Algorithms for 2-Connected Network Design and Flexible Steiner Trees with a Constant Number of Terminals

Ishan Bansal
Operations Research and Information Engineering, Cornell University, Ithaca, NY, USA

Joe Cheriyan
Department of Combinatorics and Optimization, University of Waterloo, Canada

Logan Grout
Operations Research and Information Engineering, Cornell University, Ithaca, NY, USA

Sharat Ibrahimpur
Department of Mathematics, London School of Economics and Political Science, UK

Abstract

The k-Steiner-2NCS problem is as follows: Given a constant (positive integer) k, and an undirected connected graph $G = (V, E)$, non-negative costs c on the edges, and a partition $(T, V \setminus T)$ of V into a set of terminals, T, and a set of non-terminals (or, Steiner nodes), where $|T| = k$, find a min-cost two-node connected subgraph that contains the terminals. The k-Steiner-2ECS problem has the same inputs; the algorithmic goal is to find a min-cost two-edge connected subgraph that contains the terminals.

We present a randomized polynomial-time algorithm for the unweighted k-Steiner-2NCS problem, and a randomized FPTAS for the weighted k-Steiner-2NCS problem. We obtain similar results for a capacitated generalization of the k-Steiner-2ECS problem.

Our methods build on results by Björklund, Husfeldt, and Taslaman (SODA 2012) that give a randomized polynomial-time algorithm for the unweighted k-Steiner-cycle problem; this problem has the same inputs as the unweighted k-Steiner-2NCS problem, and the algorithmic goal is to find a min-cost simple cycle C that contains the terminals (C may contain any number of Steiner nodes).

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation algorithms, Capacitated network design, Network design, Parametrized algorithms, Steiner cycle problem, Steiner 2-edge connected subgraphs, Steiner 2-node connected subgraphs

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2023.14

Category APPROX

Related Version *Previous Version*: `https://doi.org/10.48550/arXiv.2206.11807`

Funding *Joe Cheriyan*: Supported in part by NSERC, RGPIN-2019-04197.
Sharat Ibrahimpur: Received funding from the following sources: NSERC grant 327620-09 and an NSERC DAS Award, European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. ScaleOpt–757481), and Dutch Research Council NWO Vidi Grant 016.Vidi.189.087.

1 Introduction

The k-Steiner-cycle problem is as follows: Given a constant k, and an undirected connected graph $G = (V, E)$, non-negative costs c on the edges, and a partition $(T, V \setminus T)$ of V into a set of terminals, T, and a set of non-terminals (or, Steiner nodes), where $|T| = k$, find a minimum-cost simple cycle C that contains all the terminals (and any subset of Steiner nodes). Note that this is an optimization problem and not a search problem. To the best of our knowledge, no polynomial-time (deterministic or randomized) algorithm is known for finding an optimal solution of the (weighted) k-Steiner-cycle problem, even for $k = 3$; this problem has been open for several decades, see [7]. Björklund, Husfeldt, and Taslaman (SODA 2012) [3] give a randomized polynomial-time algorithm for the unweighted k-Steiner-cycle problem; also, see Taslaman's thesis [26]. Further research on the same problem is presented by Wahlström [27], and by Fafianie and Kratsch [10]. The algorithm of [3] extends easily to a randomized FPTAS for the weighted k-Steiner-cycle problem, by using techniques from Ibarra & Kim [15] and Hochbaum & Shmoys [14], see Proposition 6. All the results in our paper are based on the key result of Björklund et al. [3]. Below, in the section on related work, we discuss several papers that pertain to the k-Steiner-cycle problem, but we stress that the methods and techniques of these papers have no direct implications for the k-Steiner-cycle problem.

Network design encompasses a wide class of problems that find applications in sectors like transportation, facility location, information security, resource connectivity, etc. Due to its wide scope and usefulness, the area of network design has been studied for decades and it has spawned major algorithmic innovations. Most of the problems in network design are NP-Hard, and researchers in the area have focused on designing good approximation algorithms. The nodes of a network are designated as terminals (i.e., "essential" nodes) or non-terminals (i.e., Steiner nodes or "optional" nodes). A well-known goal of network design is to construct cheap networks that can survive the failure of one element (i.e., one edge or one node); "surviving" means that all the (remaining) terminals stay connected even after the deletion of one element, that is, there exists a path between every pair of (remaining) terminals after deleting one element.

In the *Steiner-2ECS* problem, the input is an undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$, and non-negative costs c on the edges. The goal is to find a minimum-cost 2-edge-connected subgraph containing all the terminals. The *Steiner-2NCS* problem is defined similarly, where the goal is to find a minimum-cost 2-node-connected subgraph containing all the terminals. Both these problems are NP-Hard. The best known approximation algorithms achieve an approximation ratio of two, see [29].

Another paradigm to address NP-Hard problems is to develop parameterized algorithms. In this setting, a parameter of the input is chosen (e.g., the number of terminals, or the size of an optimal solution) and the goal is to develop algorithms whose running time depends on the input size and the parameter.

Feldmann, Mukherjee and van Leeuwen [11] presented parametrized algorithms for the Steiner-2ECS and Steiner-2NCS problems (among others) where the parameter is the optimal solution size, which is denoted by ℓ. They showed that if ℓ is bounded by a constant, then these problems can be solved in polynomial time. In particular, they present a *fixed parameter tractable* (FPT) algorithm that runs in time $n^{O(1)} f(\ell)$ and computes an optimal solution, where $f(\cdot)$ denotes some computable function.

Feldmann et al. [11] recently high-lighted the following open question in the area of network design: Is there a polynomial-time algorithm for the Steiner-2NCS problem, where the number of terminals is a constant? We use the term k-Steiner-2NCS problem (respectively, k-Steiner-2ECS problem) to refer to the special case of the Steiner-2NCS problem (respectively, Steiner-2ECS problem) with k terminals. Usually, we assume that the number of terminals is a constant, i.e., $k = O(1)$. We present a randomized polynomial-time algorithm for the unweighted k-Steiner-2NCS problem. We also consider the weighted k-Steiner-2NCS problem, and we provide a randomized fully polynomial time approximation scheme (FPTAS) for this problem.

We obtain similar results for the following generalization of the k-Steiner-2ECS problem: In the k-Flexible Steiner Tree (k-FST) problem, the input consists of an undirected graph $G = (V, E)$, a partition of the edge-set E into a set $\mathcal{S}$ of *safe edges* and a set $\mathcal{U}$ of *unsafe edges*, a set of terminals $T \subseteq V$ with $|T| = k$, and non-negative costs c on the edges. The goal is to find a minimum-cost connected subgraph $H = (U, F)$ such that $T \subseteq U$ and for any unsafe edge $e \in F$, the graph $H - e$ is connected. This problem is equivalent to the capacitated k-Steiner-2ECS problem, where the input is the same as that of the k-FST problem, except that there is a a positive integral capacity u on the edges (instead of the partition $E = \mathcal{S} \bigcup \mathcal{U}$); the goal is to find a minimum-cost connected subgraph $H = (U, F)$ such that $T \subseteq U$ and for any unit-capacity edge $e \in F$, the graph $H - e$ is connected. (The two problems are equivalent because the unsafe edges of the k-FST problem correspond to the unit-capacity edges of the latter problem, and the safe edges of the k-FST problem correspond to edges of capacity at least two of the latter problem.) We present a randomized polynomial-time algorithm for the unweighted k-FST problem; this easily extends to a randomized FPTAS for the weighted k-FST problem.

