Approximation Algorithms for Connectivity Augmentation Problems

Vera Traub*

Current Themes of Discrete Optimization: Boot-camp for early-career researchers, ICERM 2023

1 Introduction

A basic property of an (undirected) graph is its *edge-connectivity*. This is the minimum number $k \in \mathbb{Z}_{\geq 0}$ for which it is *k-edge-connected*.

Definition 1.1. *Let $k \in \mathbb{Z}_{\geq 0}$. An undirected graph $G = (V, E)$ is* k-edge-connected *if $|\delta_E(U)| \geq k$ for every nonempty set $U \subsetneq V$.*

A graph with at least two vertices is k-edge-connected for $k \in \mathbb{Z}_{\geq 1}$ if and only if it remains connected after the removal of any set of (at most) $k-1$ edges. Moreover, by Menger's theorem, a graph is k-edge-connected if and only if for every two vertices $u, v \in V$, the graph contains k edge-disjoint u-v paths.

Minimum Weight k-Edge-Connected Spanning Subgraph Problem (k-WECSS)

Input: An instance $(G = (V, E), c)$ consists of

- a tree $G = (V, E)$,
- a nonnegative cost function $c\colon E \to \mathbb{R}_{\geq 0}$.

Task: Find a minimum-cost set $F \subseteq L$ such that $G = (V, F)$ is k-edge-connected (or decide that no such F exists).

k-WECSS is NP-hard for every $k \geq 2$. In fact, one can even show that there exists a constant $\varepsilon > 0$ such that, unless $\mathsf{P} = \mathsf{NP}$, there exists no $(1 + \varepsilon)$-approximation algorithm for k-WECSS.

Definition 1.2. *Let $\alpha \geq 1$. An* α-approximation algorithm *for a minimization problem is a polynomial-time algorithm that computes a feasible solution of cost at most α times the cost of an optimum solution (or decides that no feasible solution exists).*

Khuller and Vishkin [KV94] gave a 2-approximation algorithm for k-WECSS. Jain [Jai01] gave a 2-approximation algorithm for a much more general problem, known as Survivable Network Design.

*Research Institute for Discrete Mathematics, University of Bonn. Email: traub@dm.uni-bonn.de.

Survivable Network Design Problem (SNDP)

Input: An instance $(G = (V, E), c)$ consists of

- an undirected graph $G = (V, E)$,
- a nonnegative cost function $c\colon E \to \mathbb{R}_{\geq 0}$, and
- connectivity requirements $r_{vw} \in \mathbb{Z}_{\geq 0}$ for all $v, w \in V$.

Task: Find a minimum-cost set $F \subseteq L$ such that for all $v, w \in V$, the graph $G = (V, F)$ contains at least r_{vw} edge-disjoint v-w paths (or decide that no such F exists).

Jain's algorithm works also in the slightly more general setting where multiple copies of an edge can be included in F and lower and upper bounds on the number of copies are given for every edge. SNDP generalizes many important network design problems. This includes for example k-WECSS (by setting $r_{vw} = k$ for all $v, w \in V$) and the Steiner tree problem (by setting $r_{vw} = 1$ if v and w are terminals and setting $r_{vw} = 0$ otherwise).

Steiner Tree Problem

Input: An instance $(G = (V, E), c)$ consists of

- an undirected graph $G = (V, E)$,
- a nonnegative cost function $c\colon E \to \mathbb{R}_{\geq 0}$, and
- a set $T \subseteq V$ of terminals.

Task: Find a minimum-cost set $F \subseteq L$ such that all terminals are connected to each other in (V, F) (or decide that no such F exists).

Jain's algorithm uses the powerful technique of iterative rounding. It works with the following LP relaxation of SNDP:

$$\begin{array}{lrcll} \min & \displaystyle\sum_{e \in E} c(e) \cdot x_e & & & \\ \text{s.t.} & \displaystyle\sum_{e \in \delta(U)} x_e & \geq & r_{vw} & \text{for } v, w \in V \text{ and } v \in U \subseteq V \setminus \{w\} \\ & x & \in & [0,1]^E & \end{array} \tag{1}$$

The LP (1) can be solved in polynomial time using the ellipsoid method. (Here we use that given a vector $x \in \mathbb{R}^E_{\geq 0}$, we can in polynomial time either verify that x is a feasible LP solution or find a violated constraint using an algorithm for minimum s-t cuts.)

