FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Sasha Sami

Parameterized Algorithms for 2-Edge Connected Steiner Subgraphs

Department of Applied Mathematics

Supervisor of the master thesis:	Andreas Emil Feldmann, Dr.
Study programme:	Computer Science
Study branch:	Theoretical Computer Science

Prague 2022

I would like to thank my supervisor Andreas Emil Feldmann for guiding me throughout the process of this work.

Title: Parameterized Algorithms for 2-Edge Connected Steiner Subgraphs

Author: Sasha Sami

Department: Department of Applied Mathematics

Supervisor: Andreas Emil Feldmann, Dr., Department of Applied Mathematics

Abstract: In the weighted Minimum 2-Edge Connected Steiner Subgraph (2-ECSS) problem, the input is a simple undirected edge-weighted graph. The task is to find a subgraph with the least cost (sum of weights of edges), such that for each pair of vertices u, v from a distinguished subset (called *terminals*), there exist at least two edge-disjoint u-v paths in the subgraph. We give a randomized XP algorithm, parameterized by the number of terminals, for weighted Minimum 2-ECSS in case of uniform edge weights, at the heart of which lies the randomized algorithm by Björklund, Husfeldt, and Taslaman (SODA 2012), for finding a shortest cycle through a given subset of vertices.

A close variant of weighted Minimum 2-ECSS is the weighted Minimum 2-Edge Connected Steiner Multi-subgraph (2-ECSM) problem. In weighted Minimum 2-ECSM, the solution subgraph can use multiple copies of each edge in the input graph, paying separately for each copy. We show that weighted Minimum 2-ECSM is polynomially equivalent to a problem called Bi-directed Strongly Connected Steiner Subgraph (BI-SCSS), for which an FPT algorithm is known due to Chitnis et al. (TALG 2021). We show that by combining the results of Jordán (Discret. Appl. Math. 2001) and Feldmann et al. (SOSA 2022), one can obtain an FPT algorithm for weighted Minimum 2-ECSM (parameterized by the number of terminals), which, as we prove, gives a faster FPT algorithm for BI-SCSS, compared to the one by Chitnis et al.

Keywords: parameterized algorithms, XP, fixed-parameter tractability, randomized algorithms, 2-edge connected

Contents

List of Notations

Notations

$G(V, E)$	graph G with vertex set V and edge set E
$V(G)$	vertex set of graph G
$E(G)$	edge set of graph G
$d_G(v)$	degree of vertex v in graph G
$\overline{A}$	complement of set A
$\delta(A)$	(elementary) edge cut containing edges uv s.t. $u \in A$ and $v \in \overline{A}$
a-b path	path with end-points a, b
$G \backslash e$	graph obtained by deleting edge e from G
poly-time	polynomial time in the input instance
poly$(\cdot)$	polynomial function
exp$(\cdot)$	exponential function
$[n]$	set of first n natural numbers
$[a \mathinner{..} b]$	interval of all integers between a and b, including a and b

Introduction

In this work, we focus on the (weighted) Minimum k-Edge Connected Steiner Subgraph (k-ECSS) problem and the weighted Minimum k-Edge Connected Steiner Multi-subgraph (k-ECSM) problem, which arise in the design of fault-resilient communication networks [RE06]. An instance of weighted Minimum k-ECSS is a pair (G, T), where G is a simple undirected edge-weighted (with positive real weights) graph and T is a subset of vertices called the *terminals*. The goal is to find a subgraph H with the minimum cost (sum of weights of edges) s.t. any two terminals have at least k edge-disjoint paths between them in H. For the weighted Minimum k-ECSM problem, the solution H can be a *multi*-subgraph, i.e., H is allowed to contain multiple copies of any edge of the input graph, with each copy contributing towards the cost of H. Both weighted Minimum k-ECSS and k-ECSM are known to be NP-hard for $k \geq 1$ [GJ78].

Parameterized algorithms [DF13, FKLM20] are a fairly recent paradigm for "tackling" NP-hard problems. Here, the input comes together with a parameter $k \in \mathbb{N}$, which expresses some property of the input. The instances of a parameterized optimization [LPRS17] problem are pairs $(I, k) \in \Sigma^* \times \mathbb{N}$ for some finite alphabet Σ, and a solution to (I, k) is simply a string $s \in \Sigma^*$, such that $|s| \leq |I| + k$. The *value* of a solution s is given by $\Pi(I, k, s)$, where $\Pi : \Sigma^* \times \mathbb{N} \times \Sigma^* \to \overline{\mathbb{R}}$ is some computable function. Given an input instance to a parameterized minimization (or maximization) problem, the task is to find an *optimum* solution, i.e., a solution with the minimum (or maximum) possible value. A parameterized optimization problem is called *fixed-parameter tractable* (FPT), if there exists an algorithm for it, which computes an optimum solution in time $f(k)n^{\mathcal{O}(1)}$, where $f : \mathbb{N} \to \mathbb{N}$ is some computable function, and n is the input size. The corresponding algorithm is called an FPT algorithm and is said to run in FPT time in parameter k. The idea is to restrict any super-polynomial runtime to the parameter, making the problem at hand tractable for input instances with small values of k. A closely related class of algorithms is that of the slice-wise polynomial (XP) algorithms, where the running time is of the form $f(k)n^{g(k)}$, for some computable functions $f, g : \mathbb{N} \to \mathbb{N}$.

Another common approach for dealing with NP-hard problems is the use of *approximation algorithms* [HV03, WS11]. The idea here is to relax the requirement to compute an optimum solution and instead compute an α-approximation in polynomial time, i.e., a solution within a factor α of the optimum solution.

Weighted Minimum k-ECSS is a special case of the Edge Connectivity Survivable Network Design Problem (EC-SNDP) [KM05], wherein one is asked to find a subgraph H with the minimum cost s.t. for each pair of terminals $s, t \in T$ there are at least $d_{s,t}$ edge-disjoint s-t paths in H, where $\{d_{s,t}\}_{s,t \in T}$ are called demands and are a part of the input. In his seminal work, Jain [Jai01] showed that EC-SNDP admits a poly-time 2-approximation algorithm. We show that one can beat the factor of 2 for unweighted Minimum 2-ECSS (or simply Minimum 2-ECSS) where all the edges have unit weight, and compute an optimal solution with high probability, but using XP time in parameter $|T|$, the number of terminals.

Theorem 1. Minimum 2-ECSS admits a randomized XP algorithm for the parameter $\boldsymbol{k} = |\boldsymbol{T}|$, with a runtime of $n^{\mathcal{O}(k)}$ s.t. the algorithm may fail (can output "fail" or a non-optimal feasible solution) with probability at most $\exp(-\operatorname{poly}(n))$, where n is the number of vertices in the input graph.

