Primal-dual meets local search: Approximating MST's with nonuniform degree bounds [*]

Jochen Könemann
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
jochen@cmu.edu

R. Ravi
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
ravi@cmu.edu

ABSTRACT

We present a new bicriteria approximation algorithm for the degree-bounded minimum-cost spanning tree problem: Given an undirected graph with nonnegative edge weights and degree bounds $B_v > 1$ for all vertices v, find a spanning tree T of minimum total edge-cost such that the maximum degree of each node v in T is at most B_v. Our algorithm finds a tree in which the degree of each node v is $O(B_v + \log n)$ and the total edge-cost is at most a constant times the cost of any tree that obeys all degree constraints.

Our previous algorithm[9] with similar guarantees worked only in the case of uniform degree bounds (i.e. $B_v = B$ for all vertices v). While the new algorithm is based on ideas from Lagrangean relaxation as is our previous work, it does not rely on computing a solution to a linear program. Instead it uses a repeated application of Kruskal's MST algorithm interleaved with a combinatorial update of approximate Lagrangean node-multipliers maintained by the algorithm. These updates cause subsequent repetitions of the spanning tree algorithm to run for longer and longer times, leading to overall progress and a proof of the performance guarantee.

Categories and Subject Descriptors

G.2.2 [**Mathematics of Computing**]: Discrete Mathematics—*Graph Theory*; F.2.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*Nonnumerical Algorithms and Problems*

General Terms

Algorithms, Design, Theory

[*]This material is based upon work supported by the National Science Foundation under Grant No. 0105548.

Keywords

approximation algorithms, network algorithms, bicriteria approximation, spanning trees, degree-bounded spanning trees

1. INTRODUCTION

1.1 Formulation

In this paper, we address the *degree-bounded minimum-cost spanning tree problem* with nonuniform degree bounds (nBMST). Given an undirected graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}^+$ and positive integers $\{B_v\}_{v \in V}$ all greater than 1, the goal is to find a spanning tree T of minimum total cost such that for all vertices $v \in V$ the degree of v in T is at most B_v. This problem is clearly NP-hard since it generalizes a variant of the TSP problem.

1.2 Previous work and our result

Related to the nBMST problem is the problem of computing a spanning tree of a given undirected graph G with minimizing maximum degree (MDST). Fürer and Raghavachari presented an approximation algorithm for this problem with an additive performance guarantee of one [4]: i.e., they describe a polynomial-time algorithm that finds a spanning tree T of G with maximum node-degree $\Delta^* + 1$, where Δ^* denotes the minimum possible maximum degree over all spanning trees.

It should be noted that the result in [4] can be extended easily to the setting of non-uniform degree bounds: For each node v in an n-node graph G, add $n - B_v$ auxiliary nodes and connect v to all of them. Then, run the algorithm from [4] and, afterwards, remove the auxiliary nodes together with their incident edges from the tree. It is not hard to see that the degree of each node v is at most $B_v + 1$.

In the weighted setting, Ravi et al. [12, 13] showed how to compute a spanning tree T of degree $O(B_v \log n)$ for all $v \in V$ and total cost $O(\log n) \cdot \texttt{opt}$ where $\texttt{opt}$ is the minimum cost of any tree in which the degree of node v is bounded by B_v for all $v \in V$. The authors generalized their ideas to Steiner trees, generalized Steiner forests and the node-weighted case. Subsequently, Könemann and Ravi [8, 9] improved the results on the (edge-weighted) spanning tree version of the problem with uniform degree-bounds (where $B_v = B$ for all v). The main theorem from [9] is as follows:

THEOREM 1 (SEE [9]). *There is an approximation algorithm that, given a graph $G = (V, E)$, a nonnegative cost* *function $c : E \to \mathbb{R}^+$, a degree bound $B \geq 2$ and a parameter $\omega > 1$, computes a spanning tree T such that*

1. $\Delta(T) \leq \frac{\omega}{\omega-1} \cdot bB + \log_b n$ *for any arbitrary constant* $b > 1$, *and*

2. $c(T) < \omega \cdot \texttt{opt}$.

The contributions of this paper are two-fold: First, we present improved approximation algorithms for the minimum-cost degree-bounded spanning tree problem in the presence of non-uniform degree bounds. Second, our algorithm is *direct* in the sense that we do not solve linear programs. The algorithm in [9] uses Lagrangean relaxation and its starting point is a solution to a large linear program. The analysis of the algorithm's performance guarantee relies crucially on the fact that we solve the latter LP exactly.

