A $(1 + \ln 2)$-Approximation Algorithm for Minimum-Cost 2-Edge-Connectivity Augmentation of Trees with constant radius

Nachshon Cohen and Zeev Nutov

The Open University of Israel,
`nachshonc@gmail.com, nutov@openu.ac.il`

Abstract. We consider the Tree Augmentation problem: given a graph $G = (V, E)$ with edge-costs and a tree T on V disjoint to E, find a minimum-cost edge-subset $F \subseteq E$ such that $T \cup F$ is 2-edge-connected. Tree Augmentation is equivalent to the problem of finding a minimum-cost edge-cover $F \subseteq E$ of a laminar set-family. The best known approximation ratio for Tree Augmentation is 2, even for trees of radius 2. As laminar families play an important role in network design problems, obtaining a better ratio is a major open problem in network design. We give a $(1 + \ln 2)$-approximation algorithm for trees of constant radius. Our algorithm is based on a new decomposition of problem solutions, which may be of independent interest.

1 Introduction

We consider the following problem:

Tree Augmentation
Instance: A graph $G = (V, E)$ with edge-costs $\{c(e) : e \in E\}$, and a tree T on V disjoint to E.
Objective: Find a minimum-cost edge-set $F \subseteq E$ such that $T \cup F$ is 2-edge-connected.

The case when T is a path reduces to the Shortest Path problem (c.f. [7]), and the case when T is a star is equivalent to the Minimum-Cost Edge-Cover problem. Tree Augmentation is equivalent to the problem of finding a minimum cost edge-cover of a laminar set-family; namely, given a graph $G = (V, E)$ with edge-costs and a laminar set-family $\mathcal{L}$ on V, we seek a minimum cost edge-set $F \subseteq E$ such that for every $S \in \mathcal{L}$ there is $uv \in F$ with $u \in S$ and $v \notin S$. The problem is also equivalent to the problem of augmenting a k-edge-connected graph to be $(k + 1)$-connected by adding a minimum cost edge-set, for odd k; this is since when k is odd, the minimum cuts of a k-edge-connected graph form a laminar set-family. In general, laminar set-families play an important role in network design problems; see [6] and surveys in [5, 7, 8], for various network design problems and applications of laminar set-families.

Fredrickson and Jájá [4] gave a 2-approximation algorithm for Tree Augmentation, and showed that it is NP-hard even for trees of radius 2 (the radius $R = R(T)$ of a tree T is $\lceil D/2 \rceil$, where D is the diameter of T). Cheriyan, Jordán and Ravi [1] proved that Tree Augmentation is NP-hard also in the case of unit costs when E is a cycle on the leaves of T, and gave a 4/3-approximation algorithm for this version. They also conjectured that a standard LP-relaxation for the problem has integrality gap less than 2, while in [2] it is shown that the integrality gap is at least 3/2.

Achieving a ratio better than 2 for Tree Augmentation, even for the particular case of unit costs, was posed as a major open problem in connectivity network design in the survey by Khuller [7]. This open question was resolved by Nagamochi [10] that gave a $(1.875 + \varepsilon)$-approximation scheme for this version. The currently best known approximation ratio for Tree Augmentation with unit costs is 3/2, by Even, Kortsarz, and Nutov [3]. Even better ratios are known for unit costs when every edge in E connects two leaves of T [9]: 17/12 for general trees, 11/8 for trees of radius 3, and 4/3 for trees of radius 2.

However, for arbitrary costs, no ratio better than 2 is known, not even for trees of radius 2. We prove the following.

Theorem. Tree Augmentation *admits an algorithm that computes a* $(1 + \ln 2)$-*approximate solution in time* $n^{3^{h(T)}} \cdot poly(n)$, *where* $n = |V|$ *and* $h(T)$ *is the radius of* T. *In particular,* Tree Augmentation *on trees with radius bounded by a constant admits a* $(1 + \ln 2)$-*approximation algorithm.*

Our algorithm is based on a new decomposition of Tree Augmentation feasible solutions, which may be of independent interest.

2 Proof of the Theorem

2.1 A local replacement algorithm for Set-Cover

We start by describing a generic local-replacement algorithm for the following well known problem.

Set-Cover
Instance: A collection E of subsets of a groundset T with costs $\{c(e) : e \in E\}$.
Objective: A minimum-costs subcollection $F \subseteq E$ such that $T \subseteq \bigcup_{e \in F} e$.

