LP-relaxations for Tree Augmentation

Guy Kortsarz[a], Zeev Nutov[b]

[a] *Rutgers University, Camden*
[b] *The Open University of Israel*

Abstract

In the Tree Augmentation problem the goal is to augment a tree T by a minimum size edge set F from a given edge set E such that $T \cup F$ is 2-edge-connected. The best approximation ratio known for the problem is 1.5. In the more general Weighted Tree Augmentation problem, F should be of minimum weight. Weighted Tree Augmentation admits several 2-approximation algorithms w.r.t. the standard cut-LP relaxation. The problem is equivalent to the problem of covering a laminar set family. Laminar set families play an important role in the design of approximation algorithms for connectivity network design problems. In fact, Weighted Tree Augmentation is the simplest connectivity network design problem for which a ratio better than 2 is not known. Improving this "natural" ratio is a major open problem, which may have implications on many other network design problems. It seems that achieving this goal requires finding an LP-relaxation with integrality gap better than 2, which is an old open problem even for Tree Augmentation. In this paper we give the *first LP based approximation* for Tree Augmentation with ratio $1.75 < 2$. We have two algorithms with ratio 7/4: the first is *a primal fitting* algorithm and the second is a *a dual fitting algorithm*. Our main algorithm is the dual-fitting one, and it runs in $O(mn)$ time, which is by far faster than all other algorithms with ratio less than 2 for the problem.

Keywords: Tree Augmentation; LP-relaxation; Laminar family; Approximation algorithms

Email addresses: guyk@camden.rutgers.edu (Guy Kortsarz), nutov@openu.ac.il (Zeev Nutov)

[1]Supported in part by NSF grants 1218620 and 1540547

1. Introduction

1.1. Problem definition and related problems

A graph (possibly with parallel edges) is k**-edge-connected** if there are k pairwise edge-disjoint paths between every pair of its nodes. We study the following fundamental connectivity augmentation problem: given a connected undirected graph $G = (V, E_G)$ and a set of additional edges called **links** E on V disjoint to E_G, find a minimum size edge set $F \subseteq E$ such that $G + F = (V, E_G \cup F)$ is 2-edge-connected. Contracting the 2-edge-connected components of the input graph G results in a tree. Hence, our problem is:

TREE AUGMENTATION
Input: A tree $T = (V, E_T)$ and a set of links E on V disjoint to E_T.
Output: A minimum size subset $F \subseteq E$ of links such that $T \cup F$ is 2-edge-connected.

TREE AUGMENTATION can be formulated as the problem of covering the edges of a tree by paths, or by links. For $u, v \in V$ let $(u, v) \in E_T$ denote the edge in T and uv the link in E between u and v. Let $P(uv) = P_T(uv)$ denote the path between u and v in T. A link uv **covers** all the edges along the path $P(uv)$. Then TREE AUGMENTATION is the problem of finding a minimum subset of (paths of the) links that covers the edges of T.

A set family $\mathcal{T}$ is **laminar** if any two sets in the family are disjoint or one contains the other. TREE AUGMENTATION can be also formulated as the problem of covering a laminar set family. In what follows, root T at some node r. The choice of the root r defines a partial order on V: u is a **descendant** of v (or v is an **ancestor** of u) if v belongs to $P(ru)$. The **rooted subtree** of T induced by v and its descendants is denoted by T_v (v is the root of T_v). Let $\mathcal{T} = \{T_v : v \in V \setminus \{r\}\}$. The family of node sets of the trees in $\mathcal{T}$ is laminar, and $F \subseteq E$ is a feasible solution for TREE AUGMENTATION if and only if F covers $\mathcal{T}$, namely, if and only if for every $T' \in \mathcal{T}$ there is a link in F from T' to $T \setminus T'$.

TREE AUGMENTATION is also equivalent to the problem of augmenting the edge-connectivity from k to $k + 1$ for any odd k; this is since the family of minimum cuts of a k-connected graph with k odd is laminar.

In the more general WEIGHTED TREE AUGMENTATION problem, the links in E have weights $\{w_e : e \in E\}$ and the goal is to find a minimum weight augmenting edge set $F \subseteq E$ such that $T \cup F$ is 2-edge connected.

A more general problem is the 2-Edge-Connected Subgraph problem, where the goal is to find a spanning 2-edge-connected subgraph of a given weighted graph; Weighted Tree Augmentation is a particular case, when the input graph contains a connected spanning subgraph of cost zero.

In this paper we introduce new LP-relaxations for Tree Augmentation (that are also valid for Weighted Tree Augmentation) and prove that their integrality gap for Tree Augmentation is at most 1.75. We present two algorithms: a dual-fitting algorithm and a primal-fitting algorithm. We note that our paper is the *first* to give integrality gap less than 2 *with respect to an LP*. An important feature of our algorithms is that we *do not need to solve the LP*. The primal fitting algorithm is in fact very fast. Its starts with a maximal inclusion matching and set the rest of the variables with respect to this matching. Each time we find a special kind of tree; A thing that can be done in time $O(m)$. Thus the dual fitting algorithm has running time is $O(m \cdot n)$ which is much lower than the time for solving even basic LP's. All other algorithms for Tree Augmentation with ratio better than 2, including our primal fitting algorithm. In fact among all algorithms of ratio better than 2 for Tree Augmentation its likely that only our dual-fitting algorithm can be used in practice. In [3], J. Cheriyan and Z. Gao give a $1.5 + \epsilon$ approximation for Tree Augmentation. This hold for any constant $\epsilon > 0$. The ratio is with respect to an SDP relaxation obtained by additionally using the Lasserre lift and project method to tighten the Semi Definite mathematical programs. The algorithm and analysis seem quite involved and surely make the paper technically strong.

On the other hand, our goal was to find the *simplest algorithms* that give integrality gap for some *LP* with ratio less than 2. We note that our algorithms use several ideas from the papers of G. Even, J. Feldman, G. Kortsarz and Z. Nutov [7, 8], but both algorithms and their analysis are not identical to any previous algorithm.

The challenge of breaking the ratio of 2 for Weighted Tree Augmentation: There are very recent new algorithms, that came out a few months after our paper, that give an LP based better than 2 ratio, for the case that the maximum cost is bounded by a universal constant. See [1] by D. Adjiashvili,[9] by S. Fiorini, M. Gross, J. Konemann and L. Sanita, and [21] by Z. Nutov. These complex methods relay very strongly on the above property. In our opinion it seems highly unlikely that these methods can be used for breaking the ratio of 2 for Weighted Tree Augmentation. Solving this question with our methods seems hard too, but has some hope.

We have to overcome a serious difficulty. We should compute a *min cost leaf cover* and cover a rooted subtree T' with this credit, but for some technical reason, the cover should be a local ratio step. Namely the links used so far should not be able to cover any of the remaining edges. We were not able to design a local ratio, so far, even for TREE AUGMENTATION, albeit we are trying.

2. Previous and related work

It is shown by G. N. Frederickson and J. Jájá [10], that TAP is NP-hard even for trees of diameter 4. In [4], J. Cheriyan, T. Jordán, and R. Ravi show that TAP is hard even when the set E of links, forms a cycle on the leaves of T. The first 2-approximation for Weighted TAP was given 26 years ago in 1981 by Fredrickson and Jájá [10], and was simplified later by Khuller and Thurimella [16]. These algorithms reduce the problem to the Min-Cost Arborescence problem, that is solvable in polynomial time, as shown by Y. Chu and T. Liu. [5]. However, both algorithms loose a factor of 2 in the reduction. The primal-dual algorithm of M. Goemans and D. Williamson [13] gives a ratio 2 for WTAP. Similarly, the algorithm of M. Goemans, A. Goldberg, S. Plotkin, D. Shmoys, and D. Williamson [12] is another primal-dual 2-approximation algorithm for the problem. The iterative rounding algorithm of Jain [14] is an LP-based 2-approximation algorithms. These algorithms achieve ratio 2 w.r.t. to the standard **cut-LP** that seeks to minimize $\sum_{e\in E} w_e x_e$ over the following polyhedron:

$$x_e \geq 0 \quad \forall e \in E \tag{1}$$

$$x(\delta(T')) \geq 1 \quad \forall T' \in \mathcal{T} \tag{2}$$

Here $\delta(T')$ is the set of links with exactly one endnode in T', $x(F) = \sum_{e\in F} x_e$ is the sum of the variables indexed by the links in F, and $\mathcal{T}$ is the set of proper rooted subtrees of T w.r.t. the chosen root r.

