

A 1.5-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2^{*}

Guy Even
Tel-Aviv University
guy@eng.tau.ac.il

Guy Kortsarz[†]
Rutgers University, Camden
guyk@crab.rutgers.edu

Zeev Nutov
The Open University of Israel
nutov@openu.ac.il

Abstract

We present a 1.5-approximation algorithm for the following NP-hard problem: given a connected graph $G = (V, \mathcal{E})$ and an edge set E on V disjoint to \mathcal{E} , find a minimum size subset of edges $F \subseteq E$ such that $(V, \mathcal{E} \cup F)$ is 2-edge-connected. Our result improves and significantly simplifies the approximation algorithm with ratio $1.875 + \varepsilon$ of Nagamochi

1 Introduction

1.1 Problem definition and our result

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 $G = (V, \mathcal{E})$ and a set of additional edges (called “links”) E on V disjoint to \mathcal{E} , find a minimum size edge set $F \subseteq E$ so that $G + F = (V, \mathcal{E} \cup F)$ is 2-edge-connected. The 2-edge-connected components of the given graph G form a tree. It follows that by contracting these components, one may assume that G is a tree. Hence, our problem is:

Tree Augmentation Problem (TAP)

Instance: A tree $T = (V, \mathcal{E})$ and a set of links E on V disjoint to \mathcal{E} .

Objective: Find a minimum size subset $F \subseteq E$ of edges such that $T \cup F$ is 2-edge-connected.

TAP is sometimes posed as the problem of covering a laminar family (see e.g. [1]). Namely, given a laminar family \mathcal{E} on a groundset V , and an edge set E on V , find a minimum size $F \subseteq E$ such that for every $S \in \mathcal{E}$, there is an edge in F with one endpoint in S and the other in $V - S$.

^{*}Preliminary version in APPROX 2001, pp. 90-101.

[†]Partially supported by NSF grant number 0728787.

TAP 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 minimal cuts of a k -connected graph with k odd is laminar.

The first 2-approximation for TAP (which also holds for the weighted version) was given by Frederickson & Jájá [4]. TAP is APX-hard even if the set E of links forms a cycle on the leaves of T [1]. Improving the approximation ratio below 2 was a long-standing open problem posed by Khuller in [9] as one of the main open problems in connectivity augmentation. The first approximation ratio below 2 was given by Nagamochi [12]; he gave a $(1.875 + \varepsilon)$ -approximation algorithm for TAP. In the conference version [3] we presented and sketched a proof of a 1.5-approximation algorithm for the problem. Here we give a full and substantially simplified proof of this result.

Theorem 1.1 *TAP admits a 1.5-approximation algorithm.*

Several ideas introduced by Nagamochi in [12] are used in this paper (e.g., minimally leaf-closed trees and some parts of the lower bound). Key to our analysis is a credit scheme for using the different parts of the lower bound. This credit scheme has a “static” component and a “dynamic” component. The static credit is computed in the beginning of the algorithm and is distributed to different parts of the graph. The dynamic credit is available only after the algorithm “reveals” it by proving that the optimum solution had to be larger than the initial static estimate. We believe that this technique has merit of its own and may be useful for other connectivity problems.

1.2 Related work

Several particular cases of TAP were considered in the literature. Eswaran & Tarjan [2] presented a linear time algorithm for the case when any edge is legal, namely when E forms a complete graph. A more general problem than TAP is the minimum weight 2-Edge-Connected Spanning Subgraph (2-ECSS) problem, where the goal is to find a minimum weight 2-edge-connected spanning subgraph H of a given complete graph G . TAP is a particular case, when G contains a tree T (or a connected graph) of weight 0, and any other edge in G has weight in $\{1, \infty\}$ (every link has weight 1, while any other edge of G not in T has weight ∞).

A problem sandwiched between 2-ECSS and TAP is weighted TAP, when the edges in E have weights and the goal is to find a minimum weight augmenting edge set. There are few 2-approximation algorithms for weighted TAP. The first algorithm, by Frederickson and Jájá [4] was simplified later by Khuller and Thurimella [10]. These algorithms are based on constructing a directed graph and computing a minimum weight arborescence. The primal-dual algorithm of [6] is another simple combinatorial 2-approximation algorithm for the problem. The iterative rounding algorithm of Jain [7] is an LP-based 2-approximation algorithms. The approximation ratio of 2 for all these algorithms is tight even for TAP. However, while TAP admits an approximation algorithm with ratio better than 2, no such algorithm is known for weighted TAP.

The 2-ECSS problem was vastly studied. For general weights, the best known ratio is 2 by

Fredrickson and Jájá [4], which can also be achieved by the the algorithms in [10] and [7]. For particular cases, better ratios are known. Fredrickson and Jájá [5] showed that when the edge weights satisfy the triangle inequality, the Christofides heuristic leads to a 3/2-approximation algorithm. For the special case of 2-ECSS in which edges have $\{1, \infty\}$ weights, the currently best known ratio is 5/4 [8]. Surveys on broad classes of connectivity problems can be found in [9] and [11].

