Approximating Minimum Bounded Degree Spanning Trees to within One of Optimal

Mohit Singh, Microsoft Research, Redmond
Lap Chi Lau, The Chinese University of Hong Kong

In the *Minimum Bounded Degree Spanning Tree* problem, we are given an undirected graph $G = (V, E)$ with a degree upper bound B_v on each vertex $v \in V$, and the task is to find a spanning tree of minimum cost that satisfies all the degree bounds. Let OPT be the cost of an optimal solution to this problem. In this paper, we present a polynomial time algorithm which returns a spanning tree T of cost at most OPT and $d_T(v) \leq B_v + 1$ for all v, where $d_T(v)$ denotes the degree of v in T. This generalizes a result of Fürer and Raghavachari [1994] to weighted graphs, and settles a conjecture of Goemans [2006] affirmatively. The algorithm generalizes when each vertex v has a degree lower bound A_v and a degree upper bound B_v, and returns a spanning tree with cost at most OPT and $A_v - 1 \leq d_T(v) \leq B_v + 1$ for all $v \in V$. This is essentially the best possible. The main technique used is an extension of the iterative rounding method introduced by Jain [2001] for the design of approximation algorithms.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Non Numerical Algorithms and Problems—*Computations on discrete structures*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Network Problems, Trees*.

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Approximation algorithms, Spanning Trees, Bounded degree, Iterative rounding.

ACM Reference Format:
Singh, M., Lau, L. C., 2011. Approximating Minimum Bounded Degree Spanning Trees to within One of Optimal ACM V, N, Article A (January YYYY), 18 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

The *Minimum Bounded Degree Spanning Tree* (MBDST) problem is defined as follows: Given an undirected graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}$ on the edges, and a degree upper bound B_v on each vertex $v \in V$, find a spanning tree of minimum cost that satisfies all the degree bounds. When all degree bounds are two (i.e. $B_v = 2$ for all v), the MBDST problem specializes to the *Minimum Cost Hamiltonian Path* problem, and thus even the problem of checking whether there exists a feasible solution is NP-complete. In unweighted graphs, Fürer and Raghavachari [1994] gave an elegant algorithm that returns a spanning tree in which the degree of each vertex is at most $B_v + 1$, or returns a witness certifying that the degree bounds are infeasible. It was conjectured in Goemans [2006] that this result can be generalized to weighted graphs.

Author's addresses: M. Singh, Microsoft Research, 1 Microsoft Way, Redmond, WA 98004, USA.
L. C. Lau, Computer Science and Enginnering, The Chinese University of Hong Kong, Shatin, Hong Kong.

CONJECTURE 1.1. *In polynomial time, one can find a spanning tree of maximum degree at most $k+1$ whose cost is no more than the cost of a minimum cost tree with maximum degree at most k.*

Note that the above conjecture is formulated in the special case where $B_v = k$ for each vertex $v \in V$. Recently, Goemans [2006] made a major step towards proving this conjecture by giving a polynomial time algorithm that returns a tree with maximum degree $k+2$, whose cost is at most the cost of a minimum cost tree with maximum degree at most k. The algorithm also generalizes when there are different degree bounds on different vertices, where the algorithm returns a spanning tree violating the degree bounds by at most two. In this paper, we settle Conjecture 1.1 positively by proving the following result.

THEOREM 1.2. *There exists a polynomial time algorithm for the* Minimum Bounded Degree Spanning Tree *problem that returns a tree T in which each vertex $v \in V$ has degree at most B_v+1 and the cost of the tree T is at most* OPT, *where* OPT *is the minimum cost of a spanning tree which satisfies all degree bounds.*

Theorem 1.2 also generalizes to the setting when there is a degree lower bound A_v and a degree upper bound B_v for each vertex $v \in V$. In this case, the algorithm returns a spanning tree T such that $A_v - 1 \leq d_T(v) \leq B_v + 1$ for each vertex $v \in V$ and the cost of T is at most OPT, where OPT is the minimum cost of a spanning tree which satisfies all degree (upper and lower) bounds. Note that we do not assume that the cost function satisfies triangle inequalities (or even non-negativity). With this general cost function, it is not possible to obtain any approximation algorithm if we insist on satisfying all the degree upper bounds [Garey and Johnson 1979][1]. Thus, Theorem 1.2 is essentially the best possible.

1.1. Techniques

Polyhedral combinatorics has proved to be a powerful, coherent, and unifying tool in combinatorial optimization [Schrijver 2003]. In the last two decades, polyhedral methods have also been applied very successfully to the design of approximation algorithms [Vazirani 2004]. A standard approach to design approximation algorithms is to first formulate the problem as an integer program, and then use the linear relaxation of this program as a way to lower-bound the cost of an optimal solution. We shall also use this approach. Given an undirected graph $G = (V, E)$ and a subset S of vertices, we use $E(S) = \{e \in E \; : \; |e \cap S| = 2\}$ to denote the set of edges which have both endpoints in $S \subseteq V$. For any subset of edges $F \subseteq E$ and vertices $S \subseteq V$, we denote $\delta_F(S)$ to be the set of edges in F which have exactly one endpoint in S. When $F = E$, we drop the subscript F from the notation. For any vertex $v \in V$, we let $\delta_F(v)$ denote $\delta_F(\{v\})$ and $d_F(v) = |\delta_F(v)|$. For $x \; : \; E \to \mathbb{R}^+$ and $F \subseteq E$, we denote $x(F) := \sum_{e \in F} x(e)$. Following Goemans [2006], we use the following natural linear programming relaxation for the *Minimum Bounded Degree Spanning Tree* problem.

[1]Assuming P $\neq$ NP, there is no $p(n)$-approximation which satisfies the degree bound exactly for any polynomial $p(n)$ of n where n is the number of vertices.

$$\begin{array}{llll} \text{minimize} & c(x) & = & \sum_{e \in E} c_e \, x_e & \\ \text{subject to} & x(E(V)) & = & |V| - 1 & \\ & x(E(S)) & \leq & |S| - 1 & \forall\, S \subset V \\ & x(\delta(v)) & \leq & B_v & \forall\, v \in V \\ & x_e & \geq & 0 & \forall\, e \in E \end{array}$$

Using a polyhedral approach, a general strategy is to construct a spanning tree of cost no more than the optimal value of the above linear program, and in which the degree of each vertex $v \in V$ is at most $B_v + 1$. This would prove Theorem 1.2. In fact, this general strategy has been used in previous work, and different techniques have been proposed to “round” the above linear program. An important observation of Goemans is that an *extreme point solution* of the above linear program is characterized by a *laminar family* (definitions will be provided later) of *tight constraints*, inequalities that are satisfied as equalities. This fact was exploited cleverly to obtain the result in Goemans [2006].

We note that a very similar observation was made by Jain [2001] in his breakthrough work on the *Survivable Network Design* problem, where he first introduced the idea of *iterative rounding* to the design of approximation algorithms. This potential connection was initiated in Lau et al. [2009], where Jain’s iterative rounding method was extended to give the first constant factor (bi-criteria) approximation algorithm for bounded degree network design problems including the *Minimum Bounded Degree Steiner Tree* problem and the *Bounded Degree Survivable Network Design* problem. Inspired by these results, we attempted Conjecture 1.1 using the iterative rounding method.

The basic setting of the iterative rounding method for network design problems goes as follows. First, we solve the linear program to obtain an optimal extreme point solution x^*. We proceed by adding the edges with the highest fractional value to the integral solution. Then we construct the *residual problem* where the edges added previously are fixed, and update the linear program appropriately. A key feature of the iterative rounding method is to repeat this procedure: solve again the linear program for the residual problem to obtain an optimal extreme point solution (instead of using x^*), and add the edges with the highest fractional value in this new fractional solution to the integral solution. This procedure is iterated until the integral solution constructed is a feasible solution. In the *Survivable Network Design* problem, the crucial theorem in Jain’s approach is that the edges picked in each iteration have fractional value at least $1/2$, which ensures that the above algorithm has an approximation ratio of two. This theorem relies heavily on the properties of an extreme point solution, as in the work of Goemans [2006].

