@tx.technion.ac.il.

⁴ Business School, University of Michigan, Ann Arbor, MI 48103, USA. amitabh@umich.edu.

graph that spans the server node and all client nodes. We then send data packets from the root along each of its incident edges in the tree. An internal node forwards incoming information to its descendants in the tree. The number of descendants of a node in this tree is proportional to the amount of work that the node has to do and it is hence natural to aspire to compute spanning trees of low maximum degree (see also [5], [7], and [20]).

Our work is motivated by precisely these considerations. We proceed by defining our problem.

1.1. Problem Definition. Formally, we consider the following *bounded degree minimum diameter spanning tree problem* (BDST): given an undirected complete graph $G = (V, E)$ whose edges are endowed with a metric length function $\{l_e\}_{e \in E}$ and a parameter $B \geq 2$, we want to find a spanning tree T of G of maximum node-degree at most B . At the same time we want to minimize the *diameter* of T , i.e. we would like to minimize

$$\Delta(T) := \max_{u, v \in V} \text{dist}_l^T(u, v),$$

where $\text{dist}_l^T(u, v)$ denotes the l -length of the unique (u, v) -path in T .

Let the *height* of a tree T rooted at node r be the maximum number of edges on any (r, v) -path, where v is a leaf node in T and denote it by $\text{height}(T)$. We also use n and m to denote $|V|$ and $|E|$, respectively.

For $B = 2$, BDST can be approximated within a constant using approximation algorithms for the *Traveling Salesperson* problem. In this paper we consider the case $B \geq 3$.

1.2. Our Contribution. Our main result is an $O(\sqrt{\log_B n})$ approximation algorithm for BDST. The algorithm is described and analyzed in Section 2. There are two main ideas in the algorithm. First, we break up the graph into clusters of low diameter. For each cluster, we compute a balanced $(B - 1)$ -ary tree. We then compute a global tree over the clusters, and show that the resulting tree has low diameter.

Our algorithm is the first known sublogarithmic approximation for this problem. An $O(\log_B n)$ approximation is trivial; any complete balanced $(B - 1)$ -ary spanning tree of the graph will do.

Our result directly leads to an improvement of a recent paper by Arkin et al. [1] on the *Freeze-Tag problem*. Here, we are given an undirected graph $G = (V, E)$ with non-negative edge-lengths l_e for all edges $e \in E$. Initially there is an *awake* robot at a given node $v_0 \in V$ and each vertex $v \in V$ contains r_v *asleep* robots. Our model allows an awake robot at node v to traverse an edge $e \in E$ that is incident to v in time l_e . Awake robots can now wake up asleep robots by moving to their location in G . Once awake, the new robots can help in waking up other robots. The goal in the Freeze-Tag problem is to minimize the *makespan*, i.e. the time it takes to wake up all robots.

In [1] Arkin et al. present an $O(\log \Delta)$ approximation algorithm for the Freeze-Tag problem where Δ is the largest degree in the graph. The authors also show that an ω -approximation for the BDST problem implies the same guarantee also for the Freeze-Tag problem. The main insight used in this reduction is that a wake-up schedule corresponds to an arborescence in G that is rooted at v_0 and has out-degree at most $r_v + 1$ for

each node $v \in V$. A minimum-makespan wake-up schedule corresponds to a minimum diameter degree-bounded arborescence. Our algorithm therefore implies an $O(\sqrt{\log n})$ approximation algorithm for the Freeze-Tag problem.

More recently, Arkin et al. [2] improved upon [1] in several special cases. In the general Freeze-Tag problem as stated above they obtain an $O(L/d \cdot \log n + 1)$ approximation where L is the length of the longest edge in E and d is the diameter of G . Our algorithm remains the best known result for the Freeze-Tag problem in the general setting.

Finally, in [1], Arkin et al. show that it is **NP**-hard to obtain an approximation algorithm with performance guarantee better than $5/3$ for the Freeze-Tag problem. This implies that the BDST problem has no $(5/3 - \varepsilon)$ -approximation either for any $\varepsilon > 0$ unless **NP** = **P**.

1.3. Related Work. The problem considered in this paper extends a long line of previous work on constrained network design. In the following we give a survey of those results that are closely related to our work.

The most basic related problem is the *minimum degree spanning tree problem* where we are given an undirected graph $G = (V, E)$ and the goal is to find a spanning tree of G whose maximum node-degree is minimized. The best known algorithm for this problem is due to Fürer and Raghavachari [11] who show how to compute a spanning tree with maximum degree $\Delta^* + 1$ where Δ^* is the smallest maximum degree of any spanning tree of G . The algorithms in [11] extend to the Steiner case. For directed graphs, Krishnan and Raghavachari [16] present a quasi-polynomial-time algorithm that computes a directed spanning tree with maximum out-degree $O(B + \log n)$.

The *minimum diameter spanning tree problem* is the following: given an undirected graph $G = (V, E)$ and length function defined over its edge set $\{l_e\}_{e \in E}$, we want to find a spanning tree of G of minimum diameter. This problem is equivalent to finding the shortest paths tree from the absolute 1-center of G (see [15]), and, therefore, is solvable in $O(mn + n^2 \log n)$ time.