Algorithm 1: Jain's 2-approximation algorithm for SNDP

Let x^* be an optimal extreme point solution to the LP (1).
(If the LP is infeasible, the given instance of SNDP is infeasible.)

while x^* *is not integral* **do**
  For all $e \in E$ with $x^*_e \geq \frac{1}{2}$, add the constraint $x_e = 1$ to the LP (1).
  Let x^* be an optimal extreme point solution to the resulting LP.

Return $\{e \in E : x^*_e = 1\}$.

The key lemma in the analysis of Jain's algorithm is the following.

Lemma 1.3 ([Jai01])**.** *In every iteration of Algorithm 1, there exists a fractional variable x^*_e with $x^*_e \geq \frac{1}{2}$.*

Corollary 1.4. *Algorithm 1 terminates after $O(|E|)$ iterations.*

Proof. By Lemma 1.3, the number variables not fixed to 1 decreases strictly in every iteration. □

The key part of the analysis of Jain's algorithm is Lemma 1.3. Assuming that the algorithm terminates, it is not very hard to prove that it has an approximation ratio of 2.

Theorem 1.5 ([Jai01])**.** *Algorithm 1 is a 2-approximation algorithm for SNDP.*

We omit the proof of Lemma 1.3 and Theorem 1.5 here and refer to [Jai01] for details. (A proof can also be found in Chapter 20 of [KV18].)

It is a major open question whether Jain's algorithm can be improved. This is open even for many important special cases, including k-WECSS, even for $k = 2$, and the special case where all connectivity requirements r_{vw} are in $\{0, 1\}$. The latter special case is known as the Steiner Forest Problem. For some other special cases of SNDP, better-than-2 approximation algorithms are known. This includes the Steiner tree problem, where the currently best known approximation ratio is $\ln(4)+\varepsilon \approx 1.386$ (for any $\varepsilon > 0$) [BGRS13; GORZ12; TZ22b]). Moreover, very recently the first better-than-2 approximation algorithm has been found for the Weighted Connectivity Augmentation Problem [TZ22a].

Weighted Connectivity Augmentation Problem (WCAP)

Input: An instance $(G = (V, E), L, c)$ consists of

- an undirected graph $G = (V, E)$,
- a link set $L \subseteq \binom{V}{2}$, and
- a nonnegative cost function $c\colon L \to \mathbb{R}_{\geq 0}$.

Task: Let k be the edge-connectivity of G. Find a minimum-cost set $F \subseteq L$ such that the graph $G = (V, E \dot\cup F)$ is $(k + 1)$-edge-connected (or decide that no such F exists).

Note that a feasible WCAP solution exists if and only if $(V, E \dot\cup L)$ is $(k+1)$-edge-connected. This can be checked in polynomial time and thus we will in the following assume that all instances of WCAP that we encounter admit a feasible solution.

2 Weighted Tree Augmentation

The *Weighted Tree Augmentation Problem (WTAP)* is the special case of the Weighted Connectivity Augmentation Problem (WCAP), where the graph G is a tree.

Weighted Tree Augmentation Problem (WTAP)

Input: An instance $(G = (V, E), L, c)$ consists of

- a tree $G = (V, E)$,
- a link set $L \subseteq \binom{V}{2}$, and
- a nonnegative cost function $c\colon L \to \mathbb{R}_{\geq 0}$.

Task: Find a minimum-cost set $F \subseteq L$ such that $G = (V, E \dot\cup F)$ is 2-edge-connected (or decide that no such F exists).

The *Tree Augmentation Problem (TAP)* is the special case of WTAP, where all links have cost one, i.e., $c(\ell) = 1$ for all $\ell \in L$.

Note that a feasible WTAP solution exists if and only if $(V, E \dot\cup L)$ is 2-edge-connected. This can be checked in polynomial time and thus we will in the following assume that all instances of WTAP that we encounter admit a feasible solution.

2.1 Hardness of TAP

In this section we show that TAP is NP-hard even if the tree G has diameter 4 and all endpoints of links are leaves of G. We will use a reduction from the 3-Dimensional Matching Problem (3D-Matching), which has be shown to be NP-complete by Papadimitriou and Steiglitz [PS82]. (A proof can also be found in [KV18].) An instance of 3D-Matching consists of three disjoint sets X, Y, Z of equal cardinality and a set $\mathcal{T} \subset X \times Y \times Z$. The task is to decide if there exists a subset $\mathcal{T}^* \subseteq T$ of triples such that every element of $X \dot\cup Y \dot\cup Z$ is contained in exactly one of the triples in $\mathcal{T}^*$. Our proof of the hardness of TAP follows [KKL04].