1.1 Our Results and Techniques

We prove that the k-Steiner-2NCS problem can be solved in randomized slicewise polynomial time, and hence it is in the complexity class randomized **XP**.

▶ **Theorem 1.** *For any $\eta > 0$, $\epsilon > 0$ and constant k,*

(i) *there exists a randomized algorithm for the unweighted k-*Steiner-2NCS *problem that outputs an optimal solution with probability $1 - \eta$ in time $O(\binom{n}{2k} \cdot B(3k) \cdot \binom{3k}{2}^k \cdot 2^{3k} \cdot n^{O(1)} \cdot \log \frac{k}{\eta}) = O\left(n^{O(k)} \cdot \log \frac{1}{\eta}\right)$, where $B(i)$ denotes the ith ordered Bell number.*

(ii) *there exists a randomized algorithm for the weighted k-*Steiner-2NCS *problem that runs in time $O(n^{O(k)} \cdot (1/\epsilon)^{O(k)} \cdot \log(1/\eta))$ such that, with probability at least $1 - \eta$, the solution returned by the algorithm costs at most $(1 + \epsilon)$ times the cost of an optimal solution.*

Our methods build on results by Björklund, Husfeldt, and Taslaman (SODA 2012) [3] that give a randomized polynomial-time algorithm for the unweighted k-Steiner-cycle problem. Given an instance of the k-Steiner-2NCS problem, we guess an ear decomposition of an optimal solution by enumeration, and repeatedly use the algorithm in [3] to construct an optimal subgraph. It can be seen that naively attaching new ears does not lead to an optimal solution. Hence, we also keep track of the high degree nodes in the optimal subgraph. Subsequently, we obtain our randomized FPTAS for the weighted k-Steiner-2NCS problem, by using the scaling techniques from Ibarra & Kim [15] and Hochbaum & Shmoys [14]. We present similar results for the k-FST problem.

▶ **Theorem 2.** *For any $\eta > 0$, $\epsilon > 0$ and constant k,*

(i) *there exists a randomized algorithm for the unweighted k-*FST *problem that outputs an optimal solution with probability $1 - \eta$ in time $O\left(n^{O(k)} \cdot \log \frac{1}{\eta}\right)$.*

(ii) *there exists a randomized algorithm for the weighted k-*FST *problem that runs in time $O(n^{O(k)} \cdot (1/\epsilon)^{O(k)} \cdot \log(1/\eta))$ such that, with probability at least $1 - \eta$, the solution returned by the algorithm costs at most $(1 + \epsilon)$ times the cost of an optimal solution.*

Our methods here rely on the block decomposition of an optimal solution, in conjunction with our results on the k-Steiner-2NCS problem. We use Theorem 1 to find the individual blocks optimally, and then paste these blocks together using the results by Adjiashvili, Hommelsheim, Mühlenthaler, and Schaudt [1] who give a polynomial-time algorithm for finding an optimal solution to the 2-FST problem; see Proposition 1 and Theorem 5 of [1]. As a corollary, we obtain the same results for the capacitated k-Steiner-2ECS problem.

▶ **Corollary 3.**
(i) *The capacitated* k-Steiner-2ECS *problem can be solved in slicewise polynomial time with high probability, hence, it is in randomized* $\mathbf{XP}$.
(ii) *There exists a randomized FPTAS for the weighted capacitated* k-Steiner-2ECS *problem.*

1.2 Related Work

1.2.1 k-Steiner-cycle problem

One of the key applications of the Graph Minors theory of Roberston and Seymour [23] is a polynomial-time algorithm for the decision/search version of the k disjoints paths problem for constant k. In this problem, we are given an undirected graph $G = (V, E)$ and k source-sink pairs s_i, t_i, and the goal is to decide (or, find) if there exist k node-disjoint paths P_i where the end-nodes of P_i are s_i and t_i. The Graph Minors theory (as of now) cannot address the optimization version of this problem.

There are many papers on the k disjoint paths problem, and a few on problems related to the k-Steiner-cycle problem. Kawarabayashi [19] presented improved algorithms for the search version of the latter problem, by improving on the methods from Graph Minors theory. There are a few other relevant results from the last few decades; for example, Fleischner and Woeginger present results for the unweighted 3-Steiner-cycle problem, see [13].

Recently, Lochet [21] and Bentert et al. [2] presented interesting algorithms for the so-called k disjoint shortest paths problem, i.e., each of the paths P_i (of the disjoint paths problem) is required to be a *shortest* path between s_i and t_i; this problem is *not* directly related to the optimization problems of interest to us.

1.2.2 k-Steiner-2NCS problem and k-Steiner-2ECS problem

Network design problems involving finding a cheapest subgraph subject to connectivity requirements have been studied for decades. One of the simplest such problems is the *minimum spanning tree* problem which is known to have polynomial-time algorithms. However, increasing the connectivity requirements makes these problems intractable. The 2-edge-connected spanning subgraph (2-ECSS) problem is Max-SNP-Hard, see [6]. In the weighted setting, the best known approximation algorithm achieves an approximation ratio of two, see Khuller and Vishkin [20]. The best known approximation ratio for the minimum-cost 2-node-connected spanning subgraph (2-NCSS) problem is two, see the survey by Nutov [22].

The analogous problems with Steiner nodes are usually harder. For instance, the minimum-cost Steiner tree problem is already NP-Hard [18]. The best known approximation ratio for this problem is ≈ 1.39, due to Byrka et al. [5]. The best known approximation ratios for the (weighted) Steiner-2ECS/Steiner-2NCS problem is two, see [29, 16, 12].

In the context of parameterized algorithms, Dreyfus and Wagner [9] showed that the Steiner tree problem can be solved in FPT time where the parameter is the number of terminals. Feldmann et al. [11] showed that the Steiner-2ECS and Steiner-2NCS problems can be solved in FPT time where the parameter is the size of an optimal solution.