2 Preliminaries

Let $T = (V, \mathcal{E})$ be a tree. For $u, v \in V$ let $(u, v) \in \mathcal{E}$ denote the edge in T and uv the link in E between u and v . Let $p(uv)$ denote the path between u and v in T . A link uv covers all the edges (and nodes) along the path $p(uv)$. We designate a node r of T as the *root*, and refer to the pair T, r as a *rooted tree* (we do not mention the root when it is clear from the context). The choice of r defines a partial order on V : u is a *descendant* of v (or v is an *ancestor* of u) if v belongs to $p(ru)$; if, in addition, $(u, v) \in T$, then u is a *child* of v , and v is the *parent* of u . The *leaves* of T are the nodes in $V - r$ that have no descendants. We denote the leaf set of T by $L(T)$, or simply by L , when the context is clear. The *rooted subtree* of T induced by v and its descendants is denoted by T_v (v is the root of T_v). A subtree T' of T is called a *rooted subtree* of T if $T' = T_v$ for some $v \in V$. For a node set U and a link set M , $U \cap T'$ is the set of nodes in U that belong to T' , and $M \cap T'$ is the set of links in M with both endnodes in T' .

To *contract* a (not necessarily rooted) subtree T' of T is to combine all nodes in T' into a single 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, but retain their correspondence with the edge or link of the original graph.

If we add a link uv to a partial solution I , then the nodes along the path $p(uv)$ belong to the same 2-edge-connected component of the augmented graph $(V, \mathcal{E} \cup I)$. Hence, we may contract the nodes along $p(uv)$. 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. Since all contractions are induced by subsets of links, we refer to the contraction of every 2-edge-connected component of $T \cup I$ into a single node simply as the contraction of the links in I .

Definition 2.1 A link $u'v'$ is a shadow of a link uv if $p(u'v') \subseteq p(uv)$.

Every instance can be rendered closed under shadows by adding all shadows of existing links. We refer to the addition of all shadows as *shadow completion*. Shadow completion does not affect the optimal solution size, since every shadow can be replaced by some link covering all edges covered by the shadow. Thus:

Assumption 2.1 The set of links E is closed under shadows, that is, if $uv \in E$ and $p(u'v') \subseteq p(uv)$ then $u'v' \in E$.

The following reductions allow us to make some simplifying assumptions about TAP instances.

Definition 2.2 *A cover F of T is shadows minimal if, for every link $uv \in F$, replacing uv by a proper shadow of uv results in a set of links that does not cover T .*

Claim 2.2 *Let F be a shadows minimal cover of T . Then there is exactly one link in F incident to every leaf of T . In particular, the leaf-to-leaf links in F form a matching on L .*

Proof: Let a be a leaf. If $au, av \in F$ for $u \neq v$, then we can replace the link av by its shadow ending at the parent of a . Since every edge of T remains covered, this contradicts the shadow minimality of F . The statement follows. \square

Further simplifying assumptions can be made when we consider the relationship between pairs of edges in T .

Definition 2.3 *A link is maximal if it is not a shadow of any other link. An edge e of T is reducible with respect to a link set E if there is a unique maximal link that covers e , or if there exists an edge $e' \neq e$ such that every maximal link that covers e' also covers e .*

If an edge (u, v) is covered by a unique maximal link ab , then adding ab to the cover is an optimal step (i.e., the size of an optimal solution in the residual problem is smaller by one). If an edge e is covered by any maximal link that covers $e' \neq e$, then e may be contracted without affecting the sizes of the optimal solution and the computed solution. Hence, we may preprocess the instance so that:

Assumption 2.3 *There are no reducible edges in the tree T with respect to the link set E .*

The following statement shows that if there are no reducible edges, then paths in T consisting of degree 2 nodes can not end at leaves.

Claim 2.4 *If there are no reducible edges, then every parent of a leaf has at least two children.*

Proof: For the sake of contradiction, suppose that v has a unique leaf child a in T . Let u the parent of v . Thus v has degree exactly 2. Note that if there is a link $av \in E$ parallel to the edge (u, v) , then av can not be the only link incident to a , as otherwise the edge (a, v) is reducible. Hence av is not maximal. Thus any maximal link incident to a covers the edge (u, v) , which is a contradiction as T has no reducible edges. \square

For $X, Y \subseteq V$ and a link set F , let $F(X, Y) = \{xy \in F : x \in X, y \in Y\}$ denote the set of links in F that have one endpoint in X and the other in Y ; for $x \in V$ let $\deg_F(x) = |F(x, V)|$ be the degree of x w.r.t. F . For the rest of the paper fix F to be some optimum shadows minimal solution for an instance $T = (V, \mathcal{E}), E$ of TAP.

3 The lower bound and the algorithm

3.1 The Lower Bound

Definition 3.1 A node $s \in V - r$ is a stem of T if it has exactly two children and both of them are leaves. The link between the two children of a stem, if it exists, is called a twin-link.

Claim 3.1 If no edge of T is reducible, then there is a twin-link between the children of every stem.

Proof: Let s be a stem with children a, b , and let v be the parent of s . If $ab \notin E$, then since there are at least two links covering the edge (a, s) (because no edge is reducible), every maximal link that covers (a, s) also covers the edge (s, v) . Hence, (s, v) is reducible, contradicting the assumption. \square

Lemma 3.2 Let L be the set of leaves and S be the set of stems of T , let $X = V \setminus (L \cup S)$, and recall that F is any shadows-minimal cover of T . Let M be a maximum matching of leaf-to-leaf links among all matchings in E not containing twin links. Then:

$$|F| \geq \frac{2}{3}|L| - \frac{1}{3}|M| + \frac{1}{3} \sum_{x \in X} \deg_F(x). \quad (1)$$