Theorem 2.1. *TAP is* NP*-hard even when restricted to instances* (G, L, c) *where the tree has diameter* 4 *and all endpoints of links in* L *are leaves of* G*.*

Proof. Given an instance of 3D-Matching consisting of nonempty sets X, Y, Z and tuples $T \subseteq X \times Y \times Z$, we construct an instance of TAP as follows. The vertex set V of the tree G consists of the set $X \dot\cup Y \dot\cup Z$, vertices $t_{xyz}, \bar{t}_{xyz}$ for all $(x, y, z) \in \mathcal{T}$, and a root vertex r. The edge set E of the tree G consists of the edges $\{x, r\}$ for all $x \in X \dot\cup Y \dot\cup Z$ and edges $\{z, t_{xyz}\}, \{z, \bar{t}_{xyz}\}$ for all $(x, y, z) \in \mathcal{T}$. Note that the tree G has diameter 4. The link set L consists of links $\{x, t_{xyz}\}, \{y, \bar{t}_{xyz}\}$, and $\{t_{xyz}, \bar{t}_{xyz}\}$ for all $(x, y, z) \in \mathcal{T}$. See Fig. 1 for an illustration.

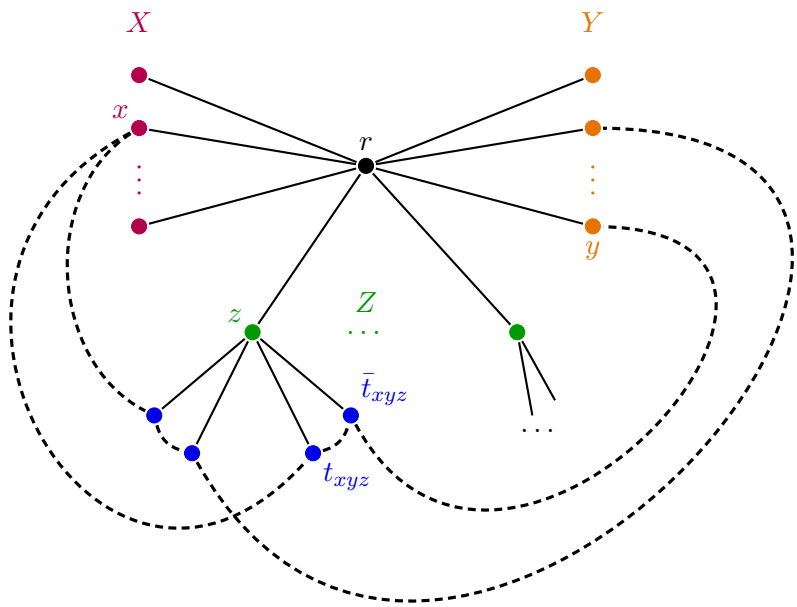

Figure 1: Illustration of the construction of the TAP instance in the proof of Theorem 2.1. Solid lines show edges of the tree G and dashed lines show examples of links.

Because the tree G has $2|X| + 2|\mathcal{T}|$ leaves, every WTAP solution must have cardinality at least $|X|+|\mathcal{T}|$. We show that this instance of TAP has a solution of cardinality exactly $|X|+|\mathcal{T}|$ if and only if there exists a subset $\mathcal{T}^* \subseteq T$ of tripels such that every element of $X \dot\cup Y \dot\cup Z$ is contained in exactly one of the triples in $\mathcal{T}^*$. First suppose such a set $\mathcal{T}^*$ of triples exists. Then the link set consisting of the links $\{x, t_{xyz}\}$ and $\{y, \bar{t}_{xyz}\}$ for $(x, y, z) \in \mathcal{T}^*$ and the links $\{t_{xyz}, \bar{t}_{xyz}\}$ for all $(x, y, z) \in \mathcal{T} \setminus \mathcal{T}^*$ is a TAP solution of cardinality $|X| + |\mathcal{T}|$.