We note that while writing this work, Bansal et al. [BCGI22] independently proved Theorem 1, along with some other additional results, including a randomized Fully Polynomial Time Approximation Scheme (FPTAS) for weighted Minimum 2-ECSS, given a *constant* value of $|T|$.

Turning to weighted Minimum 2-ECSM, we show that it is polynomially equivalent to a previously studied [CFM17] problem called the Bi-directed Strongly Connected Steiner Subgraph (BI-SCSS) problem. That is, we show that weighted Minimum 2-ECSM and BI-SCSS are polynomially reducible to each other. In BI-SCSS, given an edge-weighted *bi-directed* graph, the aim is to find a minimum cost strongly connected subgraph which contains a given subset of vertices called the terminals. A bi-directed edge-weighted graph is such, that for every directed edge in the graph there is a directed edge of the same weight in the opposite direction. It is known due to Chitnis et al. [CFM17] that BI-SCSS admits an FPT algorithm with a runtime of $2^{\mathcal{O}(|T|^2)}n^{\mathcal{O}(1)}$, where T is the terminal set and n the number of vertices in the input graph. We show, that by combining the results of Jordán [Jor03] and Feldmann et al. [FMvL21] one can obtain an FPT algorithm with a runtime of $2^{\mathcal{O}(|T|\log|T|)}n^{\mathcal{O}(1)}$ for weighted Minimum 2-ECSM. Using this, in addition to the polynomial reduction from BI-SCSS to weighted Minimum 2-ECSM, we get a faster algorithm for BI-SCSS.

Theorem 2. BI-SCSS admits an FPT algorithm (parameter $|T|$) with a runtime of $2^{\mathcal{O}(|T|\log|T|)}n^{\mathcal{O}(1)}$, where $|T|$ is the number of terminals and n is the number of vertices in the input graph.

In chapter 1 we go through the relevant preliminaries. In chapter 2 we prove Theorem 1. Using results from [BCGI22], we give an alternative proof of Theorem 1 in chapter 3. Finally, in chapter 4 we prove Theorem 2.

Related work

Both (weighted) Minimum 1-ECSS and 1-ECSM are equivalent to the Steiner Tree problem [Kar72], which has been well-studied both in terms of polynomial time approximation algorithms [CC02, BGRS13], and parameterized complexity [DW71, FKM+07, BHKK07, DF13, DFK+21].

All the vertices are terminals. In the (weighted) Minimum Spanning k-Edge Connected Subgraph (Spanning-k-ECS) problem, the aim is to find a minimum cost spanning subgraph in which *every pair* of vertices have k edge-disjoint paths between them, i.e., (weighted) Minimum Spanning-k-ECS is a special case of (weighted) Minimum k-ECSS where every vertex of the input graph is a terminal. Analogously, (weighted) Minimum Spanning k-Edge Connected Multi-subgraph (Spanning-k-ECM) is a special case of (weighted) Minimum k-ECSM. We note that both (weighted) Minimum Spanning-1-ECS and Spanning-1-ECM are equivalent to the Minimum Spanning Tree problem, which is known to admit a poly-time algorithm [Bor26].

It is known that Minimum Spanning-2-ECS is Max-SNP hard [CL99], and thus is unlikely to have a Polynomial Time Approximation Scheme (PTAS). In their ground-breaking work, Khuller and Vishkin [KV94] gave a poly-time $\frac{3}{2}$-approximation algorithm for the same, beating the previous factor of 2. Since the approximation ratio has been improved to the current best factor of $\frac{4}{3}$ [CSS01, SV14, HVV19, KK01, Ç19]. Interestingly, in planar graphs there does exist a PTAS for weighted Minimum Spanning-2-ECS (2-ECM) [BCGZ05, BG07, Kle08]. It turns out that one can, without loss of generality, take the metric closure of the input graph in case of weighted Minimum k-ECSM (see chapter 4). Thus it is known due to Fredrickson and Jájá [FJ82] that Christofides algorithm has a performance guarantee of $\frac{3}{2}$ for weighted Minimum Spanning-2-ECM.

For $k \geq 2$, the most notable result is by Gabow et al. [GGTW08] showing that Minimum Spanning-k-ECM (k-ECS) admits a poly-time algorithm with approximation ratio $1+\frac{2}{k}$, i.e., the ratio approaches 1 as $k \to \infty$. For weighted Minimum Spanning-k-ECM, in a recent work, Karlin et al. [KKGZ21] gave a randomized poly-time approximation algorithm with ratio $1+\sqrt{\frac{8\ln k}{k}}$, which is a step toward answering the open conjecture that the problem (like its unweighted counterpart) admits a $1+\frac{\mathcal{O}(1)}{k}$ poly-time approximation algorithm.

General case and parameterization. It follows from the work of Hu, Hsu, and Lin [HHL00] that weighted Minimum 2-ECSM has a poly-time $\frac{3}{2}$-approximation algorithm. In planar graphs, weighted Minimum 2-ECSM is even known to admit a PTAS[1] [BK16]. For some other special graphs, polytopes associated with the solutions have also been studied [MP00].

Coming to the world of parameterized algorithms, it follows from the work of Feldmann, Mukherjee, and van Leeuwen [FMvL21] that weighted Minimum k-ECSS admits an FPT algorithm for the parameter 'solution size' (the number of edges in the solution). Feldmann et al. [FMvL21] also proved that EC-SNDP is FPT for the parameter $D + tw$, where tw is tree-width of the input graph, and D is the sum of demands $\{d_{s,t}\}_{s,t\in T}$. This implies, that weighted Minimum k-ECSS admits an FPT algorithm for the parameter $k+|T|+tw$, as for (weighted) Minimum k-ECSS the sum of demands is $D = k\binom{|T|}{2}$. However, it remains an open question if there exists an FPT algorithm for (weighted) Minimum k-ECSS, for $k \geq 2$, for the parameter $|T|$.

[1]whether a PTAS exists for weighted Minimum 2-ECSS in planar graphs is an open question.

1. Preliminaries

We skip definitions and notations that are not specific to this work and instead refer the reader to the following texts: Deo's *Graph Theory with Applications to Engineering and Computer Science* [Deo04], *The Design of Approximation Algorithms* by David P. Williamson and David B. Shmoys [WS11], and *Parameterized Algorithms* by Marek Cygan et al. [CFK⁺15].

Unless stated explicitly, we deal only with undirected graphs. A graph is called 2-edge (2-vertex) connected if it remains connected even after an edge (vertex[1]) is removed from the graph. The following theorem gives an equivalent characterization in terms of disjoint paths.

Theorem 3 (Menger's theorem;[Men27])**.**

1. (Edge). Given vertices x and y, the size of the minimum *x-y edge cut* (edges whose removal disconnects x and y) equals the maximum number of pairwise edge-disjoint x-y paths.

