RC24347 (W0709-035) September 13, 2007
Computer Science

IBM Research Report

Additive Guarantees for Degree Bounded Directed Network Design

Nikhil Bansal, Rohit Khandekar
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Viswanath Nagarajan
Tepper School of Business
Tech & Frew Street
Pittsburgh, PA

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Additive Guarantees for Degree Bounded Directed Network Design

Nikhil Bansal* Rohit Khandekar* Viswanath Nagarajan†

Abstract

We present polynomial-time algorithms for some *directed* network design problems with degree bounds on vertices. In particular,

1. *Degree bounded arborescence problem.* Given out-degree bounds b_v on vertices, our algorithm either computes an (out-)arborescence with out-degrees $b_v + 2$ or gives a certificate of infeasibility. This is the first result with *additive constant* violation in the *directed* setting. This improves a recent result of Lau et al. [11] that incurs a factor 2 multiplicative violation, and a result of Klein et al. [9] that gives a $(1+\epsilon)b_v + O(\log_{1+\epsilon} n)$ bound on the degree for any $\epsilon > 0$. Our result is almost best possible: assuming $P \neq NP$, one can not obtain a guarantee better than $b_v + 1$ (even in the case of undirected graphs).
2. *Minimum cost degree bounded arborescence problem.* When arcs have non-negative costs, for any $0 < \epsilon \leq \frac{1}{2}$, our algorithm computes an arborescence with out-degrees $\lceil \frac{b_v}{1-\epsilon} \rceil + 4$ and cost at most $\frac{1}{\epsilon}$ times that of the optimal degree-bounded arborescence.
3. *Lower bounds on the integrality gap.* Interestingly, we also prove that the above $(\frac{b_v}{1-\epsilon} + O(1), \frac{1}{\epsilon})$ tradeoff is near-optimal for the natural LP relaxation. For every $\epsilon > 0$, we show an instance where any arborescence with out-degrees at most $\frac{b_v}{1-\epsilon} + O(1)$ has cost at least $\frac{1-o(1)}{\epsilon}$ times the optimal LP value.
4. *Intersecting supermodular connectivity with degree bounds.* We extend the above results (both unweighted and cost versions) to intersecting supermodular connectivity requirements with in-degree and out-degree bounds. In the unweighted case, we obtain the first *additive constant* approximation for the problem. Our algorithm violates the degrees by $+4$. In the weighted case, for any $0 < \epsilon \leq 1/2$, our algorithm computes a solution with out-degrees and in-degrees at most $\lceil \frac{b_v}{1-\epsilon} \rceil + 4$, and cost at most $1/\epsilon$ times the optimum cost. This extends and improves the $(3b_v+6, 4)$-approximation of Lau et al. [11, 12]. This includes, as a special case, the problem of packing the maximum number of edge-disjoint arborescences.

Our algorithms use the iterative rounding technique of Jain [8], which was recently used by Lau et al. [11] and Singh and Lau [16] in the context of degree bounded network design. It is, however, non-trivial to extend these techniques to the directed setting without incurring a *multiplicative* violation in the degree bounds. This is due to a fact that known polyhedral characterization of arborescences has the set-boundary covering constraints which, along with degree bound constraints, are unsuitable for arguing the existence of integral variables in a basic feasible solution. We overcome this difficulty by enhancing the iterative rounding steps and by means of more careful counting arguments.

Our counting technique also gives a substantially easier proof of the recent $+1$ approximation of Singh and Lau [16] for computing the (undirected) bounded-degree minimum cost spanning tree. More generally, we give a $+(r-1)$ additive approximation for the minimum crossing spanning tree problem where each edge lies in at most r-sets which substantially improves a result of Bilò et al. [1].

*IBM T.J. Watson Research, P.O. Box 218, Yorktown Heights, NY. {nikhil,rohitk}@us.ibm.com.
†Tepper School of Business, Tech & Frew Street, Pittsburgh, PA. viswa@cmu.edu.

1 Introduction

The problem of finding a minimum spanning tree that satisfies given degree bounds on vertices has received much attention in the field of combinatorial optimization recently. This problem was first considered by Fürer and Raghavachari [6]. Their motivation was to find a broadcast tree in a communication network along which the maximum overload of any node, proportional to its degree, is minimized. Assuming unit edge-costs, they gave a local-search based polynomial-time algorithm for computing a spanning tree with maximum degree at most $\Delta^* + 1$ as long as there exists a spanning tree with maximum degree at most Δ^*. This is essentially the best possible since computing the optimum is NP-hard.

Earlier in this decade, variety of techniques were developed in attempts to generalize this result to the case of arbitrary edge-weights. Ravi et al. [15], using a matching-based augmentation technique, gave a bi-criteria approximation algorithm that violates both the cost and the degree bounds by a multiplicative logarithmic factor. Könemann and Ravi [10] used a Lagrangian relaxation based method to get $O(1)$ approximation on the cost while violating the degrees by a constant factor plus an additive logarithmic term. Chaudhuri et al. [2] based their algorithms on the augmenting-path and push-relabel frameworks from the maximum flow problem and obtained either logarithmic additive violation or constant multiplicative violation on degrees. In a recent break-through result, Goemans [7] presented an algorithm, based on matroid intersection techniques, that computes a spanning tree with cost at most that of the optimum and with degrees at most the bounds *plus* 2. This line of research recently culminated in the "best possible" *plus* 1 result of Singh and Lau [16]. Their algorithm used an iterative rounding approach of Jain [8] while obtaining a spanning tree with cost at most that of the optimum while violating the degrees by at most an additive $+1$ term.

1.1 Directed network design with degree bounds

In this paper, we consider *directed* network design problems with either in-degree or out-degree (or both) constraints on the vertices. Directed graphs naturally arise in communication networks. In fact our original motivation was a problem that arose at IBM in the context of maximizing throughput in peer to peer networks. Here, we are given a network where a root node r wishes to transmit packets to all the nodes in the network. However, each node has limited network resources which determines how many packets it can transmit per unit time. It turns out that computing the maximum achievable throughput of this network is exactly equal to determining the number of r-arborescences that can be packed in the network subject to out-degree bounds at each node.

As we shall discuss below, the directed setting turns out to be substantially harder than the undirected setting, and much fewer results are known in this case. Below we discuss the relevant related work for each of the problems that we consider in this paper.

1.2 Our results and previous work

Degree bounded arborescence problem (without costs). Let $G = (V, E)$ be a directed graph with root r, and let b_v be the bounds on out-degree for each vertex v (since every vertex except the root has indegree exactly one in any arborescence, we do not consider bounds on indegree here). This problem was first considered by Fürer and Raghavachari [6] who gave a polynomial time algorithm to compute an arborescence that violates the degree bound by at most logarithmic multiplicative factor. Klein et al. [9] gave a quasi-polynomial time algorithm with degree violation $(1 + \epsilon)b_v + O(\log_{1+\epsilon} n)$ for any $\epsilon > 0$. Their algorithm starts with a solution and successively applies local improvement steps to reduce high degrees. Recently, Lau et al. [11], using an iterative rounding technique, obtained a polynomial-time algorithm that computes an arborescence with degrees at most $2 \cdot b_v + 2$[1] (their algorithm additionally

[1]Strictly speaking, the conference version [11] only shows a bound of $4b_v + 6$ on the degree and a bound of 4 on cost. These improved results were communicated to us recently [12].

guarantees that the cost is violated by a factor of at most 2). We obtain the first result with only *additive* violation in the degree bounds. In fact, assuming $P \neq NP$, our result is almost best possible.

Theorem 1 *There is a polynomial time algorithm that constructs an (out-)arborescence rooted at r such that any vertex v has out-degree at most $b_v + 2$ or else gives a certificate showing that no arborescence exists satisfying the degree bounds exactly.*

Minimum cost degree bounded arborescence problem. Suppose there is a cost function $c : E \to \mathbb{R}^+$ on the arcs (or edges). Let us assume that there exists an arborescence T that satisfies all the out-degree bounds exactly. Consider the problem of computing minimum cost arborescence that satisfies all the out-degree bounds exactly. The algorithm of Lau et al. [11] computes an arborescence with degrees at most $2 \cdot b_v + 2$ for all v and also ensures that the cost of the solution is at most 2 times that of the optimum. We extend their result as follows.

Consider the following natural integer programming formulation of the minimum cost degree bounded arborescence problem.

$$\begin{array}{lll} \min & \sum_{e \in E} c_e \cdot x_e & \\ s.t. & & \\ & x(\delta^-(S)) \geq 1 & \forall S \subseteq V \setminus \{r\} \\ & x(\delta^+(v)) \leq b_v & \forall v \in W \\ & x_e \in \{0,1\} & \forall e \in E \end{array} \tag{1}$$

Let OPT be the optimum value of the above integer program and let FRAC be the value of its linear programming relaxation in which the constraints $x_e \in \{0,1\}$ are replaced with $0 \leq x_e \leq 1$ for $e \in E$. Clearly we have FRAC $\leq$ OPT. We prove the following theorem.

Theorem 2 *There is a polynomial-time algorithm that, given any $\epsilon \in (0, \frac{1}{2}]$, constructs an arborescence rooted at r with cost at most $\frac{1}{\epsilon} \cdot$ FRAC and such that any vertex v has out-degree less than $\lceil \frac{b_v}{1-\epsilon} \rceil + 4$.*

Surprisingly, it also turns out that the above $(\frac{1}{\epsilon}, \frac{b_v}{1-\epsilon} + O(1))$ tradeoff between the cost blowup and the degree-bound violation, is near-optimal for the above LP relaxation. The following theorem captures this more formally (the $O(1)$ terms below are independent of ϵ).

Theorem 3 *For any $\epsilon > 0$, there is an instance of the minimum cost degree bounded arborescence problem such that, any arborescence with out-degrees at most $\frac{b_v}{(1-\epsilon)} + O(1)$ for all vertices v has cost at least $(\frac{1-o(1)}{\epsilon}) \cdot$ FRAC.*

Note that our dependence on ϵ in Theorems 2 and 3 is identical, (i.e. we do not have any hidden constant factors using the big-O or Ω notation), and hence we obtain the optimum possible tradeoff between cost and degree violation (modulo some additive terms). Theorem 3 suggests that computing low-cost arborescences subject to degree bounds might be an inherently harder problem in the directed case as compared with the undirected case.

Intersecting supermodular connectivity requirements with degree bounds. We also consider the network design problem in directed graphs where the connectivity requirement is specified by an arbitrary *intersecting supermodular* function [5], and there are both in-degree and out-degree bounds $\{(a_v, b_v)\}_{v \in V}$ on vertices. The intersecting supermodular requirements (defined formally later) are general enough to include the problem of packing k-edge disjoint arborescences, and choosing the minimum cost edges to increase the rooted connectivity of a digraph [5, 14]. The goal here is to find a subgraph (if it exists) that satisfies the connectivity requirement and degree bounds on vertices. We show that:

Theorem 4 *Given a directed network design problem where the connectivity requirement is specified by an intersecting supermodular function, and upper bounds on the in-degree and out-degree of vertices, there is a polynomial time algorithm that either shows that the instance is infeasible or else finds a subgraph satisfying the connectivity requirement while violating any degree bound by at most an additive constant $+4$.*

The above theorem improves upon the result of Lau et al. [11] that obtains a feasible solution with in-degree at most $3 \cdot a_v + 6$ and out-degrees at most $3 \cdot b_v + 6$. Additionally their result also guarantees that the cost is at most 3 times the optimum[2]. Moreover, their result also holds in the more general case of crossing supermodular connectivity requirements.

Our algorithm for minimum cost degree bounded arborescence also generalizes to intersecting supermodular case (in fact it strictly improves their guarantee stated above), and in particular we obtain:

Theorem 5 *Given a directed network design problem with edge-costs, an intersecting supermodular connectivity requirement, and upper bounds on the in-degree and out-degree of vertices, for any $\epsilon \in (0, \frac{1}{2}]$, there is a polynomial time algorithm that either shows that the instance is infeasible or else finds a subgraph satisfying the connectivity requirement, having cost at most $\frac{1}{\epsilon}$ times the optimum, and with each vertex v having degree at most $\lceil \frac{b_v}{1-\epsilon} \rceil + 4$.*

Again the lower bound in Theorem 3 obviously applies to this case, and hence our dependence on ϵ is optimum, based on the natural linear program.

Finally, using our counting arguments, we also give a substantially simpler proof of the +1 guarantee for the bounded degree minimum cost (undirected) spanning tree problem due to Singh and Lau [16]. In fact, the proof readily extends to the more general minimum crossing spanning tree problem defined as follows.

