A MATTER OF DEGREE: IMPROVED APPROXIMATION ALGORITHMS FOR DEGREE-BOUNDED MINIMUM SPANNING TREES [*]

J. KÖNEMANN AND R. RAVI

Abstract. In this paper, we present a new bicriteria approximation algorithm for the *degree-bounded minimum spanning tree* problem. In this problem, we are given an undirected graph, a nonnegative cost function on the edges, and a positive integer B^*, and the goal is to find a minimum cost spanning tree T with maximum degree at most B^*. In an n-node graph, our algorithm finds a spanning tree with maximum degree $O(B^* + \log n)$ and cost $O(\mathtt{opt}_{B^*})$ where $\mathtt{opt}_{B^*}$ is the minimum cost of any spanning tree whose maximum degree is at most B^*. Our algorithm uses ideas from Lagrangean duality. We show how a set of optimum Lagrangean multipliers yields bounds on both the degree and the cost of the computed solution.

Key words.

AMS subject classifications.

1. Introduction.

1.1. Motivation and formulation.

In the design of computer networks a fundamental problem is that of transmitting a single information packet from a given source-host to a set of recipient-hosts. This problem is widely known as the *broadcast* or *multicast* problem, depending on whether we want to transmit the packet to all other hosts or to a specific subset of recipients. Both problems have been widely studied [3, 6, 18]. In particular, it is believed that the multicast problem will play an increasingly important role in data networks.

A naive solution to the multicast problem would be to implement it as a series of unicasts. In other words, the source sends a single packet to every recipient host. The routing is done independently for each of the unicasts. However, the cost of this approach in terms of bandwidth consumption becomes unacceptable if the number of hosts in the multicast group grows.

Graph theoretic ideas have turned out to be essential in the design of efficient network routing protocols. A physical network can be modeled as a complete graph where each host is associated with a node and an edge represents the *virtual link* between the corresponding hosts. Usually, edges of that graph are annotated by the *transmission delay* of the corresponding virtual link. A standard solution to broadcasting and multicasting problems is then to send packets along the edges of a minimum spanning tree rooted at the source node [18]. Every internal node duplicates the incoming message and sends it to its children.

However, a spanning tree might have a high *fan-out* out at certain internal nodes. Switches in point-to-point networks may vary in their ability to support multicasting. That is, some routers may not support multicasting at all and others may only support a limited number of copies they can make of an incoming packet [20]. Bauer and Varma [1] show that it is natural to model switch capabilities by node degrees in graphs.

The preceding discussion suggests that a solution to the multicasting problem should minimize the total transmission delay *and* the maximum degree of a vertex

[*]GSIA, Carnegie Mellon University, Pittsburgh, PA 15213. EMail: {jochen,ravi}@cmu.edu. Research supported in part by NSF grant 96-25279.

in the computed solution. Traditional approaches for this kind of *bicriteria* problem often compute the minimum-cost solution under a linear combination of the two cost measures [14, 17]. However, in the case of very disparate objectives these techniques usually do not produce useful solutions.

In this paper, we address a natural budgeted version of the *degree-bounded minimum spanning tree problem* (BMST). Here, we are given an undirected graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}^+$ and a positive integer $B \geq 2$. We would like to find a spanning tree T of maximum vertex degree at most B and minimum cost. This formulation was first introduced in [17] and can be modeled by the following integer linear program.

$$\begin{aligned} \mathtt{opt}_B = \min \quad & \sum_{e \in E} c_e x_e && \text{(IP)} \\ \text{s.t} \quad & x(\delta(v)) \leq B \quad \forall v \in V && (1.1) \\ & x \in \mathrm{SP}_G \\ & x \text{ integer} \end{aligned}$$

Here, $\delta(v)$ denotes the set of all edges of E that are incident to v and SP_G is the spanning tree polyhedron, that is, the convex hull of edge-incidence vectors of spanning trees of G. We note that complete descriptions of SP_G are known in the literature ([2, 4]).

1.2. Previous work and our result. Ravi et al. [17] showed how to compute a spanning tree T of maximum degree $O(B \log(\frac{n}{B}))$ and total cost at most $O(\log n)\, \mathtt{opt}_B$. They generalize their ideas to Steiner trees, generalized Steiner forests and the node-weighted case.