Laminar set families play an important role in the design and analysis of exact and approximation algorithms for network design problems, both in the primal-dual method by and the iterative rounding method. See the book of L. C. Lau, R. Ravi, and M. Singh [18]. Weighted TAP is the simplest network design problem for which a ratio better than 2 is not known. Breaking the "natural" ratio of 2 for Weighted TAP is a major open problem in network design, that may have implications on other problems.

Khuller [15] in his survey on high connectivity network design problems, posed this problem as major open question. The question back then was only: is there a ratio better than 2 for TAP? Nagamochi [20] used a novel lower bound to achieve ratio $1.875+\epsilon$ for TAP. In [7] a sketch of an algorithm with ratio of 1.5 for TAP was given. The authors decided to wait with the publication of the full version. The idea was to simplify the full version. Indeed, at the time the full version of the 1.5 ratio had over 40 pages and used a very extensive case analysis.

Instead publishing the full version of [7], G. Even J. Feldman G, Kortsarz and Z.nutov [8], published a much simpler 1.8 ratio algorithm. Achieving a simple full version of the [7] was very hard and took many years. Finally, in 2016, Kortsarz and Nutov [17] presented a drastically simplified full version of [7]. This paper has 24 pages and is drastically simpler than the original full version of [7]. Additionally, [17] has a better ratio than [20], and in addition, [17] is much shorter and simpler than [20].

Several algorithms for Weighted TAP with ratio better than 2 are known for special cases. In [6] N. Cohen and Z. Nutov. give an algorithm with ratio $(1+\ln 2)$ for Weighted TAP, running time $n^{f(D)}$ where D is the diameter of T. In [4] by J. Cheriyan, T. Jordán, and R. Ravi, it is shown how to round a half-integral solution to the cut-LP within ratio 4/3. However, as is pointed in [4], the cut-LP LP has extreme points which are not half integral.

For the special case of TAP when every link connects two leaves, Y. Maduel and Z. Nutov. [19] obtained ratios 5/3 in the case that all links are leaf to leaf links, with respect to some LP. They also give a combinatorial 17/12 approximation. However, the analysis of [19] does not extend directly to the general TAP. Cheriyan and Gao [2] provided an SDP based algorithm with $1.8+\epsilon$ for TAP. It used a PSD mathematical program in addition to the powerful Lasserre lift and project method. Later [3] J. Cheriyan and Z. Gao improved their analysis, giving a ratio of $1.5+\epsilon$ based on similar methods. Both papers [2] and [3], did not have to solve the SDP and the SDP was used only in the analysis [2, 3]. The SDP and the analysis in [2, 3] are quite involved, and hence might be very hard to extend to Weighted TAP. The following very recent papers [1, 9, 21] give LP based better than 2 ratio, for the case the maximum weight is a constant.

Finally, we mention some work on the closely related 2-Edge-Connected Subgraph problem. This problem was also vastly studied. For general weights, the best known ratio is 2 by Fredrickson and Jájá [10], which can also be achieved by the algorithms of S. Khuller and R. Thurimella [16] and K. Jain

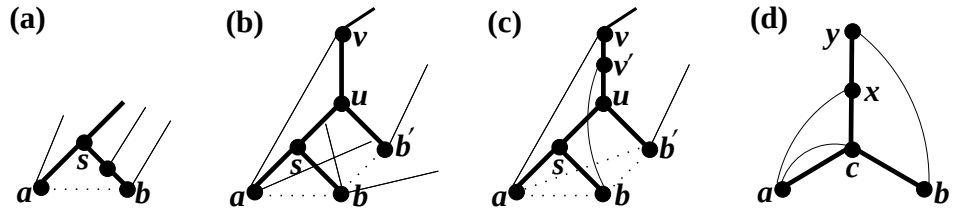

Figure 1: (a,b,c) Illustration for twin links and locking links. The twin-link and the locking link are shown by dotted lines, some other possible links in E are shown by solid thin lines. Some tree edges may be paths (the uv-path in (b,c) may have length zero). (d) A link by that overlaps a link ax.

[14]. For particular cases, better ratios are known. Fredrickson and Jájá [11] showed that when the edge weights satisfy the triangle inequality, the Christofides heuristic has ratio 3/2. For the special case when all the edges of the input graph have unit weights (the "min-size" version of the problem), the currently best known ratio is 4/3 due to Sebő and Vygen [23].

3. Fundamentals

Before we describe our new technique we need to mention some basic terminology and important structures.

Definition 3.1 (leaf, twin link, stem). *The* **leaves** *of T are the nodes in $V \setminus \{r\}$ that have no descendants. We denote the leaf set of T by $L(T)$, or simply by L, when the context is clear. A link $ab \in \delta(L, L)$ is a* **twin link** *and the least common ancestor s of a, b is a* **stem** *if the contraction of T_s results in a new leaf; such a, b are called* **twins**. *Let W denote the set of twin links, and for $e \in W$ let s_e denote the stem of e.*

See Figure 1 (a).

Intuition: the importance of stems.

The problem with a stem (see Figure 1 (a)) is that the contraction of the twin link creates a new leaf, a very problematic thing for any algorithm. Note that the simple lower on the optimum, based only on L, is $|L|/2$. If for each contraction (we use one link and) we reduce the number of leaves by 1 only, we get a solution of size $|L|$ which is twice the above simple lower bound.

Definition 3.2 (shadow, shadow-minimal cover). *Let $P(uv)$ denote the path between u and v in T. A link $u'v'$ is a* **shadow** *of a link uv if $P(u'v') \subseteq P(uv)$. A cover F of T is* **shadow-minimal** *if for every link $uv \in F$ replacing uv by any proper shadow of uv results in a set of links that does not cover T.*

Definition 3.3. *Two links* **overlap** *if their paths share an edge and one contains an end of the other.*

Since adding all the shadow does not change the value of the optimum (if a shadow is used, we can replace it by the original link) we *add all shadows of all links* to the graph. This is called *shadow completion*.

The Shadow-Completion Assumption
The set of links E is closed under shadows.

From now on, let F be any optimum *shadows minimal* solution.

Proposition 3.1. *F is not shadow minimal if and only if two links in F overlap.*

The following claim was proved in several papers. See for example [17] by Kortsarz and Nutov. It is also proven later in our paper, for completeness.

Claim 3.2. *In any shadow minimal solution F the leaves are touched by exactly one link of F. Hence the graph induced by the leaves and the links of F forms a matching. Additionally given a stem s, either the twin link belongs to F and* exactly one link *touches s, or the twins are covered by two separate links and s* has no link in *F.*

Corollary 3.3. *A stem s is touched by a* single link *in F if and only if F contains the the twin link of s.*

For $A, B \subseteq V$ and $F \subseteq E$ let $\delta_F(A, B)$ denote the set of links in F with one end in A and the other end in B, and let $\delta_F(A) = \delta_F(A, V \setminus A)$ denote the set of links in F with exactly one endnode in A. The default subscript in the above notation is E. To **contract** a subtree T' of T is to combine the nodes in T' into a new node v. The edges and links with both endpoints in T' are deleted. The edges and links with one endpoint in T' now have v as their new endpoint.

Definition 3.4. *Given a subtree T_v rooted at v, a set A is T_v-**open** (equivalently, T_v is A-open) if there is a link in E from $A \cap T'$ to $T \setminus T'$. Namely, A contains a link with one endpoint inside T' and one outside T'. Otherwise we say that either A is T_v-closed or that T_v is A-closed.*

Intuition: The above definition is crucial when we try to cover a subtree T_v. Any set A that is not T_v-open will not cover the root v of T_v and so will not be a feasible solution. The next definition appears in many papers. See for example [17].

Definition 3.5 (locked node, locking link, dangerous locking tree).