Definition 1. *Given an instance of* Set-Cover *and* $S, F \subseteq E$ *let*

$$R_F(S) = \left\{ f \in F : f \subseteq \bigcup_{e \in S} e \right\} .$$

We say that a set-family $\mathcal{S} \subseteq 2^E$ overlaps $F \subseteq E$ *if* $\bigcup_{S \in \mathcal{S}} R_F(S) = F$, *namely, if for every* $f \in F$ *there is* $S \in \mathcal{S}$ *with* $f \in R_F(S)$. *We say that* $F \subseteq E$ *is* q-overlapped *by* $F^* \subseteq E$ *if* F^* *admits a partition* $\mathcal{S}$ *into parts of size at most* q *that overlaps* F.

The following statement is immediate.

Fact 1 *Let $F \subseteq E$ be a feasible solution. Then $F \setminus R_F(S) \cup S$ is also a feasible solution for any $S \subseteq E$.*

The following technique was introduced by Zelikovsky in [12] in the context of the Steiner Tree problem. We reinterpret Zelikovsky's result in a more general Set-Cover setting.

Lemma 1. *Suppose that for an instance of* Set-Cover *we are given an α-approximate solution F_0 that is q-overlapped by some optimal solution F^*. Then the problem admits a $(1+\ln\alpha)$-approximation in $m^q \cdot poly(m)$ time, where $m = |E|$.*

Proof. Let $\mathcal{S}$ be a partition of F^* into parts of size at most q each that overlaps F. Clearly, $\sum_{S\in\mathcal{S}} c(S) = \mathsf{opt}$. For any $F \subseteq F_0$ we have $\sum_{S\in\mathcal{S}} c(R_F(S)) \geq c(F)$, hence by an averaging argument there exist $S \in \mathcal{S}$ such that

$$\frac{c(R_F(S))}{c(F)} \geq \frac{c(S)}{\mathsf{opt}} \; . \tag{1}$$

The algorithm is as follows.

```
Initialization: $I \leftarrow \emptyset$, $F \leftarrow F_0$.
While $F \neq \emptyset$ do:
        Find $S \subseteq E \setminus I$ with $|S| \leq q$ satisfying (1).
        - If $c(R_F(S)) \leq c(S)$ then STOP and Return $F \cup I$;
        - Else do $F \leftarrow F \setminus R_F(S)$ and $I \leftarrow I \cup S$.
EndWhile
Return $F \cup I$.
```

The time complexity is straightforward, and feasibility follows from Fact 1. We prove the approximation ratio. Let S_i be the set picked at iteration i and let F_i be the set stored in F after iteration i. When the algorithm stops, either $F = \emptyset$ or $c(R_F(S)) \leq c(S)$; note that the later case implies that $c(F) \leq \mathsf{opt}$, by (1). In both cases, there exist an iteration j such that $c(F_{j-1}) > \mathsf{opt} \geq c(F_j)$. Hence there exists $\theta \in (0,1]$ such that $c(F_{j-1}) - \theta \cdot c(R_{F_{j-1}}(S_j)) = \mathsf{opt}$. Note that $c(F \cup I)$ is decreasing after each iteration, hence the cost of the solution produced by the algorithm is at most

$$c(F \cup I) \leq c(F_{j-1}) - \theta \cdot c(R_{j-1}(S_j)) + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j) = \mathsf{opt} + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j) \; .$$

We can assume w.l.o.g that $c(S) \geq 1$ for all $S \in E$. Since at each iteration F_i, S_i satisfy (1), we have

$$c(F_{i+1}) = \left(1 - \frac{c\left(R_{F_i}(S_i)\right)}{c(F_i)}\right) c(F_i) \leq \left(1 - \frac{c(S_i)}{\mathsf{opt}}\right) c(F_i) \leq \left(1 - \frac{1}{\mathsf{opt}}\right)^{c(S_i)} c(F_i) \; .$$

Applying the same argument for iteration j we get

$$\begin{aligned}\mathsf{opt} = c(F_{j-1}) - \theta c(R_{F_{j-1}}(S)) &\leq \left(1 - \frac{1}{\mathsf{opt}}\right)^{\theta c(S_j)} c(F_{j-1}) \\ &\leq \left(1 - \frac{1}{\mathsf{opt}}\right)^{\sum_{i=1}^{j-1} c(S_i) + \theta c(S_j)} \cdot c(F_0) \\ &\leq \left(1 - \frac{1}{\mathsf{opt}}\right)^{\sum_{i=1}^{j-1} c(S_i) + \theta c(S_j)} \cdot \alpha \cdot \mathsf{opt}\end{aligned}$$