Another result that is related to our work is given in a paper by Khuller, Raghavachari and Young [11]. The authors show how to compute a spanning tree of n points in the plane that has degree at most 3 (4) and cost at most 1.5 (1.25) that of a minimum-cost spanning tree (without any degree constraints).

While the approximation factor of $O(\log n)$ on the cost of the solution cannot be improved substantially (via reductions from the set covering problem [12]) in the node-weighted case, improvements for the edge-weighted case were left open in [17]. Our main result is such an improvement and is stated in the following theorem. We denote the degree of a node v in tree T by $\delta_T(v)$. Let the maximum node degree in a tree T be denoted by $\Delta(T)$.

THEOREM 1.1. *There is a polynomial-time approximation algorithm that, given a graph $G = (V, E)$, a nonnegative cost function $c : E \to \mathbb{R}^+$, a constant $B^* \geq 2$ and a parameter $\omega > 0$, computes a spanning tree T such that*

1. *$\Delta(T) \leq (1 + \omega)bB^* + \log_b n$ for any arbitrary constant $b > 1$, and*
2. *$c(T) < (1 + 1/\omega)\, \mathtt{opt}_{B^*}$.*

For instance, choosing $\omega = 1/2$ and $b = 2$ would yield a tree with degree at most $3B^* + \log_2 n$ and cost at most $3\, \mathtt{opt}_{B^*}$.

1.3. Technique: Lagrangean Duality. Our algorithm builds on the Lagrangean dual of (IP) resulting from *dualizing* the degree constraints. We denote its value by $\mathtt{opt}_{LD(B)}$.

$$\mathtt{opt}_{LD(B)} = \max_{\lambda \geq 0} \min_{T \in \mathrm{SP}_G} \{c(T) + \sum_{v \in V} \lambda_v (\delta_T(v) - B)\}. \qquad \text{(LD(B))}$$

For any fixed $\lambda \geq 0$, an optimum integral solution to (IP) is a feasible candidate for attaining the inner minimum above. Since the maximum degree of such a solution is at most B and $\lambda \geq 0$, it follows that $\texttt{opt}_{LD(B)}$ is a lower bound on $\texttt{opt}_B$.

PROPOSITION 1.2. *[15]* $\texttt{opt}_{LD(B)} \leq \texttt{opt}_B$

The interesting fact is that $\texttt{opt}_{LD(B)}$ can be computed in polynomial time [15]. The result is a vector λ^B of optimum Lagrangean multipliers on the nodes and a set of optimum trees $\mathcal{O}^{\mathcal{B}}$, all of which achieve the inner minimum for this set of multipliers. In other words, every tree $T^B \in \mathcal{O}^B$ minimizes the following objective:

$$c(T^B) + \sum_{v \in V} \lambda_v^B (\delta_{T^B}(v) - B).$$

Given λ^B, the task of finding a tree T that minimizes the above objective function is called the Lagrangean subproblem of LD(B).

Using $c^{\lambda^B}(uv) = c(uv) + \lambda_u^B + \lambda_v^B$ the last expression can be restated as

$$c^{\lambda^B}(T^B) - B \sum_{v \in V} \lambda_v^B \tag{1.2}$$

Notice that for a given λ^B and B, the second term is a constant. Hence, any minimum spanning tree of G under cost c^{λ^B}, denoted by $\texttt{MST}(G, c^{\lambda^B})$, is a solution for T.

An important feature of our algorithm is to relax the degree constraints slightly from B to $(1+\omega)B$ for some fixed $\omega > 0$ and show that there is a tree $T \in \mathcal{O}^{(1+\omega)B}$ that satisfies the conditions of Theorem 1.1.

This paper is organized as follows: in Section 2 we review results on the related *minimum-degree spanning tree problem.* In particular, we present the fundamental ideas from [5, 7]. In Section 3, we state our algorithm. Finally, we give the analysis of our procedure in Section 4.

2. Minimum Degree Spanning Trees.

Related to the BMST problem is the problem of minimizing the maximum degree of a spanning tree in some graph G (MDST). This problem is $\mathcal{NP}$-hard since the *Hamiltonian path* problem is a special case. In fact, it is $\mathcal{NP}$-complete to decide for any $k \geq 2$ whether G contains a spanning tree of maximum degree k [8].