On the other hand, our new algorithm integrates elements from the primal-dual method for approximation algorithms for network design problems with local search methods for minimum-degree network problems [4]. The algorithm goes through a series of spanning trees and improves the maximum deviation of any vertex degree from its respective degree bound continuously. A practical consequence of this is that we can terminate the algorithm at any point in time and still obtain a spanning tree of the input graph (whose node-degrees, of course, may not meet the worst-case guarantees we prove).

THEOREM 2. *There is a primal-dual approximation algorithm that, given a graph $G = (V, E)$, a nonnegative cost function $c : E \to \mathbb{R}^+$, integers $B_v > 1$ for all $v \in V$ and a parameter $\omega > 1$ computes a tree T such that*

1. $\deg_T(v) \leq \max\left\{\frac{\omega}{\omega-1}, \omega\right\} \cdot B_v + 2 \cdot \log_b n + 1$ *for all* $v \in V$, *and*

2. $c(T) < \omega \cdot \texttt{opt}$

where $b > 1$ is an arbitrary constant. The running time is $O(|V|^6 \log |V|)$.

We note that in the special case of $B_v = B$ for all $v \in V$ our algorithm computes a tree of maximum degree at most $B + \log_b n + 1$ and the running time is $O(|V|^5 \log |V|)$.

The paper is organized as follows: First, we review the primal-dual interpretation of the well-known algorithm for minimum-cost spanning trees by Kruskal [11]. Subsequently, we show how to use this algorithm for the nBMST problem and present an analysis of performance guarantee and running time of our method.

2. A PRIMAL-DUAL ALGORITHM TO COMPUTE MST'S

In this section we review Kruskal's minimum-cost spanning tree algorithm. More specifically, we discuss a primal-dual interpretation of this method that follows from [2]. We start by giving a linear programming formulation of the convex hull of incidence vectors of spanning trees.

2.1 The spanning tree polyhedron

In the following, we formulate the minimum-cost spanning tree problem as an integer program where we associate a $0,1$-variable x_e with every edge $e \in E$. In a solution x, the value of x_e is one if e is included in the spanning tree corresponding to x and 0 otherwise. Our formulation relies on a complete formulation of the convex hull of incidence vectors of spanning trees (denoted by $\texttt{SP}_G$) given by Chopra [2].

Chopra's formulation uses the notion of a *feasible partition* of vertex set V. A feasible partition of V is a set $\pi = \{V_1, \ldots, V_k\}$ such that the V_i are pairwise disjoint subsets of V. Moreover, $V = V_1 \cup \ldots \cup V_k$ and the induced subgraphs $G[V_i]$ are connected. Let G_π denote the (multi-) graph that has one vertex for each V_i and edge (V_i, V_j) occurs with multiplicity $|\{(v_i, v_j) : v_i \in V_i, v_j \in V_j\}|$. In other words, G_π results from G by contracting each of the V_i to a single node. Define the *rank* $r(\pi)$ of π as the number of nodes of G_π and let Π be the set of all feasible partitions of V. Chopra showed that

$$\texttt{SP}_G = \{x \in \mathbb{R}^m : \sum_{e \in E(G_\pi)} x_e \geq r(\pi) - 1 \quad \forall \pi \in \Pi\}.$$

We now let $\delta(v)$ denote the set of edges $e \in E$ that are incident to node v and, for a subset $S \subseteq V$, we define $\delta(S)$ to be the set of edges that have exactly one endpoint in S. We obtain an integer programming formulation for our problem:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & \text{(IP)} \\ \text{s.t} \quad & \sum_{e \in E[G_\pi]} x_e \geq r(\pi) - 1 \quad \forall \pi \in \Pi & \\ & x(\delta(v)) \leq B_v \quad \forall v \in V & (1) \\ & x \text{ integer} & \end{aligned}$$

The dual of the linear programming relaxation (LP) of (IP) is given by

$$\begin{aligned} \min \quad & \sum_{\pi \in \Pi} (r(\pi) - 1) \cdot y_\pi - \sum_{v \in V} \lambda_v B_v & \text{(D)} \\ \text{s.t} \quad & \sum_{\pi : e \in E[G_\pi]} y_\pi \leq c_e + \lambda_u + \lambda_v \quad \forall e = uv \in E & (2) \\ & y, \lambda \geq 0 & \end{aligned}$$

We also let (IP-SP) denote (IP) without constraints of type (1). Let the LP relaxation be denoted by (LP-SP) and let its dual be (D-SP).