Minimum crossing spanning tree problem (MCSP). Given an undirected graph $G = (V, E)$, costs $c_e \geq 0$ on the edges $e \in E$, subsets of edges $E_i \subseteq E$ for $1 \leq i \leq k$, and integers $b_i \geq 0$ for $1 \leq i \leq k$, the MCSP is to find a minimum cost spanning tree (if it exists) in G that contains at most b_i edges from set E_i for $1 \leq i \leq k$. We show the following result.

Theorem 6 *Consider an instance of the MCSP problem. Let r be an integer such that any edge belongs to at most r subsets E_i, e.g., $r = \max_{e \in E} |\{i \mid e \in E_i, 1 \leq i \leq k\}|$. There is a polynomial-time algorithm that either computes a spanning tree with cost at most that of the optimum and which contains at most $b_i + r - 1$ edges from set E_i for $1 \leq i \leq k$; or else gives a certificate that the instance is infeasible.*

This significantly improves on the results of Bilò et al. [1], who consider an unweighted instance in which all b_i are equal to b and find a spanning tree which contains at most $O(b \cdot r \log n)$ edges in the set E_i for $1 \leq i \leq k$.

Note that if E_i denotes the set of edges incident to vertex i and b_i denotes the degree bound on vertex i, we get the bounded degree minimum spanning tree problem. Our algorithm matches the $+1$ bounds recently obtained by Singh and Lau [16].

1.3 Our approach

Our algorithms are based on the iterative rounding technique of Jain [8] which was recently used by Lau et al. [11] and Singh and Lau [16] in the context of degree bounded network design problems.

The iterative rounding technique, which has been extensively used in network design problems, proceeds as follows. First the problem is formulated as an integer program; relaxing the integrality constraints one then obtains a linear program. An extreme point solution, a.k.a. basic feasible solution, to this linear program is then computed by standard linear programming algorithms. This extreme point solution is proved to exhibit useful structural properties, for example, an existence of an integral variable. Such variables are then fixed to their integral values and the residual problem is solved iteratively. For example, Singh and Lau [16] use a clever counting argument to show that in any extreme point solution to their LP formulation of degree bounded spanning tree problem, either there is an integral edge-variable, or the degree bound constraint of some vertex can be dropped without violating it by more than $+1$ in

[2]Personal communication, strictly speaking [11] showed $(4, 4b_v + 6, 4a_v + 6)$ guarantee

the subsequent steps. The algorithm then either sets such an edge to its integral value or drops such a constraint; thereby reducing the size of the linear program and repeats.

Challenges in extension to the directed case. Consider a basic feasible solution x to the LP relaxation of (1). A basic feasible solution is uniquely defined by a basis of $|E|$ linearly independent constraints which are satisfied as equalities. By a standard uncrossing argument [3, 8], one can assume that the sets S which contribute constraints $x(\delta^-(S)) = 1$ to the basis form a laminar family. This laminar family may have up to $2|V| - 1$ sets. Although degree bounds are also present, some additional arguments can be used to show (Lau et al. [11]) that either there exists $e \in E$ such that $x_e \geq \frac{1}{2}$ or there is a vertex v with small degree in the support. Based on this, the algorithm iteratively does one of the following: rounds an edge e to 1 or drop the degree constraint of vertex v. Since we round $\frac{1}{2}$-edges to 1, we may violate the degree bounds by a *multiplicative* factor of 2, as in [11].

To overcome this problem in the *undirected* setting, Singh and Lau [16] use the following formulation of the spanning tree polytope.

$$\begin{array}{rcll} x(E(V)) & = & |V| - 1 & \\ x(E(S)) & \leq & |S| - 1 & \forall\, S : 2 \leq |S| \leq |V| - 1 \\ x_e & \geq & 0 & \forall\, e \in E \end{array} \tag{2}$$

Here $E(S) = \{(u, v) \in E \mid u, v \in S\}$. The laminar family of sets corresponding to a basic feasible solution x of this polytope can be shown to have at most $|V| - 1$ sets. This is crucial for them to argue that there exists $e \in E$ such that $x_e = 1$.

Unfortunately, in the directed setting, the arborescence polytope is not known to have a formulation similar to (2) and one has to use the set-boundary covering constraints as in (1). However even in this case, using additional steps in iterative rounding and better counting arguments, we obtain improved guarantees based on the same LP (1). In fact the guarantees we obtain are (almost) best possible using the LP. We continue to use the idea of dropping degree constraints (Lau et al. [11]); so at any iteration the degree bounds are present only at a subset W of the vertices. Our degree-bound relaxation step involves considering vertices that have small 'spare' (i.e. difference of support degree and fractional degree). In addition, we also use some new relaxation steps that involve treating edges leaving W vertices and non-W vertices differently. Finally, as is the case with iterative rounding algorithms, we need a careful counting argument to show that an improvement is possible at every iteration. We note that the counting arguments we use are global in nature.

1.4 Preliminaries

A set function $f : 2^V \to \mathbb{R}$ on ground set V is called supermodular if for any two subsets $S, T \subseteq V$, it holds that $f(S \cup T) + f(S \cap T) \geq f(S) + f(T)$. Similarly set function f is submodular if $f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$. The set function f is said to be *intersecting supermodular* if for every $S, T \subseteq V$ such that $S \cap T \neq \phi$, we have $f(S \cup T) + f(S \cap T) \geq f(S) + f(T)$.

Given a directed graph $G = (V, E)$ and a subset S of vertices, we use $\delta^-_G(S)$ (resp. $\delta^+_G(S)$) to denote the set of edges entering (resp. leaving) S. When the graph G is clear from the context, we drop the subscript G. Consider any non-negative real-value assignment $x : E \to \mathbb{R}^+$ to the edges; we use $x(\delta^-(S))$ (resp. $x(\delta^+(S))$) to denote the total x-value of the edges coming into (resp. leaving) S. The functions $x(\delta^-)$ and $x(\delta^+)$ are always submodular.

Given a directed graph $G = (V, E)$ and a set function $f : 2^V \to \mathbb{Z}^+$, a subgraph $H = (V, E')$ of G is said to be *f-connected* if $\delta^-_H(S) \geq f(S)$ for every subset $S \subseteq V$. In the basic directed network design problem, we are given an edge-weighted graph and a set function f, and the goal is to compute the minimum cost f-connected subgraph. In the degree bounded variant of network design, there are additional constraints bounding the out-degree and in-degree at each vertex. The *degree bounded directed network design* problem is the following: given an edge-weighted graph $G = (V, E)$, a set function $f : 2^V \to \mathbb{Z}^+$ and degree bounds $\{a_v, b_b\}_{v \in V}$, compute a minimum cost f-connected subgraph where each vertex v

has in-degree at most a_v and out-degree at most b_v. In this paper, we only consider problems where the set function f is intersecting supermodular.

A family of sets $\{S_1, \ldots, S_k\}$ is called *laminar* if for every two sets they are either disjoint or one is contained in the other; i.e. for every $1 \leq i, j \leq k$, either $S_i \cap S_j = \emptyset$ or $S_i \subset S_j$ or $S_j \subset S_i$.

1.5 Organization

The rest of the paper is organized as follows. In Section 2, we consider the minimum degree arborescence problem and prove Theorem 1. This result contains many of the ideas used in the rest of the paper as well. In Section 3, we generalize the arborescence result to any arbitrary intersecting supermodular connectivity requirement (Theorem 4). Then in Section 4, we consider the bounded degree arborescence problem in the presence of edge-costs, and prove Theorem 2. We also complement this approximation guarantee by showing a tight integrality gap of the natural LP relaxation for this problem (Theorem 3). In Section 5, we extend our algorithm to handle the cost version of any intersecting supermodular connectivity requirement (Theorem 5). Note that the arborescence result (Theorem 2) is implied by the intersecting supermodular result (Theorem 5). However we present them separately since the proof of Theorem 2 is simpler and contains essentially all the ideas need for the general case. Finally, in Section 6 we consider the undirected minimum crossing spanning tree problem.

2 The degree bounded arborescence problem

Given a directed graph $G = (V, E)$ with a root $r \in V$ and bounds $b_v \geq 0$ on the out-degrees of the vertices $v \in V$, the degree bounded arborescence problem is to find an (out-)arborescence T (if it exists) rooted at r such that the out-degree in T of any vertex v is at most b_v. In this section, we prove Theorem 1. This unweighted case is relatively simple and will introduce many of the ideas that we use later. Consider the following linear programming relaxation of the problem. Find x_e for each $e \in E$ such that

$$\begin{array}{rcll} x(\delta^-(S)) & \geq & 1 & \forall S \subseteq V \setminus \{r\} \quad \text{(cut constraints)} \\ x(\delta^+(v)) & \leq & b_v & \forall v \in V \quad \text{(degree constraints)} \\ x_e & \leq & 1 & \forall e \in E \\ x_e & \geq & 0 & \forall e \in E \end{array}$$

Our algorithm, given in Figure 1, proceeds in several iterations. In a general iteration of the algorithm, we denote E to be the candidate set of edges, initially containing all the edges. The set $F \subseteq E$ denotes the edges that we have already picked in our solution and the out-degree bounds constraints are present only for a subset $W \subseteq V$ of vertices. In such an iteration, we work with the following linear program with variables x_e for $e \in E \setminus F$. Let $E' = E \setminus F$. For brevity, we use δ^- (resp. δ^+) to denote $\delta^-_{E'}$ (resp. $\delta^+_{E'}$).

$$P(E, F, W): \quad \begin{array}{rcll} x(\delta^-(S)) & \geq & 1 - |\delta^-_F(S)| & \forall S \subseteq V \setminus \{r\} \quad \text{(cut constraints)} \\ x(\delta^+(v)) & \leq & b_v - |\delta^+_F(v)| & \forall v \in W \quad \text{(degree constraints)} \\ x_e & \leq & 1 & \forall e \in E' = E \setminus F \\ x_e & \geq & 0 & \forall e \in E' = E \setminus F \end{array}$$

In the beginning of the iteration, we first compute a *basic feasible solution* x in the polytope $P(E, F, W)$ using standard linear programming techniques. We then update the sets E, F, and W as explained in Figure 1. The algorithm, in the end, outputs any arborescence contained in the set of edges F.

The following lemma is simple to note.

Lemma 1 *Assume that $P(E, F, W)$ is feasible at the beginning of the algorithm. If the algorithm terminates, it outputs an arborescence T that $|T(\delta^+(v))| \leq b_v + 2$ for all $v \in V$.*

- Set $F \leftarrow \emptyset$ and $W \leftarrow V$.
- If $P(E, F, W)$ is infeasible, output "infeasible".
- Repeat while $E \setminus F \neq \emptyset$
  1. Compute a basic feasible solution x to $P(E, F, W)$.
  2. Remove from E all edges $e \in E \setminus F$ with $x_e = 0$.
  3. Add to F all edges $e \in E \setminus F$ with $x_e = 1$.
  4. For all $v \in W$ such that there are at most $b_v - |\delta_F^+(v)| + 2$ edges leaving v in $E \setminus F$,
     (a) Remove v from W.
     (b) Add to F all the out-going edges from v in $E \setminus F$.
- Output any (out-)arborescence rooted at r in F.

Figure 1: Algorithm for the degree bounded arborescence problem

Proof: Since we remove edges e from E only if $x_e = 0$ and add edges e to F only if $x_e = 1$, a feasible solution x in an iteration induces a feasible fractional solution for the linear program in the next iteration. Therefore, by induction, if $P(E, F, W)$ is feasible in the beginning, it continues to be feasible throughout the algorithm. Since $E \setminus F = \emptyset$ at the end, we conclude that the set F at the end of the algorithm indeed contains an (out-)arborescence rooted at r.

Now it is enough to argue that the set F, at the end, satisfies $|\delta_F^+(v)| \leq b_v + 2$ for all v. Note that we remove a vertex v from W (thereby effectively removing its degree-bound constraint from $P(E, F, W)$) only if the total number of edges in $E \setminus F$ leaving v is at most $b_v - |\delta_F^+(v)| + 2$. Thus the total number of edges in E leaving v is at most $b_v + 2$. Thus even if all edges leaving v get picked in F in the remainder of the algorithm, we have $|\delta_F^+(v)| \leq b_v + 2$, at the end, as desired. ■

The rest of the section is devoted to proving that the algorithm indeed terminates. We show this by proving that in each iteration either $|E|$ decreases, $|F|$ increases, or $|W|$ decreases. Since none of these three can happen an unbounded number of times, the algorithm must terminate. It in fact terminates in $m + n$ iterations where m and n are the number of edges and vertices in the given graph.

