Node-weighted Network Design in Planar and Minor-closed Families of Graphs ⋆

Chandra Chekuri, Alina Ene, and Ali Vakilian

Dept. of Computer Science, University of Illinois, Urbana, IL 61801, USA
{chekuri,ene1,vakilia2}@illinois.edu

Abstract. We consider *node-weighted* network design in planar and minor-closed families of graphs. In particular we focus on the edge-connectivity survivable network design problem (EC-SNDP). The input consists of a node-weighted undirected graph $G = (V, E)$ and integral connectivity requirements $r(uv)$ for each pair of nodes uv. The goal is to find a minimum node-weighted subgraph H of G such that, for each pair uv, H contains $r(uv)$ edge-disjoint paths between u and v. Our main result is an $O(k)$-approximation algorithm for EC-SNDP where $k = \max_{uv} r(uv)$ is the maximum requirement. This improves the $O(k \log n)$-approximation known for node-weighted EC-SNDP in general graphs [15]. Our algorithm and analysis applies to the more general problem of covering a proper function with maximum requirement k. Our result is inspired by, and generalizes, the work of Demaine, Hajiaghayi and Klein [5] who gave constant factor approximation algorithms for node-weighted Steiner tree and Steiner forest problems (and more generally covering 0-1 proper functions) in planar and minor-closed families of graphs.

1 Introduction

Network design is an important area of discrete optimization with several practical applications. Moreover, the clean optimization problems that underpin the applications have led to fundamental theoretical advances in combinatorial optimization, algorithms and mathematical programming. In this paper we consider a class of problems that can be modeled as follows. Given an undirected graph $G = (V, E)$ find a subgraph H of *minimum weight/cost* such that H satisfies certain desired *connectivity* properties. A common cost model is to assign a non-negative weight $w(e)$ to each $e \in E$ and the weight/cost of H is simply the total weight of edges in it. A number of well-studied problems can be cast as special cases. Examples include polynomial-time solvable problems such as the minimum spanning tree (MST) problem when H is required to connect all the nodes of G, and the NP-Hard Steiner tree problem where H is required to connect only a given subset $S \subseteq V$ of terminals. A substantial generalization of these problems is the *survivable network design problem* which is defined as follows. The input, in addition to G, consists of an integer requirement function $r(uv)$ for each (unordered) pair of nodes uv in G; the goal is to find a minimum-weight subgraph H that contains $r(uv)$ edge-disjoint paths between u and v for each pair uv. This problem is called the

⋆ The authors are partially supported by NSF grant CCF-1016684.

edge-connectivity SNDP (EC-SNDP) to distinguish from more general problems such as Elem-SNDP and VC-SNDP that require the paths to be element and vertex disjoint respectively. SNDP arises naturally in the design of fault-tolerant networks, and various special cases have been extensively studied. Algorithmic approaches for SNDP and related problems are based on solving a larger class of abstract network design problems such as covering proper and skew-supermodular cut-requirement functions that we describe formally later.

Node weights: The cost of a network is dependent on the application. In connectivity problems, as we remarked, a common model is the edge-weight model. A more general problem is obtained when each node v of G has a weight $w(v)$ and the weight of H is the total weight of the nodes in H[1]. Node weights are relevant in several applications, in particular telecommunication networks, where they can model the cost of setting up routing and switching infrastructure at a given node. There have also been several recent applications in wireless network design [17,16] where the weight function is closely related to that of node weights. We refer the reader to [5] for some additional applications of node weights to network formation games.

The node-weighted versions of network design problems often turn out to be strictly harder to approximate than their corresponding edge-weighted versions. For instance the Steiner tree problem admits a 1.39-approximation for edge-weights [2], however, Klein and Ravi [12] showed, via a simple reduction from the Set Cover problem, that the node-weighted Steiner tree problem on n nodes is hard to approximate to within an $\Omega(\log n)$-factor unless $P = NP$. They also described a $(2 \log k)$-approximation where k is the number of terminals. A more dramatic difference emerges if we consider SNDP. Jain gave a 2-approximation for EC-SNDP with edge-weights [10]. The best known approximation for EC-SNDP with node-weights is $O(k \log n)$ [15] where $k = \max_{uv} r(uv)$ is the maximum connectivity requirement. Nutov [15] gives evidence, via a reduction from the k-densest-subgraph problem, that for the node-weighted problem a dependence on k in the approximation ratio is necessary.

Demaine, Hajiaghayi and Klein [5] considered the approximability of the node-weighted Steiner tree problem in planar graphs. In an interesting result, they adapted the well-known primal-dual algorithm for the edge-weighted problem [1,7] to the node-weighted problem and showed that it gives a 6-approximation in planar graphs. Demaine et al. also showed that their algorithm works for a more general class of 0-1-valued proper functions (first considered by Goemans and Williamson [7]) that includes several other problems such as the Steiner forest problem ([14] claims an improved $9/4$ approximation for the Steiner forest problem). Their analysis also shows that one obtains a constant factor approximation (the algorithm is the same) for any minor-closed family of graphs where the constant depends on the family. In addition to their theoretical value, these results have the potential to be useful in practice since in many real-world networks the underlying graph G is either planar or has very few crossings.

