A 4/3 Approximation for 2-Vertex-Connectivity

Miguel Bosch-Calvo ✉
IDSIA, USI-SUPSI, Lugano, Switzerland

Fabrizio Grandoni ✉
IDSIA, USI-SUPSI, Lugano, Switzerland

Afrouz Jabal Ameli ✉
TU Eindhoven, The Netherlands

Abstract

The 2-Vertex-Connected Spanning Subgraph problem (2VCSS) is among the most basic NP-hard (Survivable) Network Design problems: we are given an (unweighted) undirected graph G. Our goal is to find a subgraph S of G with the minimum number of edges which is 2-vertex-connected, namely S remains connected after the deletion of an arbitrary node. 2VCSS is well-studied in terms of approximation algorithms, and the current best (polynomial-time) approximation factor is 10/7 by Heeger and Vygen [SIDMA'17] (improving on earlier results by Khuller and Vishkin [STOC'92] and Garg, Vempala and Singla [SODA'93]).

Here we present an improved 4/3 approximation. Our main technical ingredient is an approximation preserving reduction to a conveniently structured subset of instances which are "almost" 3-vertex-connected. The latter reduction might be helpful in future work.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Algorithm, Network Design, Vertex-Connectivity, Approximation

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.29

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: https://arxiv.org/abs/2305.02240 [3]

Funding The first 2 authors are partially supported by the SNF Grant 200021_200731 / 1.

1 Introduction

Real-world networks are prone to failures. For this reason it is important to design them so that they are still able to support a given traffic despite a few (typically temporary) failures of nodes or edges. The basic goal of survivable network design is to construct cheap networks which are resilient to such failures.

Most natural survivable network design problems are NP-hard, and a lot of work was dedicated to the design of approximation algorithms for them. One of the most basic survivable network design problems is the 2-Vertex-Connected Spanning Subgraph problem (2VCSS). Recall that an (undirected) graph $G = (V, E)$ is k-vertex-connected (kVC) if, after removing any subset W of at most $k - 1$ nodes (with all the edges incident to them), the residual graph $G[V \setminus W]$ is connected. In particular, in a 2VC graph G we can remove any single node while maintaining the connectivity of the remaining nodes (intuitively, we can tolerate a single node failure). In 2VCSS we are given a 2VC (unweighted) undirected graph $G = (V, E)$, and our goal is to compute a minimum cardinality subset of edges $S \subseteq E$ such that the (spanning) subgraph (V, S) is 2VC.

2VCSS is NP-hard: indeed an n-node graph G admits a Hamiltonian cycle iff it contains a 2VC spanning subgraph with n edges. Czumaj and Lingas [13] proved that the problem is APX-hard, hence most likely it does not admit a PTAS. A 2-approximation for 2VCSS can be obtained in different ways. For example one can compute an (open) ear decomposition of

50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).
Editors: Kousha Etessami, Uriel Feige, and Gabriele Puppis; Article No. 29; pp. 29:1–29:13
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the input graph and remove the trivial ears (containing a single edge). The resulting graph is 2VC and contains at most $2(n-1)$ edges (while the optimum solution must contain at least n edges). The first non-trivial 5/3 approximation was obtained by Khuller and Vishkin [28]. This was improved to 3/2 by Garg, Vempala and Singla [20] (see also an alternative 3/2 approximation by Cheriyan and Thurimella [11]). Finally Heeger and Vygen [24] presented the current-best 10/7 approximation[1]. Our main result is as follows (please see Section 2 for an overview of our approach):

▶ **Theorem 1.** *There is a polynomial-time $\frac{4}{3}$-approximation algorithm for 2VCSS.*

1.1 Related Work

An undirected graph G is k-edge-connected (kEC) if it remains connected after removing up to $k-1$ edges. The 2-Edge-Connected Spanning Subgraph problem (2ECSS) is the natural edge-connectivity variant of 2VCSS, where the goal is to compute a 2EC spanning subgraph with the minimum number of edges. Like 2VCSS, 2ECSS does not admit a PTAS unless $P = NP$ [13]. It is not hard to compute a 2 approximation for 2ECSS. For example it is sufficient to compute a DFS tree and augment it greedily. Khuller and Vishkin [27] found the first non-trivial 3/2-approximation algorithm. Cheriyan, Sebö and Szigeti [10] improved the approximation factor to 17/12. This was further improved to 4/3 in two independent and drastically different works by Hunkenschröder, Vempala and Vetta [25] and Sebö and Vygen [34]. The current best and very recent $\frac{118}{89} + \varepsilon < 1.326$ approximation is due to Garg, Grandoni and Jabal Ameli [19]. Our work exploits several ideas from the latter paper. The k-Edge Connected Spanning Subgraph problem (kECSS) is the natural generalization of 2ECSS to any connectivity $k \geq 2$ (see, e.g., [11, 17]).

A major open problem in the area is to find a better than 2 approximation for the weighted version of 2ECSS. This is known for the special case with 0-1 edge weights, a.k.a. the Forest Augmentation problem, by the recent work by Grandoni, Jabal-Ameli and Traub [21] (see also [2, 7, 6] for the related Matching Augmentation problem).

A problem related to kECSS is the k-Connectivity Augmentation problem (kCAP): given a k-edge-connected undirected graph G and a collection of extra edges L (*links*), find a minimum cardinality subset of links L' whose addition to G makes it $(k+1)$-edge-connected. It is known [14] that kCAP can be reduced to the case $k = 1$, a.k.a. the Tree Augmentation problem (TAP), for odd k and to the case $k = 2$, a.k.a. the Cactus Augmentation problem (CacAP), for even k. Several approximation algorithms better than 2 are known for TAP [1, 8, 9, 15, 16, 22, 29, 30, 31], culminating with the current best 1.393 approximation by Cecchetto, Traub and Zenklusen [5]. Till recently no better than 2 approximation was known for CacAP (excluding the special case where the cactus is a single cycle [18]): the first such algorithm was described by Byrka, Grandoni and Jabal Ameli [4], and later improved to 1.393 in [5]. In a recent breakthrough by Traub and Zenklusen, a better than 2 (namely 1.694) aproximation for the weighted version of TAP was achieved [35] (later improved to $1.5 + \varepsilon$ in [36]). Partial results in this direction where achieved earlier in [1, 12, 16, 22, 32].

[1] Before [24] a few other papers claimed even better approximation ratios [23, 26], however they have been shown to be buggy or incomplete, see the discussion in [24].