This implies

$$c(F \cup I) \leq \mathsf{opt} + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j) \leq \mathsf{opt} + \ln \alpha \cdot \mathsf{opt} = (1 + \ln \alpha) \cdot \mathsf{opt} \ .$$

The lemma follows. □

2.2 Algorithm for Tree Augmentation

Given an instance of Tree Augmentation, we will call the edges in E *links*, to distinguish them from the edges of the tree T. For $u, v \in V$ let T_{uv} denote the (unique) uv-path in T. We say that a link $uv \in E$ *covers* an edge $e \in T$ if $e \in T_{uv}$. It is well known and easy to see that $F \subseteq E$ is a feasible solution to an instance of Tree Augmentation if, and only if, F covers all the edges of T. Hence Tree Augmentation is equivalent to the problem of finding a minimum-cost link-set $F \subseteq E$ that covers all the edges of T; namely, Tree Augmentation can be casted as a Set-Cover problem with groundset being the edge-set of T, and the collection of sets obtained by replacing every link $e = uv \in E$ by the set $T_e = T_{uv}$ of cost $c(e)$. In this setting, the restriction of Definition 1 to Tree Augmentation can be formulated as follows.

Definition 2. *Given an instance of* Tree Augmentation *and* $S, F \subseteq E$ *let*

$$R_F(S) = \left\{ f \in F : T_f \subseteq \bigcup_{e \in S} T_e \right\} \ .$$

We say that a set-family $\mathcal{S} \subseteq 2^E$ *of links* overlaps *a set* $F \subseteq E$ *of links if* $\bigcup_{S \in \mathcal{S}} R_F(S) = F$*, namely, if for every* $f \in F$ *there is* $S \in \mathcal{S}$ *with* $f \in R_F(S)$*. We say that* $F \subseteq E$ *is* q-overlapped *by* $F^* \subseteq E$ *if* F^* *admits a partition* $\mathcal{S}$ *into parts of size at most* q *that overlaps* F*.*

To apply Lemma 1, we would like to show that given an instance of Tree Augmentation, one can find in polynomial time a 2-approximate solution F_0 that is q-overlapped by some optimal solution F^*, for $q = 3^{h(T)-1}$. However, for this to be true, we need to apply a certain transformation to modify the instance, as is explained bellow.

Definition 3. *A link $u'v'$ is a* shadow *of a link uv if $T_{u'v'}$ is a subpath of T_{uv}. We say that F is a* shadow-minimal cover *of T if for every link $uv \in F$, removing uv or replacing uv by any proper shadow of uv results in an edge-set that is not a cover of T.*

Given an instance of Tree Augmentation, we can obtain an equivalent instance by applying "shadow completion": for every existing link $e \in E$, add all its shadows, of cost $c(e)$ each (if parallel links arise, then for every inclusion-maximal set of parallel links we keep only the cheapest one). Note that shadow completion does not affect the optimal solution value, since every shadow can be replaced by some original link covering all edges of T covered by the shadow. Hence we can assume the following.

Assumption 1. *If $uv \in E$ and $u', v' \in T_{uv}$ then $u'v' \in E$ and $c(u'v') \leq c(uv)$.*

In the next section we will prove the following statement.

Lemma 2. *Under Assumption 1,* Tree Augmentation *admits a polynomial time algorithm that finds a 2-approximate solution F_0 that is $3^{h(T)-1}$-overlapped by any feasible solution F^*.*

Lemmas 1 and 2 easily imply the Theorem; note that the running time is bounded by

$$m^{3^{h(T)-1}} poly(n) \leq \left(n^2\right)^{3^{h(T)-1}} poly(n) \leq n^{3^{h(T)}} poly(n) \ ,$$

as claimed.

In the rest of this paper we prove Lemma 2.

2.3 Proof of Lemma 2

Root the tree at a center s of T (so $|T_{sv}| \leq h(T)$ for every $v \in V$). This defines an ancestor-descendant relation (partial order) on the nodes of T, where u is an *ancestor* of v (and v is a descendant of u) if $u \in T_{vs}$; if also $uv \in T$ then v is a *child* of u and u is the *parent* of v.