Sami [24], in his master's thesis, has some results related to our paper. He notes that there is a reduction from the k-Steiner-2ECS problem to the k-Steiner-2NCS problem. (We can "inflate" each node v of the graph G of the k-Steiner-2ECS problem to a complete graph, i.e., clique, C_v on $\deg_G(v)$ nodes with edges of cost zero, and replace the edges incident to v in G by edges incident to distinct nodes of C_v while preserving the edge costs.) Moreover, he shows that the FPT (in the solution size parameter ℓ) algorithm of [11] for the k-Steiner-2ECS problem can be combined with a result of Jordan [17] to give an FPT (in parameter $k = |T|$) algorithm for the k-Steiner-2ECM problem where the solution subgraph may pick multiple copies of any edge (and incurs cost c_e for each copy of e).

Borradaile and Klein [4] presented a PTAS for the planar case of the Steiner-2ECM problem (i.e., the multi-edge variant of the Steiner-2ECS problem).

2 Preliminaries

This section has definitions and preliminary results. Our notation and terms are consistent with [8], and readers are referred to that text for further information.

Let $G = (V, E)$ be a loopless multi-graph with non-negative costs $c \in \mathbb{R}^E_{\geq 0}$ on the edges. We take G to be the input graph, and we use n to denote $|V(G)|$. For a set of edges $F \subseteq E(G)$, $c(F) := \sum_{e \in F} c(e)$, and for a subgraph G' of G, $c(G') := \sum_{e \in E(G')} c(e)$.

For a positive integer k, we use $[k]$ to denote the set $\{1, \dots, k\}$.

For a graph H and a set of nodes $S \subseteq V(H)$, $\Gamma_H(S) := \{w \in V(H) \setminus S \,:\, v \in S, vw \in E(H)\}$, thus, $\Gamma_H(S)$ denotes the set of neighbours of S.

For a graph H and a set of nodes $S \subseteq V(H)$, $\delta_H(S)$ denotes the set of edges that have one end node in S and one end node in $V(H) \setminus S$; moreover, $H[S]$ denotes the subgraph of H induced by S, and $H - S$ denotes the subgraph of H induced by $V(H) \setminus S$. For a graph H and a set of edges $F \subseteq E(H)$, $H - F$ denotes the graph $(V(H),\ E(H) \setminus F)$. We may use relaxed notation for singleton sets, e.g., we may use $\delta_H(v)$ instead of $\delta_H(\{v\})$, and we may use $H - v$ instead of $H - \{v\}$, etc.

We may not distinguish between a subgraph and its node set; for example, given a graph H and a set S of its nodes, we use $E(S)$ to denote the edge set of the subgraph of H induced by S.

2.1 2EC, 2NC and related notions

A multi-graph H is called k-edge connected if $|V(H)| \geq 2$ and for every $F \subseteq E(H)$ of size $< k$, $H - F$ is connected. Thus, H is 2-edge connected if it has ≥ 2 nodes and the deletion of any one edge results in a connected graph. A multi-graph H is called k-node connected if $|V(H)| > k$ and for every $S \subseteq V(H)$ of size $< k$, $H - S$ is connected. We use the abbreviations *2EC* for "2-edge connected," and *2NC* for "2-node connected."

For any instance H, we use $\textsc{opt}(H)$ to denote the minimum cost of a feasible subgraph (i.e., a subgraph that satisfies the requirements of the problem). When there is no danger of ambiguity, we use $\textsc{opt}$ rather than $\textsc{opt}(H)$.

By a *bridge* we mean an edge of a connected (sub)graph whose removal results in two connected components, and by a *cut-node* we mean a node of a connected (sub)graph whose deletion results in two or more connected components. A maximal connected subgraph without a cut-node is called a *block*. Thus, every block of a given graph G is either a maximal 2NC subgraph, or a bridge (and its incident nodes), or an isolated node. For any node v of G, let $\Gamma_G^{blocks}(v)$ denote the set of 2NC blocks of G that contain v.

2.2 Ear decompositions

An *ear decomposition* of a graph is a partition of the edge set into paths or cycles, $P_0, P_1, \ldots, P_\ell$, such that P_0 is the trivial path with one node, and each P_i $(1 \le i \le \ell)$ is either (1) a path that has both end nodes in $V_{i-1} = V(P_0) \cup V(P_1) \cup \ldots \cup V(P_{i-1})$ but has no internal nodes in V_{i-1}, or (2) a cycle that has exactly one node in V_{i-1}. For an ear P_i, let $\text{int}(P_i)$ denote the set of nodes $V(P_i) \setminus V_{i-1}$. Each of $P_1, \ldots, P_\ell$ is called an *ear*; note that P_0 is not regarded as an ear. We call $P_i, i \in \{1, \ldots, \ell\}$, an *open ear* if it is a path, and we call it a *closed ear* if it is a cycle. An *open* ear decomposition $P_0, P_1, \ldots, P_\ell$ is one such that all the ears $P_2, \ldots, P_\ell$ are open. (The ear P_1 is always closed.)

▶ **Proposition 4** (Whitney [28]).
(i) *A graph is 2EC* $\iff$ *it has an ear decomposition.*
(ii) *A graph is 2NC* $\iff$ *it has an open ear decomposition.*

2.3 Algorithms for basic computations

There are well-known polynomial-time algorithms for implementing all of the basic computations in this paper, see [25]. We state this explicitly in all relevant results, but we do not elaborate on this elsewhere.

3 FPTAS for k-Steiner-cycle

Björklund, Husfeldt, and Taslaman [3] presented a randomized algorithm for finding a min-cost simple cycle that contains a given set of terminals T of an unweighted, undirected graph $G = (V, E)$ with a running time of $2^k n^{O(1)}$, where $k = |T|$ and $n = |V|$. In other words, they present a randomized **FPT**-algorithm for the unweighted k-Steiner-cycle problem.

▶ **Theorem 5.** *Consider a graph $G = (V, E)$ and a set of terminals $T \subseteq V$ of size k. Let $\eta > 0$ be a parameter. A minimum-size* k-Steiner-cycle *can be found, if one exists, by a randomized algorithm in time $2^k n^{O(1)} \log \frac{1}{\eta}$ with probability at least $1 - \eta$.*

We present a simple (randomized) FPTAS for the weighted k-Steiner-cycle problem, based on the algorithm of [3].