1.2 Preliminaries

We use standard graph notation. For a graph $G = (V, E)$, we let $V(G) = V$ and $E(G) = E$ denote its nodes and edges, resp. For $W \subseteq V$ and $F \subseteq E$, we use the shortcuts $G \setminus F := (V, E \setminus F)$ and $G \setminus W := G[V \setminus W]$. For a subgraph G', a node v and an edge e, we also use the shortcuts $v \in G'$ and $e \in G'$ meaning $v \in V(G')$ and $e \in E(G')$, resp. Throughout this paper we sometimes use interchangeably a subset of edges F and the corresponding subgraph (W, F), $W = \{v \in V : v \in f \in F\}$. The meaning will be clear from the context. For example, we might say that $F \subseteq E$ is 2VC or that F contains a connected component. In particular, we might say that $S \subseteq E$ is a 2VC spanning subgraph. Also, given two subgraphs G_1 and G_2, by $G' = G_1 \cup G_2$ we mean that G' is the subgraph induced by $E(G_1) \cup E(G_2)$. We sometimes represent paths and cycles as sequence of nodes. A k-vertex-cut of G is a subset W of k nodes such that $G[V \setminus W]$ has at least 2 connected components. A node defining a 1-vertex-cut is a cut vertex.

By $\mathsf{OPT}(G) \subseteq E(G)$ we denote an optimum solution to a 2VCSS instance G, and let $\mathsf{opt}(G) := |\mathsf{OPT}(G)|$ be its size. All the algorithms described in this paper are deterministic.

The proofs that are omitted here due to space constraints will appear in the journal version of the paper (see also [3]).

2 Overview of Our Approach

In this section we sketch the proof of our 4/3-approximation (Theorem 1). The details and proofs which are omitted here will be given in the following technical sections.

Our result relies on 3 main ingredients. The first one is an approximation-preserving (up to a small additive term) reduction of 2VCSS to instances of the same problem on properly *structured* graphs, which are "almost" 3VC in a sense described later (see Section 2.1).

At this point we compute a minimum-size 2-edge-cover H similarly to prior work: this provides a lower bound on the size of the optimal solution. For technical reasons, we transform H into a *canonical* form, without increasing its size (see Section 2.2).

The final step is to convert H into a feasible solution S. Starting from $S = H$, this is done by iteratively adding edges to and removing edges from S in a careful manner. In order to take the size of S under control, we assign 1/3 credits to each edge of the initial S, and use these credits to pay for any increase in the number of edges of S (see Section 2.3). We next describe the above ingredients in more detail.

2.1 A Reduction to Structured Graphs

Our first step is an approximation-preserving (up to a small additive factor) reduction of 2VCSS to instances of the same problem on properly *structured* graphs. This is similar in spirit to an analogous reduction for 2ECSS in [19]. In particular we exploit the notion of irrelevant edges and isolating cuts defined in that paper. We believe that our reduction might be helpful also in future work.

In more detail, we can get rid of the following *irrelevant* edges.

▶ **Lemma 2** (irrelevant edge). *Given a 2VC graph G, let $e = uv \in E(G)$ be such that $\{u, v\}$ is a 2-vertex-cut (we call e* irrelevant*). Then every optimal 2VCSS solution for G does not contain e.*

Proof. We will need the following observation:

▶ **Fact 3.** *Suppose that a minimal solution S to 2VCSS on a graph G contains a cycle C. Then S does not contain any chord f of C. Indeed, otherwise consider any open ear decomposition*[2] *of S which uses C as a first ear. Then f would be a trivial ear (consisting of a single edge) of the decomposition, and thus $S \setminus \{f\}$ would also be 2VC, contradicting the minimality of S.*

Let $H \subseteq E$ be any optimal (hence minimal) solution to 2VCSS on G. Assume by contradiction that H contains an irrelevant edge $e = uv$. Removing u and v splits H into different connected components $C_1, \ldots, C_k$, with $k \geq 2$. Each one of those components has edges $u_i u$, $v_i v$ in H, where $u_i, v_i \in C_i$ for $i \in \{1, \ldots, k\}$, otherwise H would contain a cut vertex. Let P_1 be a path from u_1 to v_1 in C_1, and P_2 be a path from v_2 to u_2 in C_2. Then e is a chord of the cycle $P_1 \cup P_2 \cup \{uu_1, v_1v, vv_2, u_2u\}$, contradicting the minimality of H by Fact 3. ◀

We can enforce (see later) that our graph G is "almost" 3VC, in the sense that the only 2-vertex-cuts of G are a very specific type of *isolating* cuts defined as follows.

▶ **Definition 4** (isolating cut)**.** *Given a 2-vertex-cut $\{u, v\}$ of a graph G, we say that this cut is* isolating *if $G \setminus \{u, v\}$ has exactly two connected components, one of which consisting of* 1 *node. Otherwise the cut is* non-isolating.

Assuming that there are no non-isolating cuts, we can avoid the following local configuration: this will be helpful in the rest of our analysis.

▶ **Definition 5** (removable 5-cycle)**.** *We say that a 5-cycle C of a 2VC graph G is* removable *if it has at least two vertices of degree 2 in G.*

▶ **Lemma 6.** *Given a 2VC graph G without non-isolating cuts and with at least* 6 *nodes. Let C be a removable 5-cycle of G. Then in polynomial time one can find an edge e of C such that there exists an optimum solution to 2VCSS on G not containing e (we say that e is a* removable *edge).*

Proof. Assume $C = v_1v_2v_3v_4v_5$. If C has two vertices of degree 2 that are adjacent in C, namely v_1 and v_2, then $\{v_3, v_5\}$ is a non-isolating cut of G, a contradiction. Thus we can assume that C has exactly two non-adjacent vertices of degree 2, say v_1 and v_3 w.l.o.g.

We will show that the edge $e = v_4v_5$ is the desired removable edge. Let H be an optimal 2VCSS solution for G that uses the edge v_4v_5. Observe that in this case since v_1 and v_3 have degree 2, then H must contain all the edges of C.

To complete the argument we show that there exists an edge $f \in E(G) \setminus E(H)$, such that v_4v_5 is a chord of a cycle in $H' := H \cup \{f\}$: hence we can remove v_4v_5 from H' using Fact 3 to obtain an alternative optimum solution not containing v_4v_5.