Definition 4. *We say that a link is an* up-link *if one of its endnodes is an ancestor of the other. We say that a cover F of T is an* up-cover *of T if every link in F is an up-link.*

The following statement is known, and was implicitly proved in [4]. For completeness of exposition, we provide a proof-sketch.

Lemma 3. *Under Assumption 1, for any cover F^* of T there exists an up-cover F of T such that $c(F) \leq 2c(F^*)$. Furthermore, a minimum-cost up-cover can be computed in polynomial time. Consequently, there exists a shadow-minimal up-cover F_0 of cost at most 2opt, and such cover be computed in polynomial time.*

Proof. Let F be obtained from F^* by replacing every link $e = uv \in F$ by the two links ua, va, where a is the least common ancestor of u, v in T. By Assumption 1, the links ua, va exist, and the cost of each of them is at most $c(uv)$. Hence $c(F) \leq 2c(F^*)$. It is easy to see that F is a feasible solution, which concludes the proof of the first statement of the lemma. The problem of computing a minimum-cost up-cover is reduced to the Minimum-Cost Arborescence problem (which is solvable in polynomial time, c.f. [11]) as follows. It is known and easy to see [4] that F is an-up cover of T if, and only if, the directed graph obtained by directing the edges of T towards the root s and directing the links in F from ancestors to descendants, has a path from s to every other node. Hence to compute a minimum-cost up-cover do the following. Direct the edges of T towards the root, remove all links that are not up-links, and direct all up-links from ancestors to descendants. Then in the obtained directed graph compute a minimum-cost arborescence. The set of links (that are not edges of T) in the underlying graph of the computed arborescence, is a minimum-cost up-cover of T.

The last statement of the lemma follows by observing that under Assumption 1 we can replace any up-cover F of T by a shadow-minimal up-cover F_0 of no greater cost. □

Lemma 4. *Let uv, xy be up-links such that T_{uv} and T_{xy} have an edge in common but none of them is a subpath of the other. Then one of uv, xy has a proper shadow that together with the other link they cover all edges in $T_{uv} \cup T_{xy}$.*

Proof. As T_{uv} and T_{xy} intersect, either y is an ancestor of v, or v is an ancestor of y. Assume w.l.o.g. that y is an ancestor of v. Let z be the lowest (farthest from root) node that T_{xy}, T_{uv} have in common. Than uz is a proper shadow of uv, and $\{xy, uz\}$ cover all edges in $T_{uv} \cup T_{xy}$. As the link xy covers an edge on T_{uv}, one of x, y, say x, must be an internal node of the path T_{uv}. Since none of T_{uv}, T_{xy} is a subpath of the other, y must be either a proper ancestor of v or a proper descendant of u. In the former case vy is a proper shadow of xy and $\{uv, vy\}$ cover all edges in $T_{uv} \cup T_{xy}$. In the latter case yu is a proper shadow of xy and $\{uv, yu\}$ cover all edges in $T_{uv} \cup T_{xy}$. □

From Lemma 4 we deduce the following.

Corollary 1. *Let F be a shadow-minimal up-cover of T. Then every edge of T is covered by a unique link in F.*

Definition 5. *The* height $h(v)$ of a node $v \in V$ *is the distance from v to the farthest descendant of v in T (note that by our choice of s, $h(s)$ is the radius $h(T)$ of T). For $u, v \in V$ let $\mathsf{lca}(u, v)$ denote the least common ancestor of u and v in T. For a link $e = uv$ let $\mathsf{lca}(e) = \mathsf{lca}(u, v)$. The* height $h(e)$ of a link $e = uv$ *is $h(\mathsf{lca}(e))$.*

In the rest of this section we will prove the following statement, that together with Lemma 3 and Corollary 1 implies Lemma 2.

Lemma 5 (The Decomposition Lemma). *Let F^* be a cover and F an up-cover of a tree T rooted at s, such that every edge of T is covered by a unique link in F. Then F is $3^{h(s)-1}$-overlapped by F^*, namely, there exist a partition $\mathcal{S}$ of F^* into parts of size at most $3^{h(s)-1}$ each, such that for every $f \in F$ there exists $S \in \mathcal{S}$ with $f \in R_F(S)$.*

The bound $3^{h(s)-1}$ in Lemma 5 is tight, as is shown in the next section. Namely, for any integer $h \geq 1$, there exists a tree T rooted at s and F, F^* as in Lemma 5, such any partition of F^* that overlaps F has a part of size at least $3^{h(s)-1}$.