The problem of computing diameter-constrained trees has also been studied empirically. We point the reader to a recent paper by Gouveia and Magnanti [13] and the references therein.

Hassin and Levin [14] considered the *hop-constrained spanning tree problem*: given an undirected graph $G = (V, E)$, costs c_e for all edges $e \in E$, and a symmetric requirement matrix $(u_{ij}) \in N^{n \times n}$; the goal is to find a minimum-cost spanning tree T in G such that for all $i, j \in V$, the unique (i, j) -path in T has at most u_{ij} edges. The authors consider the special case of this problem where $u_{ij} \in \{1, 2, \infty\}$, for all $i, j \in V$, and present a constant factor approximation algorithm for this case.

Minimizing the diameter of a tree is closely related to minimizing the so-called *maximum dilation*. Let $G = (V, E)$ be an undirected graph with non-negative lengths l_e for all edges $e \in E$. Let $d_G(u, v)$ be the length of a minimum-length (u, v) -path in G . Consider a spanning tree T of graph G . The dilation of the pair of nodes $u, v \in V$ in T is defined to be the ratio $d_T(u, v)/d_G(u, v)$. A tree T is called a *k-tree-spanner* if the dilation of all pairs of nodes $u, v \in V$ is at most k . In [6] Cai and Corneil showed how to compute a 1-tree-spanner in polynomial time if it exists. The authors also give a polynomial-time algorithm to compute a 2-tree-spanner in the case where every edge has unit length.

Low-dilation trees also occur in the context of *metric space approximation*. Any general metric space can be thought of as a pair (G, l) of an undirected graph $G = (V, E)$ and a vector l of non-negative lengths for all edges in E . The distance between two points $u, v \in V$ in the corresponding metric space is then given by $d_G(u, v)$. Bartal [3], [4] considered the question of approximating a general metric space by a *tree metric* (T, l) . In particular he showed that for a given metric (G, l) there is a probability distribution over tree metrics \mathcal{T} such that the expected dilation of any pair of nodes $u, v \in V[G]$ is $O(\log(n) \log \log(n))$. Subsequently, Fakcharoenphol et al. [10] improved upon [3], and Bart98 and showed that for any general metric (G, l) there is a probability distribution over tree metrics with expected maximum dilation $O(\log n)$.

Whereas the common criteria in low dilation trees and metric space approximation is to bound the maximum dilation (i.e. the distances in the resulting tree with respect to the original distances), we are concerned with bounding the performance of the tree with respect to the optimal tree. This significant difference allows us to use a better lower bound, and to derive a better approximation algorithm.

In [18] Ravi considered the problem of broadcasting a bit of information from a root node to all other nodes in a given undirected graph. As a subproblem he considered the BDST problem in (non-complete) undirected graphs $G = (V, E)$ with non-negative lengths l_e for all edges $e \in E$. The goal is to compute a spanning tree T in G of minimum diameter whose maximum node-degree is bounded by a given parameter $B > 0$. The paper shows how to compute a tree whose diameter is $O(\log n)$ times that of any spanning tree with maximum degree B and whose maximum degree is $O(B \cdot \log^2 n)$.

We also direct the reader to a paper by Marathe et al. [17] that introduced a formal model for network design optimization problems with two criteria. The authors study spanning trees and Steiner trees under combinations of diameter, degree, and cost constraints.

2. Algorithm and Analysis

2.1. Overview. The main idea behind our algorithm is *filtering*. Let $\alpha > 0$ be a threshold, where distances more than α are called *long* and distances less than α are *short*. We partition the node set of G into clusters such that the diameter of each cluster is low, but the number of clusters is also small. We do this by filtering the node set so that we retain one representative node for each cluster, and define an artificial degree bound for this representative node to account for the degree capacity of the entire cluster.

We obtain our performance guarantee from the following two observations. Since the number of clusters is small, any balanced tree which spans the representatives has a small number of long edges. Moreover, since each cluster has a small diameter, the overhead added to any path by the expansion of the representative nodes into trees spanning the clusters is also small. The rest of this paper shows that such a threshold exists and yields our claimed performance guarantee.

A graphic illustration of the algorithm on a sample input instance is shown in Figure 1.

2.2. Algorithm. Given an appropriately chosen threshold α , the first step of our algorithm is to find representatives $R = \{v_1, \dots, v_q\} \subseteq V$ and a partition of V into pairwise