▶ **Proposition 6.** *Consider a graph $G = (V, E)$ with nonnegative costs $c \in \mathbb{R}^E_{\ge 0}$ on the edges, and a set of k terminals $T \subseteq V$. Let $\varepsilon, \eta > 0$ be some parameters. There is a randomized algorithm that finds a $(1 + \varepsilon)$-approximate* k-Steiner-cycle, *if one exists, with probability at least $1 - \eta$. The running time of the algorithm is $O\left(2^k \cdot n^{O(1)} \cdot \left(\frac{1}{\varepsilon}\right)^{O(1)} \cdot \log \frac{1}{\eta}\right)$.*

Proof. Let $E = \{e_1, e_2, \ldots, e_m\}$ where $c_{e_1} \le c_{e_2} \le \cdots \le c_{e_m}$. Let $\eta' := \eta/2$. Let $j \in [m]$ denote the smallest index such that the graph $(V, \{e_1, \ldots, e_j\})$ contains a k-Steiner-cycle. Note that if G does not have a k-Steiner-cycle, then the weighted-version of the problem is trivially infeasible. Using at most m applications of Theorem 5 with the η-parameter set to η', we can find the index j with probability at least $1 - \eta/2$. Suppose that we have the correct index j. Let $\beta := c(e_j)$. Let Q^* denote an optimal k-Steiner-cycle in G, and $\textsc{opt} := c(Q^*)$ denote the optimal cost. By the definition of j, $\beta \le \textsc{opt} \le n\beta$. In particular, every edge in Q^* has cost at most $n\beta$. We now describe our randomized algorithm for obtaining a k-Steiner-cycle with cost at most $(1 + \varepsilon)\textsc{opt}$. First, we discard all edges e of G with cost $c_e > n\beta$. Let $\mu := \varepsilon\beta/n$; this is our "scaling parameter". For each edge e, define $\tilde{c}_e := \mu \cdot \max(1, \lceil c_e/\mu \rceil)$. Note that $\tilde{c}_e = \mu$ if $c_e = 0$. (Observe that this rounding introduces errors, but the total error incurred on any cycle is $\leq n\,\mu \leq \epsilon\beta \leq \epsilon\text{OPT}$.) Consider the graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ obtained from G by replacing each edge e by a path of $\tilde{c}_e/\mu$ edges (of unit cost). Note that $|\widetilde{V}| \leq |V| + |E| \cdot (n\beta)/\mu = O(mn^2/\varepsilon)$. Using a single application of Theorem 5, we can obtain a minimum-size k-Steiner-cycle $\widetilde{Q} \subseteq \widetilde{E}$ with probability at least $1 - \eta/2$ in $O\Big(2^k \cdot (\frac{n^2 m}{\varepsilon})^{O(1)} \cdot \log \frac{1}{\eta}\Big)$ time. Let Q denote the k-Steiner-cycle in G corresponding to $\widetilde{Q}$. By our choice of $\tilde{c}$, we have $c(Q) \leq \tilde{c}(Q) \leq \mu \cdot |\widetilde{Q}|$. Since the optimal k-Steiner-cycle Q^* consists of at most n edges each with cost at most $n\beta$, the (unweighted) k-Steiner-cycle $\widetilde{Q}^*$ in $\widetilde{G}$ corresponding to Q^* satisfies $\mu|\widetilde{Q}^*| \leq \tilde{c}(Q^*) \leq c(Q^*) + n\mu \leq \text{OPT}(1+\varepsilon)$. By the above discussion, we can obtain a k-Steiner-cycle Q satisfying $c(Q) \leq \mu|\widetilde{Q}| \leq \mu|\widetilde{Q}^*| \leq (1+\varepsilon)\text{OPT}$ with probability at least $1-\eta$. Clearly, the overall running time is $O\Big(2^k \cdot n^{O(1)} \cdot (\frac{1}{\varepsilon})^{O(1)} \cdot \log \frac{1}{\eta}\Big)$. ◀

4 Algorithms for k-Steiner-2NCS

In this section, we consider the k-Steiner-2NCS problem. First, we present a randomized polynomial-time algorithm for finding an optimal subgraph for the special case of unweighted k-Steiner-2NCS; then, using the method from Section 3 we extend our algorithm to a (randomized) FPTAS for weighted k-Steiner-2NCS.

We denote an instance of the k-Steiner-2NCS problem by $(G = (V, E),\ c \in \mathbb{R}^E_{\geq 0},\ T \subseteq V)$; G is the input graph with non-negative edge costs c, and T is the set of terminals, $|T| \geq 3$ (we skip the easy case of $|T| = 2$). We assume (w.l.o.g.) that G is a feasible subgraph, that is, all terminals are contained in one block of G.

For any graph H, let $D_3(H)$ denote the set of nodes that have degree ≥ 3 in H.

▶ **Lemma 7.** *Let $H = (V', E')$ be an (edge) minimal 2NC subgraph that contains T. Then H has an open ear decomposition $P_0, P_1, \ldots, P_\ell$ such that*

(i) *each of the ears P_i ($i \in [\ell]$) contains a terminal as an internal node (i.e., $\text{int}(P_i) \cap T \neq \emptyset$), and P_0 contains a terminal,*

(ii) *$|D_3(H)| \leq 2(|T| - 2)$.*

Proof.

(i) Pick any terminal to be P_0. Suppose we have constructed open ears $P_1, \ldots, P_{i-1}$ and that each $\text{int}(P_j)(j \in [i-1])$ contains a terminal. Let $F = \cup_{j=1}^{i-1} E(P_j)$. Let t be a terminal in $T \setminus V(F)$ (we have the required ear decomposition, if $T \subseteq V(F)$). Suppose $i = 1$; then, G has two openly disjoint paths between t and P_0, and we take P_1 to be the edge-set of these two paths. Suppose $i \geq 2$; then, G has a two-fan P between t and $V(F)$ (i.e., P is the union of two paths between t and $V(F)$ that have only the node t in common); we take P_i to be P.

(ii) Clearly, $\ell \leq |T| - 1$ for the ear decomposition of part (i), and each of the ears $P_2, \ldots, P_\ell$ contributes at most 2 (new) nodes to $D_3(H)$. ◀

The next lemma states an extension of Proposition 4.

▶ **Lemma 8.** *Let $G = (V, E)$ be a graph, and let $H = (V', E')$ be a 2NC subgraph of G. Let P be a path of G that has both end nodes in V'. Then, $H \cup P = (V' \cup V(P), E' \cup E(P))$ is a 2NC subgraph of G.*

Each set $S \subseteq V$ of size $\leq 2|T| - 4$ is a candidate for $D_3(H)$ for a 2NC subgraph H that contains T, and we call $T \cup S$ the set of *marker* nodes.

Our algorithm has several nested loops. The outer-most loop picks a set $S \subseteq V$ of size $\leq 2|T| - 4$, and then applies the following main loop. Each iteration of the main loop attempts to construct a 2NC subgraph that contains the set of marker nodes $T \cup S$, by iterating over all ordered partitions $(\widetilde{T}_1, \widetilde{T}_2, \ldots, \widetilde{T}_r)$ of $T \cup S$ such that $|\widetilde{T}_1| \geq 2$ and the number of sets in the partition, r, is a positive integer, $r \leq k = |T|$.