Fürer and Raghavachari presented an approximation algorithm with an additive performance guarantee of one [7]: i.e., they describe a polynomial time algorithm that finds a spanning tree T of G such that $\Delta(T) \leq \Delta^* + 1$, where Δ^* denotes the minimum possible maximum degree over all spanning trees. In the same paper the authors also give a local search algorithm for the MDST problem. This approach yields a tree with maximum degree at most $b\Delta^* + \log_b n$ for any constant $b > 1$. Later, Fischer noted that this procedure can be adapted to find a minimum-cost spanning tree of approximately minimum maximum degree in an edge-weighted graph [5].

The local search algorithms from [5, 7] play an important role in this paper. In this section we show a minor strengthening of these results that is crucial to the analysis of our algorithm.

2.1. A local improvement algorithm.

In this section, we explain the basic ideas from the local search algorithm for the MDST problem. We state the algorithm since we use it later. The procedure starts with a spanning tree T and tries to improve it by *swapping* non-tree edges against tree edges.

Definition 2.1. *Given a tree T and a non-tree edge uv, let $\mathcal{C}(uv)$ be the unique cycle in $T \cup \{uv\}$ and let $wz \in \mathcal{C}(e)$. We call the swap $\langle uv, wz\rangle$ an* improvement for *w if*

$$\max\{\delta_T(u), \delta_T(v)\} + 1 < \delta_T(w).$$

If an edge swap $\langle uv, wz\rangle$ is an improvement step for either w or z then the maximum degree of the nodes u, v, w and z decreases as a result of the swap; We call such a swap simply an improvement.

The algorithm in [7] performs improvement steps as long as possible. In fact, it is not hard to see that starting with an arbitrary tree, the number of possible improvements is finite. We end up with a *locally optimal tree.*

Definition 2.2. *A tree T is called* locally optimal *(LOT) if no vertex degree can be decreased by applying an improvement swap.*

Computing a locally optimal tree might be too ambitious a goal however. In fact, it is not known how to do this in polynomial time. However, the analysis in [7] shows that it is enough to compute a *pseudo-optimal* tree.

Definition 2.3. *A tree T of maximum degree $\Delta(T)$ is called* pseudo-optimal *(POT) if for all vertices v with $\Delta(T) - \lceil \log_b n \rceil \leq \delta_T(v) \leq \Delta(T)$, no improvement step for v is applicable. Here b is an arbitrary constant bigger than one.*

Fischer's adaptation [5] of the algorithm from [7] computes a minimum-cost spanning tree of approximately minimum maximum degree. To obtain his algorithm we have to make two small changes to the procedure from [7]. First, instead of starting with an arbitrary spanning tree, we start with a minimum-cost spanning tree. Second, an improvement step must be *cost neutral.* That is, for an improvement step $\langle uv, wz\rangle$ to be applicable we must have $c_{uv} = c_{wz}$. Algorithm 1 states the procedure.

Algorithm 1 The algorithm `PLocal` computes a pseudo-optimal tree.

```
1: Given: graph $G = (V, E)$ and cost function $c : E \to \mathbb{R}^+$
2: $T \leftarrow$ MST$(G, c)$
3: while $T$ is not pseudo optimal do
4:    Identify cost neutral improvement $\langle uv, wz\rangle$
5:    $T \leftarrow T - wz + uv$
6: end while
```

2.2. Analysis and running time. In what follows we highlight and strengthen the major ideas of the analysis from [5, 7]. The strengthening is due to Éva Tardos [19] and leads to a shorter and simpler proof of Lemma 4.5 than the one that appeared in the extended abstract [13].

The fundamental underlying proof idea for the unweighted problem is based on an averaging argument that we present here. Let a set $W \subseteq V$ be such that for a given graph $G = (V, E)$, the graph $G[V - W]$ has t connected components. A spanning tree of G has to connect these t components *and* the nodes from W. We need exactly $t + |W| - 1$ edges going between the nodes of W and the t connected components to accomplish this. Each of these edges must be incident to a node from W. Hence averaging yields a lower bound of $(t + |W| - 1)/|W|$ on the maximum degree Δ^* of T.