Let $H'' = H \setminus \{v_4v_5\}$. There is no cycle C' in H'' that contains both v_4 and v_5, otherwise v_4v_5 is a chord of C' in H, contradicting the minimality of H by Fact 3. Therefore if we remove v_2 from H'', there must be no paths from v_4 to v_5. This means that there is a partition of $V(G) \setminus \{v_2\}$ into non-empty sets V_1 and V_2 such that, $\{v_3, v_4\} \in V_1$, $\{v_1, v_5\} \in V_2$ and there is no edge in H'' between V_1 and V_2. Since $|V(G)| \geq 6$, then we can assume w.l.o.g that $|V_1| \geq 3$.

[2] An ear-decomposition of an undirected graph G is a sequence of paths or cycles $P_0, \ldots, P_k$ (ears) spanning $E(G)$ such that P_0 is a cycle and P_i, $i \geq 1$, has its internal nodes disjoint from $V_{i-1} := V(P_0) \cup \ldots \cup V(P_{i-1})$ and its endpoints (or one node if P_i is a cycle) in V_{i-1}. We say that an ear-decomposition is open if P_i is a path, for $i \geq 1$. Every 2VC graph admits an open ear decomposition [33, Chapter 15].

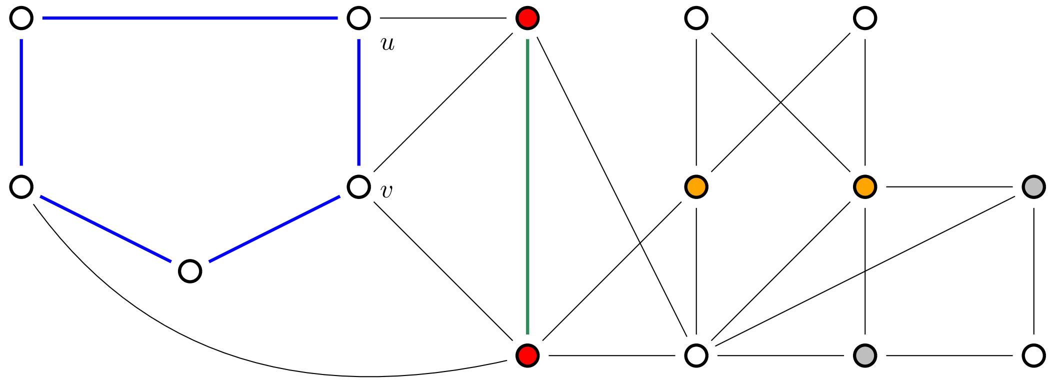

Figure 1 The cycle induced by the blue edges is a removable cycle, since it has two vertices of degree 2 in G. The edge uv is removable. The red and orange (resp. gray) pairs of vertices form a non-isolating (resp. isolating) cut. The green edge is irrelevant.

There must be an edge $f = u_1u_2 \in E(G)$ such that $u_1 \in V_1 \setminus \{v_3, v_4\}$ and $u_2 \in V_2$, otherwise $\{v_2, v_4\}$ is a non-isolating cut in G, a contradiction. Now we show that f is the desired edge. We claim that there exists a path P_1 in $H[V_1 \setminus \{v_3\}]$ between u_1 and v_4. Since H is 2VC, there exists a path P_1 between u_1 and v_4 not using v_2. Such path does not use v_3 either since this node is adjacent only to v_2 and v_4, and $u_1 \notin \{v_3, v_4\}$. If P_1 is not contained in $H[V_1]$, it would need to use at least two edges between V_1 and V_2 in H, however we argued before that H contains only one such edge, namely v_4v_5. Symmetrically, we claim that there exists a path P_2 in $H[V_2 \setminus \{v_1\}]$ between u_2 and v_5. Notice that $u_2 = v_5$ is possible, in which case the claim trivially holds. Hence next assume $u_2 \neq v_5$. Observe that $u_2 \neq v_1$ since u_2 is adjacent to $u_1 \notin \{v_2, v_5\}$. Thus, the claim about P_2 follows symmetrically to the case of P_1. Altogether, v_4v_5 is a chord of the cycle $P_1 \cup P_2 \cup \{f\} \cup C \setminus \{v_4v_5\}$ in $H' = H \cup \{f\}$, which implies the lemma. ◀

We are now ready to define a structured graph and to state our reduction to such graphs.

▶ **Definition 7** (structured graph). *A 2VC graph G is structured if it does not contain: (1) Irrelevant edges; (2) Non-isolating cuts; (3) Removable* 5*-cycles.*

▶ **Lemma 8.** *Given a constant $1 < \alpha \leq \frac{3}{2}$, if there exists a polynomial-time algorithm for 2VCSS on a structured graph G that returns a solution of cost at most* $\max\{\mathsf{opt}(G), \alpha \cdot \mathsf{opt}(G) - 2\}$*, then there exists a polynomial-time α-approximation algorithm for 2VCSS.*

We remark that any $\alpha - \varepsilon$ approximation of 2VCSS on structured graphs, for an arbitrarily small constant $\varepsilon > 0$, immediately implies an algorithm of the type needed in the claim of Lemma 8: indeed, instances with $\mathsf{opt}(G) \leq \max\{\frac{2}{\varepsilon}, \frac{2}{\alpha-1}\}$ can be solved exactly in constant time by brute force.

The algorithm at the heart of our reduction is algorithm RED given in Algorithm 1. Lines 1-2 solve by brute force instances with few nodes. Lines 3-4, 5-10, and 11-12 get rid recursively of irrelevant edges, non-isolating vertex cuts and removable 5-cycles, resp. When Line 13 is reached, the graph is structured and therefore we can apply a black-box algorithm ALG for structured instances of 2VCSS.

It is easy to see that the algorithm runs in polynomial time.

Algorithm 1 Reduction from arbitrary to structured instances of 2VCSS. Here G is 2VC and ALG is an algorithm for structured instances that returns a solution of cost at most $\max\{\mathsf{opt}(G), \alpha\cdot\mathsf{opt}(G)-2\}$ for some $1 < \alpha \le \frac{3}{2}$.