2. (Vertex). For two non-adjacent vertices x and y, the size of the minimum *x-y vertex cut* (vertices distinct from x and y, whose removal disconnects x and y) is equal to the maximum number of pairwise internally vertex-disjoint x-y paths.

It follows from Theorem 3 that a graph is 2-edge (vertex) connected if and only if every pair of vertices has at least 2 edge (internally vertex) disjoint paths between them.

Definition 1 (Minimally $(2, T)$-edge connected;[Jor03])**.** A graph $G(V, E)$ is called $(2, T)$-edge connected, if for all u, v $(u \neq v)$ in $T \subseteq V$, there exist 2 edge-disjoint paths between u and v in G. A $(2, T)$-edge connected graph is called minimally $(2, T)$-edge connected if it contains no isolated vertices, and after removal of any edge, it no longer remains $(2, T)$-edge connected.

We now formally define the Minimum 2-ECSS problem.

Problem 1 (Minimum 2-ECSS)**.** Given as input an unweighted *simple* graph $G(V, E)$ with a set of distinguished vertices (terminals) $T \subseteq V$, find a subgraph H of G s.t.

1. H is $(2, T)$-edge connected.

2. $|E(H)|$ is minimized.

If H satisfies only condition 1, it is said to be a *feasible solution*, and if it meets both conditions, it is said to be an optimal solution. Vertices in the set $V \backslash T$ are called *Steiner* vertices.

Analogous to Definition 1 and Problem 1 we can define minimally $(2, T)$-vertex connected graphs (by changing the requirement from edge-disjoint to internally

[1]and the graph has at least 3 vertices.

vertex-disjoint paths), and the Minimum 2-Vertex Connected Steiner Subgraph (2-VCSS) problem, by requiring H to be $(2, T)$-vertex connected in condition 1.

For weighted Minimum 2-ECSS instead of minimizing the number of edges in H (condition 2), the aim is to minimize the sum of weights of edges in H. In weighted Minimum 2-ECSM, H is allowed to have multiple copies of any edge of the input graph, i.e., H is a *multi*-subgraph (instead of a subgraph) of the input graph.

Moving forward, we assume $|T| > 1$, and that the input graph G is $(2, T)$-edge connected (this can be checked in poly-time using Theorem 3 and a *max-flow* algorithm [Din06]) in case of (weighted) Minimum 2-ECSS. Otherwise, one can output "no feasible solution". In the case of weighted Minimum 2-ECSM, it is trivial to check that a feasible solution exists if and only if all the terminals belong to the same connected component. Thus we assume that the input graph to weighted Minimum 2-ECSM is connected.

Definition 2 (Ear decomposition;[Nar16]; see Figure 1.1)**.** An ear of G is a path P of length at least 1, such that the end-points of P may coincide, but every other pair of vertices of P are distinct. An ear-decomposition of G is a sequence $P_0, P_1, \cdots, P_k$, where P_0 is a vertex and $P_1, \cdots, P_k$ are ears such that P_i shares *exactly* its two end-points with the vertices of $P_0 \cup P_1 \cup \cdots \cup P_{i-1}$. An open ear decomposition is an ear decomposition in which the two end-points of each ear P_i ($\boldsymbol{i \geq 2}$) are distinct.

Ear decomposition gives us an alternative way of defining 2-edge (2-vertex) connected graphs, which is crucial to the algorithm in Theorem 1.

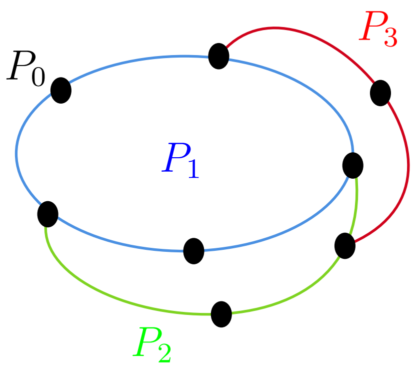
Theorem 4 ([Rob39a])**.** A graph is 2-edge connected if and only if it has an ear decomposition.

Theorem 5 ([Whi32])**.** A graph $G(V, E)$ with $|V| > 2$ is 2-vertex connected if and only if it has an open ear decomposition.

As the following theorem states, an ear decomposition of a 2-edge (vertex) connected graph can be computed efficiently.

Theorem 6 ([Sch12])**.** An ear (open ear) decomposition of a 2-edge (vertex) connected graph can be found in poly-time.

Figure 1.1: An example of an (open) ear decomposition with vertex P_0 and ears P_1, P_2 and P_3.



Definition 3 (Extended edge)**.** An extended edge between vertices u and v ($u \neq v$) in graph G, denoted by uv_{ex} is a path $u, w_1, \cdots, w_i, \cdots, v$ such that for all the internal vertices w_i, $d_G(w_i) = 2$. An edge uv is an extended edge with no internal vertices.

Definition 4 (Chord-path)**.** Let u, v ($u \neq v$) be two vertices that lie on a cycle C. Let uv_{ex} denote an extended uv edge. Then uv_{ex} is called a chord-path (simply chord, if uv_{ex} is just an edge) of C if it does not share any edges or vertices with C (apart from u, v).

We formally define the problems BI-SCSS and EC-SNDP, as we would require these definitions in chapter 4.

Problem 2 (BI-SCSS;[CFM17])**.** Given a *directed* graph $G(V, E)$ with weight function $w : E \rightarrow \mathbb{R}^+$ and a subset of vertices $T \subseteq V$ (called terminals), s.t. for every edge $\overrightarrow{uv}$ (directed edge from u to v) there is an edge $\overrightarrow{vu}$ of the same weight in G, the task is to find a subgraph H of minimum cost (sum of weights of edges) s.t. $T \subseteq V(H)$ and H is a strongly connected component.

Problem 3 (EC-SNDP;[FMvL21])**.** We are given a (multi-) graph $G(V, E)$ with positive real edge-weights, together with a terminal set $T \subseteq V$, and a non-negative integer demand $d_{s,t} \in \mathbb{N}_0$ for each terminal pair $s, t \in T$. The aim is to find a subgraph H so that each terminal pair $s, t \in T$ is connected by at least $d_{s,t}$ edge-disjoint paths in H, while minimizing the sum of weights of edges in H.

2. Randomized XP algorithm

In sections 2.1 and 2.2, we make some observations about the structure of an optimal solution to Minimum 2-ECSS, essentially showing that an optimal solution "decomposes into a tree-like structure". We then use these observations to describe the randomized XP algorithm for Minimum 2-ECSS in section 2.3.

2.1 Structure of an optimal solution

First, we make the following three observations about a minimally $(2,T)$-edge connected graph, which will be useful going forward.