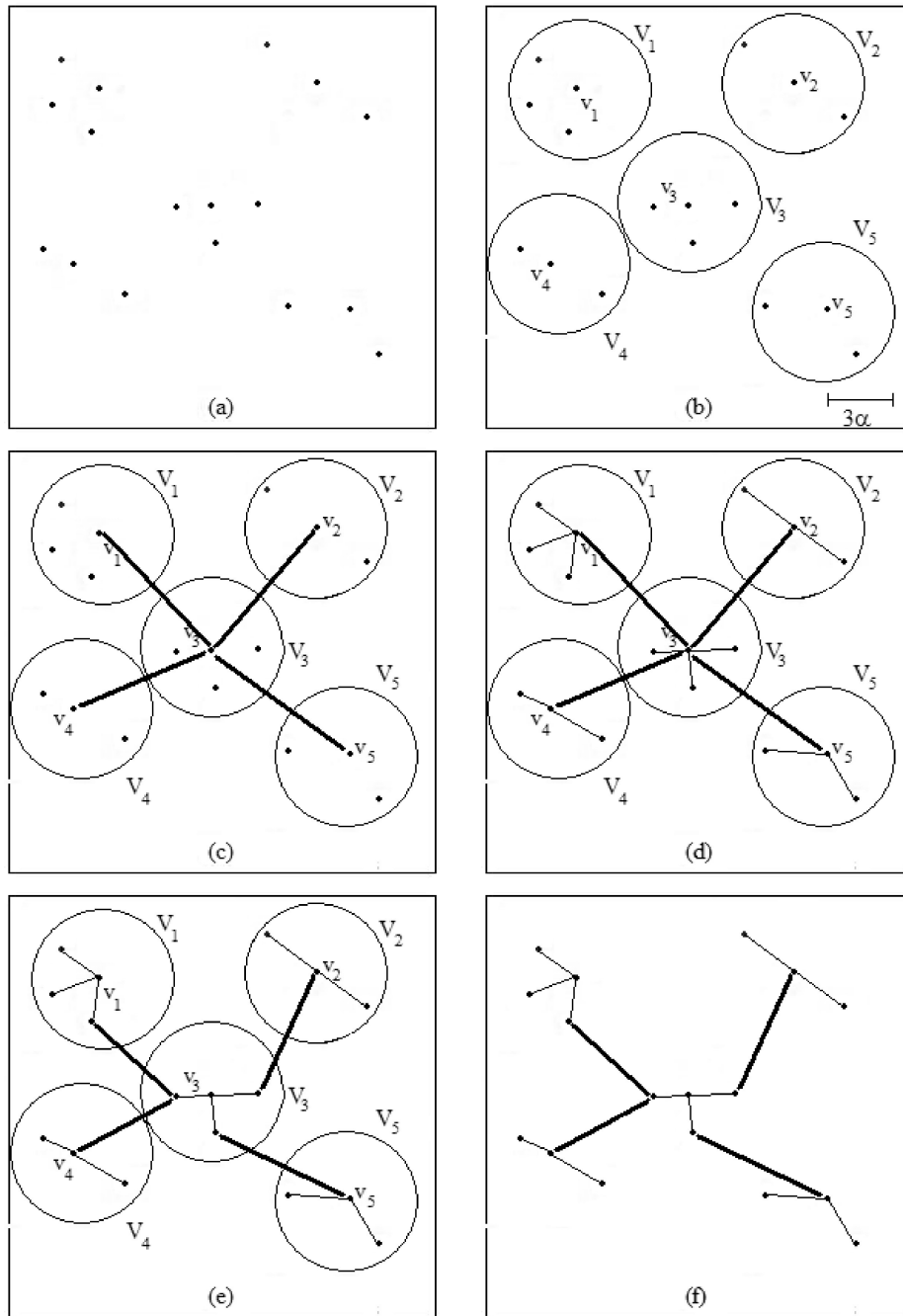


Fig. 1. (a) An input instance, with $B = 3$. (b) Partition into clusters V_1, \dots, V_5 with representatives $R = \{v_1, \dots, v_5\}$. (c) Global tree T^S spanning R , shown by thick edges. (d) Local trees T_1, \dots, T_5 on clusters V_1, \dots, V_5 shown using thin edges. (e) Edges of T^S redistributed to leaves of T_i to preserve degree bound of 3 at each node. (f) Final solution.

Algorithm 1—`GlobTree`($R, \{B_v\}_{v \in R}$): Compute a tree T on the nodes in R such that node $v \in R$ has node degree at most B_v for all $v \in R$.

```

1: Let  $R = \{v_1, \dots, v_q\}$  such that  $B_{v_1} \geq \dots \geq B_{v_q}$ 
2:  $T \leftarrow \emptyset$ 
3:  $d_j \leftarrow B_{v_j}$  for all  $1 \leq j \leq q$ 
4: for  $i = 2$  to  $q$ 
5:   Let  $1 \leq j \leq i$  be smallest with  $d_j > 0$ .
6:   Add edge  $(v_j, v_i)$  to  $T$ .
7:    $d_j \leftarrow d_j - 1$ .
8:    $d_i \leftarrow d_i - 1$ .
9: end for
10: return Tree  $T$  with root  $v_1$ .

```

disjoint sets:

$$(1) \quad V = V_1 \cup \dots \cup V_q$$

such that $v_i \in V_i$ and $\text{dist}_l(v_i, u) \leq 3 \cdot \alpha$ for all $1 \leq i \leq q$ and for all $u \in V_i$. Roughly speaking, we then construct a low-degree and low-diameter tree on the nodes of R . This tree determines the global structure of our solution. In addition we construct low-diameter degree- B -bounded trees for the nodes of each set V_i , $1 \leq i \leq q$. We finish by replacing the nodes from R in the global solution by the respective spanning trees.