We now argue that if $|E|$ and $|F|$ do not change in Steps 2 and 3, $|W|$ must decrease in this iteration. Assume that the conditions in Steps 2 and 3 do not hold, i.e., all $e \in E'$ satisfy that $0 < x_e < 1$. In such a case, all the tight constraints in the basic feasible solution x come from the cut-conditions and the degree constraints. Moreover, since all edges leaving v are added to F as soon as v is removed from W, every edge in $E \setminus F$ must be out-going from a vertex in W. The following lemma is standard and obtained by using the fact that the RHS of the cut-constraints is a supermodular set function.

Lemma 2 ([11]) *For any basic solution x to $P(E, F, W)$ such that $0 < x_e < 1$ for all $e \in E'$, there exists a set $T \subseteq W$ and a laminar family $\mathcal{L}$ of subsets of V such that x is the unique solution to the linear system:*

$$\begin{array}{ll} x(\delta^-(S)) = 1 & \forall S \in \mathcal{L} \\ x(\delta^+(v)) = b_v - |\delta_F^+(v)| & \forall v \in T \end{array}$$

Furthermore, the following two conditions are satisfied

1. *The ($|E'|$-dimensional) characteristic vectors $\{\chi_{\delta^-(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{\delta^+(v)} \mid v \in T\}$ are linearly independent.*
2. *The size of the support is equal to $|E'| = |T| + |\mathcal{L}|$.*

For $v \in W$, we define the *spare* of a vertex $\mathrm{Sp}(v)$ as

$$\mathrm{Sp}(v) = \sum_{e \in \delta^+(v)} (1 - x_e) = |\delta^+(v)| - \sum_{e \in \delta^+(v)} x_e.$$

For $v \in W$, let $d_v = b_v - |\delta_F^+(v)|$ be the current degree bounds on v. Clearly we have $\mathrm{Sp}(v) \geq |\delta^+(v)| - d_v$. Note that $\mathrm{Sp}(v)$ is an upper bound on the degree violation of vertex v if its degree bound is dropped.

To complete the proof of Theorem 1, we prove the following lemma that states that if neither Step 2 or Step 3 in the algorithm is applicable, then there exists a vertex $v \in W$ that satisfies the conditions in Step 4.

Claim 1 *If neither tep 2 or Step 3 is applicable, then there exists $v \in W$ such that $|\delta^+(v)| - d_v \leq 2$.*

Proof: We first argue that it is enough to show that

$$|\mathcal{L}| < \sum_{e \in E'} x_e + 2|W|. \tag{3}$$

Consider the quantity $\sum_{v \in W} \mathrm{Sp}(v)$. As each (u, v) in E' has its tail u in W, it follows that $\sum_{v \in W} \mathrm{Sp}(v) = |E'| - \sum_{e \in E'} x_e$. Since $\mathrm{Sp}(v) \geq \delta^+(v) - d_v$ this implies that

$$\begin{aligned}
\sum_{v \in W} (\delta^+(v) - d_v) &\leq |E'| - \sum_{e \in E'} x_e \\
&= |\mathcal{L}| + |T| - \sum_{e \in E'} x_e \quad \text{(by Lemma 2)} \\
&\leq |\mathcal{L}| + |W| - \sum_{e \in E'} x_e \\
&< 3|W| \qquad\qquad \text{(by (3))}
\end{aligned}$$

This in turn implies that there exists $v \in W$ such that $|\delta^+(v)| - d_v < 3$. Since $|\delta^+(v)| - d_v$ is an integer, it must be at most 2.

The proof of (3) is based on a counting argument, as is common in iterative rounding. We assign x_e units of "tokens" to each $e \in E'$ and two "tokens" to each $v \in W$. We will show that all but one of these tokens can be "assigned" to sets $S \in \mathcal{L}$ such that each token gets assigned to at most one set in $\mathcal{L}$ and each set in $\mathcal{L}$ gets at least one token, thereby proving that $|\mathcal{L}|$ is strictly smaller than the total number of tokens $\sum_{e \in E'} x_e + 2|W|$.

The laminar family $\mathcal{L}$ naturally defines a forest $\mathcal{T}$ with $S \in \mathcal{L}$ as nodes. We call a node $S \in \mathcal{L}$ *marked* if there is some vertex $w \in W \cap S$; otherwise we call S *unmarked*. Recall that every edge in E' leaves a W-vertex; hence if S is an unmarked node, no edge of E' leaves a vertex in S and in particular, no edge of E' is contained in S. From Lemma 2, for any set $S \in \mathcal{L}$, $x(\delta^-(S)) = 1$. The assignment of tokens to nodes of $\mathcal{T}$ is done as follows.

1. Leaf nodes in $\mathcal{T}$. Let $S \in \mathcal{L}$ be a leaf in $\mathcal{T}$. Recall that $x(\delta^-(S)) = 1$. The tokens of edges $e \in \delta^-(S)$, which sum up to 1, are assigned to S. Clearly in this case no token is assigned to more than one set.

2. Unmarked non-leaf nodes in $\mathcal{T}$. We in fact show that unmarked non-leaves do not exist in $\mathcal{T}$ at all. Let on the contrary, $S \in \mathcal{L}$ be such a node, and $C_1, \cdots, C_t \subset S$ with $t \geq 1$ be its children in $\mathcal{T}$. Since S is unmarked, there is no edge of E' inside S, which implies that $\delta^-(C_i) \subseteq \delta^-(S)$ for all i. Recall that $x(\delta^-(S)) = x(\delta^-(C_i)) = 1$ for all i. This implies that $t = 1$ and $\chi_{\delta^-(S)} = \chi_{\delta^-(C_1)}$. This contradicts that they are linearly independent as given in Lemma 2.

3. Marked nodes in $\mathcal{T}$. Let $\mathcal{M} \subseteq \mathcal{T}$ denote the sub-forest consisting of only marked nodes in $\mathcal{T}$. Call a node $S \in \mathcal{M}$ *high-degree* if S has at least 2 children in $\mathcal{M}$; *low-degree* if S has exactly 1 child in $\mathcal{M}$; all other nodes are leaves in $\mathcal{M}$.

   Since leaves in $\mathcal{M}$ correspond to disjoint sets, every such node contains at least one *distinct* vertex of W. We next argue that each low-degree node in $\mathcal{M}$ also contains a *distinct* vertex of W, distinct also from the vertices of W contained in the leaves of $\mathcal{M}$. Let $S \in \mathcal{M}$ be a low-degree node in $\mathcal{M}$, and $C \in \mathcal{M}$ be its unique child in $\mathcal{M}$. To establish the above property, it is enough to show that $W \cap (S \setminus C) \neq \emptyset$. Assume on the contrary that this is not the case. As $S \setminus C$ does not contain any vertex of W, there are no edges from $S \setminus C$ to C; so $\delta^-(C) \subseteq \delta^-(S)$. Recalling that $x(\delta^-(C)) = x(\delta^-(S)) = 1$, we get $\chi_{\delta^-(S)} = \chi_{E'(\delta^-(C))}$ contradicting the linear independence condition in Lemma 2.

   Thus we proved that the total number of leaves and low-degree vertices in $\mathcal{M}$ is at most $|W|$.

   Now observe that the number of high-degree nodes in $\mathcal{M}$ is *strictly less* than the number of leaves in $\mathcal{M}$. Therefore the total number of nodes in $\mathcal{M}$ is strictly less than twice the number of leaves and low-degree nodes in $\mathcal{M}$, and hence strictly less than $2|W|$. Assign each node in $\mathcal{M}$ a distinct token out of $2|W|$ tokens from vertices in W leaving at least one token unassigned.

By the token assignment given above, each set in $\mathcal{L}$ gets at least one token with one token unassigned. Thus the proof is complete.

■

3 General connectivity requirements with degree bounds

We now consider the network design problem in directed graphs where the connectivity requirement is specified by an arbitrary *intersecting supermodular* function [5], and there are upper bounds on in-degree and out-degree $\{(a_v, b_v)\}_{v \in V}$ on the vertices. The goal here is to find a subgraph (if it exists) that satisfies the connectivity requirement and degree bounds on vertices. In this section, we prove Theorem 4.

We work with the following linear relaxation $P(E, F, A, B)$ in each iteration. In a generic iteration, E denotes the current set of edges in the graph, $F \subseteq E$ the set of edges that have been fixed to value 1, $A \subseteq V$ the vertices for which there is an in-degree bound, and $B \subseteq V$ the vertices for which there is an out-degree bound. Again we abbreviate $E' = E \setminus F$, $\delta^+(\cdot) = \delta^+_{E'}(\cdot)$ and $\delta^-(\cdot) = \delta^-_{E'}(\cdot)$.

$$P(E,F,A,B): \quad \begin{array}{ll} x(\delta^-(S)) \geq f(S) - |\delta^-_F(S)| & \forall S \subseteq V \\ x(\delta^+(v)) \leq b_v - |\delta^+_F(v)| & \forall v \in B \\ x(\delta^-(v)) \leq a_v - |\delta^-_F(v)| & \forall v \in A \\ x_e \leq 1 & \forall e \in E \setminus F \\ x_e \geq 0 & \forall e \in E \setminus F \end{array}$$

In any iteration, let x denote an optimal basic feasible solution. The algorithm is similar to the one for minimum degree arborescence, and performs one of the following steps in each iteration where $E' \neq \phi$:

1. If there is an edge $e \in E'$ with $x_e = 0$, set $E \leftarrow E \setminus \{e\}$.
2. If there is an edge $e \in E'$ with $x_e = 1$, set $F \leftarrow F \cup \{e\}$.
3. If there is an edge $(u, v) \in E'$ with $u \notin B$ *and* $v \notin A$, set $F \leftarrow F \cup \{(u, v)\}$.
4. If there is a vertex $v \in B$ with at most $b_v - |\delta^+_F(v)| + 4$ support edges leaving it, set $B \leftarrow B - \{v\}$.
5. If there is a vertex $v \in A$ with at most $a_v - |\delta^-_F(v)| + 4$ support edges entering it, set $A \leftarrow A - \{v\}$.

If at least one of these conditions holds at each iteration, the algorithm results in a solution satisfying the connectivity requirement while violating each degree bound by at most an additive 4. The rest of this section proves that one of the above conditions is always true. In particular, we show that if none of the

conditions (1)-(3) are satisfied, then at least one of (4) and (5) must be true. We assume henceforth that none of (1)-(3) are satisfied. As in the previous section, since conditions (1) and (2) do not hold: all the tight constraints in a basic feasible solution x come from the cut-conditions and the degree constraints. Based on standard uncrossing arguments, we have the following.

Lemma 3 ([11]) *For any basic solution x to (DLP) with no variable fixed to 0 or 1, there exist sets $T' \subseteq A$, $T'' \subseteq B$, and a laminar family $\mathcal{L}$ of subsets of V such that x is the unique solution to the linear system:*

$$\begin{array}{ll} x(\delta^-(v)) = a_v - |\delta_F^-(v)| & \forall v \in T' \\ x(\delta^+(v)) = b_v - |\delta_F^+(v)| & \forall v \in T'' \\ x(\delta^-(S)) = f(S) - |\delta_F^-(S)| & \forall S \in \mathcal{L} \end{array}$$

Furthermore, the following conditions hold:

1. *For every $S \in \mathcal{L}$, $f(S) - |\delta_F^-(S)| \geq 1$ and integral.*
2. *The characteristic vectors $\{\chi_{\delta^-(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{\delta^-(v)} \mid v \in T'\} \cup \{\chi_{\delta^+(v)} \mid v \in T''\}$ are linearly independent; and*
3. *The size of the support $|E'| = |T'| + |T''| + |\mathcal{L}|$.*

Let $W = A \cup B$. Note that each edge in the support either leaves a B-vertex or enters an A-vertex. We now classify the various types of edges in the support E':

1. Let E_1 denote the set of edges (u, v) such that $u \in B$ and $v \notin W$. Note that we only require that u lie in B, and in particular, (u, v) will lie in E_1 even if u lies in both A and B.
2. Let E_2 denote that set of edges (u, v) such that $v \in A$ and $u \notin W$ (again v is allowed to lie in both A and B).
3. Let E_3 denote the set of edges (u, v) such that $u \in B$ and $v \in A$.
4. Let E_4 denote the set of edges (u, v) such that $u \in B$ and $v \in \overline{A} \cap B$.
5. Let E_5 denote the set of edges (u, v) such that $u \in A \cap \overline{B}$ and $v \in A$.