Proposition 2.4. *[7] Let W be a set of size w whose removal splits G into t components. Then $\Delta^* \geq \lceil \frac{w+t-1}{w} \rceil$.*

We now turn to the weighted case, i.e. the *minimum-degree minimum-cost spanning tree* problem. The above mentioned strengthening of the results from [5] is based on the following definitions.

DEFINITION 2.5. *Given an undirected graph $G = (V, E)$ and a non-negative cost function c on the edges, let $\mathcal{O}^c$ be defined as*

$$\mathcal{O}^c = \{T : T \textit{ is an MST under cost } c\}.$$

In the following we will be talking about convex combinations of spanning trees. Hence we introduce some further simplifying notation.

DEFINITION 2.6. *Let $T_c^\alpha = \sum_{T \in \mathcal{O}^c} \alpha_T T$ be a convex combination of minimum-cost spanning trees of G with respect to cost function c, i.e. $\alpha_T \geq 0$ for all T and $\sum_{T \in \mathcal{O}^c} \alpha_T = 1$. We denote the* fractional degree *of vertex v in T_c^α by*

$$\delta_c^\alpha(v) = \sum_{T \in \mathcal{O}^c} \alpha_T \delta_T(v).$$

Finally we define the minimum maximum degree of convex combinations of spanning trees.

DEFINITION 2.7. *Given $G = (V, E)$ and a non-negative cost function c on the edges, let Δ_c^* denote the minimum maximum degree of any convex combination of minimum-cost spanning trees, i.e.*

$$\Delta_c^* = \min_{\textit{convex comb. } \alpha} \max_{v \in V} \delta_c^\alpha(v).$$

The following easy proposition will be used in the later analysis.

PROPOSITION 2.8. *[7] For any constant $b > 1$ and a tree T, let S_d be the set of nodes that have degree at least d in T. Then, there is a*

$$d \in \{\Delta_T - \lceil \log_b n \rceil + 1, \ldots, \Delta_T\}$$

such that $|S_{d-1}| \leq b|S_d|$.

The main theorem is the following.

THEOREM 2.9. *[5, 7] If T is a pseudo-optimal MST, then $\Delta_T < b\Delta_c^* + \lceil \log_b n \rceil$ for any constant $b > 1$. Moreover, a pseudo-optimal MST can be computed in polynomial time.*

Proof. Given a constant $b > 1$, choose d as in Proposition 2.8. That is, we have $|S_{d-1}| \leq b|S_d|$. Recall that S_d contains the nodes of degree at least d in the tree T.

Removing S_d from T leaves us with a forest F. Let $\widehat{G}$ be obtained from G by contracting each connected component of F. It is now easy to see that every minimum-cost spanning tree of G contains a minimum-cost spanning tree of $\widehat{G}$ (e.g., every edge added by Kruskal's algorithm for finding a minimum-cost spanning tree for G is feasible for a minimum-cost spanning tree of $\widehat{G}$ if it were not contracted in the formation of $\widehat{G}$).

Let $(u, v) \in E - T$ be an edge that connects two components of F such that $u, v \notin S_{d-1}$, i.e. both u and v have degree at most $d - 2$. We claim that such an edge cannot be included in any minimum spanning tree of $\widehat{G}$. To see that, let $P_{u,v}^T$ be the edges of the unique u, v-path in T and let $\widehat{P_{u,v}^T}$ be the subset of the edges of $P_{u,v}^T$ that are in $\widehat{G}$.

It follows from the pseudo-optimality of T that the cost of edge (u,v) must be higher than the cost of each edge from $\widehat{P^T_{u,v}}$. For otherwise, (u,v) can be swapped in place of another edge of the same or higher cost in $\widehat{P^T_{u,v}}$ and all such edges are incident to at least one node in S_{d-1}, leading to an improvement. This means (u,v) cannot be a part of any minimum spanning tree of $\widehat{G}$. Equivalently, a minimum-cost spanning tree of G must use edges incident to S_{d-1} to connect the components of F and the nodes of S_d.