In the following we assume that we have a guess for the optimum diameter Δ . This is justified since the diameter of an optimum tree is within the interval $[\max_{e \in E} l_e, n \cdot \max_{e \in E} l_e]$ and we can perform a binary search in order to find a proper approximate guess (i.e. a guess within twice the optimum diameter).

We now detail the process of finding the partition from (1). We proceed in iterations: in iteration $1 \leq i \leq q$, we compute the set V_i and its representative v_i . For ease of notation, we use U_i^γ to denote the set of nodes that are at a distance of at least γ from the first $i - 1$ representatives $\{v_1, \dots, v_{i-1}\}$. In order to define these sets formally, let $\text{cov}_\gamma(v, U) = \{u \in U : \text{dist}_l(v, u) \leq \gamma\}$ be the set of nodes in U that are within a distance of γ of vertex v (we also say that v γ -covers the nodes in $\text{cov}_\gamma(v, U)$). Then we let $U_1^\gamma = V$ for all $\gamma > 0$. For $i > 1$ we define $U_i^\gamma = V \setminus \bigcup_{1 \leq j \leq i-1} \text{cov}_\gamma(v_j, V)$.

Let α be a given threshold. In iteration i we then pick vertex $v_i \in U_i^\alpha$ that α -covers most nodes in $U_i^{3\alpha}$, i.e. we let $v_i = \text{argmax}_{v \in U_i^\alpha} |\text{cov}_\alpha(v, U_i^{3\alpha})|$, and $V_i = \text{cov}_{3\alpha}(v_i, U_i^{3\alpha})$.

The algorithm stops as soon as all nodes in V are within a distance of at most 3α from some representative. We assume that this happens after q iterations. We have $U_{q+1}^{3\alpha} = \emptyset$ and $U_i^{3\alpha} \neq \emptyset$ for all $1 \leq i \leq q$. Figure 1(b) shows the partition with $q = 5$ on the input instance shown in Figure 1(a).

In order to compute the final tree, we go through two main steps:

Global structure. Let $u \in R$ be a representative and let $U \subseteq V$ be its set from the computed partition. We let the degree bound for node u be

$$(2) \quad B_u = |U| \cdot (B - 2) + 2.$$

Algorithm 2— $\text{BDST}(G, \Delta)$: Compute a degree B tree of diameter no more than $O(\sqrt{\log_B n})\Delta$.

```

1:  $\alpha \leftarrow \Delta / \sqrt{\log_B n}$ .
2:  $i \leftarrow 0, R \leftarrow \emptyset$  [Fig. 1(a)]
3:  $U_1^{3\alpha} \leftarrow V$ 
4: while  $U_i^{3\alpha} \neq \emptyset$ 
5:    $v_i \leftarrow \operatorname{argmax}_{v \in U_i^{3\alpha}} |\operatorname{cov}_\alpha(v, U_i^{3\alpha})|$ .
6:    $V_i \leftarrow \operatorname{cov}_{3\alpha}(v_i, U_i^{3\alpha})$ .
7:    $B_{v_i} \leftarrow |V_i|(B - 2) + 2$ 
8:    $R \leftarrow R \cup \{v_i\}$ .
9:    $i \leftarrow i + 1$ .
10: end while [Fig. 1(b)]
11:  $T^g \leftarrow \mathbf{GlobTree}(R, \{B_v\}_{v \in R})$ . [Fig. 1(c)]
12: for  $1 \leq i \leq q$ 
13:    $T_i \leftarrow$  Tree spanning  $V_i$  of degree at most  $B$  and minimum height. [Fig. 1(d)]
14:   Replace  $v_i$  by  $T_i$ , and distribute the edges in  $T^g$  incident on  $v_i$  over the nodes of  $T_i$  so that the maximum degree of any node in  $T_i$  is minimized. [Fig. 1(e)]
15: end for
16: return resulting tree  $T^{\text{apx}}$ . [Fig. 1(f)]

```

We then compute a tree $T^g = \mathbf{GlobTree}(R, \{B_v\}_{v \in R})$ on the nodes R of low height. See Algorithm 1 for the details, and Figure 1(c) for an illustration on the input instance of Figure 1(a).

Local structure. Let $u \in R$ be a representative node in R and let U be its corresponding set in the partition. We then let T_u be a tree spanning with the nodes of U of minimum height, i.e. all internal nodes in T_u except for those with only leaves as children have degree B . This is shown in Figure 1(d).

Finally, we compute the final tree T^{apx} by taking the global tree T^g and replacing each node $v \in R$ by the tree T_v . We distribute the edges that are incident to v in T^g over the nodes of T_v evenly, such that the maximum degree of any node of T_v is as small as possible. This is shown in Figure 1(e), with the final solution in Figure 1(f).

A listing outline of the algorithm is shown in Algorithm 2. We will show that its output is always a tree of degree no more than B . We do a binary search over Δ to obtain a tree of minimum diameter. In the following we analyze the performance of the algorithm, assuming the correct value for Δ is fed to Algorithm 2.