Now suppose the TAP instance has a solution $F \subseteq L$ of cardinality $|X| + |\mathcal{T}|$. Because every leaf of G must be incident to a link from F and because G has $2 \cdot (|X| + |\mathcal{T}|) = 2 \cdot |F|$ many leaves, every leaf of G must be incident to exactly one link from F. Therefore, for every triple $(x, y, z) \in \mathcal{T}$, either $\{t_{xyz}, \bar{t}_{xyz}\} \in F$ or both $\{x, t_{xyz}\}$ and $\{y, \bar{t}_{xyz}\}$ are contained in F. We claim that the set $\mathcal{T}^*$ of triples $(x, y, z) \in \mathcal{T}$ with $\{x, t_{xyz}\}, \{y, \bar{t}_{xyz}\} \in F$ are a solution to 3D-Matching. Because every leaf of G is incident to exactly one link from F, every element from

$X \cup Y$ is contained in exactly one of the tripels in $\mathcal{T}^*$. In particular, $|\mathcal{T}^*| = |X| = |Y| = |Z|$. Thus, it suffices to show that every element $z \in Z$ is contained in at least one of the triples in $\mathcal{T}^*$. This holds because otherwise the edge $\{r, z\}$ would be a bridge in $(V, E \cup F)$. $\square$

One can use a similar construction to show that there exists a constant $\varepsilon > 0$ such that it is NP-hard to find a solution for TAP of cost at most $(1 + \varepsilon) \cdot c(\mathrm{OPT})$, i.e., the existence of an $(1 + \varepsilon)$-approximation algorithm for TAP would imply $\mathsf{P} = \mathsf{NP}$ ([KKL04]). This also holds when restricting TAP to the special case where G has diameter 4 and all endpoints of links are leaves of G.

The special case of WTAP where the tree G has diameter at most 3 can be solved in polynomial time using an algorithm for the minimum-weight perfect matching problem or the minimum-cost edge cover problem.

2.2 Shadow completeness

In this section we explain how we can view WTAP as a covering problem and introduce the concept of shadows of links which will be useful for designing approximation algorithms for WTAP.

Definition 2.2. *Given a WTAP instance $(G = (V, E), L, c)$, we say that a link $\{u, v\} \in L$* covers *an edge $e \in E$ if e is contained in the (unique) u-v path in the tree G. We also say that a link set $F \subseteq L$* covers *an edge $e \in E$ if F contains at least one link covering e.*

Using the above definition, we can now formulate WTAP as a covering problem.

Lemma 2.3. *Let $(G = (V, E), L, c)$ be an instance of WTAP. A link set $F \subseteq L$ is a feasible WTAP solution if and only if F covers all edges in E.*

Proof. First, suppose that F is a feasible WTAP solution and let $e \in E$. Then there is a set $U \subseteq V$ with $\delta_E(U) = \{e\}$. Because F is a WTAP solution, there exists a link $\{u, v\} \in \delta_F(U)$. The u-v path in the tree G must contain e, implying that F covers e.

Now suppose $F \subseteq L$ covers all edges in E and let $\emptyset \neq U \subsetneq V$. Then because G is a tree, we have $|\delta_E(U)| \geq 1$. If $\delta_E(U)$ contains only a single edge e, consider a link $\{u, v\} \in F$ covering e. Because e is the only edge in $\delta_E(U)$, we have $\{u, v\} \in \delta_F(U)$ and thus $|\delta_{E \dot\cup F}(U)| \geq 2$. $\square$

Next, we introduce the notion of *shadows* of a link ℓ, which are links that cover only a subset of the edges covered by ℓ.

Definition 2.4 (shadow)**.** *Let $(G = (V, E), L, c)$ be an instance of WTAP. A* shadow *of a link $\{u, v\}$ is a link $\{u', v'\} \in \binom{V}{2}$ such that both u' and v' lie on the u-v path in the tree G. A link $\{u', v'\}$ is called a* strict shadow *of $\{u, v\}$ if it is a shadow of $\{u, v\}$ and $\{u', v'\} \neq \{u, v\}$.*

Note that indeed, if ℓ' is a shadow of ℓ, then every edge covered by ℓ' is also covered by ℓ. Therefore, if a link is present in an instance of WTAP, we may assume that also all of its shadows are present and that the shadows have no higher cost. Such instances are called *shadow complete*.