By the definition of S_d, we know that F has at least

$$|S_d|d - 2(|S_d| - 1) = |S_d|(d-2) + 2$$

trees. This follows from an easy counting argument after observing that every node in S_d has degree at least d in T and there are at most $|S_d| - 1$ edges going between nodes of S_d.

This means that we need at least

$$|S_d|(d-2) + 2 + |S_d| - 1 = |S_d|(d-1) + 1$$

edges to connect up the components of F and the nodes of S_d in *any spanning tree.* By the preceeding argument each of these edges has to be incident to at least one node of degree at least $d-1$ in an MST. Hence the the average degree of a node of S_{d-1} in any MST is

$$\frac{|S_d|(d-1)+1}{|S_{d-1}|}.$$

Moreover, the average degree of a node in S_{d-1} in any *convex combination* of MSTs is also at least the above ratio. Since Δ^*_c denotes the minimum possible maximum degree of any fractional MST, it follows using our choice of index d from Proposition 2.8 that

$$\Delta^*_c > \frac{d-1}{b}.$$

Using the range of d we obtain $\Delta(T) < b\Delta^*_c + \lceil \log_b n \rceil$. The results in [5, 7] show a lower-bound on the degree of any MST. The extention to fractional MST's is the mentioned strengthening [19] of the previous ideas.

For the running time we follow [7]. Note that each improvement step can be implemented in polynomial time. We need to bound the number of iterations. The proof uses a potential function argument. Define the potential of a vertex v as

$$\Phi(v) = 3^{\delta_T(v)}$$

where T is the current tree. The total potential is the sum over all vertex potentials, that is

$$\Phi(T) = \sum_{v \in V} \Phi(v).$$

Now, an improvement step in Algorithm 1 improves the degree of a vertex $v \in S_d$ with $\delta_T(v) = d$ and $d \geq \Delta(T) - \lceil \log_b n \rceil + 1$. Hence, the reduction in the potential is going to be at least

$$(3^d + 2 \cdot 3^{d-2}) - 3 \cdot 3^{d-1} = 2 \cdot 3^{d-2}.$$

Using the range of d we can lower bound the right hand side of the last equality by

$$3^{\Delta(T)-\log_b n-1} = \Omega\left(\frac{3^{\Delta(T)}}{n}\right).$$

The potential $\Phi(T)$ of the tree T is at most $n3^{\Delta(T)}$. This implies that the overall decrease of the potential due to the improvement step is

$$\Omega\left(\frac{\Phi(T)}{n^2}\right)$$

In other words, we reduce the potential by at least a polynomial factor in each iteration. In $O(n^2)$ iterations the reduction is by a constant factor. Hence, the algorithm needs $O(n^3)$ improvement steps in total. □

3. The BMST-Algorithm. In this section, we describe our algorithm for the BMST problem. It uses the Lagrangean formulation LD(B) from the introduction and Algorithm 1.

The input to our algorithm consists of a graph G, a non-negative cost function c, a degree bound B^* and a positive constant ω. Let $B = (1+\omega)B^*$.

Algorithm 2 Our algorithm for the BMST problem

```
1: Given: graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}^+$, a degree bound $B^* \geq 2$
   and a parameter $\omega > 0$.
2: $B \leftarrow (1+\omega)B^*$
3: $\lambda^B \leftarrow$ Solve($LD(B)$)
4: $\overline{T} \leftarrow$ PLocal($G, c^{\lambda^B}$)
```

Since the optimum Lagrange multipliers and pseudo-optimal MSTs can be computed in polynomial time [7, 15], Algorithm 2 runs in polynomial time.

Recall that c^{λ^B} denotes the original cost function c *augmented* by the Lagrangean multipliers λ^B, i.e. $c_{uv}^{\lambda^B} = c_{uv} + \lambda_u + \lambda_v$. We use $\mathcal{O}^B$ to denote the set of all minimum-cost spanning trees of G for cost function c^{λ^B}.

4. Analysis. In this section we prove Theorem 1.1. First we show that the cost $c(\overline{T})$ of the tree output by Algorithm 2, $\overline{T}$, is small. Then, we prove that $\overline{T}$ has low maximum degree. Our proofs rely on techniques from Lagrangean duality.