[1] For many problems of interest, including Steiner tree and SNDP, the version with weights on both edges and nodes can be reduced to the version with only node weights; sub-divide an edge e by placing a new node v_e and set the weight of v_e to be that of e.

Our Results: In this paper we consider node-weighted network design problems in planar graphs for higher connectivity. In particular we consider EC-SNDP and show that the insights in [5] can be used to develop improved approximation algorithms for this more general problem as well. However, the results require non-trivial technical work that we explain after we state the results. The algorithm works for any graph but the ratio is constant for planar graphs and more generally graphs from any minor-closed family; we articulate the precise dependence of the ratio on the family in later sections.

Our main result is the following.

Theorem 1. *There is an $O(k)$-approximation for node-weighted EC-SNDP in planar graphs where k is the maximum requirement.*

The above theorem extends to a more general problem that we describe now. An integer valued set function $f : 2^V \to \mathbb{Z}_+$ on the vertex set of G is said to be proper if it satisfies the following conditions: (i) f is symmetric, that is, $f(S) = f(V - S)$ for all S, and (ii) f is maximal, that is, $f(A \cup B) \leq \max\{f(A), f(B)\}$ for any two disjoint sets A, B. Given a proper function f on V (by a value oracle) and a graph G on V, the f-covering problem is to find a subgraph H of minimum weight such that $|\delta_H(S)| \geq f(S)$ for all S[2]. EC-SNDP is a special case of this problem [18]. We obtain an $O(k)$-approximation for the node-weighted version of this problem in planar graphs where $k = \max_S f(S)$.

Overview of Technical Ideas and Contribution: The two main algorithmic approaches for SNDP are the following. The first is the augmentation approach pioneered by Williamson et al. [18] in which the required network is built in k phases. At the end of the first $(i-1)$ phases the connectivity of a pair uv is at least $\min\{r(uv), i - 1\}$. Thus the i'th phase is required to increase the connectivity of some of the pairs by 1 by adding additional edges; the advantage of this approach is that we now work with a 0-1 covering problem. On the other hand the covering problem is no longer so simple. The function that we need to cover falls into the more general class of *uncrossable* functions: A requirement function $f : 2^V \to \{0, 1\}$ is uncrossable if for any sets $A, B \subseteq V$, $f(A) = f(B) = 1$ implies $f(A \cap B) = f(A \cup B) = 1$ or $f(A - B) = f(B - A) = 1$. Williamson et al. [18] showed that a primal-dual algorithm achieves a 2-approximation for the edge-weighted version of covering uncrossable functions. Nutov [15] gave an $O(\log n)$-approximation for the node-weighted case. These results for uncrossable functions, when combined with the augmentation framework, give a $2k$ and an $O(k \log n)$ approximation for the edge-weighted and node-weighted versions of EC-SNDP in general graphs[3]. The second approach for SNDP is the powerful iterated rounding technique pioneered by Jain which led to a 2-approximation for EC-SNDP [10] and also for covering a certain class of skew-supermodular functions[4]. Iterated rounding does not quite apply to node-weighted problems for various technical reasons.

---

[2] We work with node-induced subgraphs H of G in which case H may not contain all the nodes of a set $S \subset V$. In that case $\delta_H(S)$ denotes the edges of H with exactly one endpoint in S.

[3] The approximation for the edge-weighted version can be improved to $2H_k$ by doing the augmentation in the reverse [6].

[4] A function $f : 2^V \to \mathbb{Z}$ is skew-supermodular if for all $A, B \subseteq V$, $f(A) + f(B) \leq \max\{f(A \cap B) + f(A \cup B), f(A - B) + f(B - A)\}$. A skew-supermodular function f with $f(A) \leq 1$ for all A gives rise to an uncrossable function.

We follow the augmentation approach. Demaine et al. adapted the primal-dual algorithm for edge-weighted 0-1-proper functions to the node-weighted case. The novel technical ingredient in their analysis is to understand properties of *node-minimal* feasible solutions instead of edge-minimal feasible solutions. For the most part, problems captured by 0-1-proper functions are very similar to the Steiner forest problem, a canonical problem in this class. In this setting it is possible to visualize and understand node-minimal solutions through connected components and basic reachability properties. In the augmentation approach for higher-connectivity, as we remarked, the problem in each phase is no longer that of covering a proper function but belongs to the richer class of covering uncrossable functions. The primal-dual analysis for this class of functions is more subtle and abstract [18] and proceeds via uncrossing arguments and laminar witness families.