2.2 A primal-dual interpretation of Kruskal's MST algorithm

Kruskal's algorithm can be viewed as a continuous process over *time*: We start with an empty tree at time 0 and add edges as we go along. The algorithm terminates at time t^* with a spanning tree of the input graph G. In this section we show that Kruskal's method can be interpreted as a primal-dual algorithm (see also [7]). At any time $0 \leq t \leq t^*$ we keep a pair (x_t, y_t), where x_t is a partial (possibly infeasible) primal solution for (LP-SP) and y_t is a feasible dual solution for (D-SP). Initially, we let $x_{e,0} = 0$ for all $e \in E$ and $y_{\pi,0} = 0$ for all $\pi \in \Pi$.

Let E_t be the forest corresponding to partial solution x_t, i.e. $E_t = \{e \in E : x_{e,t} = 1\}$. We then denote by π_t the partition induced by the connected components of $G[E_t]$. At time t, the algorithm then increases y_{π_t} until a constraint of type (2) for edge $e \in E \setminus E_t$ becomes tight. Assume that this happens at time $t' > t$. The dual update is

$$y_{\pi_t, t'} = t' - t.$$

We then include e into our solution, i.e. we set $x_{e,t'} = 1$. If more than one edge becomes tight at time t', we can process these events in any arbitrary order; Thus, note that we can pick any such tight edge first in our solution. Subsequently, we will use MST to refer to the above algorithm.

The proof of the following lemma is folklore. We supply it for the sake of completeness.

LEMMA 1. *At time t^*, Algorithm* MST *finishes with a pair (x_{t^*}, y_{t^*}) of primal and dual feasible solutions to (IP-SP) and (D-SP), respectively, such that*

$$\sum_{e \in E} c_e x_{e,t^*} = \sum_{\pi \in \Pi} (r(\pi) - 1) \cdot y_{\pi,t^*}$$

PROOF. Notice, that for all edges $e \in E_{t^*}$ we must have $c_e = \sum_{\pi : e \in E[G_\pi]} y_{\pi,t^*}$ and hence, we can express the cost of the final tree as follows:

$$c(E_{t^*}) = \sum_{e \in E_{t^*}} \sum_{\pi : e \in E[G_\pi]} y_{\pi,t^*} = \sum_{\pi \in \Pi} |E_{t^*} \cap E[G_\pi]| \cdot y_{\pi,t^*}.$$

By construction E_{t^*} is a tree and we must have that the set $E_{t^*} \cap \pi$ has cardinality exactly $r(\pi) - 1$ for all $\pi \in \Pi$ with $y_{\pi,t^*} > 0$. We obtain that $\sum_{e \in E} c_e x_{e,t^*} = \sum_{\pi \in \Pi} (r(\pi) - 1) \cdot y_{\pi,t^*}$ and this finishes the proof of the lemma. □

3. MINIMUM-COST DEGREE-BOUNDED SPANNING TREES

In this section, we propose a modification of the above algorithm for approximating degree-bounded spanning trees of low total cost (for suitably weakened degree bounds). Our algorithm goes through a sequence of spanning trees $E^0, \ldots, E^t$ and associated pairs of primal (infeasible) and dual feasible solutions x^i, (y^i, λ^i) for $0 \le i \le t$. The idea is to reduce the degree of nodes $v \in V$ whose degree is substantially higher than their associated bound B_v, as we proceed through this sequence, while keeping the cost of the associated primal solution (tree) bounded with respect to the corresponding dual solution.

To begin, our algorithm first computes an approximate minimum-cost spanning tree using the Algorithm MST. This yields a feasible primal solution x^0 for (LP-SP) and a feasible dual solution y^0 for (D-SP). Notice that y^0 also induces a feasible solution for (D) by letting $\lambda_v^0 = 0$ for all $v \in V$ while x^0 potentially violates constraints of type (1).

We introduce the notion of *normalized degree* of a node V in a tree T and denote it by

$$\texttt{ndeg}_T(v) = \max\{0, \deg_T(v) - \beta_v \cdot B_v\} \qquad (3)$$

where $\beta_v > 0$ are constants for all $v \in V$ to be specified later. Our algorithm successively computes pairs of spanning trees and associated dual solutions to (D), i.e.

$$(x^1, \{y^1, \lambda^1\}), (x^2, \{y^2, \lambda^2\}), \ldots, (x^t, \{y^t, \lambda^t\}).$$

From one such pair to the next, we try to reduce the degree of nodes of high normalized degree. Specifically, our algorithm runs as long as there is a node in the current tree with $\texttt{ndeg}(v) \ge 2\log_b(n)$ for some constant $b > 1$.