Proof: Let \hat{M}_F be the set of twin links in F and let M_F be the set of leaf-to-leaf links in F that are not twin links. Note that $\deg_F(a) = 1$ for every $a \in L$, by Claim 2.2. In particular, $M_F \cup \hat{M}_F$ is a matching, and $|M_F| \leq |M|$ (since M is a maximum matching without twin links). Partition F into $M_F \cup \hat{M}_F$, and all other possible combinations of endpoints of F in L, S and X to get:

$$|F| = |M_F| + |\hat{M}_F| + |F(L, S)| + |F(L, X)| + |F(S, S)| + |F(S, X)| + |F(X, X)| .$$

Clearly

$$|L| = 2(|M_F| + |\hat{M}_F|) + |F(L, S)| + |F(L, X)| . \quad (2)$$

Note that if a, b are children of a stem s and $ab \in \hat{M}_F$, then $\deg_F(s) \geq 1$. Therefore,

$$|\hat{M}_F| \leq |F(L, S)| + 2|F(S, S)| + |F(S, X)| . \quad (3)$$

Multiply (2) by $2/3$, multiply (3) by $1/3$, and then add and rearrange the terms to get:

$$\begin{aligned} \frac{2}{3}|L| - \frac{1}{3}|M_F| &\leq |M_F| + |\hat{M}_F| + |F(L, S)| + \frac{2}{3}|F(L, X)| + \frac{2}{3}|F(S, S)| + \frac{1}{3}|F(S, X)| \leq \\ &\leq |F| - \frac{1}{3}(|F(L, X)| + |F(S, X)| + 2|F(X, X)|) = |F| - \frac{1}{3} \sum_{x \in X} \deg_F(x) . \end{aligned}$$

Thus, since $|M_F| \leq |M|$ we get:

$$|F| \geq \frac{2}{3}|L| - \frac{1}{3}|M_F| + \frac{1}{3} \sum_{x \in X} \deg_F(x) \geq \frac{2}{3}|L| - \frac{1}{3}|M| + \frac{1}{3} \sum_{x \in X} \deg_F(x) .$$

\square

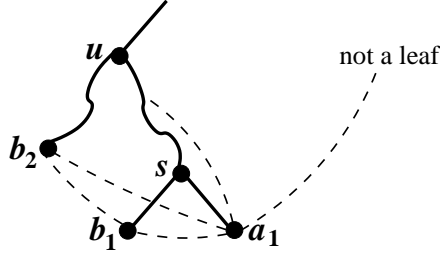


Figure 1: Illustration to Definition 3.2. Links are shown by dashed lines and paths by wiggly lines.

The lower bound in Lemma 3.2 can be used to obtain a 1.8-approximation algorithm. We now present the lower bound we use to obtain a 1.5-approximation algorithm.

Definition 3.2 A leaf a_1 is locked by a link b_1b_2 , and b_1b_2 is a locking link, if the following holds (see Figure 1). The nodes b_1, b_2, a_1 are leaves of a rooted proper subtree T_u of T (so $u \neq r$) so that b_1, a_1 are children of a stem s that is a proper descendant of u , and any leaf to leaf link in E incident to a_1 has its other endnode in $\{b_1, b_2\}$.

Lemma 3.3 Let F, M_F, X be as in Lemma 3.2. Let M be a maximum matching among all matchings in E without twin links and without locking links. Let J be the set of links in F that are not incident to a leaf locked by a link in F . Then:

$$|F| \geq \frac{2}{3}|L| - \frac{1}{3}|M| + \frac{1}{3} \sum_{x \in X} \deg_J(x). \quad (4)$$

Proof: Let M'_F be the set of locking links in M_F and note that $M_F \setminus M'_F$ is a matching without locking links and without twin links, of size $|M_F| - |M'_F|$. As M is a maximum matching of this type, we have $|M| \geq |M_F| - |M'_F|$. Thus $|M_F| \leq |M| + |M'_F|$. Substituting the last inequality in (1) gives

$$\begin{aligned} |F| &\geq \frac{2}{3}|L| - \frac{1}{3}(|M| + |M'_F|) + \frac{1}{3} \sum_{x \in X} \deg_F(x) \\ &= \frac{1}{3}|L| - \frac{1}{3}|M| + \frac{1}{3} \left(\sum_{x \in X} \deg_F(x) - |M'_F| \right). \end{aligned}$$

Now we observe that the last term in the obtained bound equals the last term in (4), since for any link $b_1b_2 \in M'_F$ there is a unique link $a_1x \in F$ with $x \in X$. \square

The term $\frac{2}{3}|L| - \frac{1}{3}|M|$ in (4) can be computed in polynomial time. The other terms depend on F which is not known to us, and are partly revealed during the run of the algorithm.

3.2 The credit scheme and the invariants of the algorithm

We explain how we use the lower bound (4). Let U denote the set of leaves unmatched by M , so $|U| = |L| - 2 \cdot |M|$. By multiplying both sides of (4) by 1.5 and rearranging terms, we obtain

$$\frac{3}{2} \cdot |F| \geq |U| + \frac{3}{2} \cdot |M| + \frac{1}{2} \sum_{x \in X} \deg_J(x). \quad (5)$$

We prove the following statement that implies Theorem 1.1.

Theorem 3.4 *There exists a polynomial time algorithm that for an instance of TAP computes a solution I of size at most the right-hand side of (5). Thus $|I| \leq 1.5 \cdot |F|$.*

Initially, the algorithm assigns *coupons* to unmatched leaves and to links in M so that the total number of coupons assigned equals the sum of the first two terms in the right hand side of (5). For technical reasons, we also assign 1 coupon to the root r .

Initial assignment of coupons:

- Every unmatched leaf gets 1 coupon, and r gets 1 coupon.
- Every link in M gets 3/2 coupons.