Our main technical contribution is understanding properties of node-minimal feasible solutions for uncrossable functions. We refer the reader to Theorem 3 in Section 3 for the precise statement; the theorem holds for general graphs (not just planar graphs) and may have other applications. We remark on a crucial aspect of our algorithm and analysis. Why do our results only apply for covering proper functions and not the more general class of skew-supermodular functions? For the node-weighted problem of covering an arbitrary uncrossable function there is no natural covering LP relaxation. However, we observe that the particular uncrossable functions that arise in the augmentation framework for a proper function (including EC-SNDP) have certain additional connectivity properties that allow for an LP relaxation and the primal-dual approach. We obtain a constant factor approximation in each phase and this results in an $O(k)$-approximation overall where k is the maximum requirement.

As in [5] we use planarity only in one step of the analysis where we argue about the average degree of a certain graph that is a minor of the original graph; this is the reason that the algorithm and analysis generalize to any minor-closed family. In this paper, in the interest of clarity and exposition, we have not attempted to optimize the constants in the approximation.

Extensions: Our ideas for EC-SNDP can be extended to give an $O(k)$ approximation for node-weighted Elem-SNDP in planar graphs. We again use the augmentation approach but for Elem-SNDP we use a primal-dual algorithm and analysis with respect to the setpair relaxation [11,3]. There are however some non-trivial differences and the generalization is not immediate. An improved algorithm for node-weighted VC-SNDP in planar graphs follows from a generic reduction of VC-SNDP to Elem-SNDP [4]. A longer version of this paper will discuss these extensions.

Other related work: There is extensive literature on network design but due to space limitations we are unable to discuss it in detail. We refer the reader to [8] for a survey on primal-dual based algorithms for network design, and to recent surveys [13,9] for an overview of the known approximation results and references to related work.

Organization: Section 2 describes our algorithm based on the augmentation approach and the primal-dual algorithm for each phase of the augmentation. The analysis is done by assuming the main technical theorem on a node-minimal augmentation of the uncrossable requirement functions that arise in the augmentation framework. We state and prove this theorem in Section 3. Some of the proofs are omitted in this version. A longer version with detailed proofs as well as the claimed extensions will be made available on arXiv and the authors' web pages in the near future.

2 Algorithm for Node-weighted EC-SNDP and Proper Functions

We start by defining the node-weighted EC-SNDP problem formally. The input consists of an undirected node-weighted graph $G = (V, E)$ (weight of node v is denoted by $w(v)$) and a requirement function $r(uv)$ for each pair of nodes. The goal is to find a minimum node-weighted subgraph H of G such that H contains $r(uv)$ edge-disjoint paths for each pair uv. We use k to denote the maximum requirement. A node u is called a *terminal* if there is some node v such that $r(uv) > 0$. Since any feasible solution has to contain all terminals, we can assume without loss of generality that the weight of every terminal is zero. We define a function $f : 2^V \to \mathbb{Z}_+$ where $f(S) = \max\{r(uv) \mid u \in S, v \notin S\}$. It is well-known that f is a proper function. By Menger's theorem, solving node-weighted EC-SNDP is equivalent to finding a minimum node-weight subgraph H such that $|\delta_H(S)| \geq f(S)$ for all $S \subset V$. (Recall that $\delta_H(S)$ is the set of all edges of H with exactly one endpoint in S.) Our algorithm and analysis extend to the problem of finding a node-weighted subgraph to cover a given proper function. For an arbitrary proper function f we call a node v a terminal if $f(\{v\}) > 0$; maximality of f implies that S contains a terminal if $f(S) > 0$. Again, we can assume without loss of generality that terminals have zero weight, since they are included in any feasible solution.

We alert the reader that, in order to cover the function f, we need to pick a set of *edges*. But since the weights are (only) on the nodes, we pay for a set of nodes and we can use any of the edges in the graph induced by the nodes in order to cover the function. More precisely, our goal is to select a minimum-weight node-induced subgraph $H = G[X]$ that covers f, where X is a subset of nodes of G. We will always assume that X contains the terminals.

As we mentioned, our algorithm for covering f uses the augmentation framework introduced in [18]. Let $f_p : 2^V \to \mathbb{Z}$ be the function such that $f_p(S) = \min\{f(S), p\}$ for each set S. If f is a proper function then f_p is also a proper function. The algorithm performs k phases: for $1 \leq p \leq k$, at the end of phase p, the algorithm has a subgraph H_p that covers f_p. In phase p the algorithm starts with H_{p-1} that covers f_{p-1} and adds some additional nodes to obtain H_p that covers f_p. We can express the underlying optimization problem in phase p as follows.