Our first contribution is to extend the iterative rounding method and show that it can be used to solve problems *optimally*. This is achieved by setting *integral* variables; an variable e such that $x_e = 0$ is removed and a variable e with $x_e = 1$ is picked in the solution. One then formulates the residual problem and iterates. The technical claim is to show that one can always find an integral variable in each iteration. We apply this approach to the minimum spanning tree problem and give two different proofs of the integrality of the spanning tree polyhedron in Section 2.

For the *Minimum Bounded Degree Spanning Tree* problem, however, the direct approach of iterative rounding would not work. The standard iterative rounding framework of picking an edge e with $x_e^* \geq 1/2$ would not work because we can not guarantee the optimality (with respect to the cost of the linear program) of the solution. The latter approach of picking integral edges would not work since we do not expect the linear programming solution to only have integral values. Our algorithm uses the *iterative relaxation* framework as introduced in Lau et al. [2009]. The key insight is that if the algorithm cannot find integral edges to pick, then it can remove/relax one of the degree constraints. In particular, we find a vertex with degree upper bound B_v and with at most $B_v + 1$ edges incident on it in the support of an extreme point solution. The algorithm then removes the degree constraint of v and proceeds to re-solve the linear program. The heart of our analysis is to show that one can always find such a vertex. This is proved by a counting argument which relies heavily on the fact that an extreme point solution is characterized by a laminar family of tight constraints [Jain 2001; Goemans 2006]. Since the algorithm only picks integral edges, the optimality of the cost follows naturally. The condition when the degree constraint of a vertex is removed ensures that the degree constraint can be violated by at most one, giving us Theorem 1.2.

1.2. Related Work

The *Minimum Bounded Degree Spanning Tree* problem is a well studied problem and has been attacked using a variety of techniques. Initial efforts on the problem were concentrated on obtaining bi-criteria approximation algorithms. Let OPT be the cost of an optimal solution to the MBDST problem. An $(\alpha, f(B_v))$-approximation algorithm[2] is an algorithm which returns a spanning tree T with cost at most $\alpha \cdot \text{OPT}$ and $d_T(v) \le f(B_v)$ for all v, where $d_T(v)$ denotes the degree of v in T. Ravi et al. [1993] gave an $(O(\log n), O(B_v \log n))$-approximation for the MBDST problem using a matching-based augmentation technique. Konemann and Ravi [2002; 2005] used a Lagrangian-relaxation based approach to obtain an $(O(1), O(B_v + \log n))$-approximation algorithm. Chaudhuri et al [2009b; 2009a] presented an $(1, O(B_v + \log n))$-approximation algorithm, and an $(O(1), O(B_v))$-approximation algorithm based on the push-relabel framework developed for the maximum flow problem. Ravi and Singh [2006] considered a variant of the problem in which the tree returned must be a minimum spanning tree, and gave an algorithm that returns an MST in which the degree of any vertex v is at most $B_v + p$, where p is the number of distinct costs in any MST. Recently, Goemans [2006] presented an $(1, B_v + 2)$-approximation algorithm using matroid intersection techniques. This was the previous best guarantee for the MBDST problem. In the special case where the graph is unweighted, Fürer and Raghavachari [1994], building on the work of Win [1989], gave an algorithm that returns a spanning tree in which the degree of each vertex v is at most $B_v + 1$ or returns a witness certifying infeasibility of the degree bounds.

The iterative rounding technique that we use in our algorithm was developed in Jain [2001] for the *Survivable Network Design* problem and has later been successfully applied to various problems [Cheriyan et al. 2006; Fleischer et al. 2006]. Recently, this technique has been extended to give constant factor bi-criteria approximation algorithm for the *Bounded Degree Survivable Network Design* problem [Lau et al. 2009; Lau and Singh 2008].

1.3. Organization

The rest of the paper is organized as follows. In Section 2, we give two iterative algorithms which shows the integrality of the spanning tree polyhedron. Then, in Section 3, we present a new proof of the result of Goemans [2006] that gives an $(1, B_v + 2)$-approximation algorithm for the MBDST problem. In Section 4, we present the main algorithm and the proof of Theorem 1.2. Both the results for the MBDST problem build

[2]Notice that the first parameter is used to specify the *ratio*, while the second parameter is used to specify the *actual bound*.

on the iterative algorithms for the spanning tree problem given in Section 2. Finally, in section 5, we extend the algorithm to deal with degree lower bounds.

2. SPANNING TREE POLYHEDRON

In this section, we present two iterative arguments to show that the minimum spanning tree polytope is integral. This motivates the main result of the paper and illustrates the basic proof techniques.

2.1. Linear Program

Let $G = (V, E)$ be a graph with a cost function c on edges. A classical result of Edmonds Edmonds [1971] states that the following linear program LP-MST(G) is integral, and an optimal extreme point solution is always a minimum spanning tree.

$$\begin{array}{lrcll} \text{minimize} & c(x) & = & \sum_{e \in E} c_e x_e & & (1) \\ \text{subject to} & x(E(V)) & = & |V| - 1 & & (2) \\ & x(E(S)) & \leq & |S| - 1 & \forall\, S \subset V & (3) \\ & x_e & \geq & 0 & \forall\, e \in E & (4) \end{array}$$

2.2. Characterization of the Extreme Point Solutions

An extreme point solution is defined as the unique solution of m linearly independent tight constraints (constraints which achieve equality), where m denotes the number of variables in the linear program. We focus on extreme point solutions x^* such that $x^*_e > 0$ for each $e \in E$. We first define some basic definitions. For a set $F \subseteq E$, we define the characteristic vector of F in $\mathbb{R}^{|E|}$, $\chi(F)$, as the vector that has an 1 corresponding to each edge $e \in F$, and 0 otherwise. For any set family $\mathcal{F} \subseteq 2^V$, let $span(\mathcal{F})$ be the vector space generated by the set of vectors $\{\chi(E(S)) \mid S \in \mathcal{F}\}$. Given a set V, subsets S and T are called *intersecting* if $S \setminus T$ and $T \setminus S$ are both non-empty. A family of sets $\mathcal{L} \subseteq 2^V$ is *laminar* if any two sets in the family are not intersecting, i.e., for any two sets in the family, either one contains the other or they are disjoint.

LEMMA 2.1. *Let x^* be any extreme point solution to LP-MST(G) such that $x^*_e > 0$ for each edge $e \in E$ and let $\mathcal{F} = \{S \subseteq V \mid x^*(E(S)) = |S| - 1\}$ be the collection of sets corresponding to tight constraints. Then there exists a laminar family $\mathcal{L} \subseteq \mathcal{F}$ such that*

(1) *The set of vectors $\{\chi(E(S)) : S \in \mathcal{L}\}$ are linearly independent and $span(\mathcal{L}) = span(\mathcal{F})$.*
(2) $|\mathcal{L}| = |E|$.
(3) *x^* is the unique solution to the set of equations $\{x(E(S)) = |S| - 1 : S \in \mathcal{L}\}$.*

The proof of Lemma 2.1 is quite standard [Cornuéjols et al. 1985; Jain 2001], but we include it for completeness. It also illustrates the uncrossing technique.

PROOF. First, we need the following uncrossing lemma on intersecting sets among tight sets.

LEMMA 2.2 ([GOEMANS 2006]). *If $S, T \in \mathcal{F}$ and $S \cap T \neq \emptyset$, then both $S \cap T$ and $S \cup T$ are in $\mathcal{F}$. Furthermore, $\chi(E(S)) + \chi(E(T)) = \chi(E(S \cap T)) + \chi(E(S \cup T))$.*

PROOF. As $S \cap T \neq \emptyset$, we have:

$$\begin{aligned} |S| - 1 + |T| - 1 &= |S \cap T| - 1 + |S \cup T| - 1 \\ &\geq x^*(E(S \cap T)) + x^*(E(S \cup T)) \\ &\geq x^*(E(S)) + x^*(E(T)) \\ &= |S| - 1 + |T| - 1, \end{aligned}$$

and hence we have equality throughout. This implies that $S \cup T$ and $S \cap T$ are both in $\mathcal{F}$, and furthermore there are no edges $e \in E^*$ between $S \setminus T$ and $T \setminus S$. Therefore, $\chi(E(S)) + \chi(E(T)) = \chi(E(S \cap T)) + \chi(E(S \cup T))$. □

Now, we show that any maximal laminar family in $\mathcal{F}$ suffices for the proof.