Consider one of these ordered partitions $(\widetilde{T}_1, \widetilde{T}_2, \ldots, \widetilde{T}_r)$. We attempt to find a min-cost Steiner-cycle C_1 that contains $\widetilde{T}_1$ using the algorithm of [3]; if G has no Steiner-cycle that contains $\widetilde{T}_1$, then this iteration has failed, otherwise, we take C_1 to be the first (closed) ear of an open ear decomposition of our candidate 2NC subgraph that contains $T \cup S$. Then, for $i = 1, \ldots, r-1$, we pick a pair of nodes $s_i, t_i \in \widetilde{T}_1 \cup \cdots \cup \widetilde{T}_i$, and attempt to find a min-cost Steiner-path P_{i+1} between s_i and t_i that contains $\widetilde{T}_{i+1}$; if G has no such Steiner-path, then this iteration has failed, otherwise, we augment the current subgraph $H := C_1 \cup P_2 \cup \cdots \cup P_i$ by P_{i+1}.

The algorithm maintains an edge-set $\hat{F}$; initially, $\hat{F} = E$, and, at termination, $\hat{F}$ is the edge-set of a min-cost 2NC subgraph that contains T.

Pseudo-code for the algorithm is presented below.

We use $\mathcal{A}_{\text{BHT-cycle}}(G,\ \widetilde{T}_1,\ \eta)$ to denote a call to the Steiner-cycle algorithm of [3] where the inputs are the graph G, the terminal set $\widetilde{T}_1 \subseteq V(G)$, and the desired probability of failure η. With probability at least $1-\eta$, this call either returns the edge-set of a minimum-size cycle of G that contains all nodes of $\widetilde{T}_1$ or reports an error if G has no such cycle.

We use $\mathcal{A}_{\text{BHT-path}}(G,\ \widetilde{T}_1,\ s,\ t,\ \eta)$ to denote a call to the following subroutine that attempts to find an s,t-path of G that contains all nodes of $\widetilde{T}_1$. First, construct an auxiliary graph G' from G by adding a node u' and two edges $u's, u't$. Then call $\mathcal{A}_{\text{BHT-cycle}}(G',\ \widetilde{T}_1 \cup \{u', s, t\},\ \eta)$; report an error if the call returns an error, and, otherwise, return the path obtained by deleting the node u' (and its two incident edges) from the cycle returned by the call.

Algorithm 1 $\mathcal{A}_{2NC}(G, T, \eta)$ for the unweighted k-Steiner-2NCS problem.

```
η' ← η/k
F̂ ← E
for S ⊆ V such that |S| ≤ 2k do
    for r = 1, ..., k do
        for Ordered partitions (T̃_1, ..., T̃_r) of T ∪ S such that |T̃_1| ≥ 2 do
            for i = 1, ..., r − 1 and node pairs (s_i, t_i) ∈ ∪_{j=1}^{i} T̃_i, where s_i ≠ t_i do
                H ← A_BHT-cycle(G, T̃_1, η')  ⋃_{i=1}^{r−1} A_BHT-path(G, T̃_{i+1}, s_i, t_i, η')
                continue the loop if any call to any subroutine reports an error
                if |E(H)| < |F̂| then
                    F̂ ← E(H)
                end if
            end for
        end for
    end for
end for
return F̂
```

► **Lemma 9.** *Let $H^* = (V^*, E^*)$ be an optimal subgraph for k-*Steiner-2NCS*. Assume that each of the calls to the subroutines (namely, $\mathcal{A}_{BHT\text{-}cycle}, \mathcal{A}_{BHT\text{-}path}$) returns a valid subgraph whenever one exists. Let $H = (U, \hat{F})$ denote the output of the above algorithm. Then H is a 2NC subgraph, $U \supseteq T$, and $|\hat{F}| \leq |E^*| = \text{OPT}$.*

Proof. By Lemma 7, H^* has an open ear decomposition $P_1, P_2, \ldots, P_{r^*}$ such that each of the ears P_i contains at least one terminal as an internal node; hence, $r^* \leq k = |T|$. Let $S^* = D_3(H^*)$ be the set of nodes of degree ≥ 3 of H^*; clearly, $|S^*| \leq 2r^* \leq 2k$.

For $i = 1, \ldots, r^*$, let $T_i^* = P_i \cap (T \cup S^*)$. For $i = 1, \ldots, r^* - 1$, let (s_i^*, t_i^*) denote the end nodes of the ear P_{i+1}; clearly, $(s_i^*, t_i^*) \in \cup_{j=1}^{i}(T_i^*)$.

Now consider the loop in the algorithm where $S = S^*$, $r = r^*$, $\widetilde{T}_i = T_i^*$ for $i = 1, \ldots, r^*$ and $(s_i, t_i) = (s_i^*, t_i^*)$ for $i = 1, \ldots, r^* - 1$. Observe that the calls to the subroutines $\mathcal{A}_{\text{BHT-cycle}}$ and $\mathcal{A}_{\text{BHT-path}}$ return minimum-size subgraphs, hence, $|\mathcal{A}_{\text{BHT-cycle}}(G, \widetilde{T}_1)| \leq |P_1|$ and $|\mathcal{A}_{\text{BHT-path}}(G, \widetilde{T}_{i+1}, s_i, t_i)| \leq |P_{i+1}|$ for $i = 1, \ldots, r^* - 1$. Since $|E^*| = \sum_{i=1}^{r^*} |P_i|$, we conclude that the 2NC subgraph H found by this iteration satisfies $|E(H)| \leq |E^*|$. Thus, the algorithm outputs an optimal 2NC subgraph that contains T. ◀

4.1 Proof of Theorem 1

Proof. As seen in the proof of Lemma 9, if the subroutines $\mathcal{A}_{\text{BHT-cycle}}$ and $\mathcal{A}_{\text{BHT-path}}$ run correctly when $S = S^*$, $r = r^*$, $\widetilde{T}_i = T_i^*$ for $i = 1, \ldots, r^*$ and $(s_i, t_i) = (s_i^*, t_i^*)$ for $i = 1, \ldots, r^* - 1$ corresponding to an ear decomposition of an optimal solution H^*, then the above algorithm outputs an optimal solution. During this loop, there are at most $r^* \leq k = |T|$ calls to the subroutines $\mathcal{A}_{\text{BHT-cycle}}$ and $\mathcal{A}_{\text{BHT-path}}$. Hence, with probability at least $(1 - \eta')^k \geq 1 - \eta$, Algorithm $\mathcal{A}_{2NC}$ outputs an optimal solution.

The running time is analyzed as follows: the term $2k \cdot \binom{n}{2k} = O(\binom{n}{2k})$ comes from choosing $S \subseteq V$, $|S| \leq 2k$ (in the outer-most loop), the term $B(3k)$ comes from choosing ordered partitions of $S \cup T$, the term $\binom{3k}{2}^k$ comes from choosing the node pairs (s_i, t_i) for the $r - 1 (\leq k)$ calls to $\mathcal{A}_{\text{BHT-path}}$, and the term $2^{3k} n^{O(1)} \log \frac{k}{\eta}$ comes from the running time of the algorithm of [3] for the Steiner-cycle problem, with error probability $\frac{\eta}{k}$.