Definition 2.5 (shadow completeness)**.** *An instance (G, L, c) of WTAP is* shadow complete *if for every link $\ell \in L$, every shadow $\ell' \in \binom{V}{2}$ of ℓ is contained in L and fulfills $c(\ell') \leq c(\ell)$.*

Lemma 2.6. *Let $\alpha \geq 1$. If there is an α-approximation algorithm for shadow complete instances of WTAP, then there is an α-approximation algorithm for WTAP.*

Similarly, if there is an α-approximation algorithm for shadow complete instances of TAP, then there is an α-approximation algorithm for TAP.

Proof. Given a (W)TAP instance (G, L, c), we define a shadow complete (W)TAP instance $(G, \tilde{L}, \tilde{c})$, where

$$\tilde{L} := \left\{ \ell' \in \tbinom{V}{2} \colon \ell' \text{ is a shadow of } \ell \text{ for some } \ell \in L \right\}$$

and

$$\tilde{c}(\ell') := \min \left\{ c(\ell) \colon \ell' \text{ is a shadow of } \ell \in L \right\}. \tag{2}$$

Then $L \subseteq \tilde{L}$ and $\tilde{c}(\ell) \leq c(\ell)$ for all $\ell \in L$ because every link is a shadow of itself. This implies that the cost of an optimal solution to the new instance is no larger than the cost of an optimal solution to the original instance. Moreover, given a solution $\tilde{F}$ to the new instance $(G, \tilde{L}, \tilde{c})$, we obtain a solution F to the original instance (G, L, c) with $c(F) = \tilde{c}(\tilde{F})$ by replacing every link $\ell' \in \tilde{F}$ by a link ℓ attaining the minimum in (2). □

2.3 2-approximation algorithms for WTAP

There are many different 2-approximation algorithms for WTAP. The first 2-approximation algorithm has been found by Frederickson and JáJá [FJ81] in the early '80s. Frederickson and JáJá's procedure was subsequently simplified and significantly sped up by Khuller and Thurimella [KT93]. Moreover, many classical and very versatile techniques for network design problems developed later also lead to a 2-approximation for WTAP. This includes primal-dual approaches (see Goemans, Goldberg, Plotkin, Shmoys, Tardos, and Williamson [GGPSTW94]) and the iterative rounding technique by Jain [Jai01]. The algorithm we present here follows the approach from [KT93].

Definition 2.7. *For a fixed instance $(G = (V, E), L, c)$ of WTAP and a root $r \in V$, we call a link $\{u, v\} \in L$ an* up-link *if u is an ancestor of v, i.e., u lies on the r-v path in G or v is an ancestor of u, i.e., v lies on the r-u path in G.*

We write L_{up} to denote the set of up-links in L. If ℓ is an up-link, we denote by $\text{top}(\ell)$ the endpoint of ℓ that is closer to the root in the tree G and by $\text{bottom}(\ell)$ the other endpoint of ℓ (that is further away from the root).

Lemma 2.8. *Let $(G = (V, E), L, c)$ be a shadow complete instance of WTAP. Then there exists a WTAP solution $F \subseteq L_{\text{up}}$ with $c(F) \leq 2 \cdot c(\text{OPT})$, where* OPT *denotes an optimal WTAP solution.*

Proof. We fix an optimal WTAP solution OPT and replace every link $\{u, v\} \in \text{OPT}$ that is not an up-link by its two shadows $\{u, \text{lca(u, v)}\}$ and $\{v, \text{lca(u, v)}\}$, where lca(u, v) denotes the least common ancestor of u and v in G. Let F be the resulting link set. Then F contains only up-links. Moreover, because every edge covered by a link $\{u, v\}$ is also covered by $\{v, \text{lca(u, v)}\}$ or lca(u, v), the link set F is a WTAP solution. Using that $(G = (V, E), L, c)$ is shadow complete, we have $c(\{u, \text{lca(u, v)}\}) + \text{c}(\{\text{v}, \text{lca(u, v)}\}) \leq 2 \cdot \text{c}(\{\text{u}, \text{v}\})$ for every link $\{u, v\}$ and thus $c(F) \leq 2 \cdot c(\text{OPT})$. □

Lemma 2.9. *There is a polynomial-time algorithm that, given a WTAP instance $(G = (V, E), L, c)$ with a root $r \in V$, computes a cheapest WTAP solution consisting only of up-links (or decides that no such solution exists).*

Proof. We construct a weighted directed graph as follows. First, we orient the edges of G towards the root r, i.e., such that every edges lies on a directed path to r, and we assign cost 0 to these edges. We denote the resulting set of oriented edges by $\overrightarrow{E}$. Then for every up-link ℓ we add an arc $\overrightarrow{\ell}$ from $\text{top}(\ell)$ to $\text{bottom}(\ell)$ of cost $c(\ell)$. In this directed graph, we compute a minimum-cost spanning arborescence rooted at r and return the set of those up-links that correspond to arcs in this arborescence.