LEMMA 2.3 ([JAIN 2001]). *If $\mathcal{L}'$ is a maximal laminar subfamily of $\mathcal{F}$, then $span(\mathcal{L}') = span(\mathcal{F})$.*

PROOF. Let $\mathcal{L}'$ be a maximal laminar subfamily of $\mathcal{F}$ and assume that $\chi(E(S)) \notin span(\mathcal{L}')$ for some $S \in \mathcal{F}$. Choose one such set S that intersects as few sets of $\mathcal{L}'$ as possible. Since $\mathcal{L}'$ is a maximal laminar family, there exists $T \in \mathcal{L}'$ that intersects S. From Lemma 2.2, we have that $S \cap T$ and $S \cup T$ are also in $\mathcal{F}$ and that $\chi(E(S)) + \chi(E(T)) = \chi(E(S \cap T)) + \chi(E(S \cup T))$. Since $\chi(E(S)) \notin span(\mathcal{L}')$, either $\chi(E(S \cap T)) \notin span(\mathcal{L}')$ or $\chi(E(S \cup T)) \notin span(\mathcal{L}')$. In either case, we have a contradiction because both $S \cup T$ and $S \cap T$ intersect fewer sets in $\mathcal{L}'$ than S; this is because every set in $\mathcal{L}'$ that intersects $S \cup T$ or $S \cap T$ must intersect S. □

A maximal subfamily $\mathcal{L}$ of $\mathcal{L}'$ such that $\{\chi(E(S)) : S \in \mathcal{L}\}$ is linearly independent suffices to prove property (1). And property (2) and property (3) follow from property (1) and the fact that x^* is an extreme point solution.

Any laminar family $\mathcal{L}$ defines a directed forest L in which nodes correspond to sets in $\mathcal{L}$ and there exists an edge from set R to set S if R is the smallest set containing S. We call R the parent of S and S a child of R. For clarity, we will refer the vertices of forest L by nodes. A node with no parent is called a root and a node with no child is called a leaf. Given a node R, the subtree rooted at R consists of R and all its descendants. The following fact is standard and follows easily from induction.

FACT 2.4. *Let $\mathcal{L} \subset 2^V$ be a laminar family over ground set V. Then $|\mathcal{L}| \leq 2|V| - 1$ and if each set in $\mathcal{L}$ has size at least two then $|\mathcal{L}| \leq |V| - 1$.*

2.3. Iterative Algorithm

Iterative MST Algorithm I

```
(1) Initialization $T \leftarrow \emptyset$.
(2) While $V(G) \neq \emptyset$ do
    (a) Find an optimal extreme point solution $x^*$ of LP-MST($G$) and remove every
        edge $e$ with $x^*_e = 0$ from $G$.
    (b) Find a vertex $v$ with at most one edge $e = uv$ incident at it, and update $T \leftarrow$
        $T \cup \{e\}$, $G \leftarrow G \setminus \{v\}$.
(3) Return $T$.
```

Fig. 1. MST Algorithm I

The iterative MST Algorithm I is given in Figure 1. To prove the correctness of the algorithm, we prove the following two lemmas. First, we prove that the algorithm will terminate.

LEMMA 2.5. *For any extreme point solution x^* to the LP-MST(G) with $x^*_e > 0$ for every edge e, there exists a vertex v with $d(v) = 1$.*

PROOF. Suppose to the contrary that each vertex is of degree at least two. Then $|E| = \frac{1}{2}\sum_{v\in V} d(v) \geq |V|$. On the other hand, since there is no edge e with $x^*_e = 0$, every tight inequality is of the form $x^*(E(S)) = |S| - 1$. By Lemma 2.1, there are $|\mathcal{L}|$ linearly independent tight constraints of the form $x^*(E(S)) = |S| - 1$, where $\mathcal{L}$ is a laminar family with no singleton sets. Moreover, we have $|E| = |\mathcal{L}|$. From Fact 2.4, we have that $|\mathcal{L}| \leq |V| - 1$ which is a contradiction. □

Next, we show that the returned solution is a minimum spanning tree in the following lemma.

LEMMA 2.6. *The Iterative MST Algorithm I returns a minimum spanning tree.*

PROOF. This is proved by induction on the number of iterations of the algorithm. If the algorithm finds a vertex v of degree one (a leaf vertex) in Step 2b with an edge $e = \{u, v\}$ incident at v, then we must have $x^*_e = 1$ since $x(\delta(v)) \geq 1$ is a valid inequality of the LP (subtracting the constraint $x(E(V - v)) \leq |V| - 2$ from the constraint $x(E(V)) = |V| - 1$). Observe that the residual problem is to find a minimum spanning tree on $G' = G \setminus v$, and the same procedure is applied to solve the residual problem recursively. Since $x^*_e = 1$, the restriction of x^* to $E(G')$, denoted by x^*_{res}, is a feasible solution to the LP-MST(G'). Therefore, inductively, the algorithm will return a spanning tree T' of G' of cost at most the optimal value of the LP-MST(G'), and hence $c(T') \leq c \cdot x^*_{res}$. Therefore,

$$c(T) = c(T') + c_e \text{ and } c(T') \leq c \cdot x^*_{res},$$

which imply that

$$c(T) \leq c \cdot x^*_{res} + c_e = c \cdot x^*$$

as $x^*_e = 1$. This shows that the algorithm returns a minimum spanning tree of the graph. □

Remark 2.7. If x^* is an optimal extreme point solution to LP-MST(G), then the residual LP solution x^*_{res}, x restricted to $G' = G \setminus v$ remains an optimal extreme point solution to LP-MST(G'). Hence, in the iterative MST algorithm I, we only need to solve the original linear program once and none of the residual linear programs.

We give another algorithm which achieves the same guarantee as the previous algorithm but is even simpler. Instead of finding integral edges one by one, we can directly prove that all the variables must be integral by a similar counting argument.

Iterative MST Algorithm II

```
(1) Find an optimal extreme point solution $x^*$ to the LP-MST$(G)$ and remove every
    edge $e$ with $x^*_e = 0$ from $E$.
(2) Return $T = \{e : x^*_e > 0\}$.
```

Fig. 2. Iterative MST Algorithm II

LEMMA 2.8. *Iterative MST Algorithm II returns a minimum spanning tree of G.*

PROOF. We first claim that $|T| = |V| - 1$ where T is the set of edges returned by the algorithm. Since T is the support of x^*, and we have $x^*(E) = |V| - 1$ and $x^*_e \leq 1$ for each edge $e \in E$, we must have $|T| \geq |V| - 1$. Moreover, since x^* is extreme, we must have that the number of non-zero edges $|T|$ equals the size of a laminar family $\mathcal{L}$ defining x^* as given by Lemma 2.1. Since, $|\mathcal{L}| \leq |V| - 1$ we have $|T| \leq |V| - 1$ giving us the equality.

Then we also have that $x^*_e = 1$ for each edge $e \in T$. Thus x^* is an integral feasible solution to LP-MST(G) and therefore T is a minimum spanning tree. □

3. A +2 APPROXIMATION ALGORITHM

In this section we prove the following theorem using the iterative relaxation technique.