Observe that the sets $E_1, \ldots, E_5$ are pairwise disjoint and that each edge in the support must lie in one of the sets above, and hence $E_1, \ldots, E_5$ form a disjoint partition of E'. For an edge e, let $\mathrm{Sp}(e) = 1 - x_e$. Define

$$\mathrm{Sp}(B) = \sum_{(u,v):u \in B} \mathrm{Sp}((u, v)) = \sum_{(u,v):u \in B} (1 - x_{u,v}) = \sum_{e \in E_1 \cup E_3 \cup E_4} (1 - x_e)$$

$$\mathrm{Sp}(A) = \sum_{(u,v):v \in A} \mathrm{Sp}((u, v)) = \sum_{(u,v):v \in A} (1 - x_{u,v}) = \sum_{e \in E_2 \cup E_3 \cup E_5} (1 - x_e).$$

For a set H of edges, define

$$\mathrm{Sp}(H) = \sum_{e \in H} (1 - x_e) \quad \text{and} \quad \mathrm{Val}(H) = \sum_{e \in H} x_e.$$

Observe that,

$$\mathrm{Sp}(A) + \mathrm{Sp}(B) = \mathrm{Sp}(E_1) + \mathrm{Sp}(E_2) + 2\mathrm{Sp}(E_3) + \mathrm{Sp}(E_4) + \mathrm{Sp}(E_5) \tag{4}$$

The following is the main part of our argument (below we set $W = A \cup B$).

Claim 2

$$2|\mathcal{L}| < |E_1| + |E_2| + \mathrm{Val}(E) + 3|W|. \tag{5}$$

We first show how Claim 2 implies Theorem 4. Since $|E| = |\mathcal{L}| + |T'| + |T''|$, Claim 2 implies that

$$\begin{aligned} 2|E| &< |E_1| + |E_2| + \mathrm{Val}(E) + 3|W| + 2|T'| + 2|T''| \\ &\leq |E_1| + |E_2| + \mathrm{Val}(E) + 3|W| + 2|A| + 2|B| \\ &\leq |E_1| + |E_2| + \mathrm{Val}(E) + 5|A| + 5|B| \end{aligned}$$

The second step follows as $|T'| \leq |A|$ and $|T''| \leq |B|$ and the third step follows as $|W| \leq |A| + |B|$. As $|E| = |E_1| + |E_2| + |E_3| + |E_4| + |E_5|$, and as $\mathrm{Sp}(X) = |X| - \mathrm{Val}(X)$ for any $X \subseteq E'$, the inequality above implies that

$$\mathrm{Sp}(E) + |E_3| + |E_4| + |E_5| < 5|A| + 5|B|.$$

As $\mathrm{Sp}(X) \leq |X|$ for any subset of edges X, this implies that

$$\mathrm{Sp}(E_1) + \mathrm{Sp}(E_2) + 2\mathrm{Sp}(E_3) + 2\mathrm{Sp}(E_4) + 2\mathrm{Sp}(E_5) < 5|A| + 5|B|$$

which together with inequality (4) and the fact that $sp(X) \geq 0$ for any $X \subseteq E'$, implies $\mathrm{Sp}(A) + \mathrm{Sp}(B) < 5(|A| + |B|)$. This must imply that either $\mathrm{Sp}(A) < 5|A|$ or $\mathrm{Sp}(B) < 5|B|$. We now argue that at least one of conditions (4) or (5) must hold. Suppose that $\mathrm{Sp}(A) < 5|A|$ (the other case is identical). Rewrite $\mathrm{Sp}(A) = \sum_{v \in A}(|\delta^-(v)| - x(\delta^-(v))) \geq \sum_{v \in A}[|\delta^-(v)| - (a_v - |\delta^-_F(v)|)]$. So there is some $v \in A$ with $|\delta^-(v)| - (a_v - |\delta^-_F(v)|) < 5$, but since the LHS is integral, we have $|\delta^-(v)| \leq (a_v - |\delta^-_F(v)|) + 4$ which implies that condition (5) holds.

3.1 Token assignment: Proof of Claim 2

Consider the following token assignment scheme: We assign $1 + x_e$ tokens to each edge $e \in E_1 \cup E_2$. For an edge $(u, v) \in E_1$, the $1 + x_e$ tokens lie at the head v. For an edge $(u, v) \in E_2$, 1 unit of token lies at u, and x_e units lies at the head v. Each remaining edge $e = (u, v) \in E_3 \cup E_4 \cup E_5$ has x_e tokens; these x_e units of token are present at v. We also assign 3 tokens to each vertex in W. We will show that these tokens can be redistributed to obtain at least 2 tokens for each node $S \in \mathcal{L}$, with at least one token to spare which will imply Claim 2.

We call a node $S \in \mathcal{L}$ *marked* if $W \cap S \neq \emptyset$; otherwise S is called *unmarked*. Note that if S is an unmarked node, there is *no* edge of E' contained in S. Also, for any tight set S, $x(\delta^-(S)) \geq 1$ and is an integer. The assignment of tokens to nodes of $\mathcal{L}$ proceeds using the following steps.

1. Assignment to unmarked leaf nodes. Let $S \in \mathcal{L}$ be such a node. Since S is a tight set, we have that $x(\delta^-(S)) \geq 1$. Hence there are at least two edges of E' entering S (as each edge has $x_e < 1$). Assign the tokens at the heads of these edge to S: note that since S is unmarked, these edges must be of type E_1, and S receives at least $2 + x(\delta^-(S)) \geq 3$ tokens.
2. Assignment to unmarked non-leaf nodes. Let $S \in \mathcal{L}$ be such a node, and $C_1, \cdots, C_t \subset S$ its children. Since S is unmarked, there is no edge of E' inside S and hence $\cup_{i=1}^t \delta^-(C_i) \subseteq \delta^-(S)$. But the incidence vectors $\chi_{\delta^-(S)}, \chi_{\delta^-(C_1)}, \cdots, \chi_{\delta^-(C_t)}$ are linearly independent, so it must be that $\delta^-(S) \setminus (\cup_{i=1}^t \delta^-(C_i)) \neq \emptyset$. This implies that $x(\delta^-(S)) - \sum_{i=1}^t x(\delta^-(C_i)) > 0$. Moreover, as the quantities $x(\delta^-(S)), x(\delta^-(C_1)), \cdots, x(\delta^-(C_t))$ are all integers (because $S, C_1, \ldots, C_t$ are all tight sets), it must be that $x(\delta^-(S)) - \sum_{i=1}^t x(\delta^-(C_i)) \geq 1$. Thus $|\delta^-(S) \setminus \cup_{i=1}^t \delta^-(C_i)| \geq 2$ (as all edges have $x_e < 1$) and we assign the tokens of these edges to S. Again, as S is unmarked, these edges must be of type E_1, and S gets at least 3 tokens.
3. Assignment to marked nodes. Let $\mathcal{M} \subseteq \mathcal{L}$ denote the laminar family consisting of only marked nodes. Call a node $S \in \mathcal{M}$ *high-degree* if it has at least 2 children in $\mathcal{M}$; and *low-degree* if it has exactly 1 child in $\mathcal{M}$; all other nodes in $\mathcal{M}$ are leaf-nodes (no children). We now show how to assign tokens to each of these nodes.
   (a) High-degree nodes: Note that the number of high-degree nodes in $\mathcal{M}$ is strictly less than the number of leaf-nodes in $\mathcal{M}$. Arbitrarily assign each high-degree node in $\mathcal{M}$ two tokens from a distinct W-vertex (in a distinct leaf node of $\mathcal{M}$).

(b) Marked leaf-nodes: For each leaf node S in $\mathcal{M}$ we assign 1 token from some W-vertex contained in it. For the remaining token, we argue as follows: If S is also a leaf in $\mathcal{L}$, then S has $x(\delta^-(S))) \geq 1$ and hence S receives at least 1 unit of tokens from edges in $\delta^-(S)$ (since every edge carries at least x_e tokens at its head). If S is not a leaf in $\mathcal{L}$, then consider the subtree rooted S. This subtree has at least one unmarked node. Since each unmarked node has at least 3 tokens thus far, S borrows one token arbitrarily from one of these nodes. Note that an unmarked node cannot be used twice to borrow a token.
Note that each W-vertex has been charged at most 3 tokens so far.

(c) Low-degree marked nodes: Let $S \in \mathcal{M}$ be such a node, and $C \in \mathcal{M}$ be its unique child. Suppose that $W \cap (S \setminus C) \neq \emptyset$, and $w \in W \cap (S \setminus C)$ be such a vertex. As no node of $\mathcal{M}$ is contained in $S \setminus C$, S is the smallest set in $\mathcal{M}$ that contains w. Assign node S two tokens from vertex w. Note that this vertex w cannot be charged by more than one such set S in this step. Moreover, w could not have been used in the earlier charging to W-vertices since it is not contained in any leaf node of $\mathcal{M}$.
Henceforth we assume that $W \cap (S \setminus C) = \emptyset$. Let r denote the total number of unmarked nodes of $\mathcal{L}$ contained in $S \setminus C$. We further consider the following cases:

i. $r = 0$. We claim that there are at least two edges with end-points in $S \setminus C$. If there are none, then we would have $\delta^-(C) = \delta^-(S)$, contradicting the linear independence condition. If there is exactly one such edge, we would have $0 < |x(\delta^-(S)) - x(\delta^-(C))| < 1$ which contradicts the integrality of tight cuts. Since these edges have one endpoint in $S \setminus C$ which has no W-vertex, they can only be E_1 or E_2 edges. In either case, each of these edges contributes at least 1 token to $S \setminus C$, and hence S receives at least 2 tokens.

ii. $r \geq 2$. Consider the unmarked nodes in $\mathcal{L}$ contained in $S \setminus C$. Note that each of them has been assigned at least 3 tokens thus far (they could not have given a token to marked node in step 3b). S is assigned 2 tokens by borrowing 1 token each from some two unmarked nodes in $S \setminus C$. Again, it is easily seen that each unmarked node U contributes tokens to at most one such node S, so every unmarked node is left with at least 2 tokens after this step.

iii. $r = 1$. Let $D \in \mathcal{L}$ be this unmarked child of S in $\mathcal{L}$. Clearly, D is a leaf in the laminar family $\mathcal{L}$ (otherwise we would have $r \geq 2$). Moreover, D has at least 3 tokens from edges in $\delta^-(D)$, none of which have been used in step (3(c)ii) or (3b). We now show how to assign 2 tokens to S.

A. Suppose there is an edge with an end-point in $S \setminus (C \cup D)$: then S can be assigned 1 token from edge e and 1 token from D (then D would still have 2 tokens).

B. If there is at least one edge e from D to C, then this edge must be of type E_2 and it contributes 1 token to D in addition to the 3 tokens that D already has. These 4 tokens can be shared by S and D.

C. If the above two cases do not apply, then it must hold that $\chi_{\delta^-(S)} = \chi_{\delta^-(C)} + \chi_{E'(V \setminus S, D)}$ and $\chi_{\delta^-(C)} + \chi_{\delta^-(D)} = \chi_{\delta^-(S)} + \chi_{E'(C,D)}$. Due to the linear independence, $\chi_{\delta^- C} \neq \chi_{\delta^-(S)}$ and $E'(C, D) \neq \emptyset$. Further, due to integrality of the tight cuts, $|E'(V \setminus S, D)| \geq 2$ and $|E'(C, D)| \geq 2$. Thus D has at least 4 incoming edges (of type E_1) that contribute at least 6 tokens, which can be shared by S and D together.

Thus the proof of Claim 2 is complete.