It is convenient to assume that all of the vertices of H_{p-1} have zero weight; since we have already paid for the nodes, we can set their weight to zero at the beginning of phase p. Let $G'_p = (V, E(G) - E(H_{p-1}))$. (We emphasize that G'_p has all of the nodes of G and that the terminals and vertices of $V(H_{p-1})$ have zero weight.) Our goal is to select a minimum-weight subgraph H of G'_p that covers the following 0-1 function $h_p : 2^V \to \{0, 1\}$. For each set S, we have $h_p(S) = 1$ iff $f(S) \geq p$ and $|\delta_{H_{p-1}}(S)| = p - 1$. The function h_p is known to be an uncrossable function [18]; note that it may no longer be a proper function. We use a primal-dual algorithm to cover h_p in the graph G'_p. A 2-approximation exists for this covering problem for the edge-weighted problem and an $O(\log n)$-approximation for the node-weighted case [15]. We show that the primal-dual algorithm achieves an $O(1)$-approximation for the node-weighted case in planar graphs, however, we emphasize that it only applies for the specific uncrossable functions that arise from proper functions as above; in particular it is important that the chosen subgraphs at the end of each phase are node-induced. We describe and analyze the primal-dual algorithm below and point out the place where we need this restriction.

2.1 A primal-dual algorithm for the augmentation problem

In the following, we fix a phase p of the augmentation framework. Let $h = h_p$ and $G' = G'_p$. Recall that all of the terminals and the vertices selected in the first $p-1$ phases have zero weight. In the following, we use $\Gamma_{G'}(S)$ to denote the set of all vertices v such that $v \notin S$ but there is an edge $uv \in E(G')$ such that $u \in S$. We use a primal-dual algorithm in order to select a subgraph H of G' that covers h. The primal and dual LPs that we use are described below. We remark that the primal LP has unbounded integrality gap for an arbitrary uncrossable function[5]. However, the function h that arises from a proper function f in the augmentation framework has additional properties that allow us to avoid such examples.

Primal:

$$\min \sum_{v \in V} x(v) w(v)$$
$$\text{s.t.} \sum_{v \in \Gamma_{G'}(S)} x(v) \geq h(S) \quad \forall S \subseteq V$$
$$x(v) \geq 0 \quad \forall v \in V$$

Dual:

$$\max \sum_{S \subseteq V} y(S) h(S)$$
$$\text{s.t.} \sum_{S : v \in \Gamma_{G'}(S)} y(S) \leq w(v) \quad \forall v \in V$$
$$y(S) \geq 0 \quad \forall S \subseteq V$$

We omit the constraint $x(v) \leq 1$ in the primal since h is a 0-1 function.

The primal-dual algorithm is a "standard" one in that it is the natural adaptation to the node-weighted setting (as done in [5]) of the primal-dual algorithm for edge-weighted network design formalized by Goemans and Williamson [7]. The algorithm selects a set $X \subseteq V(G')$ of nodes such that the graph $G'[X]$ covers h. Initially, X consists of all vertices that have zero weight. We also maintain a feasible dual solution **y** that is implicitly initialized to zero. We proceed in iterations. Consider iteration i and let X_{i-1} be the set of nodes selected in the first $i-1$ iterations; the set X_0 consists of all zero-weight vertices. A set S is *violated* with respect to X_{i-1} iff $h(S) = 1$ and $\delta_{G'[X_{i-1}]}(S) = \emptyset$. A set S is a *minimal violated set* with respect to X_{i-1} iff S is a violated set and no proper subset of S is violated. Let $\mathcal{C}_i$ denote the collection of all minimal violated sets with respect to X_{i-1}. As shown in [18], no two minimal violated sets of an uncrossable function can intersect; further the collection of minimal violated sets for h arising from proper functions can be computed in polynomial time. Moreover, Lemma 1 below shows that the sets in $\mathcal{C}_i$ are subsets of X_{i-1}. If $\mathcal{C}_i$ is empty,

[5] A simple example is a function h such that there is a single set S such that $h(S) = 1$. Each vertex in S has weight 1, and each vertex in $V - S$ has weight 0. The optimum solution has value 1 since at least one node in S has to be picked but the optimum LP value is 0; note that the value is 0 even if we have integrality constraints.

$G'[X_{i-1}]$ covers h and we return $G'[X_{i-1}]$. Otherwise, we increase the dual variables $\{y(S)\}_{S\in\mathcal{C}_i}$ uniformly until a dual constraint for a vertex v becomes tight, i.e., we have $\sum_{S:v\in\Gamma_{G'}(S)} y(S) = w(v)$; we add v to X. Note that, since the components of $\mathcal{C}_i$ are contained in X_{i-1}, for each minimal violated component $C \in \mathcal{C}_i$, none of the vertices in $\Gamma_{G'}(C)$ are in X_{i-1} and thus it is possible to increase the dual variables $\{y(S)\}_{S\in\mathcal{C}_i}$.

Finally we perform a *reverse-delete* step. Let X be the set of vertices selected by the primal-dual algorithm. We select a subset Y of X as follows. We start with $Y = X$. We order the vertices of Y in the reverse of the order in which they were selected by the primal-dual algorithm. Let v be the current vertex. If $G'[Y - v]$ is a feasible cover for h, we remove v from Y.

The primal-dual algorithm described above is not well-defined for an arbitrary uncrossable function h but the following property holds for those that arise from proper functions. Using the following lemma, we can show that the algorithm is well-defined and it outputs a cover of h in polynomial time.