A node a (or a subtree T_a) is **locked** *by a link $bb' \in \delta(L, L)$ and bb' is the* **locking link of** *a if (see Fig. 1(a)) the tree obtained from T by contracting T_a into the node a has a rooted proper subtree $T' = T_{r'}$ that is a-closed such that $L(T') = \{a, b, b'\}$; such minimal T' is called the* **locking tree** *of a (note that such locking tree is unique); a locking tree is a* **dangerous locking tree** *if it is as in Fig. 2(b) with the links depicted present in E; namely, a locking tree is dangerous if there exists an ordering b, b' of the locking link endnodes such that:*

- *The contraction of ab' does not create a new leaf.*
- *$ab' \in E$.*
- *T' is b-open.*

We shall later see that such an ordering may not be unique.

Dangerous trees, locking links, intuition: The most problematic trees to cover for our algorithm are trees T' with 3 leaves. See Figure 1 (b). A dangerous tree is a tree that chose the "wrong" link into the matching. Specifically, if our solution chooses the link bb' we are in trouble.

As noted above, assuming our solution is shadows minimal, every leaf is touched by one link exactly. Hence assuming that bb' is in our solution, ab and ab' can not be in our solution. Since a is T'-closed, the link covering a will not be a leaf to leaf link. This is the reason we call the link bb' **a locking link** with respect to a: In addition, since a is T' closed, and b, b' already are touched by a link, the root will have to be covered by a link that is not touched (inside T') by a leaf. All this is very bad for our algorithm.

The definition of dangerous trees contains the solution for these difficulties as follows. We later define minimal *semi-closed* trees. Each iteration covers a minimal *semi-closed* tree. An important property of such trees is that they are closed with respect to unmatched leaves (but may be open with respect to matched leaves).

How to handle dangerous trees: We note that our algorithm does not encounter any difficulty in the covering of a non dangerous trees.

However, when all minimally sem-closed trees are dangerous, we need a special procedure to handle this case. We note that one of the invariants of our algorithm is that each new created leaf owns a coupon.

The special procedure for dangerous trees: By definition of a dangerous tree, the contraction of, say, ab' does not create a new leaf, and b is T'-open. This is used in the solution for the case all minimally semi-closed trees are dangerous. We first change the matching as described above for every dangerous tree.

Changing the matching to ab' creates an unmatched leaf b that, by the definition of dangerous tree, is T' open. As pointed above, minimal semi-closed trees are closed with respect to unmatched leaves, and the change in the matching made b an unmatched leaf **that is T' open**. Say that the next T'' contains b (it will have to contain at least one such node). Thus if the next tree contains b, will be b-closed. This means that it will contain a node that did not appear in T'. Indeed, if the link of b is bv, by definition $v \notin T'$. On the other hand, as the next tree T'' that contains b is b-closed, $v \in T''$. It is easy to see that in such a case T'' strictly contains T'. It turns out that a tree that strictly contains a dangerous tree *can not be dangerous*. Hence the above procedure finds a non dangerous tree T'', that strictly contains T'. The name *dangerous trees* seems useful, because of the special procedure required, when all minimally semi-closed trees are dangerous.

If the leaf a is an original (non compound) leaf and the tree is locking but not dangerous we say that the tree is *bad* (since there is no way to cover the tree with few links). Case (c) in Figure 1 shows a possible bad tree. The optimum solution F can not efficiently cover T'. If it will choose say ab' then b will not be T' open etc. Thus the link that covers the root has a non leaf endpoint in T'. If a is an original leaf, locked tree that are not dangerous (bad trees with a an original leaf) can be recognized before the LP is written. This fact is used in the dual-fitting solution.

To see that the ordering may not be unique, see Figure 2. Part (a) just shows a locking link. Indeed, a is $T_{r'}$ closed and the tree has 3 leaves. Both trees in (b) and in (c) are dangerous In Figure2 (b) the matching must be changed to ab'. However, Figure 2 (c) shows that the ordering as above is not always unique. The contraction of both links ab or ab' does not created a leaf, and both b, b' may be $T_{r'}$ open.

Definition 3.6. *A tree is called a* bad tree *if its a node is an original leaf, the tree is locking but not dangerous. The set $\mathcal{N}$ denoted the collection of bad trees*

4. New techniques introduced in the primal and dual fitting algorithms

4.1. The Primal fitting algorithm

1. Proving the lower bound becomes much simpler when an LP is used. The lower bound proof is remarkably short compared to some of the combinatorial lower bounds used before. In fact the proof is shorter by a few pages than the proof of lower bounds in older versions of the paper. It is even simpler to prove than the lower bound in [17] by Kortsarz and Nutov. A nice feature of the algorithm is that we do not need to solve the LP. We take part of the constraints and get a polynomial problem (a problem that can be solved by a combinatorial algorithm in polynomial time) and extend this partial solution to an LP feasible solution.
2. The following step is almost identical to a step in [17]. We multiply the LP lower bound by 1.75 and distributed the lower on vertices and links. This is an identical step to [17], except that in [17] we distributed a combinatorial lower bound.
3. We use Corollary 3.3 to say in the (primal) LP that the x value on the twin link equals the amount of flow entering s. This is a collection of valid inequalities as indicated by Corollary 3.3. These valid inequalities seem needed to give an integrality gap less than 2 in the sense that we do not know how to break the integrality gap of 2 without these constraints. This idea is completely new.
4. We change the initial LP solution in a novel way. Since twin links are harder to cover than other links, we need to transfer *part of the x values at unmatched leaves* for covering a twin link. This is done in two steps.

Note that unmatched leaves at start have ρ credits of the lower bound. Say that v is an unmatched leaf that has a link to a stem s. In the first step, we transfer $1/2$ of the credit on v to s. In a second step we transfer the credit from s to the twin link. This step is completely new. We have to take into account the lower credit on v, compared to the initial credit it was given.

5. A difficult case arises in Lemma 9.9. See Figure 3. This lemma is completely new and is the main technical challenge in the primal fitting algorithm. We need to shows that we can treat our tree as a dangerous tree. Find a way to compute a matching edge whose contraction does not create a leaf, so that the third leaf is T' open. Not all the links in Figure 3 even exists. Thus a very careful analysis is required.

The intuition for the dual fitting algorithm is given later.

Notation:

1. Let $\mathcal{N}$ denote the set of bad trees.
2. A compound node is a node created out of contraction that is not a leaf. Let C be the collection of compound modes that are not leaves. Given a tree T' we denote $C' = T' \cap C$.
3. For $s \in S$ let $\sigma(s)$ denote the set of links in $\delta(s)$ that have their other endnode not in T_s.
4. For $T' \in \mathcal{N}$ let $\zeta(T')$ denote the set of links incident to some non-leaf node of T'.
5. Let $\mathcal{O}_L = \{A \subseteq V : |A \cap L| \text{ is odd}\}$.
6. For $x \in \mathbb{R}^E$ and $F \subseteq E$ let $x(F) = \sum_{e \in F} x_e$.

Lemma 4.1. *Let F be a shadow-minimal cover of T. Let $T' \in \mathcal{T}$ be a non dangerous subtree Then the following holds:*

(i) *$\delta_F(L, V)$ is an exact edge-cover of L, namely $|\delta_F(v)| = 1$ for every $v \in L$.*

(ii) *If $e \in F \cap W$ then $|\sigma(s_e) \cap F| = 1$.*

(iii) *$\zeta(T') \cap F \neq \emptyset$ for any $T' \in \mathcal{N}$*

Proof: Part (i): As any two links incident to the same leaf overlap, (i) follows from Proposition 3.1.