We let E^i be the spanning tree corresponding to x^i and let Δ^i be the maximum normalized degree of any node in the tree E^i. The central piece of our algorithm is a *recompute step* where we raise the λ values of a carefully chosen set S_d of nodes with high normalized degree. This introduces slack in many of the constraints of type (2). Subsequently, we rerun MST and ensure that it generates a feasible dual packing that takes advantage of the newly created slack around nodes of high normalized degree. At the same time MST computes a new tree and we ensure at all times that its cost is close to the objective function value of the associated dual. We are able to show that the number of recompute steps is polynomial, by arguing that we make substantial progress in the normalized degree sequence of all nodes.

As mentioned, each recompute step takes a pair of primal infeasible and dual feasible solutions $(x^i, (y^i, \lambda^i))$ and computes a new pair of primal (infeasible) and dual feasible solutions $(x^{i+1}, (y^{i+1}, \lambda^{i+1}))$. In the following we use $\texttt{ndeg}^i(v)$ as a short for $\texttt{ndeg}_{E^i}(v)$. We then adapt the notation from [3, 4] and let

$$S_d^i = \{v \in V \,:\, \texttt{ndeg}^i(v) \ge d\}$$

be the set of all nodes whose normalized degree is at least d in the ith solution.

Algorithm 1 The algorithm for the nBMST problem attempts to reduce the maximum normalized degree of any node in a given spanning tree.

```
1: $\lambda_v^0 \leftarrow 0, \forall v \in V; \widetilde{c}_e^0 \leftarrow c_e, \forall e \in E$
2: $(x^0, y^0) \leftarrow \texttt{MST}(G, \widetilde{c}^0)$
3: $i \leftarrow 0$
4: while $\Delta^i > 2\log_b(n)$ do
5:     Choose $d^i \in \{\Delta^i - 2\log_b(n), \ldots, \Delta^i\}$ s.t. $\sum_{v \in S^i_{d^i-1}} B_v \le b \cdot \sum_{v \in S^i_{d^i}} B_v$
6:     Choose $\epsilon^i > 0$
7:     For all $v \in V$ let
           $\lambda_v^{i+1} \leftarrow \begin{cases} \lambda_v^i + \epsilon^i & : \ v \in S^i_{d^i-1} \\ \lambda_v^i & : \ \text{otherwise} \end{cases}$
8:     $\widetilde{c}^{i+1}(e) \leftarrow \widetilde{c}^i(e) + \epsilon^i$ if either $e \in E^i$ and $e \cap S^i_{d^i} \neq \emptyset$ or $e \notin E^i$ and $e \cap S^i_{d^i-1} \neq \emptyset$
9:     $(x^{i+1}, y^{i+1}) \leftarrow \texttt{MST}(G, \widetilde{c}^{i+1})$
10:    $i \leftarrow i + 1$
11: end while
```

A detailed description of the procedure is given in Algorithm 1. Recall that $\texttt{MST}(G, \widetilde{c})$ returns a pair of primal and dual optimal solutions to (LP) and (D) for cost function $\widetilde{c}$. In step 5 of Algorithm 1, we choose a suitable set of nodes whose λ-values we increase. A simple argument in [3] can be extended to show the following.

LEMMA 2. *There is a $d^i \in \{\Delta^i - 2\log_b(n), \ldots, \Delta^i\}$ such that*

$$\sum_{v \in S^i_{d^i-1}} B_v \le b \cdot \sum_{v \in S^i_{d^i}} B_v$$

for a given constant $b > 1$.

PROOF. Suppose for a contradiction that for all $d^i \in$

$\{\Delta^i - 2\log_b(n), \dots, \Delta^i\}$, we have

$$\sum_{v \in S^i_{d^i-1}} B_v > b \cdot \sum_{v \in S^i_{d^i}} B_v.$$

Note that since we may assume $B_v \leq (n-1)$ for all vertices, we must have $\sum_{v \in V} B_v \leq n(n-1)$. However, since $\sum_{v \in S^i_{\Delta^i}} B_v \geq 1$, we have in this case that

$$\sum_{v \in S^i_{\Delta^i - 2\log_b(n)}} B_v \geq b^{2\log_b(n)} = n^2$$

a contradiction. $\square$

The *low expansion* of $S^i_{d^i}$ turns out to be crucial in the analysis of the performance guarantee of our algorithm.

Intuitively, increasing λ_v for a node $v \in V$ *lengthens* edges of the form $uv \in E$; In this way, in the current packing $\{y^i_\pi\}_{\pi \in \Pi}$, we create extra slack in constraints of type (2) for edges incident to v. The hope is to increase the length of edges incident to nodes of high normalized degree until other edges that are incident to nodes of lower normalized degree are used in their place in the spanning tree.