1: **if** $|V(G)| < \max\{6, \frac{2}{\alpha-1}\}$ **then**
2: Compute OPT(G) by brute force (in constant time) and **return** OPT(G)
3: **if** G contains an irrelevant edge **then**
4: **return** $\mathsf{RED}(G \setminus \{e\})$
5: **if** G contains a non-isolating vertex cut $\{u, v\}$ **then**
6: let $(V_1, V_2), 2 \le |V_1| \le |V_2|$, be a partition of $V(G) \setminus \{u, v\}$ such that there are no edges between V_1 and V_2 in $G \setminus \{u, v\}$
7: let G_1 be the graph resulting from G by contracting V_2 into one node v_2 and G_2 the graph resulting from G by contracting V_1 into one node v_1 (keeping one copy of parallel edges in both cases)
8: let $H_1 = \mathsf{RED}(G_1)$ and $H_2 = \mathsf{RED}(G_2)$
9: let E_1 (resp. E_2) be the two edges of H_1 (resp., H_2) with endpoints in v_2 (resp., v_1)
10: **return** $H := (H_1 \setminus E_1) \cup (H_2 \setminus E_2)$
11: **if** G contains a removable 5-cycle **then**
12: let e be the removable edge (found via Lemma 6) in that cycle and **return** $\mathsf{RED}(G\setminus\{e\})$
13: **return** $\mathsf{ALG}(G)$

▶ **Lemma 9.** $\mathsf{RED}(G)$ *runs in polynomial time in* $|V(G)|$ *if* ALG *does so.*

Proof. Let $n = |V(G)|$. First observe that each recursive call, excluding the corresponding subcalls, can be executed in polynomial time. In particular, we can find one irrelevant edge, if any, in polynomial time by enumerating all the possible 2-vertex-cuts. Furthermore, we can find some removable 5-cycle, if any, in polynomial time by enumerating all 5-cycles. Then, by Lemma 6, we can indentify a removable edge in such cycle. We also remark that in Lines 4 and 12 we remove one edge, and we never increase the number of edges. Hence the corresponding recursive calls increase the overall running time by a polynomial factor altogether.

It is then sufficient to bound the number $f(n)$ of recursive calls where we execute Lines 6-10 starting from a graph with n nodes. Consider one recursive call on a graph G with n nodes, where the corresponding graph G_1 has $5 \le k \le n/2 + 2$ nodes. Notice that G_2 has $n - k + 4$ nodes. Thus one has $f(n) \le \max_{5 \le k \le n/2+2}\{f(k) + f(n - k + 4)\}$, which implies that $f(n)$ is polynomially bounded. ◀

Let us next show that RED produces a feasible solution.

▶ **Lemma 10.** *Given a 2VC graph* G, $\mathsf{RED}(G)$ *returns a feasible 2VCSS solution for* G.

Proof. Let us prove the claim by induction on $(|V(G)|, |E(G)|)$ in lexicographic order. The base cases are given when $\mathsf{RED}(G)$ executes Lines 2 or 13: in these cases RED clearly returns a feasible solution. Consider an instance G where $\mathsf{RED}(G)$ does not execute those lines (in the root call), and assume the claim holds for any instance G' where $(|V(G')|, |E(G')|)$ is strictly smaller than $(|V(G)|, |E(G)|)$ in lexicographic order. By Lemma 2, when RED recurses at Line 4, the graph $G \setminus \{e\}$ is 2VC, hence the recursive call returns a 2VC spanning subgraph by inductive hypothesis. A similar argument holds when Line 12 is executed, this time exploiting Lemma 6.

It remains to consider the case when Lines 6-10 are executed. Notice that both G_1 and G_2 are 2VC. In this case we can assume by inductive hypothesis that both H_1 and H_2 are 2VC. Consider any $w_1 \in V_1$. Since H_1 is 2VC, H_1 contains 2 vertex disjoint paths from w_1 to v_2. Notice that both u and v must be the second last node in exactly one such path, hence in particular there exist two (internally) vertex-disjoint paths P_{w_1u} and P_{w_1v} in H over the nodes $V_1 \cup \{u, v\}$ from w_1 to u and v, resp. Symmetrically, for each $w_2 \in V_2$ there exist two vertex disjoint paths P_{w_2u} and P_{w_2v} in H over the nodes $V_2 \cup \{u, v\}$ from w_2 to u and v, resp.

For any $w_1 \in V_1$ and $w_2 \in V_2$, the w_1-w_2 paths $P_{w_1u} \cup P_{w_2u}$ and $P_{w_1v} \cup P_{w_2v}$ in H are vertex disjoint. Similarly, for any $w_1 \in V_1$ and $w_2 \in V_2$, $P_{w_1u} \cup P_{w_1v}$ and $P_{w_2u} \cup P_{w_2v}$ are vertex disjoint u-v paths in H. Given $w_1 \in V_1$ and $w_1' \in V_1 \cup \{u, v\}$, consider the two vertex disjoint paths in H_1 between them. If these paths do not contain v_2, then they also belong to H. Otherwise exactly one of those paths contains the subpath $P' = uv_2v$: by replacing P' with $P_{w_2u} \cup P_{w_2v}$ for an arbitrary $w_2 \in V_2$, one obtains two vertex disjoint w_1-w_1' paths in H. A symmetric argument holds for $w_2 \in V_2$ and $w_2' \in V_2 \cup \{u, v\}$.

Assume to get a contradiction that H has a cut vertex w. If $w \in \{u, v\}$, then w is also a cut vertex in either H_1 or H_2. Thus we can assume w.l.o.g. $w \in V_1$. Consider the components resulting of removing the vertex w from H. If one of this components does not contain u nor v then w is also a cut vertex in H_1. Thus removing w from H yields two connected components C_u, C_v, with $u \in C_u, v \in C_v$. But since $w \in V_1$, no edge from H_2 present in H is removed by deleting w. In particular, there is a path from u to v in H, contradicting the fact that w is a cut vertex. ◀

It remains to analyze the approximation factor of RED.

▶ **Lemma 11.** $|\mathsf{RED}(G)| \leq \begin{cases} \mathsf{opt}(G), & \text{if } |V(G)| < \max\{6, \frac{2}{\alpha-1}\}; \\ \alpha \cdot \mathsf{opt}(G) - 2, & \text{if } |V(G)| \geq \max\{6, \frac{2}{\alpha-1}\}. \end{cases}$

Proof. We prove the claim by induction on $(|V(G)|, |E(G)|)$ in lexicographic order. The base cases correspond to the execution of Lines 2 and 13. Here the claim trivially holds. The claim holds by inductive hypothesis and by Lemmas 2 and 6 when Lines 4 and 12, resp., are executed. Notice that the 6 that appears in the max in the claim of the lemma is meant to guarantee that the conditions of Lemma 6 are satisfied.