2.3. Performance Ratio

THEOREM 1. *Suppose that there is a tree T^* with maximum node-degree B and diameter Δ . Then Algorithm $\text{BDST}(G, \Delta)$ produces a tree T^{apx} with maximum node-degree B and diameter $O(\sqrt{\log_B n} \cdot \Delta)$.*

Theorem 1 is the main result we are trying to prove. We prove it at the end of this section, using a sequence of lemmas which follow.

LEMMA 1. *The maximum degree of T^{apx} is no more than B .*

PROOF. Let $u \in R$ be a representative node and let U be its corresponding vertex set in the computed partition of V . Recall the definition of the degree-bound B_u for node u from (2). Algorithm **GlobTree** guarantees that vertex $u \in R$ has degree at most B_u in T^g for all $u \in R$.

The local tree T_u has $|U|$ nodes each with capacity B and there are exactly $|U| - 1$ edges in T_u . Hence the total available capacity of the nodes in U for edges outside T_u is

$$|U| \cdot B - 2 \cdot (|U| - 1) = B_u.$$

This means that there is a way of distributing the edges of T^g that are incident to node u over all nodes of T_u such that the maximum degree in T^{apx} is at most B . \square

We now prove that T^{apx} has diameter $O(\sqrt{\log_B n} \cdot \Delta)$. We say that a path P in the tree T^{apx} is an *rl-path* if P is the unique path connecting a leaf to the root in T^{apx} . We also say that an edge $uv \in E$ is *short* if $u, v \in V_i$ for some $1 \leq i \leq q$, and *long* otherwise. Our proof of the diameter bound has two parts: the first part shows that the maximum number of long edges on any rl-path in T^{apx} is $O(\sqrt{\log_B n})$. The second part shows that there are $O(\log_B n)$ short edges on any rl-path in T^{apx} . We will show that this suffices, using the facts that the length of any edge in our input graph is at most Δ , and the length of a short edge in G is at most 6α (using triangle inequality).

First, we prove that any rl-path contains at most $O(\sqrt{\log_B n})$ long edges. We begin by creating a partition of V using T^* 's structure. We root T^* at v_1^* (chosen arbitrarily), and let V_1^* be the set of nodes $u \in V$ such that the unique (v_1^*, u) -path in T^* has length at most α . We let $S^* = \{v_1^*\}$, and let the set of *uncovered nodes* be $U = V \setminus V_1^*$ initially.

We continue until there are no uncovered nodes remaining. In iteration $i > 1$, let $v_i^* \in U$ be an uncovered node of smallest height in T^* (i.e. v_i^* 's parent in T^* is already covered). We then say that a node u is *covered* by v_i^* if u is a descendant of v_i^* (in T^*) and the length of the path from v_i^* to u in T^* is at most α . We let V_i^* be the set of nodes in U that are covered by v_i^* . We remove V_i^* from U and repeat.

Assume that the final partition has sets V_1^*, \dots, V_p^* and representatives $R^* = \{v_1^*, \dots, v_p^*\}$. Since the subtree $T^*[V_i^*]$ of T^* induced by the nodes of V_i^* is connected, a counting argument shows that the nodes in V_i^* have at most

$$|V_i^*| \cdot (B - 2) + 1$$

children from $V \setminus V_i^*$ in T^* for $i \geq 2$. Similarly, V_1^* has at most $|V_1^*| \cdot (B - 2) + 2$ children in T^* . Let

$$(3) \quad B_{v_i^*}^* = |V_i^*| \cdot (B - 2) + 2$$

for all $1 \leq i \leq p$ and let T be the tree produced by **GlobTree**(R^* , $\{B_v^*\}_{v \in R^*}$).

DEFINITION 1. $\{(\bar{v}_i, \bar{V}_i)\}_{i=1}^p$ is called a *proper collection of V* for a given node set V if the following conditions hold:

1. $\bar{V}_i \subset V$ and $\bar{v}_i \in \bar{V}_i$ for all $1 \leq i \leq p$.

2. $\bar{V}_i \cap \bar{V}_j = \emptyset$ for all $1 \leq i < j \leq p$.
3. $|\bar{V}_1| \geq \dots \geq |\bar{V}_p|$.
4. $\text{dist}_l(\bar{v}_i, u) \leq \alpha$ for all $1 \leq i \leq p$ and for all $u \in \bar{V}_i$.

The following lemma is useful in order to prove that the height of the global tree T^g is at most that of T .

LEMMA 2. *Let $\{(v_i, V_i)\}_{i=1}^q$ be a partition of V together with a corresponding set of representatives created by steps 1–10 of Algorithm 2. Let $\{(\bar{v}_i, \bar{V}_i)\}_{i=1}^p$ be a proper collection of G as defined in Definition 1. We then must have*

$$(4) \quad \sum_{i=1}^j |V_i| \geq \sum_{i=1}^j |\bar{V}_i|$$

for all $1 \leq j \leq \max\{q, p\}$.

PROOF. We prove the lemma by induction over $|V| = n$. For $n = 1$ the lemma is trivially satisfied since in this case $V_1 = \bar{V}_1 = V$. For $n > 1$, assume that the lemma holds for all node sets with at most $n - 1$ nodes.