Step 6 of Algorithm 1 hides the details of choosing an appropriate ϵ^i by which edges in the current tree that are incident to nodes of normalized degree at least d^i are lengthened. Our choice of ϵ^i will ensure that there exists at least one point in time during the execution of MST in step 9 at which we now have the choice to connect two connected components using edges e^i and $\overline{e}^i$ where $e^i \in E^i$ and $\overline{e}^i \notin E^i$. We now break ties such that $\overline{e}^i$ is chosen instead of e^i and obtain a new tree E^{i+1} that differs in exactly one edge from E^i.

We show that we can choose ϵ^i and hence a pair of edges $\langle e^i, \overline{e}^i \rangle$ such that $\overline{e}^i$ is not incident on any node from $S^i_{d^i-1}$. The main idea here is to increase λ_v for nodes $v \in S^i_{d^i-1}$ by ϵ^i and increase the lengths of non-tree edges that are incident to nodes $v \in S^i_{d^i-1}$ by ϵ^i as well. In other words, the length of non-tree edges incident to nodes of normalized degree at least $d^i - 1$ increases by the same amount as the length of tree edges incident to nodes of normalized degree at least d^i. This way, we enforce that the edge we swap in touches nodes of normalized degree at most $d^i - 2$. Once we accomplish this, adapting a potential function argument from [3] we can put a polynomial upper bound on the number of such iterations (see Section 3.5).

Suppose, that our algorithm terminates after t iterations. Our goal is then to prove

$$\sum_{e \in E^i} c_e \leq \omega \sum_{\pi \in \Pi} (r(\pi) - 1) \cdot y^i_\pi - \omega \cdot \sum_{v \in V} B_v \cdot \lambda^i_v \tag{4}$$

for all $1 \leq i \leq t$. The right-hand side of (4) is ω times the dual objective function value of the feasible dual solution (y^i, λ^i). Therefore, using weak duality, (4) implies that $\sum_{e \in E^i} c_e \leq \omega \cdot \texttt{opt}$. We now describe how to choose ϵ^i so that the above conditions are satisfied.

3.1 Choosing ϵ^i

In this section we elaborate on the choice of ϵ^i in step 6 of Algorithm 1. Note that we maintain

$$c_{uv} \leq \widetilde{c}_{uv} \leq c_{uv} + \lambda_u + \lambda_v$$

for all edges uv at all times during the run of Algorithm 1. In step 8 of Algorithm 1, we increase $\widetilde{c}_{uv}$ by ϵ^i for all tree edges uv that are incident to nodes of degree at least d^i and for all non-tree edges that are incident to nodes of degree at least $d^i - 1$. Note that our algorithm increases the λ^i-value of at least one endpoint of all the edges whose $\widetilde{c}$-cost increases.

We want to choose ϵ^i such that the subsequent update of $\widetilde{c}^i$ and the following run of MST yields a new tree E^{i+1} that differs from E^i by a single edge swap: $E^{i+1} = E^i \setminus \{e^i\} \cup \{\overline{e}^i\}$. Here, the edge $e^i \in E^i$ is a tree edge that is incident to a node from $S^i_{d^i}$. On the other hand $\overline{e}^i \in E \setminus E^i$ is a non-tree edge that is not incident to any node from $S^i_{d^i-1}$. We want that $\widetilde{c}^i(e^i) \leq \widetilde{c}^i(\overline{e}^i)$ and $\widetilde{c}^i(e^i) + \epsilon^i = \widetilde{c}^i(\overline{e}^i)$. In other words, the update of λ_v for $v \in S^i_{d^i-1}$ creates one more beneficial swap.

We let $\mathcal{K}^i$ be the set of connected components of the forest $E^i \setminus S^i_{d^i}$, i.e. the forest that results from removing nodes of normalized degree at least d^i from E^i. We say that an edge $e = uv \in E$ is a *cross-edge* if

1. e is a non-tree edge, i.e. $e \in E \setminus E^i$, and
2. $u \in K_1, v \in K_2$ for $K_1, K_2 \in \mathcal{K}^i$ and $K_1 \neq K_2$.

We denote the set of cross-edges in iteration i by $\mathcal{C}^i$.

It is now clear that $E^i + e$ contains a unique cycle C^i_e for each cross-edge $e \in \mathcal{C}^i$. Furthermore, there must be at least one vertex v on C^i_e that has normalized degree at least d^i.