It remains to consider the case when Lines 6-10 are executed. Let OPT be a minimum 2VC spanning subgraph of G, and OPT_i be an optimal 2VCSS solution for G_i, $i \in \{1, 2\}$. We will later show

$$|\mathsf{OPT}| = |\mathsf{OPT}_1| + |\mathsf{OPT}_2| - 4. \tag{1}$$

Notice that since $|H_i \cap E_i| = 2$ for $i \in \{1, 2\}$ and $H_1 \setminus E_1$ and $H_2 \setminus E_2$ are edge-disjoint, we have $|H| = |H_1| + |H_2| - 4$.

Notice that, for $|V_i| \geq \frac{2}{\alpha-1}$, one has $|\mathsf{OPT}_i| \leq \alpha|\mathsf{OPT}_i| - 2$. We now distinguish a few cases.

If $|V_2| < \max\{6, \frac{2}{\alpha-1}\}$, then $|H| = |H_1| + |H_2| - 4 = |\mathsf{OPT}_1| + |\mathsf{OPT}_2| - 4 = |\mathsf{OPT}|$.

If $|V_1| \geq \max\{6, \frac{2}{\alpha-1}\}$, then $|H| = |H_1| + |H_2| - 4 \leq \alpha|\mathsf{OPT}_1| - 2 + \alpha|\mathsf{OPT}_2| - 2 - 4 \leq \alpha(|\mathrm{OPT}_1| + |\mathrm{OPT}_2|) - 8 \leq \alpha|\mathsf{OPT}| + 4\alpha - 8 \leq \alpha|\mathsf{OPT}| - 2$. The last inequality uses the assumption $\alpha \leq 3/2$.

Finally, if $|V_1| < \max\{6, \frac{2}{\alpha-1}\}$ and $|V_2| \geq \max\{6, \frac{2}{\alpha-1}\}$, we have $|H| = |H_1| + |H_2| - 4 \leq |\mathsf{OPT}_1| + \alpha|\mathsf{OPT}_2| - 2 - 4 = (1-\alpha)|\mathsf{OPT}_1| + \alpha(|\mathsf{OPT}_1| + |\mathsf{OPT}_2|) - 6 \leq (1-\alpha)|\mathsf{OPT}_1| + 4\alpha + \alpha|\mathsf{OPT}| - 6 \leq \alpha|\mathsf{OPT}| - 2$. The last inequality holds since $|\mathsf{OPT}_1| \geq |V(G_1)| \geq 5$ and $\alpha > 1$.

It remains to prove (1). Let E_1 be the two edges of G_1 with endpoints in v_2 and E_2 be the two edges of G_2 with endpoints in v_1. Observe that E_i coincides with the E_i defined in Line 9. By the same argument as in the proof of Lemma 10, one has that $(\mathsf{OPT}_1 \setminus E_1) \cup (\mathsf{OPT}_2 \setminus E_2)$ is a 2VC spanning subgraph of G. Notice that $\mathsf{OPT}_1 \setminus E_1$ and $\mathsf{OPT}_2 \setminus E_2$ are edge-disjoint and that $|E_i \cap \mathsf{OPT}_i| = |E_i| = 2$ for $i \in \{1, 2\}$. Using this two facts we get that $|\mathsf{OPT}| \leq |(\mathsf{OPT}_1 \setminus E_1) \cup (\mathsf{OPT}_2 \setminus E_2)| = |\mathsf{OPT}_1| + |\mathsf{OPT}_2| - 4$.

For the other direction, assume by contradiction that $|\mathsf{OPT}| < |\mathsf{OPT}_1| + |\mathsf{OPT}_2| - 4$. Notice that $E(G) = (E(G_1) \setminus E_1) \dot\cup (E(G_2) \setminus E_2)$ and thus $\mathsf{OPT} = ((E(G_1) \setminus E_1) \cap \mathsf{OPT}) \dot\cup ((E(G_2) \setminus E_2) \cap \mathsf{OPT})$. Thus we have that either $|(E(G_1) \setminus E_1) \cap \mathsf{OPT}| < |\mathsf{OPT}_1| - 2$ or $|(E(G_2) \setminus E_2) \cap \mathsf{OPT}| < |\mathsf{OPT}_2| - 2$. Assume w.l.o.g. that $|(E(G_1) \setminus E_1) \cap \mathsf{OPT}| < |\mathsf{OPT}_1| - 2$. Then $((E(G_1) \setminus E_1) \cap \mathsf{OPT}) \cup \{uv_2, vv_2\}$ is a 2VC spanning subgraph of G_1 of cardinality less than $|\mathsf{OPT}_1|$, a contradiction. (1) follows. ◀

2.2 A Canonical 2-Edge-Cover

It remains to give a good enough approximation algorithm for structured graphs. The first step in our algorithm (similarly to prior work on related problems [6, 19, 25]) is to compute (in polynomial time [33, Chapter 30]) a minimum-cardinality 2-edge-cover[3] H of G. It is worth to remark that $|H| \leq \mathsf{opt}(G)$: indeed the degree of each node in any 2VC spanning subgraph of G must be at least 2.

For technical reasons, we transform H, without increasing its size, into another 2-edge-cover which is *canonical* in the following sense. We need some notation first. If a connected component of H has at least 6 edges we call it a *large component*, and otherwise a *small component*. Let C be a large component of H. We call every maximal 2VC subgraph of C a *block*, and every edge of C such that its removal splits that component into two connected components a *bridge*. Notice that every edge of C is either a bridge or belongs to some block in that component. Also, every edge of C belongs to at most one block, thus there is a unique partition of the edges of C into blocks and bridges (but a node of C might belong to multiple blocks and to multiple bridges). Observe that C is 2VC iff it has exactly one block. If C is large but not 2VC we call it a *complex component*. If a block B of a complex component C contains only one cut vertex of C, we say that B is a *leaf-block* of C. Notice that since H is a 2-edge-cover, C must have at least 2 leaf blocks.