In the rest of this section we prove Lemma 5. Define an auxiliary directed graph $J = (V_J, E_J)$ as follows. For every $f \in F$, let u^f, v^f be the endnodes of f, where u^f is a descendant of v^f. Let $I^f \subseteq F^*$ be some inclusion minimal cover of the unique $u^f v^f$-path P^f in T. Let $k(f) = |I^f|$ and let $e^f_1, e^f_2, ...e^f_{k(f)}$ be an ordering of I^f obtained as follows, see Figure 1. For $i = 1, \ldots, k(f)$, e^f_i is the link in I^f that covers the lowest (farthest from the root) edge of P^f not covered by $\{e^f_1, \ldots, e^f_{i-1}\}$. The node set of J is $V_J = F^*$ and the edge set of J is $E_J = \{e^f_{i+1} e^f_i : f \in F, 1 \leq i \leq k(f) - 1\}$. We will prove:

Lemma 6. *J is a collection of node-disjoint arborescences with at most $3^{h(s)-1}$ nodes each.*

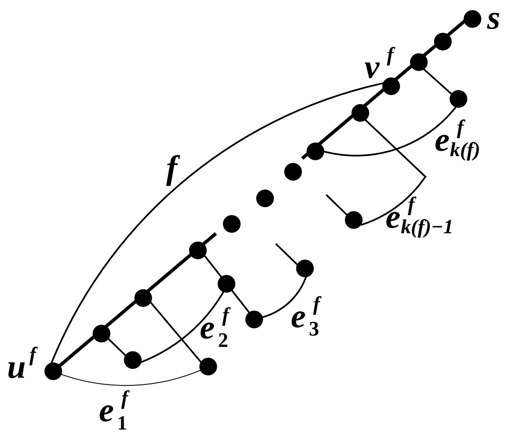

Fig. 1. Illustration to the definition of I^f and Fact 2.

Lemma 5 easily follows from Lemma 6. The desired partition $\mathcal{S}$ of F^* is the one defined by the arborescences of the auxiliary graph J. Note that by the construction, for every $f \in F$ the ordering $e^f_1, e^f_2, ...e^f_{k(f)}$ of I^f forms a directed path in J. Hence for every $f \in F$, I^f belongs to the same part (arborescence), which defines the part $S \in \mathcal{S}$ such that $f \in R_F(S)$.

In the rest of this section we prove Lemma 6. The following fact stems from the definition of I^f (see Figure 1).

Fact 2 *Let $f \in F$. Then for every $i = 1, \ldots, k(f)$, the set of edges from P^f covered by the link-set $\{e_1^f, \ldots, e_i^f\}$ is exactly the set of edges on the $u^f v_i^f$-path in T, where $v_i^f = \mathsf{lca}(e_i^f)$.*

Thus from the minimality of I^f we have the following.

Corollary 2. *Let $f \in F$ with $k(f) \geq 2$. Then for every $1 \leq i \leq k(f) - 1$, v_i^f is an inner node of P^f, $h(e_i^f) < h(e_{i+1}^f)$, and both f and e_{i+1}^f cover the parent edge of e_i^f (the edge between v_i^f and its parent) in T.*

Lemma 7. *J is acyclic and $\deg_J^{in}(e) \leq 1$ for every $e \in F^*$. Thus J is a collection of node-disjoint arborescences.*

Proof. From Corollary 2 it easily follows that J is acyclic. We will prove that $\deg_J^{in}(e) \leq 1$ for every $e \in F^*$. Suppose to the contrary that there are two edges $e'e, e''e \in E_J$ entering e in J. By the definition of J, there are two distinct links $f', f'' \in F$ such that $e = e_{i'}^{f'} \in I_{f'}$ and $e = e_{i''}^{f''} \in I_{f''}$. By Corollary 2, e has a parent edge in T, and both f', f'' cover the parent edge of e in T. This contradicts the assumption that every edge of T is covered by a unique link in F. □

Lemma 8. *For every $e \in F^*$, no three neighbors of e in J have the same height.*