4 Minimum-cost bounded-degree arborescence problem

We now consider the minimum cost arborescence problem with out-degree constraints on vertices. We present an algorithm that attains a constant factor approximation in cost while violating the degrees by a constant factor that is arbitrarily close to 1 (Theorem 2). This is based on the iterative rounding of

a natural LP relaxation. In fact, our result is the best possible (up to constant factors) using this LP (Theorem 3). In a generic iteration of rounding, we would have set some edges $F \subseteq E$ to value 1, and the out-degree constraints are only present on a subset $W \subseteq V$ of vertices. At such an iteration, we work with the following linear relaxation. We let $E' = E \setminus F$ and δ^- (resp. δ^+) denote $\delta^-_{E'}$ (resp. $\delta^-_{E'}$).

$$
\begin{array}{rll}
\min & \sum_{e \in E} c_e \cdot x_e & \\
s.t. & & \\
P(E,F,W): & x(\delta^-(S)) \geq 1 - |\delta^-_F(S)| & \forall S \subseteq V \setminus \{r\} \\
 & x(\delta^+(v)) \leq b_v - (1-\epsilon)|\delta^+_F(v)| & \forall v \in W \\
 & x_e \leq 1 & \forall e \in E \setminus F \\
 & x_e \geq 0 & \forall e \in E \setminus F
\end{array}
$$

In some iteration, let x be an optimal basic feasible solution. The algorithm works for any $0 < \epsilon \leq \frac{1}{2}$, and performs one of the following steps in each iteration where $E' \neq \emptyset$:

1. If there is an edge $e \in E \setminus F$ with $x_e = 0$, set $E = E \setminus \{e\}$.
2. If there is an edge $e \in E'$ with $x_e \geq 1 - \epsilon$, set $F \leftarrow F \cup \{e\}$.
3. If there is an edge $(u, v) \in E'$ with $u \in V \setminus W$ and $x_{u,v} \geq \epsilon$, set $F \leftarrow F \cup \{(u, v)\}$.
4. If there is a vertex $v \in W$ with strictly less than $b_v - (1-\epsilon)|\delta^+_F(v)| + 5$ edges leaving it, set $W \leftarrow W - \{v\}$.

We will show that one of these conditions holds at each iteration. In particular, if none of the conditions (1)-(3) above hold, then condition (4) holds. We first see how this implies the result. Whenever an edge e is included in F, its x_e value is at least ϵ (note that $1 - \epsilon \geq \epsilon$), thus the total cost is at most $1/\epsilon$ times the fractional cost. Now consider the out-degrees. Let $d_v = b_v - (1-\epsilon)|\delta^+_F(v)|$ be the out-degree bound of v just before v is removed from W. In the worst case, all edges incident to it in the support might be chosen, and hence its out-degree might be $|\delta^+ F(v)| + d_v + 5 - \delta$, where $0 < \delta \leq 1$ is such that $d_v + 5 - \delta$ is an integer. Here δ is strictly greater than 0 because of the strict inequality in condition 4. This is equal to $(b_v - d_v)/(1-\epsilon) + d_v + 5 - \delta$ which is at most $b_v/(1-\epsilon) + 5 - \delta$, and as the out-degree is an integer, this is at most $\lceil b_v/(1-\epsilon) \rceil + 4$.

If conditions (1) and (2) do not hold, all the tight constraints in a basic feasible solution x come from the cut-conditions and the degree constraints. The following lemma is standard and implied by the fact that the RHS of the cut-constraints is an intersecting supermodular set function.

Lemma 4 ([11]) *For any basic solution x to (DLP) with no variables set to 0 or 1, there exists a set $T \subseteq W$ and a laminar family $\mathcal{L}$ of subsets of V such that x is the unique solution to the linear system:*

$$
\begin{array}{ll}
x(\delta^+(v)) = b_v - (1-\epsilon)|\delta^+_F(v)| & \forall v \in T \\
x(\delta^-(S)) = 1 & \forall S \in \mathcal{L}
\end{array}
$$

Furthermore, (1) the characteristic vectors $\{\chi_{\delta^-(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{\delta^+(v)} \mid v \in T\}$ are linearly independent; and (2) the size of the support $|E'| = |T| + |\mathcal{L}|$.

Below we assume that none of the conditions (1)-(3) of our iterative rounding holds. Let again $\mathrm{Val}(H) = \sum_{e \in H} x_e$ for a subset H of edges. We call a vertex in W a W-vertex. We call an edge a W-edge if it leaves a W-vertex, and call it an *ordinary* edge otherwise. Let E_W be the set of W-edges and E_O be the set of ordinary edges. Clearly, $|E'| = |E_W| + |E_O|$. The key component of our proof is the following.

Claim 3

$$2|\mathcal{L}| < |E_W| + \mathrm{Val}(E_W) + 2|E_O| + 3|W|.$$

Before proving this claim, we first argue that it implies Theorem 2. Since $\mathrm{Sp}(E_W) = |E_W| - \mathrm{Val}(E_W)$ and $|E'| = |\mathcal{L}| + |T| \leq |\mathcal{L}| + |W|$, the above claim implies that

$$2|E'| - 2|W| < |E_W| + \mathrm{Val}(E_W) + 2|E_O| + 3|W|.$$

Or equivalently,

$$|E_w| - \mathrm{Val}(E_W) < 5|W| \tag{6}$$

Let $\mathrm{Sp}(W) = \sum_{u \in W}(1 - x_{u,v}) = |E_W| - \mathrm{Val}(E_W)$. Thus (6) implies that $\mathrm{Sp}(W) < 5|W|$. This in turn implies that there is a vertex $v \in W$ satisfying the condition in step 4.

4.1 Token assignment: Proof of Claim 3

Consider the following token assignment scheme. For each W-edge we assign $1 + x_e$ tokens, which are present at its head. For each ordinary edge, we give 2 tokens, $1 + x_e$ is present at its head. The remaining $1 - x_e$ token are in its middle (these tokens belong to the smallest set $S \in \mathcal{L}$ containing e). Note that either type of edge has $1 + x_e$ tokens at its head. We also assign 3 tokens to each W-vertex. We will show that the tokens can be redistributed such that each node is the laminar family gets at least two tokens, with at least one token to spare overall. It is clear that this scheme will imply Claim 3.

We call a node $S \in \mathcal{L}$ *marked* if there is some vertex $w \in W \cap S$; otherwise S is called *unmarked*. Note that if S is an unmarked node, each edge of E' contained in S is ordinary and has x-value at most ϵ. Also, for any tight set S, $x(\delta^-(S)) = 1$. The assignment of tokens to nodes of $\mathcal{L}$ proceeds using the following steps.

1. Assignment to leaf nodes. Let $S \in \mathcal{L}$ be a leaf-node. Since $x(\delta^-(S))$ corresponds to a tight constraint, we have $x(\delta^-(S)) = 1$. Hence there are at least two edges of E' entering S (as there are no 1-edges). So S gets at least $2 + x(\delta^-(S)) = 3$ tokens from the heads of these edges.
2. Assignment to unmarked non-leaf nodes. Let $S \in \mathcal{L}$ be such a node, and $C_1, \cdots, C_t \subset S$ its children. Since $S, C_1, \cdots, C_t$ are tight, $\sum_{i=1}^{t} x(\delta^-(C_i)) - x(\delta^-(S)) = (t-1)$. Since edges lying in S are ordinary and have x_e strictly less than ϵ (and hence strictly less than $1/2$ there are at least $2(t-1)+1$ edges that *belong* to S. Thus S obtains at least $2(t-1)+1-(t-1) = t$ middle tokens from these edges which is at least 2 for $t \geq 2$.

   We now consider the case of $t = 1$. Let C be the unique child of S in $\mathcal{L}$. Let $z = \sum_{e \in E'(V \setminus S, C)} x_e$ denote the total x-value entering C from outside S. If $z = 1$, then as S and C are tight, it follows that $\chi_{\delta^-(S)} = \chi_{\delta^-(C)}$ which contradicts linear independence. Thus $1-z$ units of x-value enters C from $S \setminus C$ and exactly $1-z$ units of x-value enters $S \setminus C$ from $V \setminus S$. The edges in $E'(V \setminus S, S \setminus C)$ contribute at least $1 + (1-z) = 2 - z$ units of tokens to S. The edges in $E'(S \setminus C, C)$ are ordinary edges and hence contribute at least $|E'(S \setminus C, C)| - x(E'(S \setminus C, C)) \geq 1 - (1-z) = z$ tokens. Thus S receives at least $2 - z + z = 2$ tokens overall.
3. Assignment to marked nodes. Let $\mathcal{M} \subseteq \mathcal{L}$ denote the laminar family consisting of only marked nodes. Call a node $S \in \mathcal{M}$ *high-degree* if it has at least 2 children in $\mathcal{M}$; and *small-degree* if it has exactly 1 child in $\mathcal{M}$; all other nodes in $\mathcal{M}$ are *leaf-nodes* (no children). Note that the number of high-degree nodes in $\mathcal{M}$ is strictly less than the number of leaves in $\mathcal{M}$. Arbitrarily assign each high-degree node in $\mathcal{M}$ 2 tokens from a distinct W-vertex (in a distinct leaf node of $\mathcal{M}$). Also assign each leaf node $S \in \mathcal{M}$, 1 token from some W-vertex contained in it. To see how S obtains 1 additional token, consider the following: If S is also a leaf node in $\mathcal{L}$, then it can get 3 tokens from incoming edges $\delta^-(S)$. If not, then this marked leaf node has some unmarked leaf node in the subtree rooted at it. Since unmarked leaf nodes has been assigned three tokens thus far, we borrow one of these tokens and give it to S. Note that each unmarked leaf is charged at most once like this.

   Note that each W-vertex has been charged at most 3 tokens so far (with at least one token to spare). It remains to show how to assign tokens to small-degree nodes of $\mathcal{M}$. Let $S \in \mathcal{M}$ be such a node, and $C \in \mathcal{M}$ be its unique child in $\mathcal{M}$. We have the following subcases:

(a) $W \cap (S \setminus C) \neq \emptyset$. Let $w \in W \cap (S \setminus C)$ be such a vertex. Note that since there is no node of $\mathcal{M}$ contained in $S \setminus C$, S is the smallest set in $\mathcal{M}$ that contains w. Assign node S 2 tokens from vertex w. Note that this vertex w *can not* be charged by more than one such set S, in this step. Further vertex w could not have been used in the earlier charging to W-vertices since it is not contained in any leaf node of $\mathcal{M}$.

(b) $W \cap (S \setminus C) = \emptyset$. Let r denote the total number of unmarked children of S in $\mathcal{L}$ (note that r is not the number all unmarked nodes contains in $S \setminus C$ like the previous sections). Consider the following cases:

i. $r = 0$. This case is identical to the $t = 1$ case for assignment to unmarked non-leaf nodes in step 2. That argument only used the fact that $(S \setminus C) \cap W = \emptyset$ and not $C \cap W = \emptyset$. So, at least 2 tokens assigned to S.

ii. $r \geq 2$. Consider the set of unmarked nodes in $\mathcal{L}$ that are contained in $S \setminus C$. Since $r \geq 2$, there must be at least two unmarked leaf nodes in this set. We borrow 1 token from each of these leaves and give it to S. Note that these tokens could not have been used in step 3 in the charging for marked leaf nodes, and that each unmarked leaf node is charged at most once.

iii. $r = 1$. Let D be the unmarked child of S. We first consider the simpler case when there is an incoming edge e in $S \setminus (C \cup D)$. Here, the edge e provides at least 1 token to S. For the remaining token, we observe that the subtree rooted at D in $\mathcal{L}$ has at least one unmarked leaf node (possibly D). This node has at least 3 tokens since they could not have been used earlier in steps 3 or 3(b)ii earlier. Thus S obtains at least 2 tokens overall in this case.

Henceforth, we assume that all edges from $V \setminus S$ enter C or D. Since S, C, D are all tight, we have $1 = x(\delta^-(C)) + x(\delta^-(D)) - x(\delta^-(S))$. Let z be the total amount of flow entering from $V \setminus S$ to C. Note that z cannot be 0 or 1, as it would imply that either $\chi_{\delta^-(D)} = \chi_{\delta^-(S)}$ or $\chi_{\delta^-(C)} = \chi_{\delta^-(S)}$, which would violate the linear independence. Also, it follows directly that $x(E'(V \setminus S, D)) = 1 - z$, that $x(E'(S \setminus D, D)) = z$ and that $x(E'(S \setminus C, C)) = 1 - z$.

Suppose $z \geq 1 - \epsilon$. We claim that $S \cup D$ get at least 4 tokens. Since $z \geq 1 - \epsilon$, there must be at least two edges from $S \setminus D$ to D (since the x-value of each edge is strictly less than $\max(\epsilon, 1 - \epsilon) = 1 - \epsilon$, these edges provide at least $2 + z$ tokens. Moreover the edges $E'(V \setminus S, D)$ provide at least $1 + x(E'(V \setminus S, D)) = 1 + 1 - z = 2 - z$ tokens. Thus S and D together have at least 4 tokens.