▶ **Definition 12** (Canonical 2-Edge-Cover). *A 2-edge-cover S of a graph G is* canonical *if: (1) Every small component of S is a cycle; (2) For any complex component C of S, each leaf-block B of C has at least 5 nodes.*

▶ **Lemma 13.** *Given a minimum 2-edge-cover H of a structured graph G, in polynomial time one can compute a* canonical *2-edge-cover S of G with $|S| = |H|$.*

Proof. We start with $S := H$. At each step if there are edges $e \in E(G) \setminus E(S)$ and $e' \in E(S)$, such that $S' := S \cup \{e\} \setminus \{e'\}$ is a 2-edge-cover that has fewer connected components than S or it has the same number of connected components as S but has fewer bridges and blocks in total than S, then we replace S by S'. This process clearly terminates within a polynomial number of steps, returning a 2-edge-cover S of the same size as the initial H (hence in particular S must be minimal).

[3] A 2-edge-cover H of a graph G is a subset of edges such that each node v of G has at least 2 edges of H incident to it.

Let us show that the final S satisfies the remaining properties. Assume by contradiction that S has a connected component C with at most 5 edges that is not a cycle. By a simple case analysis C must be a 4-cycle plus one chord f. However this contradicts the minimality of S by Fact 3.

Finally assume by contradiction that S has a complex component C, with a leaf-block B such that B has at most 4 nodes. By the minimality of S, B must be a 3-cycle or a 4-cycle. Let $B = v_1 \dots v_k$, $k \in \{3, 4\}$, and assume w.l.o.g. that v_1 is the only cut-vertex of C that belongs to B. In this case we show that there must exist an edge $e = uz \in E(G)$ such that $u \in \{v_2, v_k\}$ and $z \notin B$. If this is not true then for $k = 3$, v_1 is a cut-vertex in G, and for $k = 4$, $\{v_1, v_3\}$ form a non-isolating cut, leading to a contradiction in both cases. Consider $S' := S \cup \{e\} \setminus \{uv_1\}$. Note that S' is a 2-edge-cover of the same size as S. Since uv_1 belongs to a cycle of S, then the number of connected components in S' is not more than in S. If $z \notin C$ the number of connected components of S' is less than in S, which is a contradiction. Otherwise the number of connected components of S and S' is the same. Now in S' all the bridges and the blocks of S that shared an edge with any path from u to z in $S \setminus \{uv_1\}$ become part of the same block and all the other bridges and blocks remain the same. This is a contradiction as the total number of bridges and blocks of S' is less than in S. ◀

2.3 A Credit-Based Argument

Next assume that we are given a minimum-cardinality canonical 2-edge-cover H of a structured graph G. Observe that, for $|H| \leq 5$, H is necessarily a cycle of length $|H|$ by the definition of canonical 2-edge-cover and a simple case analysis. In particular H is already a feasible (and optimal) solution. Therefore we next assume $|H| \geq 6$. Starting from $S = H$, we will gradually add edges to (and sometimes remove edges from) S, until S becomes 2VC. In order to keep the size of S under control, we use a credit-based argument similarly to prior work [6, 19, 21]. At high level, the idea is to assign a certain number of credits $\mathsf{cr}(S)$ to S. Let us define the cost of S as $\mathsf{cost}(S) = |S| + \mathsf{cr}(S)$. We guarantee that for the initial value of S, namely $S = H$, $\mathsf{cost}(S) \leq \frac{4}{3}|H|$. Furthermore, during the process $\mathsf{cost}(S)$ does not increase.

During the process we maintain the invariant that S is canonical. Hence the following credit assignment scheme is valid for any intermediate S:

1. To every small component C of S we assign $\mathsf{cr}(C) = |E(C)|/3$ credits.
2. Each large component C receives $\mathsf{cr}(C) = 1$ credits.
3. Each block B receives $\mathsf{cr}(B) = 1$ credits.
4. Each bridge b receives $\mathsf{cr}(b) = 1/4$ credits.

We remark that each large connected component C of S which is 2VC, receives one credit in the role of a component, and one additional credit in the role of a block of that component. Let $\mathsf{cr}(S) \geq 0$ the total number of credits assigned to the subgraphs of S. It is not hard to show that the initial cost of S is small enough.

▶ **Lemma 14.** $\mathsf{cost}(H) \leq \frac{4}{3}|H|$.

Proof. Let us initially assign $\frac{1}{4}$ credits to the bridges of H and $\frac{1}{3}$ credits to the remaining edges. Hence we assign at most $\frac{|H|}{3}$ credits in total. We next redistribute these credits so as to satisfy the credit assignment scheme.

Each small component C retains the credits of its edges. If C is large and 2VC then it has exactly one block B. Since $|E(C)| \geq 6$, its edges have at least 2 credits, so we can assign 1 credit to C and 1 to B.

Now consider a complex component C of H. The bridges keep their own credits. Since H is a 2-edge-cover and C is complex, then C has at least 2 leaf-blocks B_1 and B_2. By the definition of canonical, B_1 and B_2 have at least 5 nodes (hence edges) each. Therefore together they have at least $\frac{10}{3} > 3$ credits, which is sufficient to assign one credit to C, B_1 and B_2. Any other block B of C (which has at least 3 edges) keeps the credits of its edges, hence at least 1 credit. Observe that $\mathsf{cost}(H) = |H| + \mathsf{cr}(H) \leq \frac{4}{3}|H|$ as desired. ◀

As mentioned before, starting from $S = H$, we will transform S without increasing its cost $\mathsf{cost}(S)$ until it becomes a single large component C that is 2VC (and thus it has exactly one block B) and therefore a 2VC spanning subgraph of G. Notice that at the end of the process $\mathsf{cr}(S) = \mathsf{cr}(C) + \mathsf{cr}(B) = 2$, hence $|S| = \mathsf{cost}(S) - 2 \leq \frac{4}{3}|H| - 2$. Combining this with the trivial case for $|H| \leq 5$, we obtain the following lemma.

▶ **Lemma 15.** *Given a canonical minimum 2-edge-cover H of a structured graph G, one can compute in polynomial time a 2VCSS solution S for G with $|S| \leq \max\{|H|, \frac{4}{3}|H| - 2\}$.*

Given the above results, it is easy to prove Theorem 1.

Proof of Theorem 1. By Lemma 8 it is sufficient to compute a solution of cost at most $\max\{\mathsf{opt}(G), \frac{4}{3} \cdot \mathsf{opt}(G) - 2\}$ on a structured graph G. We initially compute a canonical minimum 2-edge-cover H of G via Lemma 13. Then we apply Lemma 15 to obtain a 2VCSS solution S with $|S| \leq \max\{|H|, \frac{4}{3}|H| - 2\} \leq \max\{\mathsf{opt}(G), \frac{4}{3}\mathsf{opt}(G) - 2\}$. Clearly all steps can be performed in polynomial time. ◀

It remains to discuss the proof of Lemma 15 (assuming $|H| \geq 6$), which is the most technical part of our paper. The construction at the heart of the proof consists of a few stages. Recall that we start with a 2-edge-cover $S = H$, and then gradually transform S without increasing $\mathsf{cost}(S)$.