Observation 1. If a graph G is minimally $(2,T)$-edge connected, it is 2-edge connected.

Proof. See Appendix A. $\square$

Observation 2. If $G(V,E)$ is 2-edge connected, then every edge $uv \in E$ lies on some cycle.

Proof. See Appendix A. $\square$

Observation 3. Any chord-path uv_{ex} in a minimally $(2,T)$-edge connected graph $G(V,E)$ has at least one terminal in its *interior*.

Proof. By contradiction assume there exists a chord-path uv_{ex} (of some cycle C) s.t. uv_{ex} either has no internal vertices or all the internal vertices of uv_{ex} are Steiner vertices. Let $G'(V',E')$ be the graph obtained by removing all the edge(s) (and internal vertices, if any) of uv_{ex}. We prove that G' is $(2,T)$-edge connected, contradicting the minimality of G. First, we observe that all the terminals (vertices in T) and the cycle C exist in G' as we remove only edges and possibly (internal) Steiner vertices along the chord-path uv_{ex}. By contradiction, assume for some $x,y \in T$ the maximum number of edge-disjoint x-y paths in G' is less than two. Using Theorem 3 there exists an x-y cut (in G') characterized by $\emptyset \neq A \subsetneq V'$ s.t. $x \in A$, $y \in \overline{A}$ and there exist less than 2 edges in the cut $\delta(A)$. Either $u,v \in A$ or $u,v \in \overline{A}$, otherwise we would have at least 2 edges in the cut $\delta(A)$, as u,v lie on the cycle C. Without loss of generality, let $u,v \in A$. Consider the x-y cut in G characterized by $B = A \cup V_{ex}$ where V_{ex} is the set of internal vertices of uv_{ex}. As all the internal vertices of uv_{ex} have degree 2, and all of them are in B, the number of edges in the x-y cut $\delta(B)$ in G is the same as the number of edges in the x-y cut $\delta(A)$ in G', which is less than 2. Using Theorem 3, the maximum number of edge-disjoint paths between x and y in G is less than 2, which leads to a contradiction as G is minimally $(2,T)$-edge connected. $\square$

Let H^* be an optimal solution to Minimum 2-ECSS. As $|T| > 1$, without loss of generality, we assume H^* has no isolated vertices and is thus minimally $(2,T)$-edge connected. We begin by noting the following crucial theorem, which states that for a minimally $(2,T')$-edge connected *simple* graph with all the Steiner vertices having a degree of at least 3, the total number of vertices in the graph is bounded in terms of the number of terminals.

Theorem 7 ([Jor03])**.** Let $G(V, E)$ be a minimally $(2, T')$-edge connected *simple* graph s.t. for every vertex $v \in V \backslash T'$, $d_G(v) \geq 3$. Then $|V| \leq \frac{17}{3}|T'| - 8$.

The idea behind using Theorem 7 is to guess the (Steiner) vertices in an optimal solution to Minimum 2-ECSS using XP time in the number of terminals in the input graph. Thus, in light of Theorem 7, we "short-circuit" Steiner vertices of degree 2. We follow the convention that a loop contributes a degree of 2, and p copies of an edge uv ($u \neq v$) contribute p each towards degrees of u and v. Let H^*_{sc} denote the graph obtained from H^* by repeating the following operation exhaustively while there exists a Steiner vertex u of degree 2.

Op 1

- Let x and y be neighbors of u. Short-circuit u by deleting it and creating an edge between x and y, allowing multi-edges to be formed in the process.

We note that in the definition of **Op 1**, we implicitly assume that the neighbors x, y of the vertex u being short-circuited are distinct and different from u. The proof of Lemma 1 justifies this assumption.

Lemma 1. *Let the sequence of graphs created by exhaustive application of **Op** 1 be $H_0, H_1, \cdots, H_i, \cdots, H_l$ where $H_0 = H^*$ and $H_l = H^*_{sc}$. Then H_i for $0 \leq i \leq l$ is minimally $(2, T)$-edge connected.*

Proof. We give proof by induction. For $H_0 = H^*$ the lemma follows from optimality of H^* (and the assumption that H^* has no isolated vertices). Assuming the lemma holds for $0 < i < l$, we prove it holds for $i+1$ as well. By the induction hypothesis, H_i is minimally $(2, T)$-edge connected; thus, it contains no loops. Let u be the degree 2 Steiner vertex in H_i that is short-circuited to obtain H_{i+1}. Then u has distinct neighbors x and y, as otherwise, if u shared two parallel edges with some vertex w, it would contradict the minimality of H_i, as the parallel edges incident to u could be deleted (still keeping the graph $(2, T)$-edge connected). First we observe H_{i+1} is $(2, T)$-edge connected. For arbitrary $a, b \in T$ let p_1 and p_2 be two edge-disjoint a-b paths in H_i. If none of these paths contain edges in $\{ux, uy\}$, then they are edge-disjoint a-b paths in H_{i+1} as well. Otherwise, without loss of generality let p_1 contain edge xu. As u has a degree of 2 and is a Steiner vertex (so $u \notin \{a, b\}$), p_1 contains the edge uy as well. More precisely, p_1 contains the sub-path xuy. As p_2 is edge-disjoint from p_1 it does not contain edges in $\{ux, uy\}$. We can substitute the sub-path xuy in p_1 with the "new" (the only, if edge xy did not exist previously) copy f of edge xy (obtained by short-circuiting u), getting 2 edge-disjoint a-b paths in H_{i+1}. Now assume by contradiction that H_{i+1} is not minimally $(2, T)$-edge connected. Then, there exists an edge e in H_{i+1} such that $H'_{i+1} := H_{i+1} \backslash e$ is $(2, T)$-edge connected (note, H_{i+1} has no isolated vertices, as by the induction hypothesis H_i has no isolated vertices, and u has distinct neighbors).

Case (1): $e = f$. For arbitrary $a, b \in T$ let p_1, p_2 be two edge-disjoint a-b paths in H'_{i+1}. Then p_1, p_2 exist in H'_i as well, where H'_i is the graph obtained from H_i by deleting edges ux, uy. This contradicts the minimality of H_i.

Case (2): e is an edge that exists in H_{i+1} and H_i. For arbitrary $a, b \in T$ let p_1, p_2 be two edge-disjoint a-b paths in H'_{i+1}. If none of p_1, p_2 contain the edge f, then p_1, p_2 exist in $H_i \backslash e$ as well. Otherwise, without loss of generality let p_1 contain f. As e cannot be either of ux, uy (as they do not exist in H_{i+1}), we can substitute f by path xuy in p_1 to obtain p_1'. Now p_1',p_2 are edge-disjoint a-b paths in $H_i \backslash e$. This again contradicts the minimality of H_i. $\square$

Corollary 1. H^*_{sc} does not contain loops and has at most 2 copies of any edge.