Compound nodes. Our algorithm iteratively contracts certain subtrees of T . We refer to the nodes created by contraction as *compound nodes*. Compound nodes always own 1 coupon. Non-compound nodes are referred to as *original nodes* (of T). Each time a contraction takes place, the new compound node gets 1 coupon, which together with the links added is paid by the total credit of the contracted subtree. For technical reasons, r is also considered as a compound node. Our algorithm is designed to maintain the following invariant:

Invariant 3.5 (Coupons Invariant) *Every unmatched leaf and every compound node of T/I (including r) owns 1 coupon, and every link in M owns 3/2 coupons.*

Tickets. To charge the last term $\frac{1}{2} \sum_{x \in X} \deg_J(x)$ in (5), we introduce *tickets*. A ticket worth 1/2 coupon. If we can prove that some $x \in X$ is an endnode of a link in J , then we may claim a ticket (1/2 coupon) at x . In fact, we will often claim tickets in a subtree T' of T without specifying exactly the link $e \in M$ or the node $x \in X$ on which the ticket is claimed. All we care about is that T' contains a ticket. No ticket is claimed twice, if we follow the following simple rule.

Rule for claiming tickets:

Immediately after claiming tickets in T' , the whole tree T' is contracted into a compound node.

This indeed implies that no ticket is claimed twice. If a ticket is claimed at a node x , then x is contracted into a compound node, while tickets are never claimed on compound nodes.

Summarizing, we use the following notation for the credit distributed in a rooted subtree T' of T/I . Let $\text{coupons}(T')$ denote the total number of coupons owned by T' ; this includes the coupons owned by nodes of T' (every unmatched leaf and every compound node of T' owns 1 coupon), and $3/2$ coupons for every link in M with both endnodes in T' . Let $\text{tickets}(T') = \sum_{x \in X \cap T'} \deg_J(x)$ denote the number of tickets in T' . Let $\text{credit}(T') = \text{coupons}(T') + \frac{1}{2}\text{tickets}(T')$.

The high level idea of our algorithm is to repeatedly find a tree T' in T/I and a cover B' of T' so that $\text{credit}(T') \geq |B'| + 1$. In such a case we have enough credit in T' to pay for the $|B'|$ links to cover and contract T' , and to *assign a coupon on the resulting compound node*, to satisfy the Coupons Invariant. If we iteratively cover trees in this way until T is completely covered, then the number of links we use is at most the right hand side of Inequality (5), as claimed in Theorem 3.4. One such simple operation, that relies on coupons only, is as follows.

Definition 3.3 *If there exists a link uv so that $\text{coupons}(p(uv)) \geq 2$ then adding uv to the partial solution, and assigning one coupon to the obtained compound node is called a greedy contraction.*

A greedy contraction, if exists, can be found in polynomial time. One of the steps of the algorithm is to apply all possible greedy contractions exhaustively.

Lemma 3.6 *If all greedy contractions are exhausted, then contraction of a link in M does not create a new leaf.*

Proof: Let $ab \in M$. Note that in the original tree T the contraction of ab does not create a leaf, since M contains no twin links. Thus, if the contraction of ab creates a leaf in T/I , then $p(ab)$ contains a compound node in T/I . This holds as some structure that hanged out of a node of $p(ab)$ and contained a leaf is now gone (this structure was avoiding the contraction of ab in T from creating a new leaf). But then ab gives a greedy contraction, contradicting the assumption. \square

Invariant 3.7 (Matching Invariant) *M is a matching on the original leaves, and a contraction of a link in M does not create a new leaf.*

3.3 The algorithm

Our approach is iteratively to exhaust greedy contractions, and then to find a rooted subtree T' of T/I with its cover B' , to contract T' with B' , and to leave a coupon on the created compound node (in order to satisfy the Coupons Invariant 3.5). If we require not to over spend the credit provided by the right hand side of (5), we need

$$\text{credit}(T') \geq |B'| + 1 \tag{6}$$

to hold. In the next section we will prove the following statement:

Lemma 3.8 *Suppose that Invariants 3.5 and 3.7 hold for T and I . Then there exists a polynomial time algorithm that finds a rooted subtree T' of T/I and a cover $B' \subseteq E$ of T' so that (6) holds, namely: $\text{credit}(T') \geq |B'| + 1$.*

Algorithm *Approx*($T = (V, \mathcal{E}), E$) (A 1.5-approximation algorithm)

Initialization:

- $I \leftarrow \emptyset$.
- Contract reducible edges of T and update I accordingly.
- $M \leftarrow$ maximum matching among leaf-to-leaf non-twin and non-locking links in T/I .

While T/I has more than one node *do:*

- Exhaust greedy contractions and update I accordingly.
- Find a subtree T' of T/I and a cover B' of T' as in Lemma 3.8.
- Contract T' , assign 1 coupon on the new compound leaf, and set $I \leftarrow I \cup B'$.

EndWhile

Algorithm *Approx* initiates $I \leftarrow \emptyset$ as a partial cover. It applies the reductions from Section 2 (specifically, we need Assumptions 2.1 and 2.3), computes a maximum matching M on the leaves among the non-twin and non-locking links, and initiates the credit scheme as described in Section 3.2. In the main loop, the algorithm iteratively exhausts greedy contractions, computes T', B' as in Lemma 3.8, adds B' to I , contracts T' , and leaves a coupon on the created compound leaf. The stopping condition is when I covers T , namely, when T/I is a single node.