In the first stage of our construction we remove from S all the small components with the exception of the following type of 4-cycles that require a separate argument in the following.

▶ **Definition 16** (pendant 4-cycle)**.** *Let S be a 2-edge-cover of a graph G and C' be a large component of S. We say that a connected component C of S is a* pendant 4-cycle *(of C') if C is a 4-cycle and all the edges of G with exactly one endpoint in C have the other endpoint in C'.*

▶ **Lemma 17.** *Let G be a structured graph and H be a canonical minimum 2-edge cover of G, with $|H| \geq 6$. In polynomial time one can compute a canonical 2-edge-cover S of G such that the only small components of S are pendant 4-cycles and $\mathsf{cost}(S) \leq \mathsf{cost}(H)$.*

In the second stage of our construction we reduce to the case where S consists of large 2VC components only.

▶ **Lemma 18.** *Let G be a structured graph and S be a canonical 2-edge-cover of G such that the only small components of S are pendant 4-cycles. In polynomial time one can compute a canonical 2-edge-cover S' of G such that all the connected components of S' are 2VC and large, and $\mathsf{cost}(S') \leq \mathsf{cost}(S)$.*

At this point we can exploit the following definition and lemma from [19] to construct the desired 2VC spanning subgraph.

▶ **Definition 19** (Nice Cycle)**.** *Let $\Pi = (V_1, \ldots, V_k)$, $k \geq 2$, be a partition of the node-set of a graph G. A nice cycle N of G w.r.t. Π is a subset of edges with endpoints in distinct subsets of Π such that: (1) N induces one cycle of length at least 2 in the graph obtained from G by collapsing each V_i into a single node; (2) given the two edges of N incident to some V_i, these edges are incident to distinct nodes of V_i unless $|V_i| = 1$.*

▶ **Lemma 20** ([19]). *Let $\Pi = (V_1, \ldots, V_k)$, $k \geq 2$, be a partition of the node-set of a 2VC graph G. In polynomial time one can compute a nice cycle N of G w.r.t. Π.*

▶ **Lemma 21.** *Let G be a structured graph and S be a 2-edge-cover of G such that all the connected components of S are 2VC and large. In polynomial time one can compute a 2VCSS solution S' for G with $\mathsf{cost}(S') \leq \mathsf{cost}(S)$.*

Proof. Initially set $S' = S$. Consider the partition $\Pi = (V_1, \ldots, V_k)$ of $V(G)$ where V_i is the set of vertices of the 2VC component C_i of S'. If $k = 1$, S' already satisfies the claim. Otherwise, using Lemma 20 we can compute a nice cycle N of G w.r.t. Π. Let us replace S' with $S'' := S' \cup N$. W.l.o.g assume N is incident to $V_1, ..., V_r$ for some $2 \leq r \leq k$. Then in S'' the nodes $V_1 \cup \ldots \cup V_r$ belong to a unique (large) 2VC connected component C'. Furthermore $\mathsf{cost}(S') - \mathsf{cost}(S'') = \sum_{i=1}^{r}(\mathsf{cr}(C_i) + \mathsf{cr}(B_i)) - \mathsf{cr}(C') - \mathsf{cr}(B') - r = 2r - 2 - r \geq 0$, where B_i is the only block of the component C_i and B' the only block of C'. By iterating the process for a polynomial number of times one obtains a single 2VC component, hence the claim. ◀

The proof of Lemma 15 follows by chaining Lemmas 17, 18, and 21, and by the previous simple observations.

References

1 David Adjiashvili. Beating approximation factor two for weighted tree augmentation with bounded costs. In Philip N. Klein, editor, *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. SIAM, 2017. `doi:10.1137/1.9781611974782.157`.

2 Étienne Bamas, Marina Drygala, and Ola Svensson. A simple LP-based approximation algorithm for the matching augmentation problem. In Karen Aardal and Laura Sanità, editors, *Integer Programming and Combinatorial Optimization - 23rd International Conference, IPCO 2022, Eindhoven, The Netherlands, June 27-29, 2022, Proceedings*, volume 13265 of *Lecture Notes in Computer Science*, pages 57–69. Springer, 2022. `doi:10.1007/978-3-031-06901-7_5`.

3 Miguel Bosch-Calvo, Fabrizio Grandoni, and Afrouz Jabal Ameli. A 4/3 approximation for 2-vertex-connectivity, 2023. `arXiv:2305.02240`.

4 Jaroslaw Byrka, Fabrizio Grandoni, and Afrouz Jabal Ameli. Breaching the 2-approximation barrier for connectivity augmentation: a reduction to steiner tree. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 815–825. ACM, 2020. `doi:10.1145/3357713.3384301`.

5 Federica Cecchetto, Vera Traub, and Rico Zenklusen. Bridging the gap between tree and connectivity augmentation: unified and stronger approaches. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 370–383. ACM, 2021. `doi:10.1145/3406325.3451086`.

6 Joe Cheriyan, Jack Dippel, Fabrizio Grandoni, Arindam Khan, and Vishnu V. Narayan. The matching augmentation problem: a 7/4-approximation algorithm. *Math. Program.*, 182(1):315–354, 2020. `doi:10.1007/s10107-019-01394-z`.

7 Joseph Cheriyan, Robert Cummings, Jack Dippel, and J. Zhu. An improved approximation algorithm for the matching augmentation problem. *CoRR*, abs/2007.11559, 2020. `arXiv:2007.11559`.

8 Joseph Cheriyan and Zhihan Gao. Approximating (unweighted) tree augmentation via lift-and-project, part I: stemless TAP. *Algorithmica*, 80(2):530–559, 2018. `doi:10.1007/s00453-016-0270-4`.

9 Joseph Cheriyan and Zhihan Gao. Approximating (unweighted) tree augmentation via lift-and-project, part II. *Algorithmica*, 80(2):608–651, 2018. doi:10.1007/s00453-017-0275-7.