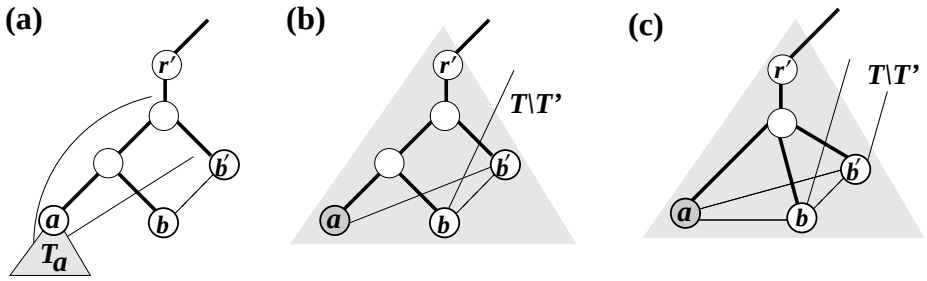

Figure 2: (a) A locking tree; (b,c) Dangerous trees; solid thin lines show links that must exist in E. The endnodes b, b' of the locking link are original leaves; in (a), a is an original leaf, and in (b),(c) the subtree T_a is contracted into a, so a may be a compound node or an original leaf. Some of the edges of T can be paths.

Part(2): Now let $e \in F \cap W$ and consider a link f that covers the parent edge of the stem s_e of e. Clearly, e and f do not overlap. However, every two links in $\sigma(s_e)$ overlap. This implies (ii).

Part (iii). For this part consider Figure 2. This part speaks of a tree T' that looks like a dangerous tree but in fact it is not dangerous. These trees belongs to $\mathcal{N}$. And (iii) says that a non-leaf node in T' is linked in F. For the sake of contradiction, assume $\zeta(T) \cap F = \emptyset$. In such a case we show that T' is a dangerous tree, deriving a contradiction. Consider a link $e = au$ that covers the parent edge of a and a link $e' = u'v$ that covers the parent edge of T', where $v \notin T'$. Note that $e \neq e'$, since T' is a-closed. As $\zeta(T) \cap F = \emptyset$, one of b, b' must cover the parent link of T' and another must cover the link of a. Thus $\{u, u'\} \subseteq \{b, b'\}$. If $u = b$ it can not be that the contraction of ab creates a new leaf. Indeed, since $deg_F(b) = deg_F(a) = 1$, this new leaf will not be covered by a or b. Therefore, this implies that $\zeta(T) \cap F) \neq \emptyset$.

Thus regardless of who covers a, contracting au does not create a leaf and the other b node covers the parent link of T'. This means that T' is dangerous. □

Now we present our new valid inequalities for TAP.

Lemma 4.2. *Suppose that the Shadow-Completion Assumption holds, and let x be the characteristic vector of a shadow minimal cover F of T. Then x*

satisfies the following constraints

$$\begin{aligned} x(\sigma(s_e)) - x_e &\geq 0 && \forall e \in W && (3)\\ x(\zeta(T')) &\geq 1 && \forall T' \in \mathcal{N} && (4)\\ x(\delta(v)) &= 1 && \forall v \in L && (5)\\ x(\delta(A,V)) &\geq \left\lceil \frac{|A \cap L|}{2} \right\rceil && \forall A \in \mathcal{O}_L && (6) \end{aligned}$$

Proof: Consider the polyhedron Π_L defined by (1), (5), and (6). Then Π_L is the convex hull of the exact edge-covers of L, see [22, Theorem 34.2] in the book of A. Schrijver. thus by Lemma 4.1(i), these constraints are valid. The validity of the constraints (3) follows from Lemma 4.1(ii) (in fact, $x(\sigma(s_e)) = x_e$ holds), and the validity of the constraints (4) follows from Lemma 4.1(iii). □

In subsequent sections we will consider two LPs, where both have the constraints (1), (2), and (3). One LP has an additional constraint (4), while the other LP has additional constraints (5) and (6) instead.

5. The algorithms

For a set of links $I \subseteq E$, let T/I denote the tree obtained by contracting every 2-edge-connected component of $T \cup I$ into a single node. We often refer to the contraction of every 2-edge-connected component of $T \cup I$ into a single node as the contraction of the links in I. Our algorithm iteratively contracts certain subtrees of T/I. Recall that C are compound nodes of T/I, that are not leaves. Non-compound nodes are referred to as **original nodes** (of T). Compound nodes in C are non leaves that contains more than one node (due to contraction). Given T', we denote by C' the non leaves compound nodes inside T'. For technical reasons, the root r is also considered as a compound node, hence initially $C = \{r\}$.

Our algorithms start with a partial solution $I = \emptyset$ and with a certain matching $M \subseteq \delta(L, L) \setminus W$. We denote by U the set of leaves of T/I unmatched by M. The algorithm iteratively finds a subtree T' of T/I and a cover I' of T', and **contracts** T' **with** I', which means adding I' to I and contracting T' into a new compound node. To use the notation T/I properly, we will assume that I' is an exact cover of T', namely, that the set of edges of T/I that is covered by I' equals the set of edges of T' (this is possible due to shadow completion).

Another property of a contracted tree T' is given in the following definition.

Definition 5.1 (M-compatible subtree). *Let M be a matching on the leaves of T/I. A subtree T' of T/I is* **M-compatible** *if for any $bb' \in M$ either both b, b' belong to T', or none of b, b' belongs to T'. We say that a contraction of T' with I' is M-compatible if T' is M-compatible.*

Assuming all compound nodes were created by M-compatible contractions, then the following type of contractions is also M-compatible.

Definition 5.2 (greedy contraction). *Adding to the partial solution I a link with both endnodes in U is called a* **greedy contraction**. *Note that a greedy contraction can not make the tree M-incompatible as it just contracts part of the graph into a single compound node. This node will not be a matched node. And in fact, every compound leaf can not be a matched leaf. All matching links are of original links. Hence contracting a subtree can not give a M-incomputable link.*

One of the steps of the algorithm is to apply greedy contractions exhaustively; clearly, this can be done in polynomial time.

We now describe a more complicated type of M-compatible contractions.

Definition 5.3 (semi-closed tree). *Let M be a matching on the leaves of T/I. A rooted subtree T' of T/I is* **semi-closed** *(w.r.t. M) if it is M-compatible and closed w.r.t. its unmatched leaves. T' is* minimally semi-closed *if T' is semi-closed but any proper subtree of T' is not semi-closed.*

For a semi-closed subtree T' of T/I let us use the following notation:

- M' is the set of links in M with both endnodes in T'.
- U' is the set of leaves of T' unmatched by M.

Our algorithms maintain the following invariant:

Partial Solution Invariant.
The partial solution I is obtained by sequentially applying a greedy contraction or contracting a minimally semi-closed tree T' into a leaf.

Definition 5.4 (dangerous semi-closed tree). *A semi-closed subtree of T/I is dangerous (w.r.t. a matching M) if it is as in Definition 3.5 with $bb' \in M$.*

In [8, 17] the following is proved:

Lemma 5.1. [8, 17] *Suppose that the Partial Solution Invariant hold for T, M, and I, and that T/I has no greedy contraction. Then there exists a polynomial time algorithm that finds a non-dangerous semi-closed tree T' of T/I and an exact cover $I' \subseteq E$ of T' of size $|I'| = |M'| + |U'|$.*

A formal description of the algorithms is given in Algorithms 1 and 2. Algorithm 1 and its dual-fitting analysis are our main results, since this analysis is completely new. Our algorithms differ from previous algorithms in the matching M computed at step 2. In Algorithm 1 the matching M is only required to be *inclusion maximal*, while all previous algorithms computed a *maximum size* matching in $\delta(L, L) \setminus W$. This is a substantial difference, since otherwise, to perform an LP-based analysis, one needs to add the constraints (6), as we will do in the analysis of Algorithm 2.