It is easy to verify that all actions needed to be taken by the algorithm can be implemented in polynomial time. The credit scheme used implies that the algorithm computes a solution I as in Theorem 3.4, namely $|I|$ is at most 1.5 times the right-hand size of (5). Hence the approximation ratio of 1.5 follows from the lower bound (4), and the proof of Theorem 1.1 is now complete.

4 Proof of Lemma 3.8

4.1 Semi-closed trees and their properties

The *up-link* incident to a node u is defined as the link uv such that v is as close as possible to the root; under Assumption 2.1, v is an ancestor of u . We denote the highest link incident to u by $up(u)$. For a node set $U \subseteq V$, we let $up(U) = \{up(u) : u \in U\}$.

Definition 4.1 ([12]) *Let U be a subset of nodes of T . A rooted subtree T' of T is U -closed if there is no link in E from $U \cap T'$ to $T \setminus T'$. T' is leaf-closed if it is $L(T)$ -closed. A leaf-closed T' is minimally leaf-closed if any proper proper rooted subtree of T' is not leaf-closed.*

Proposition 4.1 ([12]) *A minimally leaf-closed subtree T' of T is covered by $up(L(T'))$.*

A naive approach to find T', B' as in Lemma 3.8 is by taking T' to be minimally leaf-closed and $B' = \text{up}(L(T'))$. However, we do not have enough credit for that, since $|L(T')|$ can be much larger than $\text{coupons}(T')$, not to mention the one extra unit of credit. This is since every link $b_1 b_2 \in M$ owns $3/2$ coupons, while taking the up-links of b_1, b_2 requires 2 coupons. Hence we define T' and its cover B' so that $\text{coupons}(T') \geq |B'|$ holds, and one extra unit of credit is provided with the help of tickets. Our main new idea is to use minimally semi-closed trees, defined below, which admit such a cover B' , assuming invariants from Section 3.2 hold.

Definition 4.2 (Semi-closed tree) *A rooted subtree T' of T/I is semi-closed (w.r.t. a proper matching M) if the following holds:*

- For any $b_1 b_2 \in M$ either both b_1, b_2 belong to T' , or none of b_1, b_2 belongs to T' .
- T' is 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.

Note that a semi-closed T' is *not* leaf-closed (this why we called it “semi-closed”); T' is closed with respect to every unmatched leaf, but matched leaves may have links to nodes outside T' . The following statement explains how we intend to cover *minimally* semi-closed trees.

Lemma 4.2 *For a semi-closed subtree T' of T w.r.t. M let $B(T')$ be the union of $M \cap T'$ and the up-links of the unmatched leaves of T' . If T' is minimally semi-closed then $B(T')$ covers T' .*

Proof: Let T'' be obtained by contracting $M \cap T'$. Note that the leaves of T'' are the unmatched leaves of T' . Otherwise, if T'' has a leaf a that is not a leaf of T' , then the subtree of T' that was contracted into a is a semi-closed tree (with no unmatched leaves), contradicting the minimality of T' . Note also that T'' is minimally leaf-closed, since T' is minimally semi-closed. Thus the up-links of the unmatched leaves of T' cover T'' (if T' has no unmatched leaves then T'' is a single node), by Proposition 4.1. The statement follows. \square

Let T' be a (not necessarily minimally) semi-closed tree and let $C(T')$ be the set of non-leaf compound nodes of T' (recall that this includes r , if $r \in T'$). Note that $|B(T')| = |L(T')| + |M \cap T'|$ and $\text{coupons}(T') = |L(T')| + |C(T')| + \frac{3}{2}|M \cap T'|$. Hence:

$$\text{coupons}(T') - |B(T')| = \frac{1}{2} \cdot |M \cap T'| + |C(T')|. \quad (7)$$

We will show that except one special type (see Definition 4.3), a semi-closed tree T' and $B(T')$ satisfy (6). We will also show that if all semi-closed trees are of this special type, then we can find a “larger” tree T' and its cover B' that satisfy (6).

In what follows, let T' be a (not necessarily minimally) semi-closed tree and let v be the root of T' . Unless stated otherwise, we assume that $v \neq r$.

4.2 Characterizing semi-closed trees with $\text{credit}(T') < |B(T')| + 1$

Intuitively, the next lemma says that if T' has no stems, then every link in F incident to a node in $U \cap T'$ contributes a ticket, unless its other endnode is a matched leaf.

Lemma 4.3 *Suppose that all the compound nodes of T' are leaves, and that T' has no stem. Then:*

$$\text{tickets}(T') \geq |U \cap T'| - 2 \cdot |M \cap T'| + 1 . \quad (8)$$

Furthermore, if T' is leaf-closed then $\text{tickets}(T') \geq 1$.

Proof: T' has no locked leaf, since T' has no stem. Note that there is no link between two nodes in U (as otherwise a greedy contraction applies), and that T' is U -closed. Thus a link in F covering a leaf $a \in U \cap T'$ contributes a ticket, unless its other endnode is a matched leaf. E.g., if $a \in U \cap T'$ and $ax \in F$, and if x is not a matched leaf, then $x \in X$, and as T' is U -closed, $x \in X \cap T'$ and so T' owns a ticket. In addition, as we assume that $v \neq r$, there must be a link from T' to $T - T'$ covering the edge between v and its parent. This link cannot have an endnode in $U \cap T'$, as T' is U -closed. Thus the endnode of this link in T' contributes a ticket, unless this endnode is a matched leaf. In particular, T' has a ticket if T' is leaf-closed. Summarizing, $|U \cap T'| + 1$ links are needed to cover the unmatched leaves and the edge between v and its parent, and every such link contributes a ticket unless it is incident to a matched leaf. Now recall that there is exactly one link in F incident to every matched leaf, by the Matching Invariant 3.7 and Claim 2.2. Thus exactly $2 \cdot |M \cap T'|$ links in F are incident to the matched leaves. The statement follows. \square