Proof. Let e', e'' be two distinct neighbors of e in J with $h(e') = h(e'')$. By the definition of J, there exist distinct $f', f'' \in F$ such that $e' = e_{i'}^{f'}$, $e'' = e_{i''}^{f''}$, and $e = e_{i'+1}^{f'} = e_{i''+1}^{f''}$. By Corollary 2, f' covers the parent edge of e' and f'' covers the parent edge of e''. Since f' and f'' are distinct and since every edge of T is covered by a unique link in F, the parent edges of e' and e'' are also distinct Since e', e'' have the same height, the parent edges of e' and e'' also have the same height. By Fact 2, e covers the parent edge of each of e', e''. Hence, e cannot have a third neighbor in J, since then e will cover three distinct edges in T of the same height, but no link can cover three distinct edges in T of the same height. □

Corollary 3. *For $e \in F^*$ let A_e be the set of nodes in J reachable from e. Then $|A_e| \leq 3^{h(e)-1}$. Thus every arborescence in J has at most $3^{h(s)-1}$ nodes.*

Proof. We prove the statement by induction on $h(e)$. If $h(e) = 1$ then e has no neighbors in J, by Corollary 2. Hence $|A_e| = 1$ and $3^{h(e)-1} = 3^0 = 1$, and the statement is valid in this case. Suppose that $h(e) \geq 2$ and that any arborescence A' in J with root e' and $h(e') \leq h(e) - 1$ has at most $3^{h(e')-1}$ nodes.

Let e' be a neighbor of e in J. By Corollary 2, $h(e') \leq h(e) - 1$. Hence by the induction hypothesis, $|A_{e'}| \leq 3^{h(e')-1}$. By Lemma 8, no three distinct neighbors of e have the same height. Thus we get:

$$|A_e| \le 1 + 2 \cdot \sum_{i=1}^{h(e)-1} 3^{i-1} = 1 + 2 \cdot \frac{1 - 3^{h(e)-1}}{1-3} = 1 + 3^{h(e)-1} - 1 = 3^{h(e)-1} \ .$$

□

This finishes the proof of Lemma 6, and thus also the proof of the Theorem is now complete.

2.4 A tight example for the Decomposition Lemma

Here we show for any integer $h \ge 1$, there exists a tree T rooted at s and F, F^* as in Lemma 5, such any partition of F^* that overlaps F has a part of size at least $3^{h(s)-1}$.

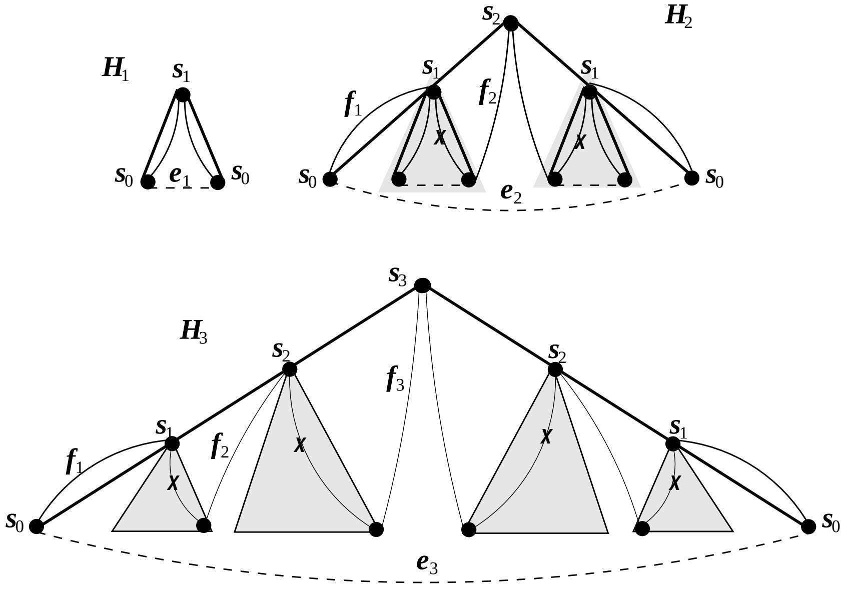

Fig. 2. Construction of a tight example for Lemma 5 with $h(s) = 1, 2, 3$. T-edges are shown by bold lines, F^*-edges are shown by dashed lines, and F-edges are shown by thin lines.

In our example we will show that any partition $\mathcal{S}$ of F^* that overlaps F must consist of one part $\mathcal{S} = \{F^*\}$. Define a sequence graphs H_i, each with spanning tree T_i rooted at s_i, and two edge subsets F_i, F_i^* as follows.

H_1 is depicted in Figure 2.