4.1. The cost of $\overline{T}$. Recall that $\texttt{opt}_{LD(B)} \leq \texttt{opt}_B$ from Proposition 1.2. Unfortunately, $\texttt{opt}_{LD(B)} = \texttt{opt}_B$ is not true in general. There might be a *duality gap*. However, it can be shown that $\texttt{opt}_{LD(B)}$ equals the optimum objective function value of the relaxation of (IP). The proof is insightful and hence we present it here.

THEOREM 4.1. *[15]* $\texttt{opt}_{LD(B)} = \min\{c(T) : T \in SP_G, \forall v \in V : \delta_T(v) \leq B\}$

Proof. We can restate (LD(B)) as the following linear program in variables η and λ. Recall that we denote its maximum objective function value by $\texttt{opt}_{LD(B)}$.

$$\begin{aligned} \max \quad & \eta \\ \text{s.t.} \quad & \eta \leq c(T) - \sum_{v \in V} \lambda_v (B - \delta_T(v)) \quad \forall T \in \mathrm{SP}_G \\ & \lambda \geq 0 \end{aligned} \tag{4.1}$$

The first block of constraints states that η is at most the cost of any spanning tree T of G with respect to the Lagrangean function (1.2). The maximization objective of (4.1) forces η to attain the best possible cost. Writing down the dual of the last program yields

$$\begin{aligned} \min \quad & c(\sum_{T\in \mathrm{SP}_G} \alpha_T T) && (4.2)\\ \text{s.t.} \quad & \sum_{T\in \mathrm{SP}_G} \alpha_T = 1 \\ & \sum_{T\in \mathrm{SP}_G} \alpha_T \delta_T(v) \leq B \sum_{T\in \mathrm{SP}_G} \alpha_T = B \quad \forall v \in V \\ & \alpha \geq 0 \end{aligned}$$

Note that $T^\alpha = \sum_{T\in \mathrm{SP}_G} \alpha_T T$ is a convex combination of trees in SP_G. Also, observe that $\sum_{T\in \mathrm{SP}_G} \alpha_T\delta_T(v)$ is precisely the degree $\delta^\alpha(v)$ of this fractional tree at node v. These observations yield the theorem. ▯

The theorem has two nice corollaries that we use. In the following, let λ^B denote the vector of optimum Lagrangean multipliers for (LD(B)). Recall that $\mathcal{O}^B$ is the set of minimum-cost spanning trees under c^{λ^B}.

COROLLARY 4.2. *There exists a convex combination* $T^\alpha = \sum_{T\in\mathcal{O}^B} \alpha_T T$ *such that*

1. $\forall v \in V : \delta^\alpha_{c^{\lambda^B}}(v) \leq B$ *and*
2. $\lambda^B_v > 0$ *only if* $\delta^\alpha_{c^{\lambda^B}}(v) = B$.

Proof. This follows from complementary slackness applied to the optimum solutions of the dual linear programs (4.1) and (4.2). ▯

The second corollary gives a bound on $\Delta^*_{c^{\lambda^B}}$.

COROLLARY 4.3. $\Delta^*_{c^{\lambda^B}} \leq B$

Proof. By Corollary 4.2, we know that there is a convex combination T^α of trees from $\mathcal{O}^B$ such that $\delta^\alpha_{c^{\lambda^B}}(v) \leq B$ for all v. Hence

$$\Delta^*_{c^{\lambda^B}} = \min_\alpha \max_{v\in V} \delta^\alpha_{c^{\lambda^B}}(v) \leq B.$$

▯

We now prove that $c(\overline{T})$ is small.

LEMMA 4.4. *For all trees* $T \in \mathcal{O}^B$ *we have* $c(T) < (1 + 1/\omega)\, \texttt{opt}_{B^*}$.

Proof. Recall that we defined $B = (1 + \omega)B^*$

The following inequality holds for every $T \in \mathcal{O}^B$:

$$\begin{aligned} \sum_{v\in V} \lambda^B_v(\delta_T(v) - B^*) &\leq c(T) + \sum_{v\in V} \lambda^B_v(\delta_T(v) - B^*) && (4.3)\\ &\leq \texttt{opt}_{LD(B^*)} \end{aligned}$$

In the first inequality we just added $c(T)$. Note, that the right hand side of the first line is just the Lagrangean objective function (1.2) for B^*. Recall that T is a minimum spanning tree with respect to c^{λ^B}. Moreover, λ^B is a feasible set of multipliers for (LD(B^*)). Hence, the second inequality follows.