Lemma 1. *Every minimal violated component $C \in \mathcal{C}_i$ is a subset of X_{i-1}.*
Proof: Consider $C \in \mathcal{C}_i$ and suppose $C \not\subseteq X_{i-1}$. Let $C' = C \cap X_{i-1}$. We observe that $f_p(C \setminus C') = 0$ since all the terminals are in X_{i-1}. Since f_p is maximal, we have $f_p(C) \leq \max\{f_p(C'), f_p(C \setminus C')\} = \max\{f_p(C'), 0\} = f_p(C')$. Since $C \in \mathcal{C}_i$, we have $f_p(C) = p$ and $|\delta_{G[X_{i-1}]}(C)| = p - 1$. Therefore $f_p(C') \geq f_p(C) = p$. Additionally, $\delta_{G[X_{i-1}]}(C) = \delta_{G[X_{i-1}]}(C')$, since $G[X_{i-1}]$ does not have any edges incident to vertices in $V \setminus X_{i-1}$. It follows that C' is violated with respect to X_{i-1}, which contradicts the minimality of C. □

Now we turn our attention to the analysis of the primal-dual algorithm. In the following, we show that the algorithm achieves an $O(1)$ approximation for the augmentation problem when the graph G is from a minor-closed family of graphs $\mathcal{G}$; the constant depends on the family $\mathcal{G}$.

Theorem 2. *If G is a graph from a minor-closed family of graphs $\mathcal{G}$, the weight of the set Y is $O(\mathrm{OPT}_h)$, where OPT_h is the optimum solution to the LP relaxation for covering h.*

The dual variables are grown uniformly in each iteration and the standard primal-dual analysis [7] gives a condition under which the approximation ratio can be upper bounded. This is encapsulated in the lemma below.

Lemma 2. *Let $B_i = Y - X_{i-1}$. Suppose there exists a γ such that, for each iteration i of the primal-dual algorithm, $\sum_{C\in\mathcal{C}_i} |B_i \cap \Gamma_{G'}(C)| \leq \gamma|\mathcal{C}_i|$. Then the weight of Y is at most $\gamma\mathrm{OPT}_h$, where OPT_h is the value of an optimal solution to the LP relaxation.*

The content of the above lemma is the following. Consider the minimal violated sets in $\mathcal{C}_i$. The set $B_i = Y - X_{i-1}$ forms a *node-minimal* set that together with X_{i-1} covers h (minimality follows from the reverse delete step). We are interested in γ, the "average degree"[6] of the components in $\mathcal{C}_i$, with respect to nodes in B_i. In general graphs γ can

[6] Here we are abusing the term slightly and we refer to the ratio $\sum_{C\in\mathcal{C}_i} |B_i \cap \Gamma_{G'}(C)|/|\mathcal{C}_i|$ as the average degree of the components in $\mathcal{C}_i$. One can view the ratio as the average degree of the components if we shrink each of the components in $\mathcal{C}_i$ to a single vertex and we remove parallel edges.

be $\Omega(n)$ in the worst case which does not give a useful bound. However, planar graphs are sparse. Thus one can bound the average degree if one can bound the number of *nodes* in B_i that are adjacent to components in $\mathcal{C}_i$. This was done in [5] for 0-1 proper functions but the case of uncrossable functions is more involved and it is our main technical contribution. Theorem 3 is stated in a general and useful form and proved in Section 3. Assuming the theorem, we finish the analysis as follows. The following lemma upper bounds the number of nodes in B_i that are adjacent to components in $\mathcal{C}_i$.

Lemma 3. *Let B'_i be the set of all vertices $u \in B_i$ such that $u \in \Gamma_{G'}(C)$ for some component $C \in \mathcal{C}_i$. We have $|B'_i| \leq 4|\mathcal{C}_i|$.*

In order to take advantage of the fact that planar and minor-closed graphs are sparse, we need the following technical ingredient. The proof of Lemma 4 follows from the maximality of f_p and it is similar to the proof of Lemma 1.

Lemma 4. *For each component $C \in \mathcal{C}_i$, the graph $G[C]$ is connected.*

In order to finish the average degree argument, we shrink each component $C \in \mathcal{C}_i$ into a single node and we use Lemma 3 and the fact that, for a graph K from a minor-closed family $\mathcal{G}$ there is a constant c' that depends only on the family such that $|E(K)| \leq c'|V(K)|$.

Lemma 5. *Let $B_i = Y - X_{i-1}$. If G is a graph from a minor-closed family of graphs $\mathcal{G}$, we have $\sum_{C \in \mathcal{C}_i} |B_i \cap \Gamma_{G'}(C)| \leq c|\mathcal{C}_i|$, where c is a constant that depends only on the family $\mathcal{G}$.*

Theorem 2 follows from Lemma 2 and Lemma 5. Theorem 2 together with the augmentation framework gives an $O(k)$-approximation for finding a minimum node-weighted subgraph to cover a proper function with maximum requirement k. The result for EC-SNDP is a special case of this result.