THEOREM 3.1 ([GOEMANS 2006]). *There exists a polynomial time algorithm for the* Minimum Bounded Degree Spanning Tree *problem that returns a spanning tree T such that $c(T) \leq opt$ and $d_T(v) \leq B_v + 2$ for each vertex $v \in V$, where opt is the cost of the optimal solution satisfying all degree bounds exactly.*

3.1. Linear Programming Relaxation

We use the following standard linear programming relaxation for the MBDST problem, which we denote by LP-MBDST(G, W). In the following, we assume that degree bounds are given for vertices only in a subset $W \subseteq V$.

$$\begin{aligned}
\text{minimize} \quad & c(x) = \sum_{e \in E} c_e x_e && && (5)\\
\text{subject to} \quad & x(E(V)) = |V| - 1 && && (6)\\
& x(E(S)) \leq |S| - 1 && \forall\, S \subset V && (7)\\
& x(\delta(v)) \leq B_v && \forall\, v \in W && (8)\\
& x_e \geq 0 && \forall\, e \in E && (9)
\end{aligned}$$

Observe that LP-MBDST(G, W) has an exponential number of constraints. Cunningham [1984] gave a polynomial time procedure to separate over constraints (6)-(7) and (9). Separating over constraints (8) is clearly in polynomial time. Hence, using the ellipsoid algorithm one can optimize over LP-MBDST(G, W) in polynomial time.

3.2. Characterization of Extreme Point Solutions

We show the following lemma which characterizes any extreme point solution with a family of tight constraints.

LEMMA 3.2. *Let x^* be an extreme point solution of LP-MBDST(G, W) with support E^*. Then there exists a set $X \subseteq W$ and a laminar family $\mathcal{L}$ such that x^* is the unique solution to the following linear system.*

$$\begin{cases} x^*(\delta(v)) = B_v & \forall v \in X \\ x^*(E(S)) = |S| - 1 & \forall S \in \mathcal{L} \end{cases}$$

Moreover, the characteristic vectors $\{\chi(E(S)) : S \in \mathcal{L}\} \cup \{\chi(\delta(v)) : v \in X\}$ are linearly independent. Furthermore, $|E^*| = |\mathcal{L}| + |X|$.

PROOF. The proof follows the same lines as the proof of Lemma 2.1. Let $\tau = \{S \subseteq V : x^*(E(S)) = |S| - 1\}$ be the set of all tight constraints from (6) and (7). Following the same uncrossing arguments as in Lemma 2.1, we obtain that there is a laminar family $\mathcal{L} \subseteq \tau$ such that $span(\tau) = span(\mathcal{L})$ and constraints corresponding to sets in $\mathcal{L}$ are linearly independent. Let X be a maximal set of vertices such that $x^*(\delta(v)) = B_v$ for each $v \in X$ and the constraints in X are linearly independent with constraints in

$\mathcal{L}$. By construction, the constraints corresponding to sets in $\mathcal{L}$ and vertices in τ are maximally linearly independent and therefore $|E^*| = |\mathcal{L}| + |X|$. □

3.3. Iterative Algorithm

Our algorithm proving Theorem 3.1 is given in Figure 3. It is a simple iterative algorithm and an extension of Iterative MST Algorithm I. The main addition is Step (2c) that iteratively relaxes degree constraints.

MBDST Algorithm I

```
(1) Initialization $T \leftarrow \emptyset$, $W \leftarrow V$.
(2) While $V(G) \neq \emptyset$ do
    (a) Find an optimal extreme point solution $x^*$ to LP-MBDST$(G, W)$ and remove
        every edge $e$ with $x^*_e = 0$ from $G$. Let the support of $x^*$ be $E^*$.
    (b) If there exists a vertex $v \in V$ with at most one edge $e = uv$ incident at $v$ in $E^*$,
        then update $T \leftarrow T \cup \{e\}$, $G \leftarrow G \setminus \{v\}$, $W \leftarrow W \setminus \{v\}$, and $B_u \leftarrow B_u - 1$.
    (c) If there exists a vertex $v \in W$ with $d_{E^*}(v) \leq 3$, then update $W \leftarrow W \setminus \{v\}$.
(3) Return $T$.
```

Fig. 3. MBDST Algorithm I

The correctness of the algorithm follows from the following two lemmas.

LEMMA 3.3. *Suppose that the MBDST Algorithm I returns T in Step (3). Then T is a spanning tree of cost at most $c(x^*)$ where x^* is the optimal LP solution to LP-MBDST(G, V), and the degree of vertex v in T is at most $B_v + 2$ for each vertex $v \in V$.*

PROOF. Assuming that the algorithm returns a solution T in Step (3), then the fact that T is a spanning tree of cost at most $c(x^*)$ follows from an argument identical to that in the proof of Lemma 2.6. The degree bound follows simply as well. While a vertex is in W, the degree constraint is satisfied exactly by the current fractional solution. If a vertex v is removed from W in Step (2c), it has only three edges incident at it and the residual degree bound B_v is at least one. Thus, even if the algorithm picks all three edges incident at v, the degree bound is violated by at most two. □

In the next lemma, we prove that in each iteration we can proceed by applying either Step 2b or Step 2c; this will ensure that the algorithm terminates.

LEMMA 3.4. *Any extreme point solution x^* to LP-MBDST(G, W) with support E^* must satisfy one of the following.*
(a) There is a vertex $v \in V$ with $d_{E^*}(v) = 1$.
(b) There is a vertex $v \in W$ with $d_{E^*}(v) \leq 3$.

PROOF. Suppose by contradiction that both (a) and (b) are not satisfied. Then every vertex has at least two edges incident on it and every vertex in W has at least four edges incident on it. Therefore, $|E^*| \geq (2(n - |W|) + 4|W|)/2 = n + |W|$, where $n = |V(G)|$.
By Lemma 3.2, there is a laminar family $\mathcal{L}$ and a set $X \subseteq W$ of vertices such that $|E^*| = |\mathcal{L}| + |X|$. As $\mathcal{L}$ contains subsets of size at least two, $|\mathcal{L}| \leq |V| - 1$. Hence, $|E^*| = |\mathcal{L}| + |X| \leq |V| - 1 + |X| \leq |V| - 1 + |W|$, a contradiction. □

In each iteration, we either remove a degree constraint or include an edge in our solution. Therefore, in a total of at most $|V| + |V| - 1 = 2|V| - 1$ iterations, we construct a spanning tree. These steps provide a simple $(1, B_v + 2)$-approximation algorithm for the MBDST problem.

4. A +1 APPROXIMATION ALGORITHM

In this section, we give an algorithm that proves Theorem 1.2. The algorithm is an extension of the Iterative MST Algorithm II and is given in Figure 4. In each step of the algorithm, we find an appropriately chosen vertex whose degree constraint is present and remove the degree constraint. Finally, when all the degree constraints are removed, we return the support of the linear programming solution. This is guaranteed to be a spanning tree following from the analysis of Iterative MST Algorithm II in Lemma 2.8.

```
MBDST Algorithm II
(1) Let $W \leftarrow V$.
(2) While $W \neq \emptyset$ do
    (a) Find an optimal extreme point solution $x^*$ to LP-MBDST$(G, W)$ and remove
        every edge $e$ with $x^*_e = 0$ from $G$. Let the support of $x^*$ be $E^*$.
    (b) If there exists a vertex $v \in W$ such that $d_{E^*}(v) \leq B_v + 1$ then update $W \leftarrow$
        $W \setminus \{v\}$.
(3) Find an optimal extreme point solution $x^*$ of LP-MBDST$(G, \emptyset)$ and remove every
    edge $e$ with $x^*_e = 0$ from $G$.
(4) Return $T$, the support of $x^*$.
```

Fig. 4. MBDST Algorithm II

We argue the correctness of the algorithm in the following two lemmas. In the first part, we assume that the algorithm does terminate and returns the set T when $W = \emptyset$, and show that the solution returned satisfies the claimed guarantee. In the next lemma, we show that in each iteration, the algorithm finds a vertex $v \in W$ satisfying the condition in Step (2b).