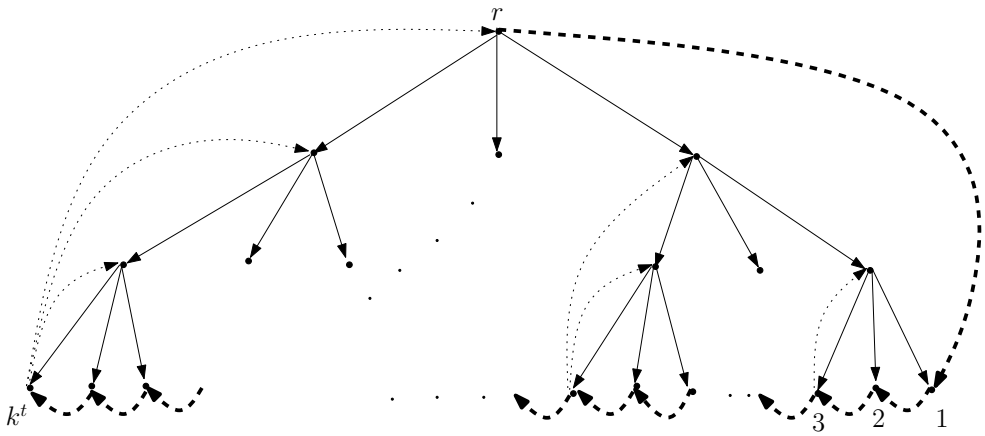
Now if $z < 1 - \epsilon$, we have that $x(E'(S \setminus C, C) = 1 - z > \epsilon$ and hence $|E'(S \setminus C, C)| \geq 2$ since each such edge is an ordinary edge ($S \setminus C$ has no W-vertex). Thus S obtains $|E'(S \setminus C, C)| - x(E'(S \setminus C, C)) \geq 2 - (1 - z) = 1 + z$ middle tokens form them. Also the edges from $S \setminus D$ to D contribute $|E'(S \setminus D, D)| + x(E'(S \setminus D, D)) \geq 1 + z$ and edges from $V \setminus S$ to D contribute $|E'(V \setminus S, D)| + x(E'(V \setminus S, D)) \geq 1 + (1 - z) = 2 - z$ tokens. Thus S and D together receive at least $(1 + z) + (1 + z) + (2 - z) = 4 + z \geq 4$ tokens.

Thus the proof is complete.

4.2 Integrality gap

In this section, we show that our result for minimum-cost bounded-degree arborescence is in fact best possible based on the linear relaxation for this problem. In the following, $O(1)$ denotes a constant independent of ϵ.

Theorem 7 *For every $\epsilon > 0$, there exists an instance of minimum cost bounded degree arborescence problem such that, any arborescence having degree at most $b_v/(1 - \epsilon) + O(1)$ at each vertex v has cost at least $\frac{1 - o(1)}{\epsilon}$ times the optimal LP value.*



Normal arcs (on complete t-level k-ary tree T) have cost 0, and LP value $1-\delta$.

Dotted arcs have cost 0, and LP value δ.

Heavy dashed arcs have cost 1, and LP value δ.

Figure 2: The integrality gap instance with $k = 3$, $t = 3$.

Proof: Given an arbitrarily small but fixed constant $\epsilon > 0$, set $\delta = \epsilon + \epsilon^c$ where c is arbitrarily large constant independent of ϵ. Consider the directed graph $G(\delta)$ constructed as follows; we refer the reader to Figure 2 for clarity. Start with a complete k-ary outward directed tree T rooted at vertex r, with t levels (the solid edges in Figure 2), where we set $k = 1/\delta^{2c}$ and $t = c\delta^{c+1}\ln(2/\delta)$. These tree edges have cost 0, and we call them T-edges. Consider the natural drawing of the tree on the plane (as in Figure 2) and label the leaves from right to left from $1, \ldots, k^t$. We say the root is at level 0 and the leaves are at level t. We also label the vertices on level i from $1, \ldots, k^i$ in the right to left order. For a vertex v, let T_v denote the subtree rooted at v and let r_v and l_v denote the smallest and largest indices of leaves in T_v (formally if v is the jth node from the right on level i, then $l_v = jk^{t-i}$ and $r_v = (j-1)k^{t-i} + 1$).

We add the following additional edges to obtain $G(\delta)$. For each internal vertex v, we add an edge from the leaf l_v to v (these are light dotted edges in Figure 2). All these edges also have cost 0. Finally, we add the path from the root, visiting the leaves in the order $1, \ldots, k^t$ (these edges are denoted by the heavy dashed edges) and each of these edges has cost 1.

Consider the problem of constructing the minimum cost arborescence rooted at r, where each internal vertex has an upper bound of $b = (1-\delta)k$ on the outdegree. Consider the fractional assignment to edges where each (solid) edge corresponding to the tree T has value $x_e = 1-\delta$ and every other edge has value $x_e = \delta$. Observe that each vertex has 1 unit of flow from the root and the fractional out-degree of each internal vertex is $(1-\delta)k$ and hence this a feasible LP solution with cost $LP^* = \delta k^t$.

We now show that any integral solution I where the degree at each internal vertex is at most $b/(1-\epsilon) + O(1)$ has cost at least $(1-o(1))LP^*/\epsilon$. The crucial observation is the following: Suppose a leaf ℓ does not have a path from root to itself in I using only T-edges, then the edge $(\ell-1, \ell)$ must necessarily lie in I. To see this, consider the unique path from r to ℓ in T and let (u, v) be some edge along this path that does not lie in I (such an edge must exist since ℓ is unreachable from r using T-edges). Let L denote the set of leaves $\{\ell, \ldots, l_v\}$, and let S_v denote the set of all nodes in T_v from which some vertex in L can be reached using T-edges. We claim that the (thick) edge $(\ell-1, \ell)$ is only edge in I entering the set S_v. Indeed, no T-edge enters S_v since $(u, v) \notin I$, similarly by construction of the graph $G(\delta)$ only one thick edge $(\ell-1, \ell)$ enters S_v . Moreover, no dotted line enters S_v since such edge must be of the form (ℓ', w) where ℓ' is a leaf not in S_v and hence $\ell' \in \{r_v, \ldots, \ell-1\}$ and $w \in S_v$. Now by the construction of dotted edges in $G(\delta)$ this means that $\ell' = \ell_w$, and hence $\ell_w \in \{r_v, \ldots, \ell-1\}$. But the

only leaves reachable by T-edges from w have indices at most ℓ_w which is at most $\ell - 1$; this implies that none of the leaves in L can be reached by w which contradicts that $w \in S_v$. Thus, $(\ell - 1, \ell)$ is unique edge entering S_v and must necessarily lie in I.

To finish the proof, consider the solution I where each internal vertex has degree at most $b/(1-\epsilon) + O(1) = (1-\delta)k/(1-\epsilon)+O(1) = (1-(\epsilon^c/(1-\epsilon)))k+O(1)$ which is at most $(1-\delta^{c+1})k$.. Thus the total number of leaves that have a path from root r using T-edges is at most $(1-\delta^{c+1})^t k^t \leq (\delta/2)^c k^t < \epsilon^c k^t$. Thus by the above claim, at least $(1-\epsilon^c)k^t$ cost 1 edges must lie in I, which implies that the total cost is at least $(1-\epsilon^c)k^t = ((1-\epsilon^c)LP^*)/\delta = (1-\epsilon^c)LP^*/(\epsilon+\epsilon^c) \geq ((1-2\epsilon^{c-1})LP^*/\epsilon)$. Since c is arbitrarily large, this implies the result.

■

As a corollary of this theorem, we see that in order to achieve a purely additive $O(1)$ guarantee on the degree using the LP, the cost would have to be violated by a factor at least $\Omega(\log n)$, where n is the number of vertices in the graph.

5 General connectivity requirements with costs

We work with the following linear relaxation (DLP) in each iteration. Let $F \subseteq E$ denote the set of edges that have been fixed to value 1, $A \subseteq V$ the vertices for which there is an in-degree bound, and $B \subseteq V$ the vertices for which there is an out-degree bound at some generic iteration. This section proves Theorem 5.

$$
\begin{array}{rll}
\min & \sum_{e \in E \setminus F} c_e x_e & \\
s.t. & & \\
 & x(\delta^-(S)) \geq f(S) - |\delta_F^-(S)| & \forall S \subseteq V \\
 & x(\delta^+(v)) \leq b_v - (1-\epsilon)|\delta_F^+(v)| & \forall v \in B \\
P(E,F,A,B) & x(\delta^-(v)) \leq a_v - (1-\epsilon)|\delta_F^-(v)| & \forall v \in A \\
 & x_e \leq 1 & \forall e \in E \setminus F \\
 & x_e \geq 0 & \forall e \in E \setminus F
\end{array}
$$

In any iteration, let x denote an optimal basic feasible solution, and let $E' = E \setminus F$. The algorithm is similar to the one for minimum cost arborescence with bounded degrees. It works with a parameter $0 < \epsilon \leq 1/2$ and performs one of the following steps in each iteration where $E' \neq \emptyset$:

1. If there is an edge $e \in E'$ with $x_e = 0$, set $E \leftarrow E \setminus \{e\}$.
2. If there is an edge $e \in E'$ with $x_e \geq 1 - \epsilon$, set $F \leftarrow F \cup \{e\}$.
3. If there is an edge $e = (u, v) \in E'$ with $u \notin B$ *and* $v \notin A$ and $x_e \geq \epsilon$, set $F \leftarrow F \cup \{(u, v)\}$.
4. If there is a vertex $v \in B$ with strictly less than $b_v - (1-\epsilon)|\delta_F^+(v)| + 5$ support edges leaving it, set $B \leftarrow B - \{v\}$.
5. If there is a vertex $v \in A$ with strictly less than $a_v - (1-\epsilon)|\delta_F^-(v)| + 5$ support edges entering it, set $A \leftarrow A - \{v\}$.

If at least one of these conditions holds at each iteration, the algorithm results in a solution satisfying the connectivity requirement, of cost at most $\frac{1}{\epsilon}$ times the optimal, while having degree at most $\lceil \frac{b_v}{1-\epsilon} \rceil + 4$ at each vertex $v \in V$. The rest of this section proves that one of the above conditions is always true. In particular, we show that if none of the conditions (1)-(3) are satisfied, then at least one of (4) and (5) must be true. We assume henceforth that none of (1)-(3) are satisfied. As in the previous section, since conditions (1) and (2) do not hold: all the tight constraints in a basic feasible solution x come from the cut-conditions and the degree constraints. Based on standard uncrossing arguments, we have the following.

Lemma 5 ([11]) *For any basic solution x to (DLP) with no variable fixed to 0 or 1, there exist sets $T' \subseteq A$, $T'' \subseteq B$, and a laminar family $\mathcal{L}$ of subsets of V such that x is the unique solution to the linear system:*

$$\begin{array}{ll} x(\delta^-(v)) = a_v - (1-\epsilon)|\delta_F^-(v)| & \forall v \in T' \\ x(\delta^+(v)) = b_v - (1-\epsilon)|\delta_F^+(v)| & \forall v \in T'' \\ x(\delta^-(S)) = f(S) - |\delta_F^-(S)| & \forall S \in \mathcal{L} \end{array}$$

Furthermore, the following two conditions hold:

1. *For every $S \in \mathcal{L}$, $f(S) - |\delta_F^-(S)| \geq 1$ and integral.*
2. *The characteristic vectors $\{\chi_{\delta^-(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{\delta^-(v)} \mid v \in T'\} \cup \{\chi_{\delta^+(v)} \mid v \in T''\}$ are linearly independent; and*
3. *The size of the support $|E'| = |T'| + |T''| + |\mathcal{L}|$.*

Let $W = A \cup B$. We now classify the various types of edges in the support into subsets $E_1, \ldots, E_5$ just as in Section 3 with the addition: let E_0 denote the set of edges (u, v) such that $u \notin B$ and $v \notin A$.

Observe that the sets $E_0, E_1, \ldots, E_5$ are pairwise disjoint and that each edge in the support must lie in one of the sets above, and hence $E_0, E_1, \ldots, E_5$ form a disjoint partition of E'. We use the same notation, in particular definitions of Sp and Val, from Section 3.