To see that this indeed yields a cheapest up-link solution, we show that a set $F \subseteq L_{\text{up}}$ is a WTAP solution if and only if $\overrightarrow{F} \cup \overrightarrow{E}$ contains an arborescence rooted at r, where $\overrightarrow{F} := \{\overrightarrow{\ell} : \ell \in F\}$.

If F is not a WTAP solution, there is a cut $\delta(W)$ with $\emptyset \neq W \subseteq V \setminus \{r\}$ that contains only a single edge in $E \cup F$. Because we oriented the edges of G towards r, we have $|\delta^+_{\overrightarrow{E}}(W)| \geq 1$ and thus $\delta^-_{\overrightarrow{E} \cup \overrightarrow{F}}(W) = \emptyset$, implying that $\overrightarrow{E} \cup \overrightarrow{F}$ contains no arborescence rooted at r.

Now let $F \subseteq L_{\text{up}}$ be a WTAP solution and suppose for the sake of deriving a contradiction that $\overrightarrow{F} \cup \overrightarrow{E}$ contains no spanning arborescence rooted at r. Let $v \in V$ be a vertex that is not reachable from r in $\overrightarrow{F} \cup \overrightarrow{E}$ and among all such vertices is closest to r in G. Then all strict ancestors of v in G, i.e., all vertices on the r-v path in G except for v, are reachable from r in $\overrightarrow{F} \cup \overrightarrow{E}$. Moreover, because we oriented G towards r, none of the descendants of v in G is reachable from r in $\overrightarrow{F} \cup \overrightarrow{E}$ as otherwise v would be also reachable. Let $W \subseteq V$ be the set of descendants of v in G, including v itself. Then, using that F contains only up-links, we get $\delta^-_{\overrightarrow{F}}(W) = \emptyset$, implying $\delta_F(W) = \emptyset$ and thus contradicting the fact that F is a WTAP solution. (Here, we used that the definition of W implies $|\delta_E(W)| = 1$.) □

One can also prove Lemma 2.9 by showing that the following LP relaxation is integral:

$$\begin{array}{lrcll} \min & \displaystyle\sum_{\ell \in L_{\text{up}}} c(\ell) \cdot x_\ell & & & \\ \text{s.t.} & \displaystyle\sum_{\ell \in L_{\text{up}} \colon \ell \text{ covers } e} x_\ell & \geq & 1 & \text{for all } e \in E \\ & x & \in & [0,1]^{L_{\text{up}}} & \end{array}$$

Another possibility to prove Lemma 2.9 is via a dynamic programming algorithm. As a direct consquence of Lemma 2.8 and Lemma 2.9 we obtain a 2-approximation algorithm for WTAP.

Corollary 2.10. *There is a 2-approximation algorithm for WTAP.*

Next, we present a strengthening of Lemma 2.8 by showing that every WTAP solution consisting only of up-links can be turned into a solution where every edge is covered by at most one link. For a link $\ell \in L_{\text{up}}$, we denote by $P_\ell \subseteq E$ the set of edges covered by ℓ.

Lemma 2.11. *Let $(G = (V, E), L, c)$ be a shadow complete WTAP instance, and let $F \subseteq L_{\text{up}}$ be a WTAP solution. Then we can in polynomial time transform F into a WTAP solution for which the paths P_ℓ with $\ell \in F$ are disjoint by replacing some links $\ell \in F$ by one of its shadows and possibly removing some links from F.*