LEMMA 4.1. *Suppose the MBDST Algorithm II returns T in Step (4), then T is a spanning tree of cost at most $c(x^*)$ where x^* is the optimal LP solution to LP-MBDST(G, V) and degree of vertex v in T is at most $B_v + 1$ for each vertex $v \in V$.*

PROOF. First we show that T is a tree. At the end of the last iteration, let G' denote the current graph where edges with fractional value zero have been removed in previous iterations. The algorithm then solves LP-MBDST$(G', \emptyset)$ which is exactly the same as linear program LP-MST(G'). Thus, from Lemma 2.8, the optimal solution to the linear program is integral and T is a spanning tree. It is straightforward to check that the optimal linear programming solution of any iteration remains feasible for the next iteration, since we either remove a constraint or remove edges which are set to zero by the fractional solution. Thus the cost of the optimal solution in next iteration does not increase. Therefore, the cost of T is at most the cost of the optimal solution to the initial linear program LP-MBDST(G, W).

Next, we argue that the degree of each vertex $v \in V$ in the returned solution T is at most $B_v + 1$. Let v be any vertex and consider the iteration when the degree constraint of v is removed. Let E^* be the support of the linear programming solution in that iteration. We have $d_{E^*}(v) \leq B_v + 1$ and in any further iteration we only remove edges incident at this vertex and never add any other edge. Thus the final solution $T \subseteq E^*$ and therefore $d_T(v) \leq B_v + 1$ as claimed. $\square$

In the next lemma, we show that the algorithm does perform correctly and returns a solution. We describe a simpler proof due to Bansal et al. [2008].

LEMMA 4.2. *Let x^* be an extreme point solution to **LP-MBDST**(G, W) and $E^* = \{e : x^*_e > 0\}$. If $W \neq \emptyset$ then there exists a $v \in W$ such that $d_{E^*}(v) \leq B_v + 1$.*

PROOF. Let $X \subseteq W$ be the tight degree constraints and $\mathcal{L}$ be laminar family as given by Lemma 2.1. If $X = \emptyset$, then Lemma 2.8 implies that E^* is a tree and $x^*_e = 1$ for each $e \in E^*$. Thus, $d_{E^*}(v) = x^*(\delta(v)) \leq B_v$ for each $v \in W$ and we are done. In the rest we assume that $X \neq \emptyset$.

Suppose to the contrary that $d_{E^*}(v) \geq B_v + 2$ for each $v \in W$. We now show a contradiction by a counting argument. We give one token to each edge in E^*. We then redistribute the token such that each vertex in X and each set in $\mathcal{L}$ gets one token and we still have extra tokens left. This will contradict $|E^*| = |X| + |\mathcal{L}|$. The token redistribution is as follows. Each edge $e \in E^*$ gives x^*_e fractional token to the smallest set in $\mathcal{L}$ containing both endpoints of e, and $(1 - x^*_e)/2$ fractional token to each of its endpoints for the degree constraints, if present.

We show that each set $S \in \mathcal{L}$ obtains one token. S receives x^*_e token for each edge e such that S is the smallest set containing both endpoints of e. Let $R_1, \ldots, R_k$ be the children of S in the laminar family $\mathcal{L}$ where $k \geq 0$. We have

$$\begin{aligned} x^*(E(S)) &= |S| - 1, \text{ and} \\ x^*(E(R_i)) &= |R_i| - 1 \text{ for each } 1 \leq i \leq k \\ \implies x^*(E(S)) - \sum_{i=1}^{k} x^*(E(R_i)) &= |S| - 1 - \sum_{i=1}^{k}(|R|_i - 1) \\ \implies x^*(A) &= |S| - 1 - \sum_{i=1}^{k}(|R_i| - 1), \end{aligned}$$

where $A = E(S) \setminus (\cup_{i=1}^{k} E(R_i))$. Observe that S receives x^*_e tokens for each edge $e \in A$ for a total of $x^*(A)$ tokens which is an integer by the above equation. If $x^*(A) = 0$ then $A = \emptyset$ and therefore, $\chi(E(S)) = \sum_{i=1}^{k} \chi(E(R_i))$ which contradicts the linear independence of the constraints. Hence, each set S receives at least one token.

Now, we show that each vertex with a tight degree constraint gets one token. Let $v \in X$ be such a vertex. Then v receives $(1 - x^*_e)/2$ token for each edge incident at v for a total token of value

$$\sum_{e \in \delta_{E^*}(v)} \frac{1 - x^*_e}{2} = \frac{d_{E^*}(v) - B_v}{2} \geq 1,$$

where the first equality holds since $\sum_{e \in \delta_{E^*}(v)} x^*_e = B_v$ and the inequality holds since $d_{E^*}(v) \geq B_v + 2$.

To finish the proof, we argue that there is some extra token left for contradiction. If $V \notin \mathcal{L}$, then there exists an edge e which is not contained in any set of $\mathcal{L}$ and the x^*_e token for that edge gives us the contradiction. Similarly, if there is a vertex $v \in W \setminus X$ then v also collects one token which it does not need and we get the desired contradiction. Moreover, if there is a vertex $v \in V \setminus X$ then each edge e incident at v must have $x^*_e = 1$; otherwise the $(1 - x^*_e)/2 > 0$ token is extra. Note that $\chi(e) \in span(\mathcal{L})$ for each e with $x^*_e = 1$, since e is a tight set of size two. We have

$$2\chi(E(V)) = \sum_{v \in V} \chi(\delta(v)) = \sum_{v \in X} \chi(\delta(v)) + \sum_{v \in V - X} \chi(\delta(v)) = \sum_{v \in X} \chi(\delta(v)) + \sum_{v \in V - X} \sum_{e \in \delta(v)} \chi(e).$$

We have argued that $V \in \mathcal{L}$ and $\chi(e) \in span(\mathcal{L})$ for each edge $e \in \delta(v)$ for $v \in V - X$. Since $X \neq \emptyset$, this implies the linear independence of the tight constraints in X and those in $\mathcal{L}$, giving us the contradiction. □

5. UPPER AND LOWER DEGREE BOUNDS

In this section, we consider an extension of the MBDST problem in which a degree lower bound A_v and a degree upper bound B_v are given for each vertex v, and the goal is to find a tree of minimum cost satisfying both upper and lower degree bounds. We prove the following theorem.

THEOREM 5.1. *There is a polynomial time algorithm for the* Minimum Bounded Degree Spanning Tree *problem with upper and lower degree constraints which returns a tree T such that $A_v - 1 \leq d_T(v) \leq B_v + 1$ for each $v \in V$ and $c(T) \leq opt$, where opt is the cost of the optimal tree satisfying degree bounds exactly.*

5.1. Linear Programming Relaxation

We assume the degree bounds are given on a subset of vertices $W \subseteq V$. The following is a linear programming relaxation for the MBDST problem, which is denoted by LP-MBDST(G, W, F). The edge set F denotes the set of edges which the linear program has assigned a value of one. We maintain that in all further iterations edges in F will be chosen integrally.

$$\begin{array}{lrcll}
\text{minimize} & c(x) & = & \sum_{e \in E} c_e x_e & \\
\text{subject to} & x(E(V)) & = & |V| - 1 & \\
& x(E(S)) & \leq & |S| - 1 \quad \forall\, S \subseteq V & \\
& x(\delta(v)) & \geq & A_v & \forall\, v \in W \\
& x(\delta(v)) & \leq & B_v & \forall\, v \in W \\
& x_e & = & 1 & \forall\, e \in F \\
& x_e & \geq & 0 & \forall\, e \in E
\end{array}$$

5.2. Characterization of Extreme Point Solutions

We give the following characterization of the extreme point solutions of LP-MBDST(G, W, F). The proof of the following lemma is similar to the proof of Lemma 3.2.