Remark 1. For planar graphs, we get a 10-approximation for the augmentation problem and a $10k$-approximation for the EC-SNDP problem. Demaine et al. [5] get a 6-approximation for planar graphs when $k = 1$, and thus our ratio is slightly weaker. Our analysis in Lemma 5 could be tightened in several ways. We believe that the analysis in Theorem 3 and consequently Lemma 3 can be improved to obtain a factor of 3 instead of 4. The analysis uses the maximality of f but not symmetry and hence our results hold for a larger class of functions than proper functions.

3 Proof of Theorem 3

Let $G = (V, E)$ be a graph. Let $h : 2^V \to \{0, 1\}$ be a requirement function. A set S is *violated* if $h(S) = 1$. A set C is a *minimal violated component* of h if C is violated and no proper subset of C is violated. Let H be a subgraph of G. The graph H is a *feasible cover* for h if, for any set $S \subseteq V$ such that $h(S) = 1$, there is at least one edge of H leaving S; in other words, $|\delta_H(S)| \geq h(S)$. We say that H is a *node-minimal* feasible cover for h if, for any vertex $v \in V(H)$, $H - v$ is not a feasible cover for h.

Now we are ready to state our main theorem.

Theorem 3. *Let $h : 2^V \to \{0,1\}$ be an uncrossable function. Let $\mathcal{C}$ be the minimal violated components of h. Let H be a node-minimal feasible cover for h. Let X be the set of all vertices $v \in V(H)$ such that v is not in the union of the components in $\mathcal{C}$ and there is an edge of H connecting v to a component of $\mathcal{C}$. Then $|X| \leq 4|\mathcal{C}|$.*

We devote the rest of this section to the proof of Theorem 3. A basic property of uncrossable functions [18] is stated below.

Lemma 6. *Let h be an uncrossable function. The minimal violated components of h are disjoint. Moreover, if S is a violated set and C is a minimal violated component, S and C do not properly intersect.*

We start with a high-level overview of the proof. The main idea is to pick a subset M of the edges of H such that M is an *edge-minimal* feasible cover for h. Such a minimal cover has nice properties that were pointed out and used in the analysis for edge-weighted problems [18]. More precisely, for each edge $e \in M$, we can pick a "witness set" that is a violated set such that e is the only edge of M that is leaving the set. Moreover, we can pick a family of witness sets, one for each edge of M, such that the family is laminar[7]. This laminar family can be used to upper bound the number of edges of M that are incident to the components of $\mathcal{C}$.

We are interested in analyzing a node-minimal cover H which is not necessarily edge-minimal; there can be a node u that is adjacent to components in $\mathcal{C}$ but it is possible that an edge-minimal cover M does not contain any of the edges connecting u to components of $\mathcal{C}$. Thus we cannot use the witness family to count such vertices. We address this issue by counting them separately using a witness family for a *different* set of edges.

We now turn our attention to the formal proof of the theorem. We refer to the vertices in X as *critical* vertices. We refer to edges connecting a critical vertex to a component $C \in \mathcal{C}$ as *red edges*, and we refer to all other edges of H as *blue edges*.

We define two subsets of edges F_1 and F_2 as follows. We start with $F_1 = E(H)$ and we remove some of the edges as follows. We order the *blue edges* arbitrarily. We consider the blue edges in this order. Let e be the current edge. If $F_1 - e$ is a feasible solution for h, we remove e from F_1. At the end of this process, each red edge is in F_1 and each blue edge in F_1 is necessary to cover h. We refer to critical vertices that are incident to at least one blue edge of F_1 as *regular* vertices; critical vertices that are not regular are referred to as *special* vertices. As we will see shortly, we can use the blue edges in F_1 to upper bound the number of regular vertices.

In order to count the special vertices, we pick a subset F_2 of F_1 as follows. We start with $F_2 = F_1$. We consider the *red edges* of F_2 in some order. Let e be the current edge. If $F_2 - e$ is a feasible cover, we remove e from F_2. We can use the red edges in F_2 to upper bound the number of special vertices. Since H is a node-minimal cover for h, each special vertex is incident to at least one red edge of F_2.

Note that F_2 is an edge-minimal feasible cover for h while F_1 is a feasible cover but is not necessarily edge-minimal. The difficulty is in counting the regular vertices via F_1. We consider the regular and special vertices separately. Theorem 3 follows from the following two lemmas.

[7] A set family $\mathcal{F}$ is laminar iff no two sets in $\mathcal{F}$ properly intersect.

Lemma 7. *The number of regular vertices is at most* $2|\mathcal{C}|$.

Lemma 8. *The number of special vertices is at most* $2|\mathcal{C}|$.

Our counting arguments are based on the laminar witness family approach of Williamson et al. More precisely, we define a witness set as follows.