Proof. Absence of loops follows trivially from minimality of H^*_{sc} (which follows from Lemma 1). H^*_{sc} cannot have more than 2 copies of an edge uv, as two copies of uv can be treated as a cycle and the third copy as a chord, which using Observation 3 contradicts the minimality of H^*_{sc}. $\square$

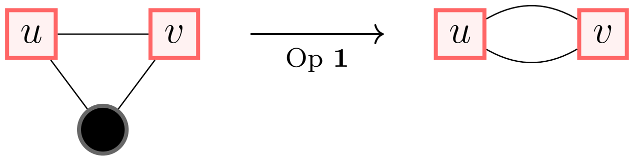


Figure 2.1: Example of application of **Op 1**. Red (square) vertices are terminals, and the black vertex is a Steiner vertex.

It follows from the construction of H^*_{sc} and Lemma 1 that H^*_{sc} is minimally $(2, T)$-edge connected, and does not contain Steiner vertices of degree less than 3. But we cannot directly apply Theorem 7 to H^*_{sc}, as it can be a multi-graph (follows from Corollary 1; see Figure 2.1). Thus we "break" H^*_{sc} into sub-parts and analyze them separately. Before doing so, we make two observations, which later help us argue about the structure of H^* from that of H^*_{sc}. The first observation (Corollary 2) states that the set of vertices short-circuited by exhaustive application of **Op 1** is exactly the set of degree 2 Steiner vertices *in* $\boldsymbol{H^*}$. The second observation (Observation 5) broadly states that each edge uv in H^*_{sc} corresponds to an extended edge uv_{ex} in H^*, such that, the internal vertices of uv_{ex} are from the set of short-circuited vertices.

Observation 4. Let the sequence of graphs created by exhaustive application of **Op 1** be $H_0, H_1, \cdots, H_i, \cdots, H_l$ where $H_0 = H^*$ and $H_l = H^*_{sc}$. Then for $0 \le i \le l$ the following holds for every vertex v in H_i : $d_{H_i}(v) = d_{H^*}(v)$.

Proof. We give proof by induction. The claim trivially holds for $i = 0$. Assuming it holds for $0 < i < l$, we prove it for $i + 1$. Let u be the degree 2 Steiner vertex in H_i, which is short-circuited to obtain H_{i+1}. Using Lemma 1, we know H_i is minimally $(2, T)$-edge connected. Thus u has two distinct neighbors x, y (in H_i). The loss of a degree of 1 (for x and y) by deleting edges ux and uy is offset by creating an edge xy. As degrees of other vertices remain unchanged while going from H_i to H_{i+1}, using the induction hypothesis we get that for each vertex v in H_{i+1}, $d_{H_{i+1}}(v) = d_{H_i}(v) = d_{H^*}(v)$. $\square$

Corollary 2. Let S denote the set of degree 2 Steiner vertices in H^*, then $V(H^*_{sc}) = V(H^*) \backslash S$.

Proof. Let u_i $(0 \leq i < l)$ denote the degree 2 Steiner vertex in H_i, that is short-circuited to obtain H_{i+1}. Using Observation 4, $d_{H_i}(u_i) = d_{H^*}(u_i)$, i.e. u_i is a degree 2 Steiner vertex in H^*. Conversely, assume by contradiction that a degree 2 Steiner vertex u in H^* exists in H^*_{sc} as well. Again using Observation 4, we get $d_{H^*_{sc}}(u) = d_{H^*}(u) = 2$. But this leads to a contradiction as **Op 1** can still be applied to H^*_{sc}. □

Observation 5. Let the sequence of graphs created by exhaustive application of **Op 1** be $H_0, H_1, \cdots, H_i, \cdots, H_l$ where $H_0 = H^*$ and $H_l = H^*_{sc}$. Let u_i $(0 \leq i < l)$ be the vertex short-circuited to obtain H_{i+1} from H_i. Also, let $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{u_{i-1}\}$ for $1 \leq i \leq l$. Then for $0 \leq i \leq l$, H^* can be obtained from H_i by replacing each edge uv in H_i by an extended edge uv_{ex} in H^* s.t. the following holds:

1. $I(uv_{ex}) \subseteq S_i$ where $I(uv_{ex})$ represents the set of internal vertices of uv_{ex}.
2. For any two distinct edges (parallel edges are considered to be distinct) $e^1, e^2 \in H_i$, if the extended edges they are replaced with are e^1_{ex} and e^2_{ex} respectively, then $I(e^1_{ex}) \cap I(e^2_{ex}) = \emptyset$.

Proof. We give proof by induction. For $H_0 = H^*$, edge uv in H_0 can be replaced by itself i.e. $uv_{ex} = uv$. Trivially the resulting graph is H^*. As $I(uv) = \emptyset$, the observation follows. Assuming the observation holds for $0 < i < l$ we prove it holds for $i+1$ as well. It follows from Lemma 1 that u_i has two distinct neighbors x, y in H_i. While going from H_i to H_{i+1}, edges u_ix, u_iy are deleted, and a *new* xy edge is created; let it be denoted by f. If edge $e \in H_i \cap H_{i+1}$ is replaced by extended edge e_{ex} in H_i (to obtain H^*), then we replace e by e_{ex} in H_{i+1} as well. Let u_ix, u_iy be replaced by extended edges u_ix_{ex}, u_iy_{ex} respectively in H_i. Then we replace f by the path $f_{ex} := u_ix_{ex}, u_iy_{ex}$ in H_{i+1}. Using the induction hypothesis and the fact that u_i is a degree 2 (Steiner) vertex in H^* (as by Observation 4 we have $d_{H_i}(u_i) = d_{H^*}(u_i) = 2$), it follows that f_{ex} is an extended xy edge in H^*. It is not hard to see that using the mapping mentioned above from edges of H_{i+1} to extended edges of H^*, we obtain H^*. Lastly, we prove that conditions 1 and 2 are met. Using the induction hypothesis, we know that for every edge $e \in H_i \cap H_{i+1}$, $I(e_{ex}) \subseteq S_i \subseteq S_{i+1}$, and $I(f_{ex}) \subseteq S_{i+1}$ as $I(u_ix_{ex}), I(u_iy_{ex}) \subseteq S_i \subseteq S_{i+1}$ and $u_i \in S_{i+1}$. Again using the induction hypothesis, we know that for any two distinct edges $e^1, e^2 \in H_i \cap H_{i+1}$, $I(e^1_{ex}) \cap I(e^2_{ex}) = \emptyset$, and for any edge $e \in H_i \cap H_{i+1}$, $I(e_{ex}) \cap I(f_{ex}) = \emptyset$, since $I(e_{ex}) \cap (I(u_ix_{ex}) \cup I(u_iy_{ex})) = \emptyset$, and $u_i \notin S_i \supseteq I(e_{ex})$. □

Coming back to the idea of breaking H^*_{sc} into smaller sub-parts, we first show that, $\widetilde{H^*_{sc}}$, the *simple* graph underlying H^*_{sc} (see Definition 6) "decomposes into a tree-like structure" (Observation 9).