Algorithm 1: DUAL-FITTING$(T = (V, \mathcal{E}), E)$ (ratio: $\rho = 7/4$)

```
1 initialize: I ← ∅, C ← {r}.
2 M ← maximal matching in δ(L, L) \ W, U ← leaves unmatched by M.
3 Contract every link ab ∈ W with a, b ∈ U.
4 Exhaust greedy contractions and update I, C accordingly.
5 while T/I has more than one node do
6     Find T', I' as in Lemma 5.1.
7     Contract T' with I'.
8     Exhaust greedy contractions and update I, C accordingly.
9 return I
```

Algorithm 2: PRIMAL-FITTING($T = (V, \mathcal{E}), E$) (ratio: $\rho = 7/4$)

initialize: $C \leftarrow \{r\}$.
$F_L \leftarrow$ min-w-weight exact edge-cover of L,

$$w_e = \begin{cases} \rho & e \in \delta(L, L) \setminus W \\ \rho - \frac{1}{2} & e \in \delta(L, V \setminus L) \\ \rho + \frac{1}{2} & e \in W \end{cases}$$

$M \leftarrow \delta_{F_L}(L, L)$, $U \leftarrow$ the set leaves of T unmatched by M
$I \leftarrow M \cap W$, $M \leftarrow M \setminus W$.
Exhaust greedy contractions and update I, C accordingly.
while T/I has more than one node **do**
    Find T', I' as in Lemma 5.1.
    Contract T' with I'.
    Exhaust greedy contractions and update I, C accordingly.
return I

Our algorithms are supplemented by an *LP-based analysis* to achieve ratios better than 2 w.r.t. to the following two linear programs (LP1) and (LP2), where

- (LP1) is defined by the constraints (1), (2), (3), and (4).
- (LP2) is defined by the constraints (1), (2), (3), (5), and (6).

$$\begin{array}{lll} & \min & x(E) \\ \textbf{(LP1)} & \text{s.t.} & x_e \geq 0 \qquad\qquad\qquad\qquad \forall e \in E \qquad (1) \\ & & x(\delta(T')) \geq 1 \qquad\qquad\qquad \forall T' \in \mathcal{T} \qquad (2) \\ & & x(\sigma(s_e)) - x_e \geq 0 \qquad\qquad \forall e \in W \qquad (3) \\ & & x(\zeta(T')) \geq 1 \qquad\qquad\qquad \forall T' \in \mathcal{N} \qquad (4) \end{array}$$

$$\begin{array}{lll} & \min & x(E) \\ \textbf{(LP2)} & \text{s.t.} & x_e \geq 0 \qquad\qquad\qquad\qquad \forall e \in E \qquad (1) \\ & & x(\delta(T')) \geq 1 \qquad\qquad\qquad \forall T' \in \mathcal{T} \qquad (2) \\ & & x(\sigma(s_e)) - x_e \geq 0 \qquad\qquad \forall e \in W \qquad (3) \\ & & x(\delta(v)) = 1 \qquad\qquad\qquad\quad \forall v \in L \qquad (5) \\ & & x(\delta(A, V)) \geq \lceil |A \cap L|/2 \rceil \quad \forall A \in \mathcal{O}_L \qquad (6) \end{array}$$

Theorem 5.2. *Algorithm 1 computes a solution I of size at most $7/4$ times the optimal value of (LP1).*

Theorem 5.3. *Algorithm 2 computes a solution I of size at most $7/4$ times the optimal value of (LP2).*

6. Dual-fitting analysis of Algorithm 1 (Theorem 5.2)

For a link $e \in E$ let us use the following notation:

- $\delta_{\mathcal{T}}^{-1}(e) = \{T' \in \mathcal{T} : e \in \delta(T')\}$; recall that $\mathcal{T}$ is the family of proper rooted subtrees of T.
- $\sigma_S^{-1}(e) = \{s \in S : e \in \sigma(s)\}$; recall that S is the set of stems of T.
- $\zeta_{\mathcal{N}}^{-1}(e) = \{T' \in \mathcal{N} : e \in \zeta(T')\}$; recall that $\mathcal{N}$ is the family of bad trees.

With this notation, the dual LP of (LP1) is:

$$
\textbf{(D)} \qquad
\begin{array}{lll}
\max & y_{T'} + q(\mathcal{T}) & \\
\text{s.t.} & y(\delta_{\mathcal{T}}^{-1}(e)) + z(\sigma_S^{-1}(e)) - |\{e\} \cap W| z_e + q(\zeta_{\mathcal{N}}^{-1}(e)) \leq 1 & \forall e \in E \\
& y_{T'} \geq 0 & \forall T' \in \mathcal{T} \\
& z_w \geq 0 & \forall w \in W \\
& q_{T'} \geq 0 & \forall T' \in \mathcal{N}
\end{array}
$$

We rewrite Algorithm 1 with the updates of the dual variables as Algorithm 3.

Algorithm 3: DUAL-UPDATE($T = (V, \mathcal{E}), E$) (ratio: $\rho = 7/4$)

initialize: $C \leftarrow \{r\}$;
$z_e \leftarrow \rho - 1$ for every link $e = ab \in W$ with $a, b \in U$.
Exhaust greedy contractions and update I, C accordingly.
$y \leftarrow 0$, $z \leftarrow 0$, $q \leftarrow 0$.
$M \leftarrow$ maximal matching in $E(L, L) \setminus W$, $U \leftarrow$ leaves unmatched by M.
$y_v \leftarrow 1$ if $v \in U$, $y_v \leftarrow \rho - 1$ if $v \in L \setminus U$.
$I \leftarrow M \cap W$, $M \leftarrow M \setminus W$. **while** T/I has more than one node **do**
    Find T', I' as in Lemma 5.1.
    Case 1: $|C'| = 0$ and either: $|M'| = 0$ or $|M'| = 1, |U'| \geq 2$
        $y_{T'} \leftarrow \rho - 1$
        $y_v \leftarrow \rho - 1$ if $v \in U'$ and $y_v \leftarrow 0$ if $v \in L' \setminus U'$.
    Case 2: $|C'| = 0$ and $|M'| = |U'| = 1$ (so $T' \in \mathcal{N}$)
        $q_{T'} \leftarrow \rho - 1$
    Contract T' with I'.
    Exhaust greedy contractions and update I, M, C accordingly.
return I

7. Intuition for the analysis of the dual-fitting algorithm

Recall that C' is the non leaf compound nodes in T'.
We define a **dual load invariant** for every edge e.

$$\mu(e) = y(\delta_{\mathcal{T}}^{-1}(e)) + z(\sigma_S^{-1}(e)) - |\{e\} \cap W| z_e + q(\zeta_{\mathcal{N}}^{-1}(e)) \leq \rho \ .$$

When covering T' the dual load of T' is raised from 0 to $\rho - 1$ (and later added to the dual load of the new leaf c created). This creates a problem that needs to carefully be checked since $y_{T'}$ is part of the load of every link with at least one endpoint in T'. The dual load invariant requires that we show that $y_v + y_u + y_{T'} \leq \rho$.

Initially unmatched leaves get dual load 1, while unmatched leaves get a load of $\rho - 1$ Since there is no greedy step we do not have two linked leaves with load 1 each, and so its easy to see that for every $e = uv$ at the beginning, the dual load invariant holds. When a matched node enters T' his dual load is set to 0 and when an unmatched node enters T' we set its dual load to $\rho - 1$.

After a tree T' is covered and gets dual load $y_{T'} = \rho - 1$ and need to show that if $e = uv$, $y_{T'} + y_u + y_v \leq \rho$. Consider for example a matched node v linked to an unmatched node u outside T' and let us bound the dual load on $e = vu$. Since the dual load of v is zeroed the total dual load is at most $y_{T'} y_v + y_u \leq \rho - 1 + 0 + 1 \leq \rho$. Other cases are similar.

Dual credit: As we increase the dual, we may distribute credit on nodes and links. The credit we have is the value of the dual optimization function (which is a lower bound on the optimum) but we need to show that enough credit is left after we pay for links taken into the solution. In particular, the first term in the dual objective function is spread as $\pi(v)$ on vertices so that the sum of $\pi(v)$ exactly equals the first term in the dual LP.

The dual LP collects credit *in advance* from bad trees in $\mathcal{N}$. Recall that such trees can not be covered efficiently. See the q variable in the first line of the dual LP. This variable is raised when we find a bad tree, and indeed we know that there is no efficient way to cover this tree.

We need the invariant that the dual credit $\pi(v) \geq 1$, for every node v. This is needed for example for greedy operation. There are several cases to check after a tree T' is covered but the calculations are simple.

There is no relation whatsoever between the dual load and the dual credit.