For each cross-edge $e \in \mathcal{C}^i$, we now let

$$\epsilon^i_e = \min_{e' \in C^i_e, e' \cap S^i_{d^i} \neq \emptyset} \left(\widetilde{c}^i(e) - \widetilde{c}^i(e') \right).$$

We then let $\epsilon^i = \min_{e \in \mathcal{C}^i} \epsilon^i_e$.

In the following, we let $\langle e^i, \overline{e}^i \rangle$ be the *witness pair* for ϵ^i. In other words, for all edges $e' \in C^i_{\overline{e}^i}$ such that e' is incident to a node of normalized degree at least d^i, we have $\widetilde{c}^i_{e'} + \epsilon^i \leq \widetilde{c}^i_{\overline{e}^i}$ and equality holds for $e' = e^i$.

Note that ϵ^i can be 0. Such a step can be viewed as a *local-improvement* step along the lines of [4]. We do not modify the dual solution but decrease the normalized degree of a node of high normalized degree.

We now show that (4) is maintained throughout the execution of our algorithm.

3.2 Analysis: Performance guarantee

Assume that Algorithm 1 terminates after iteration t^*. In this section we prove that (4) must hold for all $0 \leq i \leq t^*$. Observe that (4) holds for $i = 0$ by Lemma 1. We concentrate on the case $i \geq 1$.

Growing λ^i_v by ϵ^i at nodes $v \in S^i_{d^i-1}$ decreases the right-hand side of (4) by $\omega\epsilon^i \cdot \sum_{v \in S^i_{d^i-1}} B_v$. Still the cost of the spanning tree E^{i+1} is potentially higher than the cost of the old tree E^i. We must show that the first term on the right hand-side of (4), i.e. $\omega \cdot \sum_{\pi \in \Pi} (r(\pi) - 1) y^i_\pi$ grows sufficiently to compensate for the decrease in the second term and the increased spanning tree cost.

To do this, we maintain the following invariant inductively for all $0 \leq i \leq t^*$:

$$\omega \cdot \sum_{v \in V} B_v \lambda^i_v \leq (\omega - 1) \cdot \sum_{\pi \in \Pi} (r(\pi) - 1) \cdot y^i_\pi. \tag{Inv}$$

Since, $\lambda^0_v = 0$ for all $v \in V$ by definition, (Inv) holds for $i = 0$.

The following claim that proves the inductive step of (Inv) is the essential insight that ultimately yields (4).

CLAIM 1. *Choosing $\beta_v \geq b\alpha + 1/B_v$ for all $v \in V$ in the definition of normalized degree yields*

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1} \geq \sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i} + \epsilon^i\alpha \cdot \sum_{v\in S^i_{d^i-1}} B_v$$

for all $0 \leq i \leq t^*$.

PROOF. Let $E^i = \{e_1^i, \ldots, e_{n-1}^i\}$ and let t_j^i be the time at which MST included edge e_j^i. W.l.o.g., assume that $t_1^i \leq \cdots \leq t_{n-1}^i$. From the description of MST we can rewrite

$$\begin{aligned}\sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^i &= \sum_{j=1}^{n-1}(t_j^i - t_{j-1}^i)\cdot(n-j) \quad (5)\\ &= \sum_{j=1}^{n-1} t_j^i\left((n-j+1)-(n-j)\right)\\ &= \sum_{j=1}^{n-1} t_j^i\end{aligned}$$

where we define $t_0^i = 0$.

In fact, we can use (5) to quantify the change in dual in iteration i:

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot\left(y_\pi^{i+1}-y_\pi^i\right) = \sum_{j=1}^{n-1}\left(t_j^{i+1}-t_j^i\right) \quad (6)$$

In iteration i, we lengthen all edges $e \in E^i$ that are incident to nodes of normalized degree at least d^i by ϵ^i. Hence, all of these edges become tight ϵ^i time units later. Together with (6) we obtain

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot\left(y_\pi^{i+1}-y_\pi^i\right) \geq \epsilon^i \cdot \left|E\left(S^i_{d^i}\right)\cap E^i\right| \quad (7)$$

where $E\left(S^i_{d^i}\right)$ denotes the set of edges in E that are incident to nodes from $S^i_{d^i}$. (Note that we include in $E(S)$ edges with both endpoints in S).