Claim 4.4 *If $\text{credit}(T') < |B(T')| + 1$ then $C(T') = \emptyset$ (in particular, $r \notin T'$) and $|M \cap T'| = 1$.*

Proof: If $|C(T')| \geq 1$ or if $|M \cap T'| \geq 2$ then $\text{coupons}(T') - |B(T')| \geq 1$, by (7). Hence the case to consider is $C \cap T' = \emptyset$ and $M \cap T' = \emptyset$. Note that since T' has no matched pair, then T has no link between its leaves, as otherwise greedy contraction applies. In particular, T' has no stems. Thus $\text{tickets}(T') \geq |L(T')| + 1 \geq 2$, by Lemma 4.3, and the statement follows. \square

Claim 4.5 *If $\text{credit}(T') < |B(T')| + 1$ then T' has no stem, and thus also has no locking link.*

Proof: Suppose to the contrary that T' has a stem s with children a_1, b_1 . As no greedy contraction applies, one of a_1, b_1 is matched by M , say $b_1 b_2 \in M$. By Claim 4.4 $M \cap T' = \{b_1 b_2\}$. Consider a link from a_1 to a leaf b . We cannot have $b \notin T'$ since T' is $\{a_1\}$ -closed, and we cannot have $b \in T' - \{b_1, b_2\}$ since then a greedy contraction applies with a_1, b . Thus we must have $b = b_1$ or $b = b_2$; this implies that $b_1 b_1$ locks a_1 , contradicting that M has no locking links. \square

Claim 4.6 *If $\text{credit}(T') < |B(T')| + 1$ then T' has exactly 3 leaves.*

Proof: By Claims 4.4 and 4.5, T' has exactly one matched pair. If T' has no unmatched leaves, then the contraction of the matched pair creates a leaf, contradicting that M is proper. If T' has at least 4 leaves, then $|U \cap T'| \geq 4 - 2 = 2$, hence $\text{tickets}(T') \geq |U \cap T'| - 2 \cdot |M \cap T'| + 1 \geq 2 - 2 + 1 = 1$. In this case $\text{credit}(T') - |B(T')| \geq \frac{1}{2}(|M \cap T'| + \text{tickets}(T')) \geq 1$. \square

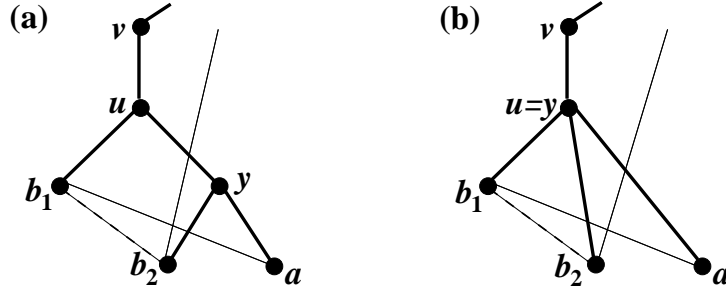


Figure 2: Deficient trees; tree edges are shown by bold lines, matching links by dashed lines. Edges vu, us in the figure can in fact be paths, possibly of the length zero; y is not a stem of the original tree T , and a may not be a leaf of T so its degree in F might be large, but b_1, b_2 are original leaves.

The following type of trees is the only type of semi-closed trees for which our “usual methods” above can not be used to prove that $credit(T') \geq |B(T')| + 1$.

Definition 4.3 (Deficient tree) *Suppose that T' has exactly 3 leaves and one matched pair, all its compound nodes are leaves, and that T' is not leaf closed. Let a denote the unmatched leaf of T' . Then T' is deficient (see Figure 2) if there is an ordering b_1, b_2 of its matched pair so that:*

- (i) $ab_1 \in E$ and the contraction of ab_1 does not create a leaf.
- (ii) There exists $b_2x \in E$ so that $x \in T - T'$ (namely, T' is not $\{b_2\}$ -closed).

If the ordering b_1, b_2 is not unique (this can happen if T' is as in Figure 2(b)), then we assume that if $up(b_1) = b_1u_1$ and $up(b_2) = b_2u_2$ then u_2 is an ancestor of u_1 .

Clearly, checking if T' is deficient or not can be done in polynomial time.

Lemma 4.7 *Suppose that T' has exactly 3 leaves and exactly one matched pair, all its compound nodes are leaves, and that T' is not leaf closed. If T' is not deficient, then T' has a ticket, and thus $credit(T') \geq |B(T')| + 1$.*

Proof: We prove that if T' has no ticket then T' is deficient. Let b_1, b_2 be the matched pair of T' . Let ya be the link in F covering a . If $y \notin \{b_1, b_2\}$ then $y \in T' \cap X$, and $y \in T'$ since T' is $\{a\}$ -closed. Also, y is not a compound node, by assumption. Hence in this case T' has a ticket. Thus assume $y \in \{b_1, b_2\}$. If the contraction of ay creates a leaf y , it can only happen in the case $y = b_2$ in Figure 2(a), and the link b_2a exists and $b_2a \in F$. The created leaf s must be covered in F by a link incident to a node in $p(ab_2)$. This node can not be an internal node of $p(ab_2)$ as then as it is not a compound node, it has a ticket, and it cannot be b_1 as $\deg_F(b_1) = 1$. Thus the link in F covering the creates leaf s must be incident to a , say az . As the tree is a -closed, z belongs to T' and z is not compound. We cannot have $z = b_2$, since b_2 has degree 1 in F . It can not be b_1 as b_1 needs to cover the root. Thus $z \in X$ and T' has a ticket.