8. Formal analysis

Definition 8.1. *During the algorithm, the* **dual load** $\mu(e)$ *of a link e is defined as the sum of the dual variables in the constraint of e in the dual program, namely*

$$\mu(e) = y(\delta_{\mathcal{T}}^{-1}(e)) + z(\sigma_S^{-1}(e)) - |\{e\} \cap W| z_e + q(\zeta_{\mathcal{N}}^{-1}(e)) \ .$$

The **dual credit** $\pi(v)$ *of a node v is defined as follows. Let $\pi'(v)$ be the sum of the dual variables y and q that correspond to v minus the number of links used by the algorithm to contract the corresponding tree into v. Then $\pi(v) = \pi'(v)$ if v does not contain r, and $\pi(v) = \pi'(v) + 1$ otherwise.*

The dual load lemma

Lemma 8.1. *The dual load of any link is always at most ρ.*

Proof: Let $e = uv$. Its easy to see that this holds after the initialization. Now, consider what happens after a tree T' is covered. In this case we zero

the dual variables of the endnodes of the links in M' that enter T', if any. Unmatched leaves are assigned the (possibly lower) value $\rho - 1$, and we raise the dual variable $y_{T'}$ of T' from 0 to $\rho - 1$. The problem is with the latter term as it increases the dual load on e, for every link with at least one endpoint in T'. We need to show that $y_u + y_v + y_{T'} \leq \rho$.

First consider e that has exactly one endnode u in T'. Then u must be matched (since T' is closed with respect to unmatched leaves). Recall that the dual load of nodes matched pairs is set to 0 when T' is covered. Thus $y_{T'} + y_v + y_u \leq 0 + 1 + \rho - 1 = \rho = \rho$

Consider the case that both uv are inside T. If both vertices are matched, their dual load is set to 0 and we get $y_u + y_v + y_{T'} = \rho - 1$. Note that it cant be that both u and v are unmatched as this gives a greedy step. Say that u is an unmatched leaf, and thus v is a matched leaf. Thus y_v is set to 0 and y_u is set to $\rho - 1$ and thus $y_{T'} + y_u \leq 2(\rho - 1) < \rho$. Hence at the end of the operation we have $\mu(e) \leq \rho$, and e enters the new compound node c $\square$

The credit lemma

Lemma 8.2. *we can always pay for all links added and leave credit $\pi(c) \geq 1$ on the newly created node c.*

Proof: Consider a greedy operation. This takes two nodes u and v so that $\pi(v), \pi(u) \geq 1$ and contract v, u into a new node c. Since we need to pay one unit of cost, at least one unit of credit is left at c.

An easy case is when $|C'| \geq 1$ or $|M| \geq 2$. In this case:

$$\pi(c) \geq (\pi(C') + 2(\rho - 1)|M'| + |U'|) - (|M'| + |U'|) \geq \pi(C') + |M'|/2 \geq 1 \ .$$

From now on we deal with $|M'| \leq 1$ starting with $|M'| = 1$ and $|U'| \geq 2$. In this case the dual credit is

$$\begin{aligned} \pi(c) &\geq (y_{T'} + \pi(U') + 2 \cdot (\rho - 1)|M'|) - (|M'| + |U'|) \\ &\geq (\rho - 1) + 2 \cdot (\rho - 1)|M'| - |M'| \geq 1.2 \end{aligned}$$

We used the fact that $\pi(U') \geq |U'|$.

If $|M'| = |U'| = 1$ we get a bad tree (there are 3 leaves and one is unmatched leaf a that by definition is T'-closed. Thus the matched link is locking. However, we assume that T' is not dangerous, and so its bad). In this case we only raise the dual variable $q_{T'}$ of a bad tree to $\rho - 1$. The credit left is:

$$\pi(c) \geq (q_{T'} + \pi(U') + 2(\rho - 1)|M'|) - (|M'| + |U'|) \geq 3(\rho - 1) - 1 = 1.2$$

Finally, we deal with the case $M' = \emptyset$. In this case T' is leaf-closed (namely $|U'|$-closed). Say that a new node c is created. Each node in v has $\pi(v) \geq 1$ and so we have the U' units of credit needed to cover the tree. This is a local ratio step. Namely the links used so far to cover the trees inside c can not cover any links outside c. This implies that the dual value grows by 1 (we found a new leaf that can not be covered by previous links). Hence we may assign $\pi(c) = 1$ without loss of generality.

□

The above lemma implies that at the end of the algorithm, the dual solution (y, z, q) violates the dual constraints by a factor of ρ, and thus $(y, z, q)/\rho$ is a feasible solution to the dual program. Hence by the Weak Duality Theorem, $y(\mathcal{E})+q(\mathcal{T}) \leq \rho\tau$, where τ is the optimal LP value. If $\rho \geq 1.75$, then the unique compound node (that contains r) has dual credit at least 1, and thus our dual solution fully pays for the links added, namely, $y(\mathcal{E}) + q(\mathcal{T}) \geq |I|$. Consequently, for $\rho = 1.75$ we get $|I| \leq y(\mathcal{E}) + q(\mathcal{T}) \leq \rho\tau$, as required.

9. Primal-fitting analysis of Algorithm 2 (Theorem 5.3)

9.1. Reduction to the minimum weight leaf edge-cover problem

Let Π be the polyhedron defined by the constraints of (LP1), namely:

$$\begin{aligned}
x_e &\geq 0 && \forall e \in E && (1)\\
x(\delta(T')) &\geq 1 && \forall T' \in \mathcal{T} && (2)\\
x(\sigma(s_e)) - x_e &\geq 0 && \forall e \in W && (3)\\
x(\delta(v)) &= 1 && \forall v \in L && (5)\\
x(\delta(A, V)) &\geq \lceil |A \cap L|/2 \rceil && \forall A \in \mathcal{O}_L && (6)
\end{aligned}$$

Let $\tau = \min\{x(E) : x \in \Pi\}$ be the optimal value of (LP2). Let $R = V \setminus (L \cup S)$. Let $\rho \geq 1.5$ be a parameter set later to $\rho = 7/4$. Recall the weight function w on $E(L, V)$ defined at step 2 of Algorithm 2:

$$w_e = \begin{cases} \rho & \text{if } e \in \delta(L, L) \setminus W \\ \rho - \frac{1}{2} & \text{if } e \in \delta(L, V \setminus L) \\ \rho + \frac{1}{2} & \text{if } e \in W \end{cases}$$

Lemma 9.1. *Let F_L be a minimum w-weight exact edge-cover of L and $x \in \Pi$ such that $x(E) = \tau$. Then:*

$$\rho\tau \geq w(F_L) + \frac{1}{2}\sum_{v \in R} x(\delta(v)) \tag{7}$$

Proof: Let Π_L be the polyhedron defined by the constraints (1), (5), and (6). Then Π_L is the convex hull of the exact edge-covers of L, see [22, Theorem 34.2] in the book by A. Schrijver. Let x' be defined by $x'_e = x_e$ if $e \in \delta(L, V)$ and $x'_e = 0$ otherwise. Note that $x' \in \Pi_L$, since x satisfies (1), (5), and (6). Since F_L is an optimal (integral) exact cover of L with respect to the weights w_e and $x' \in \Pi_L$, we have:

$$x' \cdot w \geq w(F_L) \ .$$

Assign ρx_e tokens to every $e \in E$. The total amount of tokens is exactly $\rho x(E) = \rho\tau$. We will show that these tokens can be moved around such that the following holds:

(i) Every $e \in \delta(L, L)$, and thus every $e \in W$, keeps its initial ρx_e tokens.

(ii) Every $e \in \delta(L, V \setminus L)$ keeps $(\rho - \frac{1}{2})x_e$ tokens from its initial ρx_e tokens.

(iii) Every $v \in R$ gets $\frac{1}{2}x_e$ token for each $e \in \delta(v)$.

(iii) Every $e \in W$ gets additional $\frac{1}{2}x_e$ token, to a total of $(\rho + \frac{1}{2})x_e$ tokens.

This distribution of tokens is achieved in two steps. In the first step, for every $e \in E$, move $\frac{1}{2}x_e$ token from the ρx_e tokens of e to each non-leaf endnode of e, if any. Note that after this step, (i), (ii), and (iii) hold. In the second step, every $e \in W$ gets $\frac{1}{2}x(\sigma(s_e))$ tokens moved at the first step to its stem s_e by the links in $\sigma(s_e)$. The amount of such tokens is at least $\frac{1}{2}x_e$, by (3). This gives an assignment of tokens as claimed. □

To prove Theorem 5.3 we prove the following.