We also have

$$\begin{aligned} c(T) &= c(T) + \sum_{v\in V} \lambda_v^B(\delta_T(v) - B^*) + \sum_{v\in V} \lambda_v^B(B^* - \delta_T(v)) \\ &\leq \texttt{opt}_{LD(B^*)} + \sum_{v\in V} \lambda_v^B(B^* - \delta_T(v)) \end{aligned}$$

where the inequality follows from (4.3). Applying Proposition 1.2 and the fact that $\delta_T(v) \geq 1$ for all $v \in V$ leads to

$$c(T) < \texttt{opt}_{B^*} + B^* \sum_{v\in V} \lambda_v^B.$$

In the remainder of this proof we will derive the inequality $B^* \sum_{v\in V} \lambda_v^B \leq \texttt{opt}_{B^*}/\omega$. This yields the lemma. From Corollary 4.2, we know that there is a convex combination

$$T^\alpha = \sum_{T\in\mathcal{O}^B} \alpha_T T$$

such that $\lambda_v^B > 0$ only if $\delta^\alpha_{c^{\lambda^B}}(v) = B$.

We derive a new inequality by summing over all $T \in \mathcal{O}^B$, α_T times the inequality (4.3) for each T. We obtain

$$\sum_{T\in\mathcal{O}^B} \alpha_T \left(\sum_{v\in V} \lambda_v^B(\delta_T(v) - B^*) \right) \leq \texttt{opt}_{LD(B^*)} \sum_{T\in\mathcal{O}^B} \alpha_T \tag{4.4}$$

The right hand side is equivalent to $\texttt{opt}_{LD(B^*)}$ because $\sum_{T\in\mathcal{O}^B} \alpha_T = 1$. Reordering the left hand side yields

$$\sum_{v\in V} \lambda_v^B \left(\left(\sum_{T\in\mathcal{O}^B} \alpha_T \delta_T(v) \right) - B^* \right)$$

Instead of summing over all $v \in V$ it suffices to sum over v, where $\lambda_v^B > 0$. For such v, we have

$$\delta^\alpha_{c^{\lambda^B}} = \sum_{T\in\mathcal{O}^B} \alpha_T \delta_T(v) = B$$

by Corollary 4.2. Using $B = (1+\omega)B^*$ it follows that the left hand side of (4.4) is equivalent to

$$\omega B^* \sum_{v\in V} \lambda_v^B$$

and this finishes the proof of the lemma. □

4.2. The Maximum Degree of $\overline{T}$. LEMMA 4.5. $\Delta_{\overline{T}} \leq (1+\omega)bB^* + \lceil \log_b n \rceil$ *for constants* $b > 1$ *and* ω.

Proof. $\overline{T}$ is a pseudo-optimal minimum-cost spanning tree with respect to cost function c^{λ^B}. From Theorem 2.9 we know that

$$\Delta_{\overline{T}} \leq b\Delta^*_{c^{\lambda^B}} + \lceil \log_b n \rceil.$$

An application of Corollary 4.3, noting $B = (1+\omega)B^*$ yields the lemma. □

5. Conclusions.

5.1. Summary and remarks. In this paper we developed an improved approximation algorithm for the degree-bounded minimum spanning tree problem. For a positive constant B^* and an n-node graph, our method computes a spanning tree whose cost is at most a constant factor worse than the cost of the optimum degree-B^*-bounded minimum spanning tree. Additionally, we proved that the maximum degree of the resulting tree is $O(B^* + \log n)$. Our procedure solves a Lagrangean relaxation of the BMST integer program for slightly relaxed degree constraints ($(1 + \omega)B^*$ instead of B^*). We showed how this slack helps to prove low cost of the resulting tree. Our algorithm also makes use of a local search technique from [5, 7]. We showed how a slight strengthening of the results in [5, 7] can be used to prove low maximum degree of the resulting tree.