Assume now, for the sake of contradiction, that the lemma does not hold. Let j be the minimum index such that $\sum_{i=1}^j |V_i| < \sum_{i=1}^j |\bar{V}_i|$. Then there must exist an index $j_0 \leq j$ such that

$$\bar{V}_{j_0} \not\subseteq \bigcup_{1 \leq i \leq j} V_i$$

and hence $\bar{V}_{j_0} \not\subseteq \text{cov}_{3\alpha}(v_i, V)$ for all $1 \leq i \leq j$. Notice that this implies $\bar{v}_{j_0} \in U_j^\alpha$.

Now consider the application of the induction hypothesis for the set of nodes $V' = V \setminus \bar{V}_{j_0}$. Since $\bar{V}_{j_0} \cap \text{cov}_\alpha(v_i, V) = \emptyset$ for all i , the application of our algorithm with V' yields the exact same set of the first $j - 1$ representatives v_1, v_2, \dots, v_{j-1} and the corresponding subsets $\{V_i \setminus \bar{V}_{j_0}\}_{i=1}^{j-1}$ of V' . Note that $\{\bar{V}_i\}_{i=1}^p \setminus \{\bar{V}_{j_0}\}$ is a proper collection of V' . Therefore, by the induction hypothesis, we conclude that

$$(5) \quad \sum_{i=1}^{j-1} |V_i \setminus \bar{V}_{j_0}| \geq \sum_{i=1}^j |\bar{V}_i| - |\bar{V}_{j_0}|.$$

Let us now lower bound the difference $\sum_{i=1}^j |V_i| - \sum_{i=1}^{j-1} |V_i \setminus \bar{V}_{j_0}|$.

This difference can be expressed as the sum of two terms: the size of the set V_j and the increase of the sizes of the first $j - 1$ sets of our partition. Hence, we obtain

$$(6) \quad \sum_{i=1}^j |V_i| \geq \sum_{i=1}^{j-1} |V_i \setminus \bar{V}_{j_0}| + \left| \bar{V}_{j_0} \cap \bigcup_{1 \leq i < j} \text{cov}_{3\alpha}(v_i, V) \right| + |V_j|.$$

Observe that in the j th iteration of our algorithm we could have chosen \bar{v}_{j_0} as a representative instead of v_j since $\bar{v}_{j_0} \in U_j^\alpha$. Therefore, we must have that

$$|V_j| = |\text{cov}_\alpha(v_j, U_j^{3\alpha})| \geq |\text{cov}_\alpha(\bar{v}_{j_0}, U_j^{3\alpha})|.$$

Using (5) together with (6) and noting that

$$(7) \quad \left| \overline{V}_{j_0} \cap \bigcup_{1 \leq i < j} \text{cov}_{3\alpha}(v_i, V) \right| + |\text{cov}_{\alpha}(\overline{v}_{j_0}, U_j^{3\alpha})| \geq |\overline{V}_{j_0}|$$

finally yields

$$\sum_{i=1}^j |V_i| \geq \sum_{i=1}^j |\overline{V}_i|.$$

This contradicts our assumption, and the lemma follows. \square

COROLLARY 1. *Let $\{V_i\}_{i=1}^q$ be the partition of V generated by steps 1–10 of Algorithm 2, and let $\{V_i^*\}_{i=1}^p$ be the partition of V generated from the optimum tree. Let π be a permutation of $\{1, \dots, q\}$ such that*

$$|V_{\pi(1)}| \geq \dots \geq |V_{\pi(q)}|.$$

For all $1 \leq j \leq \max\{l, p\}$, we have

$$(8) \quad \sum_{i=1}^j |V_{\pi(i)}| \geq \sum_{i=1}^j |V_i^*|.$$

PROOF. The statement in (8) clearly holds for the partition $\{V_i\}_{i=1}^q$ generated by steps 1–10 of Algorithm 2, noting that $\{(v_i^*, V_i^*)\}_{i=1}^q$ is a proper collection of V as defined in Definition 1.

The corollary follows by observing that reordering the sets of the partition by non-increasing size increases the left-hand side of (8) and does not change the right-hand side. \square

We can now prove that the height of the global tree T^g is at most the height of the tree T .

LEMMA 3. *When T is constructed from T^* by $\mathbf{GlobTree}(R, \{B_v^*\}_{v \in R^*})$, we have $\text{height}(T^g) \leq \text{height}(T)$.*

PROOF. We say that the *level* of node v of T is the number of edges in the unique path from the root of T to v . We now claim that the level of node v_i in T^g is at most the level of node v_i^* in T for all $1 \leq i \leq p$. We use induction over i to prove the claim.

The claim is clear for $i = 1$. For $i > 1$, assume that $\mathbf{GlobTree}$ connects node v_i^* to node v_s^* for some $s < i$. It follows from (2), (3), and Corollary 1 that

$$\sum_{j=1}^s B_j^* \leq \sum_{j=1}^s B_j$$

and hence there must exist a $1 \leq s' \leq s$ such that $d_{s'} > 0$ in $\mathbf{GlobTree}$ at the time when node v_i is connected. By the induction hypothesis, we know that the level of $v_{s'}$ in T^g is

at most that of node v_s^* in T . It follows from the definition of `GlobTree` that the level of v_s^* is at most the level of v_s^* . Hence, the level of v_i in T^g is at most the level of v_i^* in T and this finishes the induction.