Thus w.l.o.g. we assume that $ab_1 \in F$ and the contraction of ab_1 does not create a leaf. Consider any link $wx \in F$ covering v , where $w \in T'$ and $x \in T - T'$. If w is not a leaf of T' , then T' has a ticket at w . We have $w \neq b_1$ as $ab_1 \in F$. Also, $w \neq a$, as T' is $\{a\}$ -closed. It follows therefore that $w = b_2$, and the statement follows. \square

Summarizing, we obtain the following statement:

Corollary 4.8 *Let T' be a (not necessarily minimally) semi-closed tree. If $\text{credit}(T') < |B(T')| + 1$ then T' is deficient.*

4.3 Finding a good tree when all minimally semi-closed trees are deficient

Note that we can compute *all* semi-closed subtrees of T/I in polynomial time, as every semi-closed tree is a rooted subtree of T/I , and since we can check in polynomial time whether a given subtree is semi-closed. If T/I has a minimally semi-closed subtree T' that is not deficient, then T' and $B(T')$ satisfy the requirement of Lemma 3.8. Thus we may assume that all minimally semi-closed subtrees of T/I are deficient. In this case, we show that we can still find a semi-closed tree \tilde{T} and its cover \tilde{B} so that $\text{credit}(\tilde{T}) \geq |\tilde{B}| + 1$. But before that we need the following statement.

Claim 4.9 *Let T' be a deficient tree and let a, b_1, b_2 be as in Definition 4.3. If \tilde{T} is a rooted subtree of T/I containing b_2 that is $\{b_2\}$ -closed, then \tilde{T} properly contains T' and \tilde{T} is not deficient.*

Proof: As \tilde{T} is $\{b_2\}$ -closed, and there is a link $b_2x \in E$ with $x \in T - T'$, the root of \tilde{T} is a proper ancestor of the root of T' . It remains to show that if \tilde{T} is semi-closed then \tilde{T} is not deficient. Note that \tilde{T} is $\{a\}$ -closed, since T' is. Thus the only possibility for \tilde{T} to be deficient is if we are in the case of Figure 2(b), so that $b_2a \in E$ and \tilde{T} is not $\{b_1\}$ -closed. However, in Definition 4.3 we assume that b_2 has higher up-link than b_1 , which implies that \tilde{T} is $\{b_1\}$ -closed. This is a contradiction. \square

Definition 4.4 *The tree \tilde{T} and its cover \tilde{B} are defined as follows, see Figure 3. Let \tilde{M} be the matching obtained from M by replacing every link $b_1b_2 \in M$ that belongs to some minimally semi-closed deficient tree T' by the link b_1a , where a, b_1, b_2 are as in Definition 4.3; so b_1a exists and its contraction does not create a leaf, thus \tilde{M} is a proper matching. Then \tilde{T} is any minimally semi-closed tree w.r.t. the new proper matching \tilde{M} . Let $\tilde{B} = (\tilde{T} \cap \tilde{M}) \cup \text{up}(L(\tilde{T}))$ (note that \tilde{B} indeed covers \tilde{T} , by Lemma 4.2).*

Remark: Note that there might be links in $M \cap \tilde{M}$ (this is the set of links in M that do not belong to any deficient tree), see Figure 3(a). In fact, it might be that T/I has many leaves, but there is only one minimally semi-closed subtree T' , and this T' is deficient. An example is as follows. Take a path $r = v_1, \dots, v_{2k}$, and attach to every v_i a leaf u_i . Then attach to v_{2k} a tree T' as in Figure 2, so $v_{2k} = v$. The matching M is b_1b_2 plus the appropriate u_i, v_i edges. The tree T' is a unique minimally semi-closed subtree in this example, and T' is deficient.

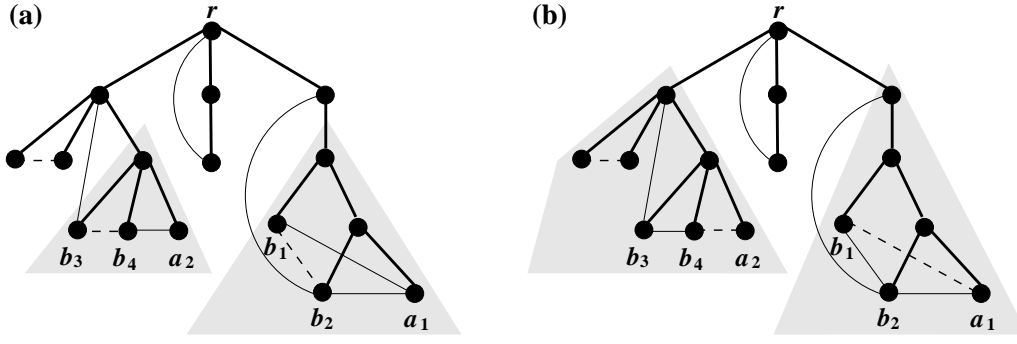


Figure 3: Illustration to Definition 4.4. Matching links are shown by dashed lines. (a) All minimally semi-closed trees are deficient. (b) The matching \tilde{M} and the minimally semi-closed trees w.r.t. \tilde{M} .