Definition 1. *Let F be a set of edges. A set $S_e \subseteq V$ is an F-**witness set** for an edge e iff $h(S_e) = 1$ and $\delta_F(S_e) = \{e\}$.*

An F-witness set S_e is a violated set; from Lemma 6 it follows that for each component $C \in \mathcal{C}$, $C \subseteq S_e$ or $C \cap S_e = \emptyset$.

Recall that a family of sets $\mathcal{L}$ is *laminar* if no two sets in $\mathcal{L}$ properly intersect; differently said, for any two sets $A, B \in \mathcal{L}$, either A and B are disjoint or one is contained in the other. The following lemma follows from [18].

Lemma 9 ([18]). *Let F be a feasible cover for an uncrossable function h. Let $M \subseteq F$ be a subset of F such that, for each edge $e \in M$, $F - e$ is not a feasible cover for h. There is a laminar family $\mathcal{L} = \{S_e \mid e \in M\}$ such that S_e is an F-witness set for e.*

Our approach is to use laminar witness families for the blue edges of F_1 and the red edges of F_2 in order to count the regular and special vertices. Before we turn our attention to the counting arguments, we describe some properties of laminar witness families that we need.

We can associate a forest $\mathcal{F}$ with a laminar set family $\mathcal{L}$ as follows. The forest $\mathcal{F}$ has a node ν_S for each set $S \in \mathcal{L}$. We add an edge between ν_A and ν_B iff A is the smallest set in $\mathcal{L}$ that contains B. Let $\mathcal{L} = \{S_e \mid e \in M\}$ be a laminar F-witness family for a set $M \subseteq F$ of edges. Let $\mathcal{T}$ be the tree associated with $\mathcal{L} \cup \{V\}$; we root $\mathcal{T}$ at the node ν_V.

We define the following bijection between the edges of the tree $\mathcal{T}$ and the edges of M. Let e be an edge of M and let S_e be the witness set for e. The node ν_{S_e} has a parent ν_A in $\mathcal{T}$, and we associate the edge $e \in M$ with the edge (ν_A, ν_{S_e}) of $\mathcal{T}$. We say that the edge e *corresponds* to the edge (ν_A, ν_{S_e}). A node ν_S of $\mathcal{T}$ *owns* a vertex $v \in V$ iff S is the smallest set in $\mathcal{L} \cup \{V\}$ that contains v.

Proposition 1. *Let $\mathcal{L} = \{S_e \mid e \in M\}$ be a laminar F-witness family for a set $M \subseteq F$ of edges. Let $\mathcal{T}$ be the tree associated with $\mathcal{L} \cup \{V\}$. For each leaf ν_S of $\mathcal{T}$ there is a distinct component $C \in \mathcal{C}$ such that $C \subseteq S$.*

The following simple observation plays a crucial role in our counting argument.

Proposition 2. *Let $\mathcal{L} = \{S_e \mid e \in M\}$ be a laminar F-witness family for a set $M \subseteq F$ of edges. Let $\mathcal{T}$ be the tree associated with $\mathcal{L} \cup \{V\}$. Let ν_S be a node of $\mathcal{T}$ and let e be an edge of $F \setminus M$. Either both endpoints of e are contained in S or neither endpoint of e is contained in S. In particular, the endpoints of e are owned by the same node of $\mathcal{T}$.*

The following lemma was proved in [18].

Lemma 10 ([18]). *Let $\mathcal{L} = \{S_e \mid e \in M\}$ be a laminar F-witness family for a set $M \subseteq F$ of edges. Let $\mathcal{T}$ be the tree associated with $\mathcal{L} \cup \{V\}$. Let e be an edge of M and let (ν_A, ν_{S_e}) be the edge of $\mathcal{T}$ corresponding to e, where S_e is the witness set for e and ν_A is the parent of ν_{S_e}. Then ν_A owns one endpoint of e and ν_{S_e} owns the other endpoint of e.*

Counting argument for regular vertices. Let $\mathcal{L}_{F_1} = \{S_e \mid e \text{ is a blue edge in } F_1\}$ be a laminar F_1-witness family for the blue edges in F_1 that is guaranteed by Lemma 9. Let $\mathcal{T}_{F_1}$ be the tree associated with the family $\mathcal{L}_{F_1} \cup \{V\}$; we view $\mathcal{T}_{F_1}$ as a rooted tree whose root is the node corresponding to V.

Recall that each regular vertex u is incident to a red edge ur; the edge ur is in F_1, since F_1 contains all the red edges. Additionally, u is incident to a blue edge $ub \in F_1$. Since r is contained in a minimal component of $\mathcal{C}$, it follows from Proposition 2 that the node of $\mathcal{T}_{F_1}$ that owns u also owns a component $C_u \in \mathcal{C}$. Our approach is to charge each regular vertex u in its subtree; more precisely, we charge u to a component $C \in \mathcal{C}$ that is owned by a node in the subtree rooted at the node that owns u and C_u.