Observe that the height of T^g is equivalent to the level of node v_q in T^g , and that the height of T equals the level of v_q^* in T . This implies the lemma. \square

LEMMA 4. *Let T^g be a tree returned by `GlobTree`($R, \{B_v\}_{v \in R}$). Then T^g must be a tree of minimum height among all trees that satisfy the given degree constraints.*

PROOF. Given a tree T , we define the following total order of the nodes in T . The order is a breadth-first-search order, with the refinement that the nodes of each level are ordered in non-increasing order of their corresponding sets V_i . In particular, the nodes of T are ordered v_1, v_2, \dots, v_q such that if $i < j$ then either $\text{level}(v_i) < \text{level}(v_j)$ or $\text{level}(v_i) = \text{level}(v_j)$ and $|V_i| \geq |V_j|$. By construction of T^g , we have that if $i < j$ in the total order of the nodes in T^g , then $|V_i| \geq |V_j|$, regardless of their levels. Moreover, every tree of minimum height for which this holds must have the same height as $\text{height}(T^g)$.

Assume for the sake of contradiction that there exists a tree T' such that $\text{deg}_{T'}(v_i) \leq B_{v_i}$ for all $1 \leq i \leq q$ and $\text{height}(T') < \text{height}(T^g)$. Let v'_1, \dots, v'_q be the total order induced by T' , as defined above. By the observation in the preceding paragraph, for some $i < j$, we have $|V'_i| < |V'_j|$. We call this an *inversion*, and, without loss of generality, assume that T' is a tree with the fewest number of inversions among all trees that satisfy the degree constraints and have height less than $\text{height}(T^g)$.

We show that we can reduce the number of inversions in T' without increasing the tree's height. This contradicts the inversion-minimality of T' .

Let $\langle v'_i, v'_j \rangle$ be an inversion in T' . We swap labels: relabel node v'_i as v'_j and relabel v'_j as v'_i . The resulting tree may now violate the degree constraints at node v'_i . We counter this by moving a sufficient number of v'_i 's children to v'_j . This does not increase $\text{height}(T')$, and reduces the number of inversions in T' , which is a contradiction. \square

LEMMA 5. *Any rl-path in T^{apx} has at most $\sqrt{\log_B n}$ long edges.*

PROOF. Let d^* denote the maximum number of long edges on any rl-path in T^* . It follows from Lemma 4 that $\text{height}(T) \leq d^*$ and hence, together with Lemma 3, we have that $\text{height}(T^g) \leq d^*$.

By the construction of the partition V_1^*, \dots, V_p^* , we know that an rl-path P in T^* that contains d^* long edges must have length at least $\alpha \cdot d^*$. Since T^* has diameter at most Δ it then follows that $d^* \leq \Delta/\alpha = \sqrt{\log_B n}$ by our choice of α . \square

Lemma 5 bounds from above the contribution of long edges to the diameter of T^{apx} . We bound the contribution of short edges in the next lemma. For an rl-path P in T^{apx} , let $|P|_s$ denote the number of short edges in P .

LEMMA 6. *Let P be an arbitrary rl-path in T^{apx} . Then*

$$|P|_s = O(\log_B n).$$

PROOF. Let P_1 and P_2 be two rl-paths in T^{apx} , and let P_1^g and P_2^g be their images in T^g , i.e. $P_1^g = \langle v_1^1, \dots, v_{l_1}^1 \rangle$ and $P_2^g = \langle v_1^2, \dots, v_{l_2}^2 \rangle$.

We define a relation $<$ on two rl-paths as follows. We say that $P_1 < P_2$ if $|V_j^1| \geq |V_j^2|$ for all $1 \leq j \leq \max\{l_1, l_2\}$, with $|V_j^i| = 0$ if V_j^i does not exist. By construction of T^g , for every two paths P_1 and P_2 at least one of the following holds: $P_1 < P_2$ or $P_2 < P_1$. Moreover, if $P_1 < P_2$ then since T^g is a minimum height spanning tree, $l_2 \leq l_1 \leq l_2 + 1$.

Recall that T_{v_i} denotes the *local* tree that spans the nodes of V_i . For the purpose of this proof, we assume that all edges of the form (v_i, v_j) in T^g such that v_i is a parent of v_j are attached to leaf nodes in T_i . This assumption only increases the number of short edges in rl-paths, and hence is valid.

Without loss of generality assume that $P_1 < P_2$. Since each T_{v_i} is a balanced $(B-1)$ -ary tree, we have $|P_2|_s \leq |P_1|_s + |P_1^g| \leq |P_1|_s + \log_B n$, where the second inequality follows because T^g is a minimum height spanning tree. We also have $|V_i^1| \leq |V_{i-1}^2|$ for $i > 1$, by construction of T^g . Therefore, $|P_1|_s \leq |P_2|_s + |P_1^g| + \text{height}(T_{v_1}) \leq |P_2|_s + 2 \log_B n$. Hence, there exists a γ such that $|P|_s \in [\gamma, \gamma + 2 \log_B n]$ for all rl-paths P in T^{apx} .