Theorem 9.2. *For $\rho = 7/4$, Algorithm 2 computes a solution I of size at most the right-hand size of (7). Thus $|I| \leq \rho\tau = \frac{7}{4}\tau$.*

9.2. Analysis of the algorithm (Proof of Theorem 9.2)

Let $M = \delta_{F_L}(L, L)$ be the set of leaf-to-leaf links in F_L and U the set of leaves unmatched by M. Then for $\rho = 7/4$ we have:

$$w(F_L) = \rho|M\backslash W| + \left(\rho - \frac{1}{2}\right)|U| + \left(\rho + \frac{1}{2}\right)|M \cap W| = \frac{7}{4}|M\backslash W| + \frac{5}{4}|U| + \frac{9}{4}|M \cap W| \,.$$

Thus (7) implies:

$$\rho\tau \geq \frac{7}{4}|M \setminus W| + \frac{5}{4}|U| + \frac{9}{4}|M \cap W| + \frac{1}{2}\sum_{v \in R} x(\delta(v)) \,. \tag{8}$$

For the analysis, we will assign tokens to nodes and edges of T according to the r.h.s. of (8), plus 1 extra token to (the compound node) r. Each time a contraction is performed (lines 3,4,7,8 in Algorithm 2), we assign 1 token to the compound node that results from the contraction. For example, every link $e \in M \cap W$ own 9/4 tokens, and when it is added to the partial solution I at step 3 of Algorithm 2, these 9/4 tokens pay both for the link addition and for the token assigned to the resulting compound node of T/I (and a spare of 1/4 token remains). After all links in $M \cap W$ are moved from M to I, we maintain the following invariant for the tree T/I and for links in M and nodes in R that are not yet contracted into compound nodes.

Tokens Invariant.

(i) Every $e \in M \setminus W$ owns $\rho = \frac{7}{4}$ tokens.

(ii) Every non-compound leaf unmatched by M owns $\rho - \frac{1}{2} = \frac{5}{4}$ tokens.

(iv) Every $v \in R$ owns $\frac{1}{2}x(\delta(v))$ tokens.

(iii) Every compound node owns 1 token.

For a subtree T' of T/I let us use the following notation:

- M' is the set of (not yet contracted) links in M with both endnodes in T'.
- U' is the set of leaves of T' unmatched by M.
- U'_0 is the set of original (non-compound) leaves of T' unmatched by M.

- C' is the set of non-leaf compound nodes of T' (this includes r, if $r \in T'$).
- R' is the set of (not yet contracted) nodes in R that belong to T'.
- $\Sigma' = \sum_{v \in R'} x(\delta(v))$

Let $tokens(T')$ denote the amount of tokens in T'; this includes the tokens on nodes of T' and tokens of links in M with both endnodes in T', namely:

$$\begin{aligned} tokens(T') &= \frac{7}{4}|M'| + (|C'| + |U'| - |U'_0|) + \frac{5}{4}|U'_0| + \frac{1}{2}\Sigma' \\ &= \frac{7}{4}|M'| + |U'| + \frac{1}{4}|U'_0| + \frac{1}{2}\Sigma' + |C'| \end{aligned}$$

If we require not to overspend the credit provided by (8), then each time we contract T' with I' we need the following property.

Definition 9.1. *A contraction of T' with I' is* legal *if $tokens(T') \geq |I'|+1$.*

This means that the set I' of links added to I and the 1 token assigned to the new compound node are paid by the total amount of tokens in T'. We do only legal contractions, which implies that at any step of the algorithm

$$|I| + tokens(T/I) \leq tokens(T) \ .$$

Thus at the last iteration, when T/I becomes a single compound node, $|I|$ is at most the right-hand side of (8).

Recall that after step 3, we have only two types of contractions of T' with I': a greedy contraction of a path by a single link between two unmatched leaves, and a contraction of a semi-closed tree with a link set of size $|I'| = |M'| + |U'|$. In the case of a greedy contraction, $tokens(T') \geq |U'| = 2$ while $|I'| = 1$; thus this contraction is legal. For a semi-closed subtree T' of T/I, we prove the following.

Lemma 9.3. *Suppose that the Partial Solution Invariant and the Tokens Invariant hold for T, M, and I, and that T/I has no greedy contraction. Then $tokens(T') \geq |M'| + |U'| + 1$ holds for any non-dangerous semi-closed subtree T' of T/I.*

9.3. Proof of Lemma 9.3

Let T' be a semi-closed subtree of T/I w.r.t. M with root r' and node set V'. Assume that $tokens(T') - (|M'| + |U'|) < 1$. We will show that T' is dangerous. This contradict the assumption in Lemma 9.3, (namely, the assumption that T is not dangerous). Note that by the Tokens Invariant:

$$tokens(T') - (|M'| + |U'|) = \frac{3}{4}|M'| + \frac{1}{4}|U'_0| + \frac{1}{2}\Sigma' + |C'| = \frac{1}{4}(3|M'| + |U'_0| + 2\Sigma') + |C'|$$

Since we assume that $tokens(T') - (|M'| + |U'|) < 1$, this immediately implies:

Lemma 9.4. *$|C'| = 0$ and $3|M'| + |U'_0| + 2\Sigma' < 4$; thus $|M'| \leq 1$, and if $|M'| = 1$ then $|U'_0| = 0$ and $\Sigma' < 1/2$.*

Let us use the following additional notation:

- L' is the set of leaves of T'.
- S' is the set of (the original) stems of T'.

Lemma 9.5. *$|S'| = 0$.*

Proof: Note that the Partial Solution Invariant implies that every stem s in T/I has exactly two leaf descendant, and they are both original leaves. Let a, b be the two leaf descendants of s, so a, b are original leaves and ab is a twin link. Since $ab \in W$, $ab \notin M'$. From the assumption that that T/I has no link greedy contraction we get that one of a, b is matched by M, as otherwise ab gives a greedy contraction. Moreover, $|M' \cap W| = 0$ and $|M'| \leq 1$ implies that $|M'| = 1$ and exactly one of a, b is matched by M. Consequently, $|M'| = |U'_0| = 1$, contradicting Lemma 9.4. □

Lemma 9.6. *$\Sigma' \geq |U'| + 1 - 2|M'|$.*

Proof: Note that no link has both endnodes in U' (since T/I has no greedy contraction), and that $\delta(U') \cap \delta(T') = \emptyset$ (since T' is U'-closed). Thus

$$x(\delta(U') \cup \delta(T')) = \sum_{v \in U'} x(\delta(v)) + x(\delta(T')) \geq |U'| + 1 \ .$$

Let $e \in \delta(U')$. Then e contributes x_e to Σ', unless e is incident to a matched leaf. However, $x(\delta(b)) = 1$ for every matched leaf b, and the number of matched leaves in T' is exactly $2|M'|$. Hence $\Sigma' \geq |U'| + 1 - 2|M'|$, as claimed. □

Lemma 9.7. *If* $|M'| = 1$ *then* $|U'| = 1$.

Proof: If $|U'| \geq 2$ then Lemma 9.6 gives the contradiction $\Sigma' \geq 1$. Suppose that $|U'| = 0$. Then $|L'| = 2$, say $L' = \{b, b'\}$, and so $M' = \{bb'\}$, since $|M'| = 1$. Consequently, the contraction of bb' creates a new leaf. We obtain a contradiction by showing that then the path between b and b' in T/I has an internal compound node. By the Partial Solution Invariant b, b' are original leaves. Note that in the original tree T the contraction of bb' does not create a new leaf, since $bb' \notin W$. This implies that in T, there is a subtree $\hat{T}$ of T hanging out of a node z on the path between b and b' in T. This subtree $\hat{T}$ is not present in T/I, hence it was contracted into a compound node during the construction of our partial solution I. Thus T/I has a compound node $\hat{z}$ that contains $\hat{T}$, and since $\hat{z}$ contains a node z that belongs to the path between b and b' in T, the compound node of T/I that contains z belongs to the path between b and b'in T/I. □

Corollary 9.8. $|C'| = |S'| = |U'_0| = 0$, $|M'| = |U'| = 1$ *(thus* T' *has* 3 *leaves), and* $\Sigma' < 1/2$.