10 Joseph Cheriyan, András Sebö, and Zoltán Szigeti. Improving on the 1.5-approximation of a smallest 2-edge connected spanning subgraph. *SIAM J. Discret. Math.*, 14(2):170–180, 2001. doi:10.1137/S0895480199362071.

11 Joseph Cheriyan and Ramakrishna Thurimella. Approximating minimum-size k-connected spanning subgraphs via matching. *SIAM J. Comput.*, 30(2):528–560, 2000. doi:10.1137/S009753979833920X.

12 Nachshon Cohen and Zeev Nutov. A (1+ln2)-approximation algorithm for minimum-cost 2-edge-connectivity augmentation of trees with constant radius. *Theor. Comput. Sci.*, 489-490:67–74, 2013. doi:10.1016/j.tcs.2013.04.004.

13 Artur Czumaj and Andrzej Lingas. On approximability of the minimum-cost k-connected spanning subgraph problem. In Robert Endre Tarjan and Tandy J. Warnow, editors, *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 17-19 January 1999, Baltimore, Maryland, USA*, pages 281–290. ACM/SIAM, 1999. URL: http://dl.acm.org/citation.cfm?id=314500.314573.

14 E. A. Dinits, A. V. Karzanov, and M. V. Lomonosov. On the structure of a family of minimal weighted cuts in a graph. *Studies in Discrete Optimization*, pages 290–306, 1976.

15 Guy Even, Jon Feldman, Guy Kortsarz, and Zeev Nutov. A 1.8 approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Trans. Algorithms*, 5(2):21:1–21:17, 2009. doi:10.1145/1497290.1497297.

16 Samuel Fiorini, Martin Groß, Jochen Könemann, and Laura Sanità. Approximating weighted tree augmentation via chvátal-gomory cuts. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 817–831. SIAM, 2018. doi:10.1137/1.9781611975031.53.

17 Harold N. Gabow and Suzanne Gallagher. Iterated rounding algorithms for the smallest k-edge connected spanning subgraph. *SIAM J. Comput.*, 41(1):61–103, 2012. doi:10.1137/080732572.

18 Waldo Gálvez, Fabrizio Grandoni, Afrouz Jabal Ameli, and Krzysztof Sornat. On the cycle augmentation problem: Hardness and approximation algorithms. In Evripidis Bampis and Nicole Megow, editors, *Approximation and Online Algorithms - 17th International Workshop, WAOA 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers*, volume 11926 of *Lecture Notes in Computer Science*, pages 138–153. Springer, 2019. doi:10.1007/978-3-030-39479-0_10.

19 Mohit Garg, Fabrizio Grandoni, and Afrouz Jabal Ameli. Improved approximation for two-edge-connectivity. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2368–2410. SIAM, 2023. doi:10.1137/1.9781611977554.ch92.

20 Naveen Garg, Santosh S. Vempala, and Aman Singla. Improved approximation algorithms for biconnected subgraphs via better lower bounding techniques. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 103–111. ACM/SIAM, 1993. URL: http://dl.acm.org/citation.cfm?id=313559.313618.

21 Fabrizio Grandoni, Afrouz Jabal Ameli, and Vera Traub. Breaching the 2-approximation barrier for the forest augmentation problem. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1598–1611. ACM, 2022. doi:10.1145/3519935.3520035.

22 Fabrizio Grandoni, Christos Kalaitzis, and Rico Zenklusen. Improved approximation for tree augmentation: saving by rewiring. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 632–645, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188898.

23 Prabhakar Gubbala and Balaji Raghavachari. Approximation algorithms for the minimum cardinality two-connected spanning subgraph problem. In Michael Jünger and Volker Kaibel, editors, *Integer Programming and Combinatorial Optimization, 11th International IPCO Conference, Berlin, Germany, June 8-10, 2005, Proceedings*, volume 3509 of *Lecture Notes in Computer Science*, pages 422–436. Springer, 2005. doi:10.1007/11496915_31.

24 Klaus Heeger and Jens Vygen. Two-connected spanning subgraphs with at most 10/7 opt edges. *SIAM J. Discret. Math.*, 31(3):1820–1835, 2017. doi:10.1137/16M1091587.

25 Christoph Hunkenschröder, Santosh S. Vempala, and Adrian Vetta. A 4/3-approximation algorithm for the minimum 2-edge connected subgraph problem. *ACM Trans. Algorithms*, 15(4):55:1–55:28, 2019. doi:10.1145/3341599.

26 Raja Jothi, Balaji Raghavachari, and Subramanian Varadarajan. A 5/4-approximation algorithm for minimum 2-edge-connectivity. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 725–734. ACM/SIAM, 2003. URL: http://dl.acm.org/citation.cfm?id=644108.644227.

27 Samir Khuller and Uzi Vishkin. Biconnectivity approximations and graph carvings. In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, editors, *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 759–770. ACM, 1992. doi:10.1145/129712.129786.

28 Samir Khuller and Uzi Vishkin. Biconnectivity approximations and graph carvings. *J. ACM*, 41(2):214–235, 1994. doi:10.1145/174652.174654.

29 Guy Kortsarz and Zeev Nutov. Lp-relaxations for tree augmentation. In Klaus Jansen, Claire Mathieu, José D. P. Rolim, and Chris Umans, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016, September 7-9, 2016, Paris, France*, volume 60 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.APPROX-RANDOM.2016.13.

30 Guy Kortsarz and Zeev Nutov. A simplified 1.5-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Trans. Algorithms*, 12(2):23:1–23:20, 2016. doi:10.1145/2786981.

31 Hiroshi Nagamochi. An approximation for finding a smallest 2-edge-connected subgraph containing a specified spanning tree. *Discret. Appl. Math.*, 126(1):83–113, 2003. doi:10.1016/S0166-218X(02)00218-4.

32 Zeev Nutov. On the tree augmentation problem. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, pages 61:1–61:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ESA.2017.61.

33 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2003.

34 András Sebö and Jens 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. *Comb.*, 34(5):597–629, 2014. doi:10.1007/s00493-014-2960-3.

35 Vera Traub and Rico Zenklusen. A better-than-2 approximation for weighted tree augmentation. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1–12. IEEE, 2021. doi:10.1109/FOCS52979.2021.00010.

36 Vera Traub and Rico Zenklusen. A $(1.5+\epsilon)$-approximation algorithm for weighted connectivity augmentation. *CoRR*, abs/2209.07860, 2022. doi:10.48550/arXiv.2209.07860.