Definition 5. Let X_{sc} denote the set of edges in H^*_{sc} that have a single copy, and Y_{sc} the set of edges that have two copies, i.e. if there are two copies of uv in H^*_{sc}, it is counted once in Y_{sc}.

For example in Figure 2.1 $X_{sc} = \emptyset$ and $Y_{sc} = \{uv\}$. Recall, using Corollary 1 we know that each edge in H^*_{sc} has at most two copies.

Definition 6. Let $\widetilde{H^*_{sc}}$ denote the simple graph obtained from H^*_{sc} by removing one copy of each edge in Y_{sc}.

For proving Observation 9, we make some auxiliary observations.

Observation 6. There exists no cycle in H^*_{sc} that contains a copy of some edge $uv \in Y_{sc}$, except for the one formed by two copies of uv (a trivial cycle).

Proof. Assume by contradiction that there exists a (non-trivial) cycle C, which contains a copy of some edge $uv \in Y_{sc}$. Then C contains exactly one copy of uv, and the other copy forms a chord of C. But, using Observation 3 this contradicts the minimality of H^*_{sc} (which follows from Lemma 1). □

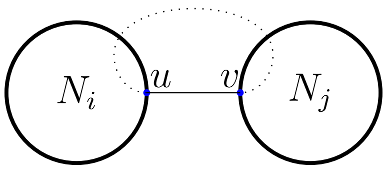
Let $N_1, N_2, \cdots N_t$ denote the maximally 2-edge connected components in $\widetilde{H^*_{sc}}$ s.t. $\bigcup_{i=1}^{t} V(N_i) = V(\widetilde{H^*_{sc}})$, where possibly N_i might consist of a single vertex. Note, by maximality of N_i, $V(N_i) \cap V(N_j) = \emptyset$ for $i \neq j$. We call N_i *nodes*. We say an edge uv in $\widetilde{H^*_{sc}}$ belongs to node N_i if $u, v \in V(N_i)$, and thus by maximality of N_i, $uv \in E(N_i)$.

Observation 7 and Observation 8 simply show that all the edges in Y_{sc} are *exactly* the bridges in $\widetilde{H^*_{sc}}$, and all edges in X_{sc} belong to some node.

Observation 7. Edges in Y_{sc} do not belong to any node.

Proof. Consider an arbitrary edge $uv \in Y_{sc}$. Assume by contradiction that it belongs to some node N_i. As N_i is 2-edge connected, by Observation 2 uv lies on some cycle C in N_i (in $\widetilde{H^*_{sc}}$). C exists in H^*_{sc} as well and contains a single copy of uv, as $\widetilde{H^*_{sc}}$ is a simple graph by construction. However, this leads to a contradiction using Observation 6, as C is a non-trivial cycle. □

Figure 2.2: Pictorial aid for Observation 8



Observation 8. Every edge in X_{sc} belongs to some node.

Proof. Assume by contradiction that there exists $uv \in X_{sc}$ s.t. uv does not belong to any node, i.e. $u \in N_i$ and $v \in N_j$ for $i \neq j$. Using Lemma 1 and Observation 1 we know that H^*_{sc} is 2-edge connected. Also, using Observation 2 we know that there exists a cycle C in H^*_{sc}, which contains the edge uv. C exists in $\widetilde{H^*_{sc}}$ as well, as C is a non-trivial cycle (as $uv \in X_{sc}$). But this contradicts maximality of N_i and N_j as N' which contains N_i, N_j and C is 2-edge connected (see Figure 2.2). Any u-v cut in N' has at least 2 edges as u, v lie on the cycle C; for any other cut characterized by $\emptyset \neq A \subsetneq V(N')$ s.t. $u, v \in A$:

- if there exists a vertex $w \in \overline{A}$ s.t. $w \in V(N_i)$ (or $V(N_j)$). Then there exist at least 2 edges in the cut $\delta(A)$ as $u, w \in V(N_i)$ (or $v, w \in V(N_j)$) and N_i (N_j) is 2-edge connected.

- otherwise, there exists a vertex $w \in \overline{A}$ s.t. $w \notin \{u, v\}$ and $w \in C$. Then there exist at least 2 edges in $\delta(A)$ as u, v, w lie on C and $u, v \in A$.

□

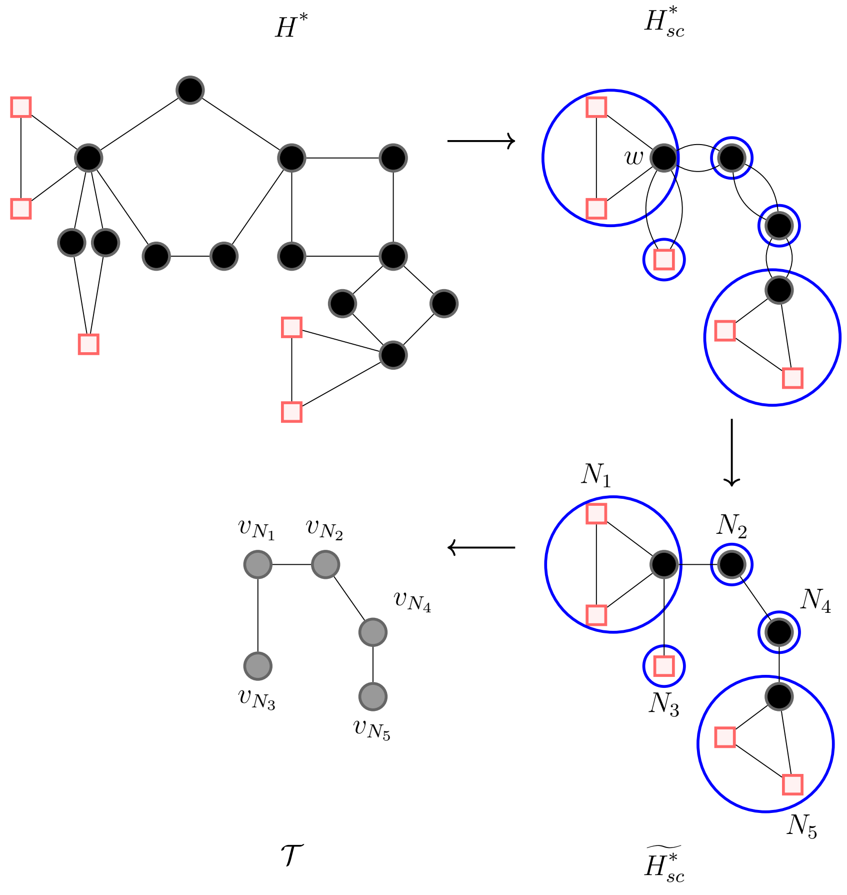
2.1.1 Summarizing the structure

Consider the graph $\mathcal{T}$ obtained by contracting each node N_i in $\widetilde{H^*_{sc}}$ to a vertex. We denote the vertex of $\mathcal{T}$ corresponding to N_i by v_{N_i}.