As a side note, the reader should notice that in Algorithm 2 we assumed the exact solution of (LD(B)). However, in an implementation, a reasonable approximation to the optimum Lagrangean multipliers will most likely be sufficient. To compute such an approximation, we could employ subgradient optimization techniques from [9, 10, 16].

5.2. Extensions and open questions. An interesting open question is whether our results extend to the case of Steiner trees and general Steiner networks. The central difficulty of such an extension stems from the fact that, in the Steiner case, the subproblem that arises from dualizing the degree constraints (the minimum cost Steiner tree problem) is $\mathcal{NP}$-hard itself.

Another avenue for extending our work is to examine if our approach capable of handling individual node degrees? In the current version, node degrees are assumed to be uniform. Lemma 4.5 relies on the pseudo-optimality of tree $\overline{T}$ from Algorithm 2 and on results from [5, 7]. These results do not apply to non-uniform degrees. Is there an extention of the known MDST algorithms to handle individual degree bounds?

We believe that the techniques used in this paper can be generalized to apply to a broader class of multicriteria problems. A central point in the development of a more general framework is the identification of key properties of suitable optimization problems; in the BMST problem, the dualization of the degree constraints yields a tractable subproblem. Furthermore, the compact form of the objective function of this subproblem proved to be a key for the analysis.

Acknowledgments We thank Éva Tardos for permitting us to include her simpler proof of Lemma 4.5 in the paper.

REFERENCES

[1] F. Bauer and A. Varma. Degree-constrained multicasting in point-to-point networks. In *Proceedings of the 14th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'95)*, pages 369–376, Los Alamitos, CA, USA, April 1995. IEEE Computer Society Press.
[2] S. Chopra. On the spanning tree polyhedron. *Operations Research Letters*, 8:25–29, 1989.
[3] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei. An architecture for wide-area multicast routing. In *Proceedings, 1994 SIGCOMM Conference*, pages 126–135, London, UK, 1994.
[4] Edmonds, J. Optimum branchings. *J. Res. Nat. Bur. Standards*, B71:233–240, 1967.
[5] T. Fischer. Optimizing the degree of minimum weight spanning trees. Technical Report TR 93-1338, Dept. of Computer Science, Cornell University, Ithaca, NY 14853, 1993.
[6] S. Floyd, V. Jacobson, S. McCanne, C. G. Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, November 1996.

[7] Martin Fürer and Balaji Raghavachari. Approximating the minimum-degree Steiner tree to within one of optimal. *Journal of Algorithms*, 17(3):409–423, November 1994.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, San Francisco, 1979.

[9] M. Held and R. M. Karp. The traveling-salesman problem and minimum spanning trees: part II. *Math. Programming*, 1:6–25, 1971.

[10] Michael Held, Philip Wolfe, and Harlan P. Crowder. Validation of subgradient optimization. *Mathematical Programming*, 6(1):62–88, 1974.

[11] Samir Khuller, Balaji Raghavachari, and Neal Young. Low-degree spanning trees of small weight. *SIAM Journal on Computing*, 25(2):355–368, April 1996.

[12] P.N. Klein and R. Ravi. A nearly best-possible approximation for node-weighted Steiner trees. *J. Algorithms*, 19:104–115, 1995.

[13] J. Könemann and R. Ravi. A matter of degree: Improved approximation algorithms for degree-bounded minimum spanning trees. In *Proceedings, ACM Symposium on Theory of Computing*, 2000.

[14] Madhav V. Marathe, R. Ravi, Ravi Sundaram, S. S. Ravi, Daniel J. Rosenkrantz, and Harry B. Hunt III. Bicriteria network design problems. *Journal of Algorithms*, 28(1):142–171, July 1998.

[15] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.

[16] B.T. Polyak. A general method of solving extremum problems. *Doklady Akademmi Nauk SSSR*, 174(1):33–36, 1967.

[17] R. Ravi, M.V. Marathe, S.S.Ravi, D.J. Rosenkrantz, and H.B. Hunt. Many birds with one stone: Multi-objective approximation algorithms. In *Proceedings, ACM Symposium on Theory of Computing*, pages 438–447, 1993.

[18] A.S. Tanenbaum. *Computer Networks*. Prentice Hall, 1996.

[19] E. Tardos. Personal communication. April 2000.

[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.