Claim 4

$$2|\mathcal{L}| < 2|E_0| + |E_1| + |E_2| + \mathrm{Val}(E) - \mathrm{Val}(E_0) + 3|W|. \tag{7}$$

We first see how this claim implies Theorem 5. Since $|E| = |\mathcal{L}| + |T'| + |T''|$, (7) implies that

$$\begin{aligned} 2|E| &< 2|E_0| + |E_1| + |E_2| + \mathrm{Val}(E_1) + \mathrm{Val}(E_2) + \mathrm{Val}(E_3) + \mathrm{Val}(E_4) + \mathrm{Val}(E_5) \\ &\quad +3|W| + 2|T'| + 2|T''| \\ &\leq 2|E_0| + |E_1| + |E_2| + \mathrm{Val}(E_1) + \mathrm{Val}(E_2) + \mathrm{Val}(E_3) + \mathrm{Val}(E_4) + \mathrm{Val}(E_5) \\ &\quad +3|W| + 2|A| + 2|B| \\ &\leq 2|E_0| + |E_1| + |E_2| + \mathrm{Val}(E_1) + \mathrm{Val}(E_2) + \mathrm{Val}(E_3) + \mathrm{Val}(E_4) + \mathrm{Val}(E_5) + 5|A| + 5|B| \end{aligned}$$

The second step follows as $|T'| \leq |A|$ and $|T''| \leq |B|$ and the third step follows as $|W| \leq |A| + |B|$. As $|E| = |E_0| + |E_1| + |E_2| + |E_3| + |E_4| + |E_5|$, and as $\mathrm{Sp}(X) = |X| - \mathrm{Val}(X)$, we obtain that

$$\mathrm{Sp}(E_1) + \mathrm{Sp}(E_2) + \mathrm{Sp}(E_3) + \mathrm{Sp}(E_4) + \mathrm{Sp}(E_5) + |E_3| + |E_4| + |E_5| < 5|A| + 5|B|.$$

As $\mathrm{Sp}(X) \leq |X|$ for any subset of edges X, this implies that

$$\mathrm{Sp}(E_1) + \mathrm{Sp}(E_2) + 2\mathrm{Sp}(E_3) + 2\mathrm{Sp}(E_4) + \mathrm{Sp}(E_5) < 5|A| + 5|B|$$

which implies $\mathrm{Sp}(A) + \mathrm{Sp}(B) < 5(|A| + |B|)$ by inequality (4). This in turn implies that one of conditions in steps 4 or 5 holds.

5.1 Token assignment: Proof of Claim 4

Consider the following token assignment scheme: We assign 2 tokens to each edge $e = (u, v) \in E_0$. Of these, $1 + x_e$ units lie at the head v, and $1 - x_e$ tokens lie in the middle (these will belong to the smallest set in $\mathcal{L}$ that contains e). We assign $1 + x_e$ tokens to each edge $e \in E_1 \cup E_2$. For an edge $(u, v) \in E_1$, the $1 + x_e$ tokens lie at the head v. For an edge $(u, v) \in E_2$, 1 unit of token lies at u, and x_e units lies at the head v. The remaining edges $e = (u, v) \in E_3 \cup E_4 \cup E_5$ have x_e tokens, that are present at vertex v. We also assign 3 tokens to each vertex in W.

We will show that these tokens can be redistributed to obtain at least 2 tokens for each node $S \in \mathcal{L}$, with at least one token to spare. We call a node $S \in \mathcal{L}$ *marked* if there is some vertex $W \cap S \neq \emptyset$; otherwise S is called *unmarked*. Note that if S is an unmarked node, there is *no* edge of E' is contained in S. Also, for any tight set S, $x(\delta^-(S)) \geq 1$ and is an integer. The assignment of tokens to nodes of $\mathcal{L}$ proceeds using the following steps.

1. Assignment to unmarked leaf nodes. Let $S \in \mathcal{L}$ be such a node. Since S is a tight set, we have that $x(\delta^-(S)) \geq 1$. Hence there are at least two edges of E' entering S (as each edge has $x_e < 1$). Assign the tokens at the heads of these edge to S: note that since S is unmarked, these must be edges of type E_0 or E_1 , and S receives at least $2 + x(\delta^-(S)) \geq 3$ tokens.

2. Assignment to unmarked non-leaf nodes. Let $S \in \mathcal{L}$ be such a node, and $C_1, \cdots, C_t \subset S$ its children. Let $z = x(E'(V \setminus S, S \setminus \cup_{i=1}^t C_i))$ denote the total x-value entering $S \setminus \cup_{i=1}^t C_i$ from outside S.

   We first consider the case when $z = 0$. By linear independence it follows that $\sum_i \chi_{\delta^-(C_i)} \neq \chi_{\delta^-(S)}$. By the integrality of connectivity requirements and since $z = 0$, it follows that $(\sum_i x(\delta^-(C_i))) - x(\delta^-(S)) \geq 1$. Consider the edges $\cup_{i=1}^t E'(S \setminus C_i, C_i)$. They must lie in E_0 and there must be at least 3 such edges (as each has x-value strictly less than ϵ, which is at most 1/2), and hence they contribute at least 2 tokens (this argument has been used earlier).

   We now consider the case when $z > 0$. Note that edges in $E'(V \setminus S, S \setminus \cup_{i=1}^t C_i)$ lie either in E_0 or E_1, thus if $z > 0$, then they contribute at least $1 + z$ tokens to S. Thus, if $z \geq 1$, then S obtains two tokens from them. Now, suppose that $z < 1$. By integrality of the tight cuts, it follows that there is at least z amount of x-value in $E'(S \setminus C_i, C_i)$. Since these are edges in E_0, we obtain a contribution of at least $1 - z$ middle tokens from them. Thus together S has at least $(1 + z) + (1 - z) = 2$ tokens.

3. Assignment to marked nodes. Let $\mathcal{M} \subseteq \mathcal{L}$ denote the laminar family consisting of only marked nodes. Call a node $S \in \mathcal{M}$ *high-degree* if it has at least 2 children in $\mathcal{M}$; and *low-degree* if it has exactly 1 child in $\mathcal{M}$; all other nodes in $\mathcal{M}$ are leaf-nodes (no children). We now show how to assign tokens to each of these nodes.

   (a) High-degree nodes: Note that the number of high-degree nodes in $\mathcal{M}$ is strictly less than the number of leaf-nodes in $\mathcal{M}$. Arbitrarily assign each high-degree node in $\mathcal{M}$ two tokens from a distinct W-vertex (in a distinct leaf node of $\mathcal{M}$).

   (b) Leaf-nodes: For each leaf node S in $\mathcal{M}$ we assign 1 token from some W-vertex contained in it. For the remaining token, we argue as follows: If S is also a leaf in $\mathcal{L}$, then S has at $x(\delta^-(S))) \geq 1$ and hence S received at least 1 unit of tokens from edges in $\delta^-(S)$ (since every edge carries at least x_e tokens at its head). If S is not a leaf in $\mathcal{L}$, then consider the subtree rooted S. This subtree has at least one unmarked node. Since each unmarked node at at least 3 tokens thus far, S borrows one token arbitrarily from one of these nodes. Note that an unmarked node cannot be used twice to borrow a token.
   Note that each W-vertex has been charged at most 3 tokens so far.

   (c) Low-degree marked nodes: Let $S \in \mathcal{M}$ be such a node, and $C \in \mathcal{M}$ be its unique child.
   Suppose that $W \cap (S \setminus C) \neq \emptyset$, and $w \in W \cap (S \setminus C)$ be such a vertex. As no node of $\mathcal{M}$ contained in $S \setminus C$, S is the smallest set in $\mathcal{M}$ that contains w. Assign node S two tokens from vertex w. Note that this vertex w cannot be charged by more than one such set S in this step. Moreover, w could not have been used in the earlier charging to W-vertices since it is not contained in any leaf node of $\mathcal{M}$.
   Henceforth we assume that $W \cap (S \setminus C) = \emptyset$. Let r denote the total number of unmarked children of S in $\mathcal{L}$ (note that r is not the number all unmarked nodes contained in $S \setminus C$ like the previous sections). Consider the following cases:

      i. $r = 0$. Let $z = x(E'(V \setminus S, S \setminus C)$ denote the total x-value entering $S \setminus C$ from outside S. We first consider the case when $z = 0$. By linear independence it follows that $\chi_{\delta^-(C)} \neq \chi_{\delta^-(S)}$. By the integrality of connectivity requirements and since $z = 0$, it follows that $x(\delta^-(C)) - x(\delta^-(S)) \geq 1$. Consider the edges $E'(S \setminus C, C)$. They must either lie in E_0 or E_2. If they all lie in E_0 there must be at least 3 such edges, and hence they contribute at least 2 tokens (this argument has been used earlier). If at least two of them are E_2 edges, we obtain the two tokens from them for S also as each of them contributes 1 token to S. If exactly one of them is an E_2 edge, then this has x-value

strictly less than $1-\epsilon$, hence we need at least two more edges from E_0 to ensure that $x(\delta^-(C)) - x(\delta^-(S)) \geq 1$. These edges provide the two tokens for S.
We now consider the case when $z > 0$. Note that edges in $E'(V \setminus S, S \setminus C)$ lie either in E_0 or E_1, thus if $z > 0$, then they contribute at least $1+z$ tokens to S. Thus, if $z \geq 1$, then S obtains two tokens from them. Now, suppose that $z < 1$. By integrality of the tight cuts, it follows that at least z amount of x-value must also enter C from $S \setminus C$. Since these are edges in either E_0 or E_2, we obtain a contribution of at least $1-z$ tokens from them. Thus together S has at least $(1+z)+(1-z) = 2$ tokens.

ii. $r \geq 2$. Consider the unmarked leaf nodes in $\mathcal{L}$ contained in $S \setminus C$. Note that each of them has been assigned at least 3 tokens thus far (they could not have given a token to marked node in previous steps). S is assigned 2 tokens by borrowing 1 token each from some two unmarked leaf nodes in $S \setminus C$ (there are at least two since $r \geq 2$).

iii. $r = 1$. Let D be the unmarked child of S. We first consider the simpler case when there is an incoming edge e in $S \setminus (C \cup D)$. Here, the edge e provides at least 1 token to S. For the remaining token, we observe that the subtree rooted at D in $\mathcal{L}$ has at least one unmarked leaf node (possibly D). This node has at least 3 tokens since they could not have been used earlier in steps earlier. Thus S obtains at least 2 tokens overall in this case.
Henceforth, we assume that all edges from $V \setminus S$ enter C or D. Since S, C, D are all tight, we have $x(\delta^-(C)) + x(\delta^-(D)) - x(\delta^-(S)) \geq 1$. If $x(\delta^-(D)) \geq 2$, then since these are only E_1 or E_0 edges, D obtains at least 4 tokens that can be shared by S and D. Similarly if $x(\delta^-(D)) = 0$, then it must be the case that $x(E'(S \setminus (C \cup D), C)) \geq 1$. These edges must lie in either E_0 or E_2. If there are two or more E_2 edges, then we obtain at least 2 tokens. If there are no E_2 edges, then all are E_0 edges, hence there are at least three of them and they contribute at least 2 tokens. Finally, if there is exactly one E_2 edge, then there must be at least 2 edges from E_0 (since one E_0 edge and one E_2 cannot have total x-value of at least 1). This gives at least 2 tokens.
Henceforth we assume that $x(\delta^-(D)) = 1$. Let z be the total amount of flow entering from $V \setminus S$ to D. Note that $0 \leq z \leq 1$. Also, it follows by integrality of connectivity requirement for tight cuts that $x(E'(S \setminus C, C)) \geq z$ and that $x(E'(S \setminus D, D)) = 1 - z$. Suppose $z < \epsilon$. We claim that $S \cup D$ get at least 4 tokens. Since $z < \epsilon$, there must be at least two edges from $S \setminus D$ to D (since the x-value of each edge is strictly less than $\max(\epsilon, 1-\epsilon) = 1-\epsilon$, these edges provide at least $2+z$ tokens. Moreover the edges $E'(V \setminus S, D)$ provide at least $1 + x(E'(V \setminus S, D)) = 1 + 1 - z = 2 - z$ tokens. Thus S and D together have at least 4 tokens.
Now if $z \geq \epsilon$, we have that $x(E'(S \setminus C, C)) = z \geq \epsilon$ and hence $|E'(S \setminus C, C)|$ consists of at least two E_0 edges or at least one E_2 edge. Thus we obtain at least 1 token from them.
Also the edges from $S \setminus D$ to D (these must be edges in E_0 or E_1) contribute $|E'(S \setminus D, D)| + x(E'(S \setminus D, D)) \geq 1 + 1 - z = 2 - z$ and edges from $V \setminus S$ to D contribute $|E'(V \setminus S, D)| + x(E'(V \setminus S, D)) = 1 + z$ tokens. Thus S and D together receive at least 4 tokens.

Thus we have shown that (5) holds which implies the result.