Having solved the unweighted k-Steiner-2NCS problem, we can directly use the methods from Section 3 to obtain an FPTAS for the weighted k-Steiner-2NCS problem. ◀

5 FPTAS for k-FST and k-Steiner-2ECS

In this section we present a randomized polynomial-time algorithm for finding an optimal subgraph for the special case of unweighted k-FST; then, using the method from Section 3, we extend our algorithm to a (randomized) FPTAS for weighted k-FST. We assume that $k = |T| \geq 3$ is a positive integer. Note that the capacitated k-Steiner-2ECS problem can be reduced to the k-FST problem by defining the set of safe edges to be the set of edges with capacity at least 2 and defining the set of unsafe edges to be the set of edges with capacity exactly 1. Hence the results in this section can also be applied to the capacitated k-Steiner-2ECS problem.

Adjiashvili, Hommelsheim, Mühlenthaler, and Schaudt [1] give a polynomial-time algorithm for finding an optimal solution to the 2-FST problem; see Proposition 1 and Theorem 5 of [1]. We refer to their 2-FST algorithm as $\mathcal{A}_{2\text{-FST}}$. We refer to (inclusion-wise) minimal feasible solutions to a 2-FST problem on G as *1-protected paths*.

Informally speaking, our randomized polynomial-time algorithm for k-FST represents minimal feasible solutions as 2NC blocks connected together using 1-protected paths. To simplify our presentation, we first modify the k-FST instance $G = (V, \mathcal{S} \sqcup \mathcal{U}, T)$ as follows. For each terminal $v \in T$, we create a new node v' and a new safe edge vv'. Let T' denote the set of these new nodes and let E' denote the set of the new safe edges. Consider the modified instance $G' = (V \cup T', (\mathcal{S} \cup E') \sqcup \mathcal{U}, T')$. Observe that (U, F) is a feasible solution to the original instance if and only if $(U \cup T', F \cup E')$ is a feasible solution to the modified instance.

▶ **Definition 10** (Block-Tree). *A block-tree of a graph G is a tree $B(G)$ with the following properties:*

1. *The nodes of $B(G)$ are in one-to-one correspondence with the 2NC blocks of G.*
2. *If two 2NC blocks are connected by a bridge in G, then the two corresponding nodes in $B(G)$ are adjacent.*
3. *For each cut-node v of G, the subgraph of $B(G)$ induced by $\Gamma_G^{blocks}(v)$ is connected ($\Gamma_G^{blocks}(v)$ is the set of 2NC blocks of G that contain v). In other words, the unique path of $B(G)$ between any two nodes of $\Gamma_G^{blocks}(v)$ has all its internal nodes in $\Gamma_G^{blocks}(v)$.*

Informally speaking, a block-tree of a graph G represents how the 2NC blocks of G are connected together. Each edge of the block-tree either represents a bridge of G or connects a pair of 2NC blocks of G that share a common cut-node. Let H be a minimal feasible k-FST solution. Due to the modification above, we may assume that every leaf of $B(H)$ corresponds to a block of H that contains exactly one terminal. Then any path in $B(H)$ corresponds to a 1-protected path of H that connects either (i) two cut-nodes, or (ii) a cut-node and a terminal, or (iii) two terminals.

For our algorithmic application, nodes of $B(H)$ of degree two are redundant, and this motivates the notion of a "non-redundant" block-tree.

▶ **Definition 11** (Condensed Block-Tree). *A condensed block-tree of a graph G is a tree $\hat{B}(G)$ obtained from a block-tree $B(G)$ with the following properties:*

1. *The nodes of $\hat{B}(G)$ are nodes b of $B(G)$ such that $\deg_{B(G)}(b) \neq 2$.*
2. *Two nodes b_1 and b_2 are adjacent in $\hat{B}(G)$ if and only if every internal node in the path connecting b_1 and b_2 in $B(G)$ has degree two.*

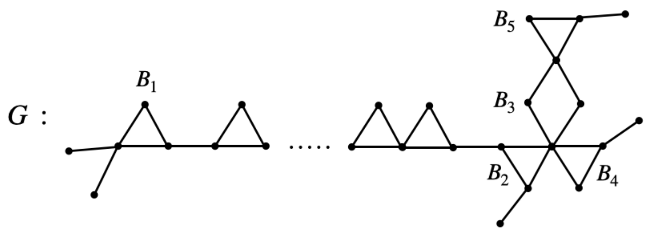

Figure 1 The original graph G.

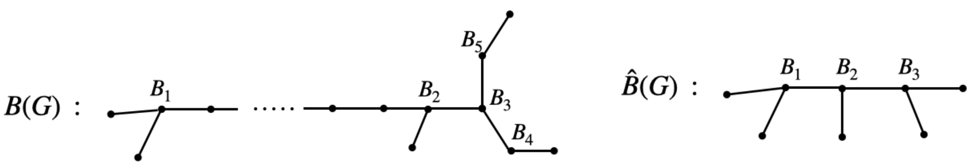

Figure 2 A block tree $B(G)$ and the corresponding condensed block tree $\hat{B}(G)$.

For any minimal feasible solution H of k-FST and a condensed block-tree $\hat{B}(H)$, we refer to the 2NC blocks of H that correspond to internal nodes of $\hat{B}(H)$ as *high-degree blocks*. The leaves of $\hat{B}(H)$ correspond to the terminals. Edges of $\hat{B}(H)$ correspond to 1-protected paths in H that connect either (i) two high-degree blocks, or (ii) a high-degree block and a terminal, or (iii) two terminals. The end-points of these 1-protected paths are either cut-nodes of H or terminals. Note that some of these 1-protected paths could be trivial paths corresponding to cut-nodes that are common to two high-degree blocks. We now state some useful lemmas that follow from the handshaking lemma applied to $\hat{B}(H)$.

▶ **Lemma 12.** *The number of internal nodes (i.e., non-leaf nodes) of $\hat{B}(H)$ is at most $k-2$ where k is the number of terminals.*

▶ **Lemma 13.** *The total number of cut-nodes (with repetitions) in high-degree blocks of H is at most $3k-6$ where k is the number of terminals.*

Now, we describe our algorithm for unweighted k-FST. We guess (via enumeration) the high-degree blocks of an optimal solution OPTSOLN corresponding to some condensed block-tree $\hat{B}(\text{OPTSOLN})$. The guess would include the number of high-degree blocks and the cut-nodes in each of these high-degree blocks. This is done by picking $3k-6$ nodes of V with replacement and then picking a partition $\widehat{\mathcal{P}}$ of these $3k-6$ nodes into at most $k-2$ sets. Let $r \le k-2$ be the number of sets in the partition $\widehat{\mathcal{P}}$. Thus, $\widehat{\mathcal{P}} = (X_1, X_2, \ldots, X_r)$. For each set X_i, we use algorithm $\mathcal{A}_{2NC}$ to find B_i, a minimum size 2NC subgraph of G containing the specified cut-nodes in X_i, possibly, with some additional Steiner nodes. Finally, we construct a tree that connects these 2NC subgraphs and terminals via 1-protected paths using the following subroutine.