Recall the definition of normalized degree in (3). Notice that it follows from the termination condition in step 4 of Algorithm 1 that $d^i > 0$. Hence, we must have that the real degree of any node $v \in S^i_{d^i}$ is at least $\beta_v \cdot B_v$. Finally, notice that it follows from the fact that E^i is a tree that there are at most $|S^i_{d^i}|-1$ edges in $E\left(S^i_{d^i}\right)$ that are incident to two nodes from $S^i_{d^i}$. We can use these observation to lower-bound the right-hand side of (7):

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot\left(y_\pi^{i+1}-y_\pi^i\right) \geq$$
$$\epsilon^i\cdot\left(\left(\sum_{v\in S^i_{d^i}}\beta_v\cdot B_v\right)-\left(|S^i_{d^i}|-1\right)\right). \quad (8)$$

Now, we choose $\beta_v \geq b\alpha + 1/B_v$ for all $v \in V$ and obtain

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot\left(y_\pi^{i+1}-y_\pi^i\right) \geq \epsilon^i\alpha b\cdot\sum_{v\in S^i_{d^i}} B_v.$$

Lemma 2 now yields the claim. $\square$

We are now ready to prove (4).

3.3 Proof of (4)

As MST finishes we obtain from Lemma 1 that

$$\widetilde{c}^{i+1}(E^{i+1}) = \sum_{e\in E^{i+1}}\widetilde{c}_e^{i+1} = \sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1}. \quad (9)$$

Observe that the real cost of the spanning tree E^{i+1} can be much smaller than $\widetilde{c}^{i+1}(E^{i+1})$. In fact, notice that we have

$$c(E^{i+1}) = c(\overline{e}^i)+c(E^i\setminus\{e_i\}) \leq \widetilde{c}^{i+1}(\overline{e}^i)+\widetilde{c}^i(E^i\setminus\{e^i\}) \quad (10)$$

where the first step follows from the fact that $E^{i+1} = E^i \setminus \{e^i\}\cup\{\overline{e}^i\}$ by the definition of ϵ^i and the way we break ties in MST. The second inequality uses the fact that we always have $c_e \leq \widetilde{c}_e^i$ for all $1 \leq i \leq t$ and for all $e \in E$.

Also, observe that

$$\widetilde{c}^{i+1}(E^i\setminus\{e^i\}) = \widetilde{c}^i(E^i\setminus\{e^i\}) + \left(\left|E\left(S^i_{d^i}\right)\cap E^i\right|-1\right)\cdot\epsilon^i \quad (11)$$

since exactly one edge from E^i that is incident to a node of normalized degree at least d^i is swapped out.

We can lower-bound $\left|E\left(S^i_{d^i}\right)\cap E^i\right|$ using the arguments from the proof of Claim 1:

$$\begin{aligned}\left|E\left(S^i_{d^i}\right)\cap E^i\right| &\geq \left(\sum_{v\in S^i_{d^i}}\beta_v\cdot B_v\right)-\left(|S^i_{d^i}|-1\right)\\ &\geq \alpha\cdot\sum_{v\in S^i_{d^i-1}}B_v+1\end{aligned}$$

where the second inequality again follows from Lemma 2 and from our choice of β_v for all $v \in V$. The last inequality together with (11) implies

$$\widetilde{c}^{i+1}(E^i\setminus\{e^i\}) \geq \widetilde{c}^i(E^i\setminus\{e^i\}) + \alpha\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}}B_v. \quad (12)$$

We now obtain

$$\begin{aligned}c(E^{i+1}) &\leq \widetilde{c}^i(E^i\setminus\{e^i\})+\widetilde{c}^{i+1}(\overline{e}^i)\\ &\leq \widetilde{c}^{i+1}(E^i)-\alpha\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}}B_v\\ &= \sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1}-\alpha\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}}B_v\end{aligned}$$

where the first inequality follows from (10), the second inequality follows from (12), and the last equality follows from (9).

Adding (Inv) to the last inequality we get

$$\begin{aligned} c(E^{i+1}) &\leq \sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1} + \\ &\quad (\omega-1)\cdot\sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^i - \\ &\quad \omega\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}} B_v - \alpha\cdot\sum_{v\in V} B_v\lambda_v^i \\ &\leq \omega\cdot\sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1} - \\ &\quad \omega\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}} B_v - \alpha\cdot\sum_{v\in V} B_v\lambda_v^i \end{aligned}$$

where the last inequality follows from the fact that

$$\sum_{\pi\in\Pi}(r(\pi)-1)y_\pi^i \leq \sum_{\pi\in\Pi}(r(\pi)-1)y_\pi^{i+1}.$$

Finally notice that $\lambda_v^{i+1}=\lambda_v^i+\epsilon^i$ if $v\in S^i_{d^i-1}$ and $\lambda_v^{i+1}=\lambda_v^i$ otherwise. Additionally, we choose $\alpha\geq\omega$ and get

$$c(E^{i+1}) \leq \omega\cdot\sum_{\pi\in\Pi}(r(\pi)-1)\cdot y_\pi^{i+1} - \omega\cdot\sum_{v\in V} B_v\lambda_v^{i+1}.$$

as required. This proves the inductive step of (4).