LEMMA 5.2. *Let x^* be any extreme point solution to LP-MBDST(G, W, F). Then there exists a set $X_L \subseteq W$, a set $X_U \subseteq W$ and a laminar family $\mathcal{L}$ such that x^* is the unique solution to the following linear system.*

$$\begin{cases} x^*(\delta(v)) = A_v & \forall v \in X_L \\ x^*(\delta(v)) = B_v & \forall v \in X_U \\ x^*(E(S)) = |S| - 1 & \forall S \in \mathcal{L} \end{cases}$$

Moreover,

(1) *the vectors $\{\chi(E(S)) : S \in \mathcal{L}\} \cup \{\chi(\delta(v)) : v \in X_L\} \cup \{\chi(\delta(v)) : v \in X_U\}$ are linearly independent.*
(2) $|E^*| = |\mathcal{L}| + |X_L| + |X_U|$.
(3) *Let $\mathcal{C}$ denote the set of vertex sets of the non-trivial connected components spanned by edges in F. Then $\mathcal{C} \subseteq \mathcal{L}$.*

PROOF. The proof follows along the same lines as in the proofs of Lemma 2.1 and Lemma 3.2. Let $\tau = \{S \subseteq V : x^*(E(S)) = |S| - 1\}$ be the set of all tight spanning tree constraints in LP-MBDST(G, W, F). By uncrossing arguments, we can choose any maximal laminar $\mathcal{L}$ such that the constraints for sets in $\mathcal{L}$ are linearly independent to ensure that $span(\mathcal{L}) = span(\tau)$. Since the constraint for each set $S \in \mathcal{C}$ is tight and all these constraints are linearly independent, we can choose a laminar family $\mathcal{L}$ such that $\mathcal{C} \subseteq \mathcal{L}$. Rest of the conditions follow simply by choosing a maximal collection of vertices with tight degree constraints X_L and X_U while ensuring linear independence. $\square$

5.3. Iterative Algorithm

The algorithm is given in Figure 5. In each iteration, we either pick an integral edge or remove one of the degree constraints. Due to presence of upper and lower degree constraints, we only remove degree constraints on vertices which have at most two strictly fractional edges incident at them.

```
MBDST Algorithm III
(1) Initialize $F \leftarrow \emptyset$, $W \leftarrow V$.
(2) While $F$ is not a spanning tree do
    (a) Find an optimal extreme point solution $x^*$ to LP-MBDST$(G, W, F)$ and remove
        every edge $e$ with $x^*_e = 0$ from $G$.
    (b) If there exists an edge $e = \{u, v\}$ such that $x^*_e = 1$, then $F \leftarrow F \cup \{e\}$.
    (c) If there exists a vertex $v \in W$ such that there are at most two edges incident at
        $v$ with $0 < x^*_e < 1$, then update $W \leftarrow W \setminus \{v\}$.
(3) Return $F$.
```

Fig. 5. MBDST Algorithm III

For the correctness of the MBDST Algorithm III, we shall prove the following key lemma, which will ensure that the algorithm terminates.

LEMMA 5.3. *An extreme point solution x^* of LP-MBDST(G, W, F) with support E^* must satisfy one of the following.*
(a) There is an edge $e \in E^* \setminus F$ such that $x^*_e = 1$ or $x^*_e = 0$.
(b) There is a vertex $v \in W$ such that there at most two edges incident at v with fractional value strictly between 0 and 1.

Before we prove Lemma 5.3, we use it to prove Theorem 5.1.

Proof of Theorem 5.1: Lemma 5.3 ensures that in each iteration we either remove a degree constraint or reduce the number of strictly fractional edges. It is also straightforward to see that the same linear programming solution remains an extreme point solution after we apply Step 2a or 2b, and we need to resolve the linear program only after we apply Step 2c. Hence, the algorithm will terminate in at most $|V|$ iterations. Moreover, the linear programming solution at any iteration is feasible to the linear program solved in the next iteration. Thus the cost of the optimal solution to the linear programming formulation can only decrease. Hence, the cost of the final solution returned is at most the cost of the optimal fractional solution to the initial linear program, which is a lower bound on the optimum.

Consider any vertex $v \in V$. We argue that both upper and lower degree constraint are satisfied within an additive error of one. Consider the iteration when the constraint for vertex v is removed. Let x^* denote the optimal fractional solution obtained in this iteration, $F' = \{e : x^*_e = 1\}$ denote the set of edges with fractional value one, and $E^* = \{e : x^*_e > 0\}$ denote the support of x^*. Since the constraint for vertex v is removed in this iteration, we have that $d_{E^* \setminus F'}(v) \leq 2$. Since every edge in F' incident at v will be present in the final tree F returned by the algorithm, $d_F(v) \geq d_{F'}(v) \geq \lfloor x^*(\delta_{E^*}(v)) \rfloor - 1 \geq A_v - 1$, where we use the fact that $\delta_{E^*}(v)$ has at most two fractional edges. Similarly,

F can contain at most two more edges incident at v that are not included in F'. Thus $d_F(v) \le d_{F'}(v) + 2 \le \lceil x^*(\delta_{E^*}(v)) \rceil + 1 \le B_v + 1$. □

It remains to complete the proof of Lemma 5.3.

Proof of Lemma 5.3: Let $\mathcal{L}$ be the laminar family and $X := X_L \cup X_U$ be the vertices defining the solution x^* as in Lemma 5.2. Suppose that both (a) and (b) of Lemma 5.3 are not satisfied. We shall derive that $|\mathcal{L}| + |X| < |E^*|$, which will contradict Lemma 5.2 and complete the proof. We also let $E' = E^* \setminus F$ be the set of edges which are fractional in the linear programming solution x^*.

As before, we do this by a counting argument. We first assign two tokens for each edge e for a total of $2|E^*|$ tokens. In the first assignment, if $e \in E'$, i.e., $x^*_{uv} < 1$, it gives one token to each of its endpoints u and v. If $e \in F$, i.e., $x^*_{uv} = 1$, it gives both of its tokens to the smallest set in $\mathcal{L}$ containing both u and v.

Let $S \in \mathcal{C}$ be the vertex set of any non-trivial connected component of F. We first show that the tokens from edges in F are enough to give two tokens to each set in $R \in \mathcal{L}$ such that $R \subseteq S$.

CLAIM 5.4. *The tokens assigned by edges in F are enough to give two tokens to each member $R \in \mathcal{L}$ such that $R \subseteq S$ for some $S \in \mathcal{C}$.*

PROOF. Observe that every edge e with both endpoints in R must be in F and therefore $x^*_e = 1$. Let $R_1, \ldots, R_k$ be the children of R in forest L. Then we have $E^*(R) \setminus \cup_i E^*(R_i) \neq \emptyset$, since the constraint for set R is independent of the constraints for set R_i's. So, any edge $e \in E^*(R) \setminus \cup_i E^*(R_i)$ gives two tokens to R as claimed. □

We now argue that enough tokens are assigned to rest of the sets in $\mathcal{L}$ and vertices in X by a careful inductive argument. We first construct a new laminar family of the remaining sets. We remove all sets from $\mathcal{L}$ that are *strictly* contained in some $S \in \mathcal{C}$. We also add all singleton sets $\{v\}$ that are not contained in any $S \in \mathcal{C}$ to the laminar family. We call the new laminar family $\mathcal{L}'$ and L' be the forest associated with laminar family $\mathcal{L}'$. By construction, leaves of $\mathcal{L}'$ are either sets of size one or sets in $\mathcal{C}$ and form a partition of V. Similarly, for any node S, the set of children of S forms a partition of S.

Observe that sets in $\mathcal{C}$ do not need to be assigned any new tokens since edges in F assign them two token by Claim 5.4. Similarly sets of size one, $\{v\}$ where $v \notin X$ do not need any tokens. But we include them in $\mathcal{L}'$ for induction. Let Y denote the set of vertices which have an edge in E' incident on them. We call the vertices in Y active. Observe that in the initial assignment each vertex in X gets at least three tokens since it has at least three edges from E' incident on it. Since we need to assign only two tokens to the degree constraint, it has at least one extra token. An active vertex not in X does not need any token. Thus every active vertex has at least one extra token. These extra tokens will be carefully reassigned to sets in $\mathcal{L}'$.

The following definition gives a characterization of those sets which need a careful analysis.