Proof: We have $|C'| = 0$ and $|M'| \leq 1$ by Lemma 9.4 and $|S'| = 0$ by Lemma 9.5. If $|M'| = 0$ then from Lemma 9.6 we get that $\Sigma' \geq 2$, contradicting Lemma 9.4. Thus $|M'| = 1$ and by Lemmas 9.4 and 9.7 we have $\Sigma' < 1/2$, $|U'| = 1$, and $|U'_0| = 0$. □

We now use the properties of T' summarized in Corollary 9.8 to show that T' must be dangerous. Let bb' be the matched pair and a the unmatched (compound) leaf of T'. Let u and u' be the least common ancestor of ab and ab', respectively, and assume w.l.o.g. that u is a descendant of u' (see Fig. 3, and note that $u = u'$ or/and $u' = r'$ may hold). Let $x_{ab} = \alpha$, $x_{bb'} = \beta$, $x_{ab'} = \gamma$, $x(\delta(b, T \setminus T') = \epsilon$, and $x(\delta(b', T \setminus T') = \theta$.

Lemma 9.9. $\alpha, \theta > 0$ *or* $\gamma, \epsilon > 0$*; if* $u \neq u'$ *then* $\gamma, \epsilon > 0$.

Proof: Consider the contribution to Σ' of links in cuts $\delta(a)$ and $\delta(T_{r'})$:

(i) Cut $\delta(a)$: $\frac{1}{2} > \Sigma' \geq x(\delta(a)) - (\alpha + \gamma) \geq 1 - (\alpha + \gamma)$; hence $\alpha + \gamma > \frac{1}{2}$.

(ii) Cut $\delta(T_{r'})$: $\frac{1}{2} > \Sigma' \geq x(\delta(T_{r'})) - (\theta + \epsilon) \geq 1 - (\theta + \epsilon)$; hence $\theta + \epsilon > \frac{1}{2}$.

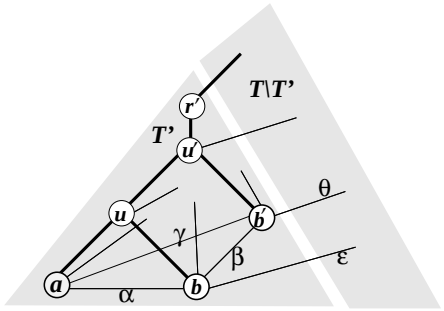

Figure 3: Illustration to the proof of Lemma 9.9.

In particular, we cannot have $\alpha, \gamma = 0$ or $\theta, \epsilon = 0$. We show that each one of the cases $\alpha, \epsilon = 0$ or $\gamma, \theta = 0$ is also not possible.

If $\alpha, \epsilon = 0$ then $\gamma, \theta > \frac{1}{2}$, giving the contradiction $1 = x(\delta(b')) \geq \gamma + \theta > 1$.

If $\gamma, \theta = 0$ then $\alpha, \epsilon > \frac{1}{2}$, giving the contradiction $1 = x(\delta(b)) \geq \alpha + \epsilon > 1$.

Now let us consider the case $u \neq u'$. Then by considering the cut $\delta(T_u)$ we get: $1/2 > \Sigma' \geq x(\delta(T_u)) - (\beta + \gamma + \epsilon) \geq 1 - (\beta + \gamma + \epsilon)$; hence $\beta + \gamma + \epsilon > 1/2$.

If $\gamma = 0$ then $\beta + \epsilon > 1/2$, and $\alpha > 1/2$ by (i); by considering the cut $\delta(b)$ we get the contradiction $1 = x(\delta(b)) \geq \alpha + \beta + \epsilon > 1/2 + 1/2 = 1$.

If $\epsilon = 0$ then $\beta + \gamma > 1/2$, and $\theta > 1/2$ by (ii); by considering the cut $\delta(b')$ we get the contradiction $1 = x(\delta(b')) \geq \beta + \gamma + \theta > 1/2 + 1/2 = 1$. □

Lemma 9.9 implies that T' is dangerous. Indeed, if $u \neq u'$, then $\gamma > 0$ implies that the link ab' exists, and $\epsilon > 0$ implies that T' is b-open. Thus, by the definition, T' is dangerous. The same holds if $u = u'$ and $\gamma, \epsilon > 0$. If $u = u'$ and $\alpha, \theta > 0$, then ab exists (since $\alpha > 0$) and T' is b'-open (since $\theta > 0$); thus by exchanging the roles of b, b' we get that T' is dangerous, by the definition.

This concludes the proof of Lemma 9.3.

References

[1] David Adjiashvili. Beating approximation factor two for weighted tree augmentation with bounded costs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2384–2399, 2017.

[2] J. Cheriyan and Z. Gao. Private communication. Manuscript, 2014.

[3] J. Cheriyan and Z. Gao. Approximating (unweighted) tree augmentation via lift-and-project, part II. Manuscript, 2015.

[4] J. Cheriyan, T. Jordán, and R. Ravi. On 2-coverings and 2-packing of laminar families. In *ESA*, pages 510–520, 1999.

[5] Y. Chu and T. Liu. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400, 1965.

[6] N. Cohen and Z. Nutov. A $(1 + \ln 2)$-approximation algorithm for minimum-cost 2-edge-connectivity augmentation of trees with constant radius. *Theoretical Computer Science*, 489-490:67–74, 2013.

[7] G. Even, J. Feldman, G. Kortsarz, and Z. Nutov. A 3/2-approximation for augmenting a connected graph into a two-connected graph. In *APPROX*, pages 90–101, 2001.

[8] G. Even, J. Feldman, G. Kortsarz, and Z. Nutov. A 1.8-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Transactions on Algorithms*, 5(2), 2009.

[9] Samuel Fiorini, Martin Groß, Jochen Könemann, and Laura Sanità. A 3/2-approximation algorithm for tree augmentation via chvátal-gomory cuts. *CoRR*, abs/1702.05567, 2017.

[10] G. N. Frederickson and J. Jájá. Approximation algorithms for several graph augmentation problems. *SIAM J. Computing*, 10:270–283, 1981.

[11] G. N. Frederickson and J. Jájá. On the relationship between the biconnectivity augmentation and traveling salesman problem. *Theoretical Computer Science*, 19(2):189–201, 1982.

[12] M. Goemans, A. Goldberg, S. Plotkin, E. Tardos D. Shmoys, and D. Williamson. Improved approximation algorithms for network design problems. In *SODA*, pages 223–232, 1994.

[13] M. Goemans and D. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Computing*, 24(2):296–317, 1995.

[14] K. Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[15] S. Khuller. Approximation algorithms for finding highly connected subgraphs (chapter 6). In *Approximation algorithms for NP-hard problems (Ed. D. S. Hochbaum)*. PWS, Boston, 1996.

[16] S. Khuller and R. Thurimella. Approximation algorithms for graph augmentation. *J. of Algorithms*, 14:214–225, 1993.

[17] G. Kortsarz and Z. Nutov. A simplified 1.5-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. To appear in *Transactions on Algorithms*, 2014.

[18] L. C. Lau, R. Ravi, and M. Singh. *Iterative Methods in Combinatorial Optimization*. Cambridge University Press, 2011.

[19] Y. Maduel and Z. Nutov. Covering a laminar family by leaf to leaf links. *Discrete Applied Mathematics*, 158(13):1424–1432, 2010.

[20] H. Nagamochi. An approximation for finding a smallest 2-edge connected subgraph containing a specified spanning tree. *Discrete Applied Math.*, 126:83–113, 2003.

[21] Zeev Nutov. A note on the tree augmentation problem. *CoRR*, abs/1703.07247, 2017.

[22] A. Schrijver. *Combinatorial Optimization, Polyhedra and Efficiency*. Springer-Verlag Berlin, Heidelberg New York, 2004.

[23] A. Sebo and J. Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, 34(5):597–629, 2014.