To obtain H_2 do the following (see Figure 2). Take a path $P_2 = s_2 - s_1 - s_0$ of T-edges, add the F-link $f_1 = s_0 s_1$, and attach a copy of H_1 via the root to s_1. This copy has an F-link from a leaf to s_1, and we replace the endnode s_1 of this link by s_2, to obtain the link f_2. Take another copy of the obtained graph and identify the node s_2 of both copies. Finally, add the F^*-link e_2 that connects the two copies of s_0 on copies of P_3, as depicted in Figure 2. The edges of T_2 are the T_1 edges in the H_1 copies plus the edges in the P_2 copies. The F^*-links are the union of the F^*-links in the copies of H_1 and the link e_2; alternatively, these are the links in H_2 that connect two leaves of T_2. The remaining links are the F-links, and these are the up-links in H_2.

In general, the construction is recursive as follows. Given $H_1, \ldots, H_{i-1}$, to obtain H_i do the following (see Figure 2 for the case $i = 3$). Take a path $P_i = s_i - s_{i-1} - \cdots - s_0$ of T-edges, add the F-link $f_1 = s_0 s_1$, and for every $j = 1, \ldots, i-1$ attach a copy of H_j via the root to s_j. Each copy H_j has an F-link from a leaf to s_j, and we replace the endnode s_j of this link by s_{j+1}, to obtain the link f_{j+1}. Take another copy of the obtained graph and identify the node s_i of both copies. Finally, add the F^*-link e_i that connects the two copies of s_0 on copies of P_i, as depicted in Figure 2. The edges of T_i are the T_j-edges in the copies H_j plus the edges in the P_i copies. The F^*-links are the union of the F^*-links in the H_j-copies and the link e_i; alternatively, these are the links in H_i that connect two leaves of T_i. The remaining links are the F-links, and these are the up-links in H_i.

It is not hard to verify that $|F_i^*| = 3^{i-1}$ and that $|F_i| = 2|F_i^*| = 2 \cdot 3^{i-1}$. Now let $\mathcal{S}$ be a partition of F_i^* that overlaps F, and let $S \in \mathcal{S}$ be the part that contains e_i. We claim that $S = F^*$. For every $j = 2, \ldots, i$, the tree edge $s_{j-1}s_j$ is covered by the link f_j, and e_i is the only link in F^* that covers this edge. This implies that we must have $f_j \in R_F(S)$. Using a similar argument, we can conclude that in any H_j-copy, the link e_j must be in S. Thus using induction we can show that in each H_j-copy, all the F^* links belong to S. Consequently, all the F^*-links in H_i belong to F^*, as claimed.

References

1. J. Cheriyan, T. Jordán, and R. Ravi. On 2-coverings and 2-packing of laminar families. In *ESA*, pages 510–520, 1999.
2. J. Cheriyan, H. Karloff, R. Khandekar, and J. Könemann. On the integrality ratio for tree augmentation. *Operations Research Letters*, 36(4):399–401, 2008.
3. G. Even, G. Kortsarz, and Z. Nutov. A 1.5 approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *Information Processing Letters*, 111(6):296–300, 2011.
4. G. N. Fredrickson and J. Jájá. On the relationship between the biconnectivity augmentation and traveling salesman problem. *Theorethical Computer Science*, 19(2):189–201, 1982.
5. M. Goemans and D. Williamson. *The primal dual method for approximation algorithms and its applications to network design problems,* Ch. 4 in *Approximation Algorithms for NP-hard problems,* D. S. Hochbaum Ed., pages 144-191. PWS, 1995.

6. K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.
7. S. Khuller. *Approximation algorithms for for finding highly connected subgraphs,* Ch. 6 in *Approximation Algorithms for NP-hard problems,* D. S. Hochbaum Ed., pages 236-265. PWS, 1995.
8. G. Kortsarz and Z. Nutov. *Approximating minimum cost connectivity problems,* Ch. 58 in *Approximation Algorithms and Metahueristics,* T. F. Gonzales ed.,. CRC, 2007.
9. Y. Maduel and Z. Nutov. Covering a laminar family by leaf to leaf links. *Discrete Applied Mathematics*, 158(13):1424–1432, 2010.
10. H. Nagamochi. An approximation for finding a smallest 2-edge connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics*, 126:83–113, 2003.
11. A. Schrijver. *Combinatorial Optimization, Polyhedra and Efficiency.* Springer-Verlag Berlin, Heidelberg New York, 2004.
12. A. Zelikovsky. Better approximation bounds for the network and euclidean steiner tree problems. Technical report, 1995.