3.4 Proof of (Inv)

It remains to show that (Inv) is maintained as well. Observe that the left hand side of (Inv) increases by

$$\omega\epsilon^i\sum_{v\in S^i_{d^i-1}} B_v.$$

We obtain from Claim 1 that

$$\sum_{\pi\in\Pi}(r(\pi)-1)\cdot(y_\pi^{i+1}-y_\pi^i) \geq \epsilon^i\alpha\cdot\sum_{v\in S^i_{d^i-1}} B_v$$

Therefore, the right hand side of (Inv) increases by $(\omega-1)\cdot\alpha\epsilon^i\cdot\sum_{v\in S^i_{d^i-1}} B_v$ which is sufficient if $(\omega-1)\alpha\geq\omega$ or equivalently, if $\alpha\geq\omega/(\omega-1)$.

The proof of the performance guarantee claimed in Theorem 2 follows by choosing $\alpha\geq\max\left\{\omega,\frac{\omega}{\omega-1}\right\}$.

3.5 Analysis: Running time

In this section, we show that Algorithm 1 terminates in polynomial time. We accomplish this by showing that there will be only a polynomial number of iterations of the main loop in Algorithm 1.

Claim 2. *Algorithm 1 terminates after $O(n^4)$ iterations.*

Proof. Following [3], we define the potential of spanning tree E^i as

$$\Phi_i = \sum_{v\in V} 3^{\mathtt{ndeg}_i(v)}$$

where $\mathtt{ndeg}_i(v)$ denotes again the normalized degree of node v in the tree E^i.

Notice that an iteration of Algorithm 1 swaps out a single edge e^i that is incident to at least one node of normalized degree at least d^i. On the other hand we swap in one edge $\overline{e}^i$ that is incident to two nodes of normalized degree at most d^i-2. The reduction in the potential hence is at least

$$(3^{d^i}+2\cdot 3^{d^i-2})-3\cdot 3^{d^i-1}\geq 2\cdot 3^{d^i-2}$$

Using the range of d^i, we can lower-bound the right hand side of the last inequality by

$$2\cdot 3^{\Delta^i-2\log_b(n)-2}=\Omega\left(\frac{3^{\Delta^i}}{n^2}\right).$$

The initial potential Φ_0 is at most $n\cdot 3^{\Delta^0}$ and the decrease in the potential Φ_i in iteration i is at least $\Omega\left(\frac{\Phi_i}{n^3}\right)$.

In other words, in $O(n^3)$ iterations, we reduce Φ by a constant factor. Hence, the algorithm runs for $O(n^4)$ iterations total. Observing that each iteration can be implemented in time $O(n^2\log n)$ [6], we see that the whole algorithm runs in time $O(n^6\log n)$. $\square$

Acknowledgments R. Ravi thanks IBM Almaden Research Center for hosting an extended visit when working on this paper.

4. REFERENCES

[1] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. In *Proceedings of the Twenty Third Annual ACM Symposium on Theory of Computing*, pages 134–144, 1991.

[2] S. Chopra. On the spanning tree polyhedron. *Operations Research Letters*, 8:25–29, 1989.

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

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

[5] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. In *Proceedings of the 3rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '92)*, pages 307–316, 1992.

[6] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24:296–317, 1995.

[7] M. X. Goemans and D. P. Williamson. *The primal-dual method for approximation algorithms and its applications to network design problems. In D.S. Hochbaum, editor, Approximation Algorithms for NP-hard Problems*, pages 144–191. PWS publishing, Boston, 1997.

[8] 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*, pages 537–546, 2000.

[9] J. Könemann and R. Ravi. A matter of degree: Improved approximation algorithms for degree-bounded minimum spanning trees. *SIAM J. Comput.*, 31(6):1783–1793, 2002.

[10] R. Krishnan and B. Raghavachari. The directed minimum-degree spanning tree problem. *FSTTCS: Foundations of Software Technology and Theoretical Computer Science*, 21, 2001.

[11] J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings, American Mathematical Society*, 7:48–50, 1956.

[12] 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.

[13] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt. Approximation algorithms for degree-constrained minimum-cost network-design problems. *Algorithmica*, 31, 2001.

[14] R. Ravi, B. Raghavachari, and P. Klein. Approximation through local optimality: Designing networks with small degree. In *Proceedings, Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 652 of *LNCS*, pages 279–290. Springer, 1992.