Proof. Let $F \subseteq L_{\text{up}}$ be a WTAP solution and suppose that there exists an edge that is covered by at least two links from F. Then let $e \in E$ be such an edge that is as close to the root r of G as possible. Let $\ell_1, \ell_2 \in F$ be two distinct links covering $e = \{v, w\}$, where w.l.o.g. v is closer to the root of G. Then by the choice of e, at least one of the links ℓ_1, ℓ_2, say ℓ_1, does not cover the last edge of the r-v path in G. Thus, because ℓ_1 is an up-link, the vertex v is one of its endpoints and more precisely, we have $v = \text{top}(\ell_1)$. Therefore, $\ell_1' := \{w, \text{bottom}(\ell_1)\}$ is a shadow of ℓ_1 and $F' := (F \setminus \{\ell_1'\}) \cup \{\ell_1\}$ is a WTAP solution. (If $w = \text{bottom}(\ell)$, $F \setminus \{\ell_1'\}$ is a feasible WTAP solution.) Because $\sum_{\ell \in F'} |P_\ell| < \sum_{\ell \in F} |P_\ell|$, we need at most $|F| \cdot |E|$ iterations of the above procedure until we obtain a WTAP solution with the desired properties. □

As a direct consequence of Lemma 2.8 and Lemma 2.11, we obtain the following strengthening of Lemma 2.8.

Corollary 2.12. *Let $(G = (V, E), L, c)$ be a shadow complete instance of WTAP. Then there exists a WTAP solution $F \subseteq L_{\text{up}}$ such that*

- *the paths P_ℓ with $\ell \in F$ are disjoint and*
- $c(F) \leq 2 \cdot c(\text{OPT})$,

where OPT *denotes an optimal WTAP solution.*

2.4 Better-than-2 approximation algorithms for WTAP

The first better-than-2 approximation for WTAP has been given in [TZ21], building on [CN13]. The currently best known approximation guarantee is $1.5 + \varepsilon$ [TZ22b]. We refer to these papers for details.

References

[BGRS13] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. "Steiner Tree Approximation via Iterative Randomized Rounding". In: *Journal of the ACM* 60.1 (2013), 6:1–6:33.

[CN13] N. Cohen and Z. Nutov. "A $(1 + \ln 2)$-Approximation Algorithm for Minimum-Cost 2-Edge-Connectivity Augmentation of Trees with Constant Radius". In: *Theoretical Computer Science* 489 (2013), pp. 67–74.

[FJ81] G. N. Frederickson and J. JáJá. "Approximation Algorithms for Several Graph Augmentation Problems". In: *SIAM Journal on Computing* 10.2 (1981), pp. 270–283.

[GGPSTW94] M. X. Goemans, A. V. Goldberg, S. Plotkin, D. B. Shmoys, É. Tardos, and D. P. Williamson. "Improved Approximation Algorithms for Network Design Problems". In: *Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 1994, pp. 223–232.

[GORZ12] M. X. Goemans, N. Olver, T. Rothvoß, and R. Zenklusen. "Matroids and Integrality Gaps for Hypergraphic Steiner Tree Relaxations". In: *Proceedings of 44th ACM Symposium on Theory of Computing (STOC)*. 2012, pp. 1161–1175.

[Jai01] K. Jain. "A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem". In: *Combinatorica* 21 (2001), pp. 39–60.

[KKL04] G. Kortsarz, R. Krauthgamer, and J. R. Lee. "Hardness of Approximation for Vertex-Connectivity Network Design Problems". In: *SIAM Journal on Computing* 33.3 (2004), pp. 704–720.

[KT93] S. Khuller and R. Thurimella. "Approximation algorithms for graph augmentation". In: *Journal of Algorithms* 14.2 (1993), pp. 214–225.

[KV18] B. Korte and J. Vygen. *Combinatorial optimization*. Springer, Sixth Edition., 2018.

[KV94] S. Khuller and U. Vishkin. "Biconnectivity Approximations and Graph Carvings". In: *Journal of the ACM* 41.2 (1994), pp. 214–235.

[PS82] C. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. 1982.

[TZ21] V. Traub and R. Zenklusen. "A Better-Than-2 Approximation Algorithm for Weighted Tree Augmentation". In: *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science, (FOCS)*. 2021, pp. 1–12.

[TZ22a] V. Traub and R. Zenklusen. *A (1.5+ε)-Approximation Algorithm for Weighted Connectivity Augmentation*. 2022. URL: https://doi.org/10.48550/arXiv.2209.07860.

[TZ22b] V. Traub and R. Zenklusen. "Local Search for Weighted Tree Augmentation and Steiner Tree". In: *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, (SODA)*. 2022, pp. 3253–3272.