Definition 5.5. A set $S \in \mathcal{L}'$ such that $S \neq V$ is *special* if:

(1) $|\delta_{E'}(S)| = 3$;
(2) $x^*(\delta_{E'}(S)) = 1$ or $x^*(\delta_{E'}(S)) = 2$;
(3) $\chi(\delta_{E'}(S))$ is a linear combination of the characteristic vectors of its descendants (including possibly $\chi(E'(S))$) and the characteristic vectors $\chi(\delta_{E'}(v))$ of $v \in S \cap X$.

From now on, to reduce notation, we drop the subscript E' in $d_{E'}$ and $\delta_{E'}$.

CLAIM 5.6. *A leaf in $\mathcal{L}'$ is special if and only if it is in one of the following two cases:*

(1) $S \in \mathcal{C}$ and contains exactly one active vertex v where $v \in X$ and $|\delta(v)| = 3$, or
(2) $S = \{v\}$ where $v \in X$ and $|\delta(v)| = 3$.

PROOF. Consider a special leaf S where $S \in \mathcal{C}$ or $S = \{v\}$. Then $\delta(S) = \cup_{u \in S \cap Y} \delta(u)$ and $\delta(u) \cap \delta(w) = \emptyset$ since there is no edge in E' with both endpoints $\{u, w\} \in S$. From Property 3 of a special set, it follows that each $v \in Y \cap S$ must be in X. Since $|\delta(S)| = 3$ and $|\delta(v)| \geq 3$ for each $v \in X$, we have that $\delta(S) = \delta(v)$ for some $v \in X \cap S$ and $|\delta(v)| = 3$ as required. Therefore, if $S \in \mathcal{C}$, there is exactly one active vertex $v \in S$ and $v \in X$ and $|\delta(v)| = 3$. Similarly if $S = \{v\}$, then $v \in X$ and $|\delta(v)| = 3$.

For the other direction, every leaf which contains exactly one active vertex where $v \in X$ with $|\delta(S)| = 3$ satisfies the first condition. The second condition is satisfied since each of the tree edges incident at it are strictly fractional and therefore $0 < x^*(\delta(v)) < 3$. Since, $v \in X$, $x^*(\delta(v))$ is an integer and therefore must equal 1 or 2 satisfying the second condition. The third condition is straightforward since $\chi(\delta(S)) = \chi(\delta(v))$. ☐

The following lemma will complete the proof of Lemma 5.3, and hence Theorem 5.1. The lemma says that inductively we can assign required number of token to each set in $\mathcal{L}'$, and at least one extra token to the root. Of course, the required number of tokens is zero for each leaf $S \in \mathcal{C}$ and $S = \{v\}$ where $v \notin X$. Rest of the sets require at least two tokens. Moreover, the root will get exactly one extra token only if it is special.

LEMMA 5.7. *For any rooted subtree of the forest $L' \neq \emptyset$ with root S, we can distribute the tokens assigned to vertices inside S such that every vertex in $X \cap S$ gets at least two tokens. Moreover, every node in the subtree which is not in $\mathcal{C}$ or $\{v\}$ where $v \notin X$ gets at least two tokens. Finally, the root gets at least one extra token and exactly one only if S is special or $S = V$.*

PROOF. First we prove a claim needed for the lemma.

CLAIM 5.8. *If $S \neq V$ and $S \in \mathcal{L}'$ then $|\delta(S)| \geq 2$.*

PROOF. Since $S \neq V$, $x^*(\delta(S)) \geq 1$ is a valid inequality of the LP. Since $S \in \mathcal{L}'$ and is not strictly contained in the vertex set of a component spanned by F, we have that $F \cap \delta(S) = \emptyset$. Thus $x^*_e < 1$ for each $e \in \delta(S)$ and $|\delta(S)| \geq 2$. ☐

Base Case: Consider a leaf S of tree L'. First consider the case when $S = \{v\}$. From Claim 5.6, if $v \in X$, we have $d(v) \geq 3$ and exactly three only if $\{v\}$ is special. In the initial assignment v is assigned $d(v)$ tokens. Since v needs to be assigned two tokens, it has at least one extra token and exactly one extra token only if $S = \{v\}$ is special. If $v \notin X$, then v receives $d(v) \geq 2$ tokens from Claim 5.8. Since it does not need any tokens, both tokens are extra tokens.

Next, consider the case when $S \in \mathcal{C}$. Then again each active vertex in $S \cap Y$ receives at least one extra token. If S contains two active vertices, we have the two extra tokens to assign to S. Recall that S itself does not need any tokens since it has been assigned two tokens from edges in F. Otherwise, consider the case when S contains exactly one active vertex, say v. If $v \notin X$, then $|\delta(S)| = |\delta(v)| \geq 2$ from Claim 5.8 and we have the two extra tokens. If $v \in X$, then $|\delta(v)| = 3$ if and only if S is special from Claim 5.6 and $|\delta(v)| \geq 4$ otherwise. Since v needs to be assigned two tokens, we can assign one extra token to S if S is special and two otherwise.

Induction Step: The following two claims are useful in the induction step. Recall that $E^*(S)$ denotes the set of edges with both endpoints in S. We denote by $D(S)$ the set of edges with endpoints in *different* children of S in the forest L'. Observe that if S is not a leaf in $\mathcal{L}'$, then $D(S) \subseteq E'$, i.e. all edges in $D(S)$ are strictly fractional.

CLAIM 5.9. *If $S \in \mathcal{L}'$ has $r \geq 1$ children, then $x^*(D(S)) = r - 1$.*

PROOF. Let the children of S be R_i, $1 \le i \le r$. We have

$$x^*(E^*(R_i)) = |R_i| - 1.$$

Observe that the above equality also holds for children R_i where $|R_i| = 1$. In this case $E^*(R_i) = \emptyset$. As $S \in \mathcal{L}$, we have

$$x^*(E^*(S)) = |S| - 1.$$

Since $S = \uplus_i R_i$, we obtain

$$\begin{aligned} x^*(D(S)) &= x^*(E^*(S)) - \sum_i x^*(E^*(R_i)) \\ &= |S| - 1 - \sum_i (|R_i| - 1) \\ &= (|S| - \sum_i |R_i|) + r - 1 = r - 1, \end{aligned}$$

because $|S| = \sum_i |R_i|$. □

CLAIM 5.10. *Suppose a set $S \neq V$ contains exactly three special children R_1, R_2, R_3 and $|D(S)| \ge 3$. Then S is a special set.*

PROOF. Note that

$$|\delta(S)| = |\delta(R_1)| + |\delta(R_2)| + |\delta(R_3)| - 2|D(S)| = 3 + 3 + 3 - 2|D(S)| = 9 - 2|D(S)|.$$

Since $S \neq V$, we have $|\delta(S)| \ge 2$ by Claim 5.8. As $|D(S)| \ge 3$, the only possibility is that $|D(S)| = 3$ and $|\delta(S)| = 3$, which satisfies the first property of a special set. Also, we have $x^*(\delta(S)) = x^*(\delta(R_1)) + x^*(\delta(R_2)) + x^*(\delta(R_3)) - 2x^*(D(S))$. As each term on the right hand side is an integer, it follows that $x^*(\delta(S))$ is an integer. Since $\delta(S) \cap F = \emptyset$, we have $x^*(\delta(S)) < |\delta(S)| = 3$ and thus $x^*(\delta(S))$ is either equal to one or two, and so the second property of a special set is satisfied. Finally, note that

$$\chi(\delta(S)) = \chi(\delta(R_1)) + \chi(\delta(R_2)) + \chi(\delta(R_3)) + \chi(E'(R_1)) + \chi(E'(R_2)) + \chi(E'(R_3)) - 2\chi(E'(S)).$$

Here, the vector $\chi(E'(R_i))$ will be the zero vector if R_i is a special vertex. Since R_1, R_2, R_3 satisfy the third property of a special set, S satisfies the third property of a special set. □

Consider the following cases for the induction step.