First, for every pair of nodes $(u, v) \in V' \times V'$, we use algorithm $\mathcal{A}_{\text{2-FST}}$ to find $G^{\min}_{uv}$, the minimum size 1-protected path connecting u and v in G'. We then construct a complete graph $K(X_1, \ldots, X_r)$ with $r+k$ nodes that has one node for each set X_i and one node corresponding to each terminal $\{t\}$. The cost of an edge between two nodes of K corresponding to node sets V_1 and V_2 is given by $\min\{|E(G^{\min}_{uv})| : u \in V_1, v \in V_2\}$. Note that if there is no 1-protected path connecting a node in V_1 to a node in V_2, then we fix the cost of the edge to be infinity. Thus an edge $\bar{e}$ of K corresponds to a subgraph $G^{\min}_{\bar{e}}$ in G' which is the minimum size 1-protected path whose end points are in V_1 and V_2 respectively. We then find a minimum spanning tree $\bar{T}$ in K. Note that if $\bar{T}$ has infinite cost, then we output an error. Else, we output the subgraph of G' defined by $G^{\min}(X_1, \ldots, X_r) := \cup_{\bar{e} \in \bar{T}} G^{\min}_{\bar{e}}$.

Algorithm 2 $\mathcal{A}_{\text{k-FST}}(G', T', \eta)$ for the unweighted k-FST problem.

$\eta' \leftarrow \eta/k$
$H \leftarrow G'$
for $S = (v_1, \ldots, v_{3k-6}) \in V^{3k-6}$ **do**
  for $r = 1, \ldots, k-2$ **do**
    for partitions $(X_1, \ldots, X_r)$ of S **do**
      $\hat{H} \leftarrow G^{\min}(X_1, \ldots, X_r) \cup_{i=1}^{r} \mathcal{A}_{2NC}(G, X_i,\ \eta')$
      continue the loop if any call to any subroutine reports an error
      if $|E(\hat{H})| < |EH|$ **then**
        $H \leftarrow \hat{H}$
      end if
    end for
  end for
end for
return H

▶ **Lemma 14.** *Let $H^* = (V^*, E^*)$ be an optimal subgraph for k-FST. Assume that each of the calls to the subroutine $\mathcal{A}_{2NC}(G, X_i, \eta')$ returns a valid subgraph $2NC(G, X_i)$ whenever one exists. Let $H = (U, F)$ denote the output of the above algorithm. Then, H is a feasible k-FST solution and $|F| \leq |E^*| = \textsc{opt}$.*

Proof. We argue that the subgraph $\hat{H}$ in any iteration of the algorithm is a feasible k-FST solution. This holds because the algorithm finds 2NC subgraphs $2NC(G, X_i)$ and then connects them to one another and to the terminals using the 1-protected paths in $G^{\min}(X_1, \ldots, X_r)$. Thus, $T \subseteq V(\hat{H})$ and any unsafe edge $e \in E(\hat{H})$ either lies in a 2NC subgraph of $\hat{H}$ or a 1-protected path in $\hat{H}$, hence, $\hat{H} - e$ is connected.

Now consider a condensed block-tree $\hat{B}(H^*)$. Let $B_1^*, \ldots, B_{r^*}^*$ be the high-degree blocks of H^* and let X_i^* be the set of cut-nodes in B_i^*. By Lemma 13, the total number of cut-nodes in all the high-degree blocks B_i^* is at most $3k - 6$. We may assume that it is exactly $3k - 6$ by duplicating a cut-node $v \in X_1^*$ multiple times. Consider the iteration of the algorithm where $r = r^*$ and $X_i = X_i^*$ for $i = 1, \ldots, r^*$. Then,

$$|E(2NC(G, X_i))| \leq |E(B_i^*)| \qquad \forall i = 1, \ldots, r.$$

Recall that the nodes of $\hat{B}(H^*)$ correspond to the high-degree blocks B_i^* (and hence to the node sets X_i^*) or to the terminals $\{t\}$. Also an edge $\bar{e}$ of $\hat{B}(H^*)$ between nodes corresponding to node sets V_1 and V_2 represents a 1-protected path $H_{\bar{e}}^*$ in H^* whose end-points lie in V_1 and V_2 respectively. Hence $\hat{B}(H^*)$ may be viewed as a subgraph of $K(X_1, \ldots, X_r)$. Furthermore, since any two nodes in H^* have a 1-protected path between them, $\hat{B}(H^*)$ must be connected. Finally, by construction of K, $|E(H_{\bar{e}}^*)| \geq c(\bar{e})$ where $c(\bar{e})$ is the cost of the edge $\bar{e}$ in K. This implies that the cost of the minimum spanning tree in K is at most $\sum_{\bar{e} \in \hat{B}(H^*)} |E(H_{\bar{e}}^*)|$. Hence,

$$|E(G^{\min}(X_1, \ldots, X_r))| \leq \sum_{\bar{e} \in \hat{B}(H^*)} |E(H_{\bar{e}}^*)|.$$

Combining the two inequalities above we obtain

$$\begin{aligned}
|E(\hat{H})| &= |E(G^{\min}(X_1, \ldots, X_r) \cup_{i=1}^{r} 2NC(G, X_i))| \\
&\leq |E(G^{\min}(X_1, \ldots, X_r))| + \sum_{i=1}^{r} |E(2NC(G, X_i))| \\
&\leq \sum_{\bar{e} \in \hat{B}(H^*)} |E(H_{\bar{e}}^*)| + \sum_{i=1}^{r} |E(B_i^*)| \\
&= |E^*|
\end{aligned}$$

The last equation holds because E^* partitions into the edge-sets of the high-degree blocks B_i^* and the edge-sets of the 1-protected paths $H_{\bar{e}}^*$. This completes the proof of the lemma. ◀

5.1 Proof of Theorem 2

Lemma 14 proves that algorithm $\mathcal{A}_{\text{k-FST}}$ outputs an optimal solution to the k-FST problem with high probability. Let α denote the running time of the algorithm $\mathcal{A}_{\text{2-FST}}$ and let β denote the running time of the algorithm $\mathcal{A}_{\text{2NC}}$. Then, the running time of the algorithm $\mathcal{A}_{\text{k-FST}}$ is bounded by

$$O(n^2 \alpha \cdot n^{3k-6} 2^k n^2 \beta \cdot k^2 \log k) = O\left(\alpha \cdot \beta \cdot n^{3k}\right).$$

Since $\mathcal{A}_{\text{2-FST}}$ has runtime $n^{O(1)}$ and $\mathcal{A}_{\text{2NC}}$ has runtime $n^{O(k)}$, we can conclude that the running time of the algorithm $\mathcal{A}_{\text{k-FST}}$ is $O\left(n^{O(k)} \cdot \log \frac{1}{\eta}\right)$.