Recall that we use the notation $B(T') = (T' \cap M) \cup \text{up}(L(T'))$ for any semi-closed tree T' , and that by Lemma 4.2 $B(T')$ covers T' if T' is minimally semi-closed. Here and everywhere, unless stated otherwise, “semi-closed” means semi-closed w.r.t. M . We will prove that \tilde{T} is semi-closed, but this does not imply that $B(\tilde{T})$ covers \tilde{T} , since \tilde{T} is not *minimally* semi-closed. Instead, we use \tilde{B} to cover \tilde{T} , and note that $\tilde{B} \neq B(\tilde{T})$. We will prove that $|\tilde{B}| = |B(\tilde{T})|$. As \tilde{T} is semi-closed, then by Corollary 4.8 (which is valid for *any* semi-closed tree) $\text{credit}(\tilde{T}) \geq |B(\tilde{T})| + 1 = |\tilde{B}| + 1$, unless \tilde{T} is deficient. Consequently, the following statement finishes the proof of Lemma 3.8.

Lemma 4.10 *Suppose that all semi-closed subtrees of T/I are deficient, and let \tilde{T} and \tilde{B} be as Definition 4.4. Then \tilde{T} is semi-closed (w.r.t. M), \tilde{T} is not deficient, and $|\tilde{B}| = |B(\tilde{T})|$. Consequently, $\text{credit}(\tilde{T}) \geq |\tilde{B}| + 1$.*

Proof: We prove that if \tilde{T} intersects a deficient tree T' with leaves a, b_1, b_2 as in Definition 4.3, then \tilde{T} is $\{b_2\}$ -closed. As T', \tilde{T} are rooted trees, they share a leaf. If this leaf is b_2 , we are done. If this leaf is b_1 or a , then both $b_1, a \in \tilde{T}$, since \tilde{T} is semi-closed with respect to \tilde{M} . Thus the least common ancestor u of b_1, a is in \tilde{T} . This implies $b_2 \in \tilde{T}$, since b_2 is a descendant of u (see Figure 2).

Clearly, \tilde{T} intersects some deficient tree T' . Thus by Claim 4.9, \tilde{T} is not deficient. We show that \tilde{T} is semi-closed. Note that $b_1 b_2 \in M \setminus \tilde{M}$ if, and only if, b_1, b_2 belong to some deficient tree T' ; thus either $b_1, b_2 \in \tilde{T}$ (if T' is a subtree of \tilde{T}), or none of b_1, b_2 in \tilde{T} (otherwise). The same holds for $b_1 b_2 \in M \cap \tilde{M}$, by the definition of \tilde{T} . It remains to prove that if $a \in \tilde{T}$ is a leaf unmatched by M then \tilde{T} is $\{a\}$ -closed. If a is unmatched by \tilde{M} , this follows from the definition of \tilde{T} . Otherwise (a is matched by \tilde{M}), a is the unmatched leaf of a deficient T' . Then T' is $\{a\}$ -closed, T' is a subtree of \tilde{T} , and thus \tilde{T} is also $\{a\}$ -closed.

To see that $|\tilde{B}| = |B(\tilde{T})|$, note that \tilde{B} is obtained from $B(\tilde{T})$ by replacing every $b_1 b_2 \in M \setminus \tilde{M}$ that belongs to a deficient tree T' contained in \tilde{T} by the link $b_1 a$ (T', a, b_1, b_2 as in Definition 4.3). Hence \tilde{B} and $B(\tilde{T})$ coincide on links not belonging to deficient trees, while for links belonging to deficient trees there is a bijective correspondence between links in \tilde{B} and $B(\tilde{T})$.

The last statement follows from Corollary 4.8 and Claim 4.9. Indeed, as $|\tilde{B}| = |B(\tilde{T})|$ and \tilde{T} is semi-closed, we can have $\text{credit}(\tilde{T}) < |\tilde{B}| + 1$ only if \tilde{T} is deficient, by Corollary 4.8. But we proved that \tilde{T} is not deficient, thus $\text{credit}(\tilde{T}) \geq |\tilde{B}| + 1$, as claimed. \square

Acknowledgment: We thank two anonymous referees for many useful comments, and Jon Feldman for some useful discussions. We thank Samir Khuller for simplifying the proof of Lemma 3.3.

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] K. Eswaran and R. Tarjan. Augmentation problems. *SIAM J. Computing*, 5:653–665, 1976.
- [3] 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.
- [4] G. N. Frederickson and J. Jájá. Approximation algorithms for several graph augmentation problems. *SIAM J. Computing*, 10:270–283, 1981.
- [5] 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.
- [6] M. Goemans, A. Goldberg, S. Plotkin, E. T. D. Shmoys, and D. Williamson. Improved approximation algorithms for network design problems. In *SODA*, pages 223–232, 1994.
- [7] K. Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.
- [8] R. Jothi, B. Raghavachari, and S. Varadarajan. A $5/4$ -approximation algorithm for minimum 2-edge-connectivity. In *SODA*, pages 725–734, 2003.
- [9] 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.
- [10] S. Khuller and R. Thurimella. Approximation algorithms for graph augmentation. *J. of Algorithms*, 14:214–225, 1993.
- [11] G. Kortsarz and Z. Nutov. Approximating minimum cost connectivity problems (chapter 58). In *Handbook of Approximation Algorithms and Metaheuristics (Ed. T. F. Gonzales)*. Chapman & Hall/CRC, 2007.
- [12] H. Nagamochi. An approximation for finding a smallest 2-edge connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics*, 126:83–113, 2003.