We charge each regular vertex u as follows. Recall that there is a blue edge $ub \in F_1$ that is incident to u. Let ν_A and ν_B be the nodes of $\mathcal{T}_{F_1}$ that own u and b, respectively. By Lemma 10, one of ν_A, ν_B is the parent of the other.

Suppose that ν_A is the parent of ν_B. Since each leaf owns a component of $\mathcal{C}$ (from Proposition 1), there is a descendant of ν_B (possibly ν_B itself) that owns a component of $\mathcal{C}$. Let ν_S be the closest such descendant, i.e., a descendant whose distance to ν_B is minimized. (If there are several descendants whose distance to ν_B is minimum, we pick one of them arbitrarily.) We charge u to one of the components of $\mathcal{C}$ that ν_S owns; we refer to this charge as a *subtree charge* (since u is charged in a subtree rooted at a child of the node ν_A that owns u). Since a regular vertex v and its component C_v are owned by the same node of the tree, the components C_v serve as sentinels that ensure that there is at most one subtree charge to each component of $\mathcal{C}$.

Suppose that ν_A is a child of ν_B. We charge u to the component C_u; we refer to this charge as a *parent charge* (since the charge corresponds to the tree edge connecting the node ν_A that owns C to its parent). Since each node has at most one parent edge, there is at most one parent charge to each component of $\mathcal{C}$.

Proposition 3. *There is at most one subtree charge to each component $C \in \mathcal{C}$.*

Proposition 4. *There is at most one parent charge to each component $C \in \mathcal{C}$.*

Proof of Lemma 7: Each component of $\mathcal{C}$ is charged at most twice and thus the number of regular vertices is at most $2|\mathcal{C}|$. □

Counting argument for special vertices. Recall that F_2 is an edge-minimal cover of h. Moreover, a critical vertex v is special only if there is an edge $e \in F_2$ (in fact a red edge) such that e connects v to a minimal violated component C. Thus, the total number of special vertices is upper bounded by $\sum_{C \in \mathcal{C}} |\delta_{F_2}(C)|$. Williamson et al. [18] show that for any edge-minimal cover of an uncrossable function this is upper bounded by $2|\mathcal{C}|$. Thus we can upper bound the number of special vertices by $2|\mathcal{C}|$ which proves Lemma 8. We remark that some of the regular vertices are counted in this step as well, but this can only help us.

References

1. A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.
2. J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. An improved LP-based approximation for Steiner tree. In *Proc. of ACM STOC*, pages 583–592, 2010.
3. J. Cheriyan, S. Vempala, and A. Vetta. Network design via iterative rounding of setpair relaxations. *Combinatorica*, 26(3):255–275, 2006.
4. J. Chuzhoy and S. Khanna. An $O(k^3 \log n)$-approximation algorithm for vertex-connectivity survivable network design. In *Proc. of FOCS*, pages 437–441. IEEE, 2009.
5. E. Demaine, M.T. Hajiaghayi, and P. Klein. Node-weighted Steiner tree and group Steiner tree in planar graphs. *Proc. of ICALP*, pages 328–340, 2009.
6. M.X. Goemans, A.V. Goldberg, S. Plotkin, D.B. Shmoys, E. Tardos, and D.P. Williamson. Improved approximation algorithms for network design problems. In *Proc. of ACM-SIAM SODA*, pages 223–232, 1994.
7. M.X. Goemans and D.P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24:296, 1995.
8. M.X. Goemans and D.P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In *Approximation algorithms for NP-hard problems*, pages 144–191. PWS Publishing Co., 1996.
9. A. Gupta and J. Könemann. Approximation algorithms for network design: A survey. *Surveys in Operations Research and Management Science*, 16(1):3–20, 2011.
10. K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.
11. K. Jain, I. Mandoiu, V.V. Vazirani, and D.P. Williamson. A primal-dual schema based approximation algorithm for the element connectivity problem. *Journal of Algorithms*, 45(1):1–15, 2002.
12. P. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted Steiner trees. *Journal of Algorithms*, 19(1):104–115, 1995.
13. G. Kortsarz and Z. Nutov. Approximating minimum cost connectivity problems. In T. F. Gonzalez, editor, *Handbook on Approximation Algorithms and Metaheuristics*. Chapman and Hall/CRC, 2007.
14. C. Moldenhauer. Primal-dual approximation algorithms for node-weighted Steiner forest on planar graphs. *Proc. of ICALP*, pages 748–759, 2011.
15. Z. Nutov. Approximating Steiner networks with node-weights. *SIAM Journal of Computing*, 39(7):3001–3022, 2010.
16. Z. Nutov. Approximating Steiner network activation problems. *Proc. of LATIN*, 2012.
17. D. Panigrahi. Survivable network design problems in wireless networks. In *Proc. of ACM-SIAM SODA*, 2011.
18. D.P. Williamson, M.X. Goemans, M. Mihail, and V.V. Vazirani. A primal-dual approximation algorithm for generalized Steiner network problems. *Combinatorica*, 15(3):435–454, 1995.