Having solved the unweighted k-FST problem, we can directly use the methods from Section 3 to obtain an FPTAS for the weighted k-FST problem.

References

1 David Adjiashvili, Felix Hommelsheim, Moritz Mühlenthaler, and Oliver Schaudt. Fault-Tolerant Edge-Disjoint s-t Paths — Beyond Uniform Faults. In *Proceedings of the 18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 227, pages 5:1–5:19, 2022. doi:10.4230/LIPIcs.SWAT.2022.5.

2 Matthias Bentert, André Nichterlein, Malte Renken, and Philipp Zschoche. Using a Geometric Lens to Find k Disjoint Shortest Paths. In *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 198, pages 26:1–26:14, 2021. doi:10.4230/LIPIcs.ICALP.2021.26.

3 Andreas Björklund, Thore Husfeldt, and Nina Taslaman. Shortest Cycle Through Specified Elements. In *Proceedings of the 23rd Symposium on Discrete Algorithms (SODA)*, pages 1747–1753, 2012. doi:10.1137/1.9781611973099.139.

4 Glencora Borradaile and Philip Klein. The Two-Edge Connectivity Survivable-Network Design Problem in Planar Graphs. *ACM Transactions on Algorithms*, 12(3):30:1–30:29, 2016. doi:10.1145/2831235.

5 Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanita. An Improved LP-based Approximation for Steiner Tree. In *Proceedings of the 42nd Symposium on Theory of Computing*, pages 583–592, 2010. doi:10.1145/1806689.1806769.

6 Artur Czumaj and Andrzej Lingas. On Approximability of the Minimum-Cost k-Connected Spanning Subgraph Problem. In *Proceedings of the 10th Symposium on Discrete Algorithms (SODA)*, pages 281–290, 1999. URL: https://dl.acm.org/doi/pdf/10.5555/314500.314573.

7 Nathaniel Dean. Open Problems. In Neil Robertson and Paul D. Seymour, editors, *Graph Structure Theory, Proceedings of a AMS-IMS-SIAM Joint Summer Research Conference on Graph Minors*, volume 147 of *Contemporary Mathematics*, pages 677–688. American Mathematical Society, 1991.

8 Reinhard Diestel. *Graph Theory*. Graduate Texts in Mathematics. Springer, 2017. doi:10.1007/978-3-662-53622-3.

9 Stuart E Dreyfus and Robert A Wagner. The Steiner Problem in Graphs. *Networks*, 1(3):195–207, 1971. doi:10.1002/net.3230010302.

10 Stefan Fafianie and Stefan Kratsch. An Experimental Analysis of a Polynomial Compression for the Steiner Cycle Problem. In *Proceedings of the 14th International Symposium on Experimental Algorithms (SEA)*, volume 9125 of *Lecture Notes in Computer Science*, pages 367–378, 2015. doi:10.1007/978-3-319-20086-6_28.

11 Andreas Emil Feldmann, Anish Mukherjee, and Erik Jan van Leeuwen. The Parameterized Complexity of the Survivable Network Design Problem. In *Proceedings of the 5th Symposium on Simplicity in Algorithms (SOSA)*, pages 37–56, 2022. doi:10.1137/1.9781611977066.4.

12 Lisa Fleischer, Kamal Jain, and David P. Williamson. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *Journal of Computer and System Sciences*, 72(5):838–867, 2006. doi:10.1016/j.jcss.2005.05.006.

13 Herbert Fleischner and Gerhard J. Woeginger. Detecting cycles through three fixed vertices in a graph. *Information Processing Letters*, 42(1):29–33, 1992. doi:10.1016/0020-0190(92)90128-I.

14 Dorit S. Hochbaum and David B. Shmoys. A Unified Approach to Approximation Algorithms for Bottleneck Problems. *Journal of the ACM*, 33(3):533–550, 1986. doi:10.1145/5925.5933.

15 Oscar H. Ibarra and Chul E. Kim. Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems. *Journal of the ACM*, 22(4):463–468, 1975. doi:10.1145/321906.321909.

16 Kamal Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica*, 21(1):39–60, 2001. doi:10.1007/s004930170004.

17 Tibor Jordán. On minimally k-edge-connected graphs and shortest k-edge-connected Steiner networks. *Discrete Applied Mathematics*, 131(2):421–432, 2003. doi:10.1016/S0166-218X(02)00465-1.

18 Richard M Karp. On the Computational Complexity of Combinatorial Problems. *Networks*, 5(1):45–68, 1975. doi:10.1002/net.1975.5.1.45.

19 Ken-ichi Kawarabayashi. An Improved Algorithm for Finding Cycles Through Elements. In *Proceedings of the 13th Integer Programming and Combinatorial Optimization (IPCO)*, volume 5035, pages 374–384. Springer, 2008. doi:10.1007/978-3-540-68891-4_26.

20 Samir Khuller and Uzi Vishkin. Biconnectivity Approximations and Graph Carvings. *Journal of the ACM*, 41(2):214–235, 1994. doi:10.1145/174652.174654.

21 William Lochet. A Polynomial Time Algorithm for the k-Disjoint Shortest Paths Problem. In *Proceedings of the 32nd Symposium on Discrete Algorithms (SODA)*, pages 169–178, 2021. doi:10.1137/1.9781611976465.12.

22 Zeev Nutov. Node-Connectivity Survivable Network Problems. In Teofilo F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics, Second Edition, Volume 2: Contemporary and Emerging Applications*. Chapman and Hall/CRC, 2018.

23 Neil Robertson and Paul D Seymour. Graph Minors. XIII. The Disjoint Paths Problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995. doi:10.1006/jctb.1995.1006.

24 Sasha Sami. *Parameterized Algorithms for 2-Edge Connected Steiner Subgraphs*. Univerzita Karlova, Matematicko-fyzikální fakulta, 2023. URL: https://dspace.cuni.cz/handle/20.500.11956/179482.

25 Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.

26 Nina Taslaman. *Exponential-Time Algorithms and Complexity of NP-Hard Graph Problems*. Number 83 in ITU-DS. IT-Universitetet i København, 2013. URL: https://pure.itu.dk/ws/portalfiles/portal/39516792/nina_thesis.pdf.

27 Magnus Wahlström. Abusing the Tutte Matrix: An Algebraic Instance Compression for the K-set-cycle Problem. In *Proceedings of the 30th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 20, pages 341–352, 2013. doi:10.4230/LIPIcs.STACS.2013.341.

28 Hassler Whitney. Non-Separable and Planar Graphs. *Transactions of the American Mathematical Society*, 34:339–362, 1932. doi:10.1090/S0002-9947-1932-1501641-2.

29 David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011. URL: http://www.cambridge.org/knowledge/isbn/item5759340/.