Observation 9. $\mathcal{T}$ is a tree (see Figure 2.3).

Proof. Using Lemma 1, we know that H^*_{sc} is minimally $(2, T)$-edge connected and hence connected. As $\widetilde{H^*_{sc}}$ is obtained from H^*_{sc} by removing one copy of every edge (and leaving the other) in Y_{sc}, $\widetilde{H^*_{sc}}$ is connected as well. Using Observation 8 and Observation 7, we know that all the edges in X_{sc} belong to some node, and edges in Y_{sc} do not belong to any node, and thus are exactly the edges of $\mathcal{T}$, in the sense that edge $uv \in Y_{sc}$, where $u \in N_i$ and $v \in N_j$, becomes edge $v_{N_i}v_{N_j}$ in $\mathcal{T}$. As a result, edges of $\mathcal{T}$ do not form a cycle, as it would imply the existence of a non-trivial cycle in H^*_{sc} containing a single copy of some edge in Y_{sc}, which using Observation 6 would lead to a contradiction. □

Figure 2.3: An example showing how $\mathcal{T}$ is obtained from H^*. Red (square) vertices are terminals, and black vertices are Steiner vertices. The blue circles denote the nodes. The figure also shows a boundary vertex w of node N_1.



We now summarize what the observations so far imply about the structures of H^*_{sc} and H^*. As $\widetilde{H^*_{sc}}$ is obtained from H^*_{sc} by keeping only one copy of every edge in Y_{sc}, using Observation 7 and Observation 8 it follows that, $\boldsymbol{H^*_{sc}}$ exactly comprises of nodes N_i, and a pair of parallel edges corresponding to *each* edge of $\mathcal{T}$; with edge $v_{N_i}v_{N_j}$ corresponding to two parallel edges between u and v, such that $u \in V(N_i)$ and $v \in V(N_j)$. Also, using Observation 5 we have, that corresponding to every edge uv with two copies (pair of parallel edges) in H^*_{sc}, there exist two (internally) vertex-disjoint u-v paths in H^*. Furthermore, as each node by definition is 2-edge connected (possibly a single vertex), using Theorem 3 and Observation 5 it is not hard to see that, every node N_i (in H^*_{sc}) corresponds to a $(2, V(N_i))$-edge connected subgraph of H^*. Using Corollary 2, it also follows that any vertex in the subgraph which is *not* in $V(N_i)$, is a degree 2 Steiner vertex of H^*.

2.2 Looking at parts of the tree structure

Recall that by construction, the nodes are simple graphs. In Lemma 2, we show that every node N_i is minimally $(2, T_i \cup B_i)$-edge connected (or contains a single vertex); with T_i and B_i being defined in Definition 8. Also, Observation 10 tells us, that the degree of every vertex $s \in V(N_i)\backslash(T_i \cup B_i)$, in N_i, is at least 3. This gives us all the ingredients to apply Theorem 7 to every node.

Definition 7. A vertex $v \in N_i$ is called a boundary vertex (of N_i), if there exists an edge uv (in $\widetilde{H^*_{sc}}$) s.t. $u \notin N_i$.

We note that a boundary vertex (see Figure 2.3) can be a terminal or a Steiner vertex.

Definition 8. Let B_i and T_i denote the set of boundary vertices and terminals in node N_i.

Observation 10. For any *Steiner* vertex $s \in V(N_i)\backslash B_i$ (i.e. a non-boundary Steiner vertex of N_i) we have $d_{N_i}(s) \geq 3$.

Proof. As s is a non-boundary vertex, $d_{N_i}(s) = d_{H^*_{sc}}(s)$, and $d_{H^*_{sc}}(s) \geq 3$. The latter follows from exhaustive application of **Op 1** and minimality of H^*_{sc} (Lemma 1). $\square$

Lemma 2. *Every node N_j is either minimally $(2, T_j \cup B_j)$-edge connected or contains a single vertex.*

Proof. Assume by contradiction that there exists an index i s.t. N_i contains more than one vertex and is not *minimally* $(2, T_i \cup B_i)$-edge connected. We note that by using Theorem 3, N_i is $(2, T_i \cup B_i)$-edge connected, as N_i is 2-edge connected by definition. It means that there exists an edge e in N_i s.t. $N'_i \coloneqq N_i\backslash e$ is still $(2, T_i \cup B_i)$-edge connected (note, N_i has no isolated vertices). We prove $H \coloneqq H^*_{sc}\backslash e$ is $(2, T)$-edge connected which contradicts the minimality of H^*_{sc}. Let $M_j \coloneqq N_j$ for $j \in [t]\backslash\{i\}$ and $M_i \coloneqq N'_i$ (recall t is the number of nodes). Now consider arbitrary terminals $a, b \in T$.

Case (1): $a, b \in V(M_x)$ for some x. Then, there exist 2 edge-disjoint paths between them, as M_x is $(2, T_x \cup B_x)$-edge connected (and $a, b \in T_x$).

Case (2): $a \in M_x$ and $b \in M_y$ $(x \neq y)$. It follows from the tree structure of $\mathcal{T}$, that there exists a path $P = v_{N_x}, \cdots, v_{N_y}$ in $\mathcal{T}$. But this implies that there exist 2 edge-disjoint paths between a and b (in H), as (1) for each edge $v_{N_j}v_{N_z}$ in P, there exist vertices p, q such that there are two parallel edges between p and q in H^*_{sc} (and hence H)[1], where p and q are boundary vertices of N_j and N_z respectively, and (2) for all $j \in [t]$, there exist two edge-disjoint paths in M_j between any two vertices in $T_j \cup B_j$. □

Finally, we apply Theorem 7 to every node, bounding the total number of vertices in a node in terms of the number of terminals and boundary vertices in it.

Lemma 3. *For a node N_i, the total number of vertices in it is upper bounded by* $6 \cdot |T_i \cup B_i|$.