Observe that on any rl-path P in T^{apx} , all but at most $O(\log_B n)$ of the short edges must be incident to nodes of degree B . This follows from the fact that T^g has $O(\log_B n)$ levels. Since there are n nodes in our graph, we must have that $\gamma = O(\log_B n)$. This finishes the proof of the lemma. \square

We are now ready to prove Theorem 1.

PROOF OF THEOREM 1. Lemma 1 shows that T^{apx} has maximum degree B .

Let P be a path in T^{apx} . We now bound the total length of P . The total length of P is the sum of the total length of long edges that belong to P , and the total length of the short edges that belong to P . By the triangle inequality a long edge in T^{apx} has length no more than Δ , since the graph has a spanning tree of diameter Δ and we are assuming we have the correct guess of Δ . By Lemma 5, the number of long edges that belong to P is at most $2\sqrt{\log_B n}$. Therefore, the contribution of long edges to the length of P is no more than $2\Delta\sqrt{\log_B n}$.

By the triangle inequality and (1), short edges in T^{apx} have length no more than $6\alpha = 6\Delta/\sqrt{\log_B n}$. Lemma 6 bounds the number of short edges in P to be at most $O(\log_B n)$, so the total contribution of short edges to the length of P is no more than $O(\alpha \log_B n) = O(\Delta\sqrt{\log_B n})$.

All edges must be either long or short, and therefore the length of P is $O(\Delta\sqrt{\log_B n})$. Since the above arguments hold for all paths P in T^{apx} , we conclude that the diameter of T^{apx} is at most $O(\Delta\sqrt{\log_B n})$. This completes the proof of the theorem. \square

3. Open Questions. As mentioned in the Introduction, the problem of computing a tree of minimum diameter is closely related to that of computing a tree that minimizes the maximum dilation. An approximation algorithm for degree-bounded minimum dilation spanning trees is still open.

Our algorithm crucially uses the fact that the input graph is a complete metric. In particular, our algorithm does not work if we are given an (incomplete) input graph and

a metric induced by the edge-lengths of its edges (and we are enforced to use only edges from the input graph). Thus, an improvement over the bicriteria ($O(\log n)$, $O(\log^2 n)$) approximation algorithm from [18] for this case is still open.

References

- [1] E. M. Arkin, M. A. Bender, S. P. Fekete, J. S. B. Mitchell, and M. Skutella. The freeze-tag problem: how to wake up a swarm of robots. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms*, pages 568–577, 2002.
- [2] E. M. Arkin, M. A. Bender, and D. Ge. Improved approximation algorithms for the freeze-tag problem. In *Proceedings, ACM Symposium on Parallelism in Algorithms and Architectures*, pages 295–303, 2003.
- [3] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings, IEEE Symposium on Foundations of Computer Science*, pages 184–193, 1996.
- [4] Y. Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings, ACM Symposium on Theory of Computing*, pages 161–168, 1998.
- [5] F. Bauer and A. Varma. Degree-constrained multicasting in point-to-point networks. In *Proceedings, IEEE INFOCOM*, pages 369–376, 1995.
- [6] L. Cai and D. G. Corneil. Tree spanners. *SIAM Journal on Discrete Mathematics*, 8(3):359–387, 1995.
- [7] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *Proceedings of SIGCOMM*, pages 55–68, 2001.
- [8] S. E. Deering and D. R. Cheriton. Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1990.
- [9] S. Deering, D. Estrin, and D. Farinacci. An architecture for wide-area multicast routing. In *Proceedings of SIGCOMM*, pages 126–135, 1994.
- [10] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings, ACM Symposium on Theory of Computing*, pages 448–455, 2003.
- [11] M. Fürer and B. Raghavachari. Approximating the minimum-degree Steiner tree to within one of optimal. *Journal of Algorithms*, 17(3):409–423, 1994.
- [12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, CA, 1979.
- [13] L. Gouveia and T.L. Magnanti. Network flow models for designing diameter-constrained minimum-spanning and steiner trees. *Networks*, 41:159–173, 2003.
- [14] R. Hassin and A. Levin. Minimum spanning tree with hop restrictions. *Journal of Algorithms*, 48:220–238, 2003.
- [15] R. Hassin and A. Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53(2):109–111, 1995.
- [16] R. Krishnan and B. Raghavachari. The directed minimum-degree spanning tree problem. In *Proceedings, Foundations of Software Technology and Theoretical Computer Science*, pages 232–243, 2001.
- [17] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt III. Bicriteria network design problems. *Journal of Algorithms*, 28(1):142–171, 1998.
- [18] R. Ravi. Rapid rumor ramification: approximating the minimum broadcast time. In *Proceedings, IEEE Symposium on Foundations of Computer Science*, pages 202–213, 1994.
- [19] S. Voss. The Steiner tree problem with hop constraints. *Annals of Operations Research*, 86:321–345, 1999.
- [20] W. De Zhong. A copy network with shared buffers for large-scale multicast ATM switching. *IEEE/ACM Transactions on Networking*, 1(2):157–165, 1993.