(1) S contains at least four children. Each child has at least one excess token. Therefore, S can collect at least four tokens by taking one excess token from each child. Since S needs two tokens, the other two tokens are extra.
(2) S has exactly three children. If any child has at least two excess tokens, then S can collect four tokens, and we are done. Otherwise, each child has only one excess token, and thus is special by the induction hypothesis. If $S = V$, then S can collect three tokens, and this is enough since V is the root of the laminar family. Else, we have $x^*(D(S)) = 2$ from Claim 5.9. Since there is no edge $e \in D(S)$ with $x^*_e = 1$, we must have $|D(S)| > x^*(D(S)) = 2$. Now, it follows from Claim 5.10 that S is special and it only requires three tokens.
(3) S contains exactly two children R_1, R_2. If both R_1, R_2 have at least two excess tokens, then S can collect four tokens, and we are done. Otherwise, one of the children has exactly one excess token, say R_1. Hence, R_1 is special by the induction hypothesis. We now show a contradiction to the independence of tight constraints defining x^*, and hence this case would not happen.
Since S contains two children, Claim 5.9 implies that $x^*(D(S)) = 1$. Since there is no edge $e \in D(S)$ with $x^*_e = 1$, we have $|D(S)| = |\delta(R_1, R_2)| \ge 2$. Also, R_1 is special and

thus $|\delta(R_1)| = 3$. We claim $\delta(R_1, R_2) = \delta(R_1)$. If not, then let $e = \delta(R_1) \setminus \delta(R_1, R_2)$. Then

$$x^*_e = x^*(\delta(R_1)) - x^*(\delta(R_1, R_2)) = x^*(\delta(R_1)) - x^*(D(S)).$$

But $x^*(\delta(R_1))$ is an integer as R_1 is special and $x^*(D(S)) = 1$. Therefore, x^*_e is an integer which is a contradiction. Thus $\delta(R_1, R_2) = \delta(R_1)$. But then

$$\chi(E^*(S)) = \chi(E^*(R_1)) + \chi(\delta(R_1)) + \chi(E^*(R_2))$$

if $R_2 \in \mathcal{L}$ or

$$\chi(E^*(S)) = \chi(E^*(R_1)) + \chi(\delta(R_1))$$

if R_2 is a singleton vertex. R_1 is special implies that $\chi(\delta(R_1))$ is a linear combination of the characteristic vectors of its descendants and the characteristic vectors $\{\chi(\delta(v)): v \in R_1 \cap X\}$. Hence, in either case $\chi(E^*(S))$ is spanned by $\chi(E^*(R))$ for $R \in \mathcal{L} \setminus \{S\}$ and $\chi(\delta(v))$ for $v \in S \cap X$ which is a contradiction to the linear independence of the constraints in $\mathcal{L}$.

This completes the proof of Lemma 5.7 and Theorem 5.1. □

□

6. CONCLUDING REMARKS AND OPEN QUESTIONS

In this paper, we extend the iterative rounding framework to obtain the best possible guarantee for the MBDST problem. This built on the integrality proofs of the spanning tree polyhedron for the spanning tree polytope. The iterative rounding framework can be used to obtain new proofs of integrality of linear programming formulations for many combinatorial optimization problems including matchings in bipartite and general graphs, matroid bases and matroid intersection, submodular flows, etc. While the proof ideas are similar to other general techniques like total dual integrality, the new iterative proofs are crucial for problems which can be modeled by introducing extra side constraints, for example the degree constraints in the minimum bounded degree spanning tree problem. We refer the reader to [Singh 2008; Lau et al. 2011] for further applications of the technique.

A closely related problem is the well studied travelling salesperson problem (TSP). The sub-tour elimination relaxation for TSP is very similar to the LP relaxation for the MBDST problem. Indeed our techniques can be used to give the following polyhedral result: Any solution to the sub-tour elimination polytope can be written as a convex combination of 1-trees each of maximum degree three and average degree two, improving on a similar result of Goemans Goemans [2006]. Here, an 1-tree is a tree on $V \setminus v$ along with any two edges incident at vertex v. A natural open question is whether the techniques used here can be used to obtain better approximation algorithm for TSP.

Acknowledgements

The collaboration started when the authors visited Microsoft Research in Redmond; we are very grateful to the Theory Group for their hospitality. The first author would like to thank R. Ravi for introducing him to the MBDST problem and innumerable discussions over the past few years. We also thank R. Ravi for help in the preparation of the paper.

REFERENCES

BANSAL, N., KHANDEKAR, R., AND NAGARAJAN, V. 2008. Additive guarantees for degree bounded directed network design. In *STOC*. 769–778.

CHAUDHURI, K., RAO, S., RIESENFELD, S., AND TALWAR, K. 2009a. A push-relabel approximation algorithm for approximating the minimum-degree MST problem and its generalization to matroids. *Theoretical Computer Science 410,* 44, 4489–4503.

CHAUDHURI, K., RAO, S., RIESENFELD, S., AND TALWAR, K. 2009b. What Would Edmonds Do? Augmenting Paths and Witnesses for Degree-Bounded MSTs. *Algorithmica 55,* 1, 157–189.

CHERIYAN, J., VEMPALA, S., AND VETTA, A. 2006. Network Design Via Iterative Rounding Of Setpair Relaxations. *Combinatorica 26,* 3, 255–275.

CORNUÉJOLS, G., FONLUPT, J., AND NADDEF, D. 1985. The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming 33*, 1–27. 10.1007/BF01582008.

CUNNINGHAM, W. H. 1984. Testing membership in matroid polyhedra. *Journal of Combinatorial Theory, Series B 36,* 2, 161–188.

EDMONDS, J. 1971. Matroids and the greedy algorithm. *Mathematical Programming 1*, 127–136. 10.1007/BF01584082.

FLEISCHER, L., JAIN, K., AND WILLIAMSON, D. P. 2006. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *Journal of Computer and System Sciences 72,* 5, 838–867.

FÜRER, M. AND RAGHAVACHARI, B. 1994. Approximating the Minimum-Degree Steiner Tree to within One of Optimal. *Journal of Algorithms 17,* 3, 409–423.

GAREY, M. R. AND JOHNSON, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

GOEMANS, M. X. 2006. Minimum Bounded Degree Spanning Trees. In *47th Annual IEEE Symposium on Foundations of Computer Science*. 273–282.

JAIN, K. 2001. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica 21,* 1, 39–60.

KÖNEMANN, J. AND RAVI, R. 2002. A Matter of Degree: Improved Approximation Algorithms for Degree-Bounded Minimum Spanning Trees. *SIAM Journal of Computing 31,* 6, 1783–1793.

KÖNEMANN, J. AND RAVI, R. 2005. Primal-Dual Meets Local Search: Approximating MSTs With Nonuniform Degree Bounds. *SIAM Journal of Computing 34,* 3, 763–773.

LAU, L. C., NAOR, J., SALAVATIPOUR, M. R., AND SINGH, M. 2009. Survivable Network Design with Degree or Order Constraints. *SIAM Journal of Computing 39,* 3, 1062–1087.

LAU, L. C., RAVI, R., AND SINGH, M. 2011. *Iterative Methods in Combinatorial Optimization*. Cambridge Texts in Applied Mathematics. Cambridge University Press.

LAU, L. C. AND SINGH, M. 2008. Additive approximation for bounded degree survivable network design. In *STOC*. 759–768.

RAVI, R., MARATHE, M. V., RAVI, S. S., ROSENKRANTZ, D. J., AND III, H. B. H. 1993. Many birds with one stone: multi-objective approximation algorithms. In *Twenty-Fifth Annual ACM Symposium on Theory of Computing*. 438–447.

RAVI, R. AND SINGH, M. 2006. Delegate and Conquer: An LP-Based Approximation Algorithm for Minimum Degree MSTs. In *33rd International Colloquium on Automata, Languages and Programming*. 169–180.

SCHRIJVER, A. 2003. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer.

SINGH, M. 2008. Iterative methods in combinatorial optimization. Ph.D. thesis, Carnegie Mellon University.

VAZIRANI, V. V. 2004. *Approximation Algorithms*. Springer.

WIN, S. 1989. On a connection between the existence ofk-trees and the toughness of a graph. *Graphs and Combinatorics 5,* 1, 201–205.