6 Generalized minimum crossing spanning tree problem

Given an undirected graph $G = (V, E)$, costs $c_e \geq 0$ on the edges $e \in E$, subsets of edges $E_i \subseteq E$ for $1 \leq i \leq k$, and integers $b_i \geq 0$ for $1 \leq i \leq k$, the generalized minimum crossing spanning tree problem (MCSP) is to find a minimum cost spanning tree (if it exists) in G that contains at most b_i edges from set E_i for $1 \leq i \leq k$. We prove theorem 6 which gives an additive $+(r-1)$ approximation with respect to

crossing, and achieves optimum cost.

Our result significantly improves on the results of Bilò et al. [1]. They consider an unweighted instance in which all b_i are equal to b and find a spanning tree which contains at most $O(b \cdot r \log n)$ edges in the set E_i for $1 \leq i \leq k$.

Our result contains several interesting special cases.

1. $r = 1$. If the sets E_i are pairwise-disjoint, the MCSP problem can be cast as finding a minimum cost basis in the graphic matroid for G that is independent in a partition matroid, in which, an independent set must have at most b_i elements from set E_i. This problem is an instance of the *matroid intersection problem* which is known to be solvable in polynomial time [4, 13].
2. $r = 2$. If E_i denote the set of edges incident to vertex i and b_i denote the degree bound on vertex i, we get the bounded degree minimum spanning tree problem. Our algorithm matches the $+1$ bounds recently obtained by Singh and Lau [16].

6.1 The algorithm

Our algorithm has several iterations. Consider a general iteration. With a slight abuse of notation we use E to denote the candidate edges which are not yet discarded, let $F \subseteq E$ denote the set of edges that we have already picked in our solution, and let $W \subseteq \{i \mid 1 \leq i \leq k\}$ denote the crossing constraints that we have not yet dropped. In the beginning E is the entire edge-set, $F = \emptyset$, and $W = \{i \mid 1 \leq i \leq k\}$. In a general iteration, we work with the following linear relaxation with variables x_e for $e \in E' = E \setminus F$.

$$
\begin{array}{rll}
\min & \sum_{e \in E'} c_e \cdot x_e & \\
s.t. & & \\
 & x(E'(V)) = V - 1 - |F(V)| & \\
 & x(E'(S)) \leq S - 1 - |F(S)| & \forall S : 2 \leq |S| \leq |V| - 1 \\
P(E, F, W): & x(E' \cap E_i) \leq b_i - |F \cap E_i| & \forall i \in W \\
 & x_e \leq 1 & \forall e \in E' = E - F \\
 & x_e \geq 0 & \forall e \in E' = E - F
\end{array}
$$

where $H(S)$ (for $H \subseteq E$ and $S \subseteq V$) is the set of edges in H with both end-points in S. In this iteration, the algorithm computes a basic feasible solution x to $P(E, F, W)$ and performs one of the following steps while $E' = E \setminus F \neq \emptyset$:

1. If there is an edge $e \in E'$ with $x_e = 0$, set $E = E \setminus \{e\}$.
2. If there is an edge $e \in E'$ with $x_e = 1$, set $F \leftarrow F \cup \{e\}$.
3. If for some $i \in W$, $|E' \cap E_i| \leq b_i - |F \cap E_i| + r - 1$, or equivalently $|E \cap E_i| \leq b_i + r - 1$, set $W \leftarrow W - \{i\}$.

It is clear that if the algorithm terminates, it terminates with a set F which is a spanning tree with cost at most the optimum and which contains at most $b_i + r - 1$ edges from E_i for $1 \leq i \leq k$.

We now argue that in each iteration, one of the above steps is always applicable. The following lemma follows by uncrossing.

Lemma 6 ([7, 16]) *For any basic solution x to $P(E, F, W)$, there exists a set $T \subseteq W$ and a laminar family $\mathcal{L}$ of subsets of V such that x is the unique solution to the linear system:*

$$
\begin{array}{ll}
x(E'(S)) = |S| - 1 - |F(S)| & \forall S \in \mathcal{L} \\
x(E' \cap E_i) = b_i - |F \cap E_i| & \forall i \in T
\end{array}
$$

Furthermore, ($|E'|$-dimensional) characteristic vectors $\{\chi_{E'(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{E' \cap E_i} \mid i \in T\}$ are linearly independent, and the size of the support $|E'| = |T| + |\mathcal{L}|$.

Assume that the conditions in steps (1) and (2) do not hold. The key component of our proof is the following lemma which is proved by a simple counting argument.

Claim 5 *We have $|\mathcal{L}| \leq x(E'(V))$. Moreover the equality holds if and only if each edge in E' is contained in some inclusion-wise maximal set $S \in \mathcal{L}$.*

Proof: Suppose each edge $e \in E'$ is given x_e tokens. These tokens are assigned to the sets $S \in \mathcal{L}$ as follows. Edge e is said to belong to S if S is the inclusion-wise minimal set in $\mathcal{L}$ that contains e. If e belongs to S, then x_e tokens are assigned to S. We argue that each set in the laminar family is assigned a total of unit tokens, thereby proving the claim.

Since $x_e > 0$ for all $e \in E'$, each set $S \in \mathcal{L}$ has the right-hand-side $|S| - 1 - |F(S)|$ at least 1, and hence $x(E'(S)) \geq 1$. This gives every *leaf* set $S \in \mathcal{L}$ at least a total of unit tokens. Now consider a *non-leaf* set $S \in \mathcal{L}$ with t children $C_1, \cdots, C_t \in \mathcal{L}$. Now $\chi_{E'(S)} = \sum_{j=1}^{t} \chi_{E'(C_j)} + \sum\{\chi_e \mid e \in E' \text{ belongs to } S\}$. Since $\chi_{E'(S)} \cup \{\chi_{E'(C_j)}\}_{j=1}^{t}$ is a linearly independent set, we have $\{e \mid e \in E' \text{ belongs to } S\} \neq \emptyset$. So, the right-hand-side $|S| - 1 - |F(S)|$ of the constraint for S is at least 1 more than the sum of the right-hand-sides of constraints of $\{C_j\}_{j=1}^{t}$. Thus S gets at least a total of unit tokens. ■

Now for $i \in W$, define $\mathrm{Sp}(i) = \sum_{e \in E' \cap E_i}(1 - x_e) = |E' \cap E_i| - x(E' \cap E_i)$ and for $e \in E'$, define $r(e) = |\{i \in W \mid e \in E' \cap E_i\}|$.

Lemma 7 *We have*

$$\sum_{i \in W} \mathrm{Sp}(i) < r|W|.$$

Before proving Lemma 7, we argue that it directly implies that the condition in step (3) holds. Lemma 7 implies that there exists $i \in W$ such that $\mathrm{Sp}(i) < r$. Since $x(E' \cap E_i) \leq b_i - |F \cap E_i|$, we have

$$|E' \cap E_i| = \mathrm{Sp}(i) + x(E' \cap E_i) < r + b_i - |F \cap E_i|.$$

Since $|E' \cap E_i|$ and $|F \cap E_i|$ are integers, we have $|E' \cap E_i| \leq r + b_i - |F \cap E_i| - 1$, i.e., the condition in step (3) holds for i.

Proof of Lemma 7. Lemma 6 and Claim 5 imply $\sum_{e \in E'}(1 - x_e) = |E'| - x(E'(V)) = |\mathcal{L}| + |T| - x(E'(V)) \leq |T| = |W| - |W \setminus T|$. Therefore

$$\begin{aligned} \sum_{i \in W} \mathrm{Sp}(i) = \sum_{e \in E'} r(e)(1 - x_e) &= r \sum_{e \in E'}(1 - x_e) - \sum_{e \in E'}(r - r(e))(1 - x_e) \\ &\leq r|W| - r|W \setminus T| - \sum_{e \in E'}(r - r(e))(1 - x_e). \end{aligned}$$

Moreover, the equality holds if and only if $|\mathcal{L}| = x(E'(V))$. Thus if $|\mathcal{L}| < x(E'(V))$ or if $r|W \setminus T| + \sum_{e \in E'}(r - r(e))(1 - x_e) > 0$, then we obtain that $\mathrm{Sp}(W) < r|W|$ as desired. Assume on the contrary that this is not the case. This combined with the fact that $x_e < 1$ for all $e \in E'$, we have $r(e) = r$ for all $e \in E'$, $W = T$, and by Claim 5 (equality condition) $\sum_{i=1}^{p} \chi_{E'(S_i)} = \chi_{E'}$, where $S_1, \ldots, S_p$ are the inclusion-wise maximal sets in $\mathcal{L}$. Also since $r(e) = r$ for each $e \in E'$ and $W = T$, we have $\sum_{i \in T} \chi_{E' \cap E_i} = r \cdot \chi_{E'}$. This implies that $r \cdot \sum_{i=1}^{p} \chi_{E'(S_i)} = \sum_{i \in T} \chi_{E' \cap E_i}$, contradicting to the fact that the characteristic vectors $\{\chi_{E'(S)} \mid S \in \mathcal{L}\} \cup \{\chi_{E' \cap E_i} \mid i \in T\}$ are linearly independent. Thus the proof is complete.

Concluding Remarks and Acknowledgments

We would like to thank Zhenghua Fu for suggesting the problem of packing arborescences subject to degree bounds, which started this project. We would also like to thank Mohit Singh and R. Ravi for several useful discussions.

The techniques developed in this paper can also be used to solve connectivity problems on undirected graphs with costs. For example, in the case when the connectivity requirement is specified by a

0-1 proper function, we can give O(1) additive approximations with respect to the degree, and O(1) multiplicative approximation with respect to cost. However, we have recently learnt that Lau and Singh [12] have independently obtained similar results for this problem, and hence we do not present them in this paper. We have also learnt that Lau and Singh [12] have obtained results for a generalization of the MCSP problem considered in Section 6. In particular they consider the problem of computing a minimum cost basis in a matroid $(V, \mathcal{I})$ subject to 'degree bounds' on a family of subsets of V, $\{S_i, b_i\}_{i=1}^k$ (where $S_i \subseteq V$, $b_i \in \mathbb{Z}^+$ for all i); for this problem they obtain an additive $+r$ guarantee on the degree constraints while achieving optimal cost, where r is the maximum number of sets among $\{S_i\}_{i=1}^k$ that any element of V appears in.

References

[1] Vittorio Bilò, Vineet Goyal, R. Ravi, and Mohit Singh. On the crossing spanning tree problem. In *Proceedings of APPROX-RANDOM*, pages 51–60, 2004.

[2] K. Chaudhuri, S. Rao, S. Riesenfeld, and K. Talwar. Push relabel and an improved approximation algorithm for the bounded degree MST problem. In *Proceedings of the 33rd International Colloqium on Automata Languages and Programming*, 2006.

[3] G. Cornuejols, D. Naddef, and J. Fonlupt. The travelling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, 33(3):1–27, 1985.

[4] J. Edmonds. Matroid Intersection in Discrete Optimization I. *Annals of Discrete Mathematics*, 4:3949, 1979.

[5] Andras Frank. Increasing the rooted-connectivity of a digraph by one. *Mathematical Programming (B)*, 84(3):565–576, 1999.

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

[7] Michel X. Goemans. Minimum bounded degree spanning trees. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 273–282, 2006.

[8] Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, pages 39–61, 2001.

[9] Philip N. Klein, Radha Krishnan, Balaji Raghavachari, and R. Ravi. Approximation algorithms for finding low-degree subgraphs. *Networks*, 44(3):203–215, 2004.

[10] J. Konemann and R. Ravi. A matter of degree: Improved approximation algorithms for degree bounded minimum spanning trees. *SIAM Journal on Computing*, 31(3):1783–1793, 2002.

[11] Lap Chi Lau, Joseph (Seffi) Naor, Mohammad R. Salavatipour, and Mohit Singh. Survivable network design with degree or order constraints. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 651–660, 2007.

[12] Lap Chi Lau and Mohit Singh, 2007. Personal Communication.

[13] E.L. Lawler. Matroid intersection algorithms. *Mathematical Programming*, 9:3156, 1975.

[14] Vardges Melkonian and Éva Tardos. Approximation algorithms for a directed network design problem. In *Proceedings of the 7th International Conference on Integer Programming and Combinatorial Optimization*, pages 345–360, 1999.

[15] R. Ravi, M.V. Marathe, S.S. Ravi, D.J. Rosenkrantz, and H.B. Hunt III. Many birds with one stone: Multi objective approximation algorithms. In *Proceedings of the 25th annual ACM symposium on Theory of computing*, pages 438–447, 1993.

[16] Mohit Singh and Lap Chi Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 661–670, 2007.