Proof. First, we consider the case where N_i contains a single vertex. If the sole vertex of N_i is a boundary vertex, then the lemma follows trivially. Otherwise, if N_i has no boundary vertex then, $\widetilde{H^*_{sc}}$ consists of a single node (N_i), as $\widetilde{H^*_{sc}}$ is connected (which follows from Lemma 1 and the definition of $\widetilde{H^*_{sc}}$). However, this leads to a contradiction as we assume $|T| > 1$, whereas N_i consists of a single vertex.

If N_i consists of more than one vertex, then by Lemma 2 it is minimally $(2, T_i \cup B_i)$-edge connected. Also, using Observation 10, for any non-boundary Steiner vertex s of N_i, we have that $d_{N_i}(s) \geq 3$. Lastly, we recall that N_i is a simple graph by definition. Thus, by using Theorem 7, we get

$$|V(N_i)| \;\leq\; \frac{17}{3}|T_i \cup B_i| - 8 \;\leq\; 6 \cdot |T_i \cup B_i|$$

□

Now, we bound the size of the vertex set of $\mathcal{T}$ except for a certain subset L, in terms of $\boldsymbol{k = |T|}$, the number of terminals in the input graph (Observation 14). We also bound the total number of vertices in nodes corresponding to vertices in $V(\mathcal{T})\backslash L$, in Lemma 4. As we see in the next section, this helps us guess the nodes corresponding to vertices in $V(\mathcal{T})\backslash L$, in time $n^{\mathcal{O}(k)}$, where n is the number of vertices in the input graph. In Lemma 5, we also show that the nodes corresponding to vertices (of $\mathcal{T}$) in L consist of a single (Steiner) vertex.

We root $\mathcal{T}$ at some arbitrary vertex and begin by observing that a node corresponding to a leaf of $\mathcal{T}$ contains at least one terminal. Note that we do not consider the root vertex as an internal vertex or as a leaf.

Observation 11. If v_{N_i} is a leaf in $\mathcal{T}$, then N_i contains at least one terminal.

Proof. Assume by contradiction that there exists a leaf v_{N_i} in $\mathcal{T}$ s.t. N_i does not contain any terminal. Let the edge incident to v_{N_i} (in $\mathcal{T}$) be $e = v_{N_i}v_{N_j}$. Then, by using Observation 7 and Observation 8 we know that e corresponds to two copies

[1]recall for each edge in $\mathcal{T}$ there exist parallel edges in H^*_{sc} (with end-points in different nodes). These parallel edges exist in H as well, as e belongs to a node, namely N_i.

of an edge uv in H^*_{sc} where $u \in N_i$ and $v \in N_j$ (where u is the sole boundary vertex of N_i). But this contradicts the minimality of H^*_{sc} (Lemma 1), as N_i and two copies of the edge uv could be removed from H^*_{sc}, still keeping the resulting graph $(2, T)$-edge connected. $\square$

Corollary 3. Let l denote the number of leaves in $\mathcal{T}$, and r the number of internal vertices v_{N_i} s.t. N_i contains a terminal. Then, $l + r \leq k$.

Proof. Follows from Observation 11 and (vertex) disjointness of the nodes. $\square$

Next, we observe that the number of internal vertices in $\mathcal{T}$ with at least 2 children is bounded from above by k.

Observation 12. Let s denote the number of internal vertices in $\mathcal{T}$ with at least 2 children. Then $s \leq k$.

Proof. First, we note that the number of internal vertices with at least 2 children is bounded by the number of leaves in a tree (which can be proved by induction [B2́2]). Thus, the observation follows from Corollary 3. $\square$

After Corollary 3 and Observation 12, we are left to argue about the size of set L, where L is the set of internal vertices v_{N_i} s.t. v_{N_i} has exactly 1 child, and N_i does not contain a terminal. Instead of trying to bound $|L|$ in terms of k, we short-circuit the vertices in L. Let $\mathcal{S}$ be the tree obtained from $\mathcal{T}$ by short-circuiting vertices in L (see Figure 2.4), while retaining the labels of other vertices. The following two observations show that every terminal belongs to a node corresponding to some vertex of $\mathcal{S}$ and that the number of vertices in $\mathcal{S}$ is upper bounded in terms of k.

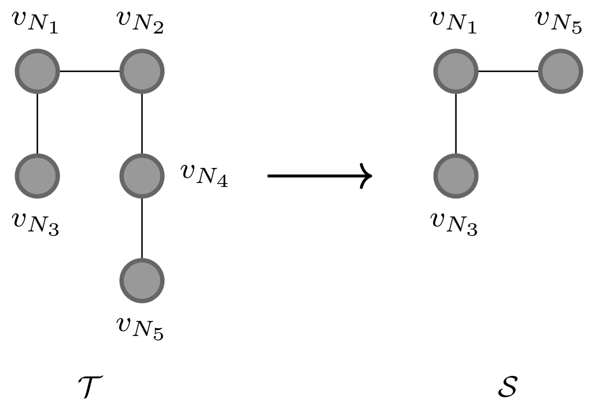
Observation 13. $T \subseteq \bigcup\limits_{i: v_{N_i} \in V(\mathcal{S})} V(N_i)$.

Proof. We know $T \subseteq \bigcup\limits_{i=1}^{t} V(N_i)$. As $V(\mathcal{S}) = \left(\bigcup\limits_{i=1}^{t} \{v_{N_i}\}\right) \setminus L$, the observation follows from the definition of L. $\square$

Observation 14. $|V(\mathcal{S})| \leq 2k + 1$.

Proof. Follows from Corollary 3, Observation 12, and the definition of $\mathcal{S}$. The plus 1 accounts for the root. $\square$

Figure 2.4: Example showing how $\mathcal{S}$ is obtained for $\mathcal{T}$ as per the example in Figure 2.3. We root $\mathcal{T}$ at v_{N_1} and short-circuit vertices v_{N_2} and v_{N_4}.



In the following lemma, we bound the total number of vertices in nodes corresponding to vertices of $\mathcal{S}$.

Lemma 4. *The total size of all the nodes corresponding to the vertices of* $\mathcal{S}$ *is upper bounded by* $30k$*, i.e.,* $\left(\sum_{i:v_{N_i}\in V(\mathcal{S})} |V(N_i)|\right) \leq 30k$.

Proof. We claim that the total number of boundary vertices across all the nodes corresponding to vertices of $\mathcal{S}$ is bounded by $4k$, i.e. $\left(\sum_{i:v_{N_i}\in V(\mathcal{S})} |B_i|\right) \leq 4k$. Consider an arbitrary node N_i. Let u be an arbitrary boundary vertex of N_i (if one exists). Then, by the definition of a boundary vertex, there exists an edge incident to u (in $\widetilde{H}^*_{sc}$) s.t. the other end-point of the edge does not belong to N_i. Thus, the number of boundary vertices in N_i is upper bounded by the number of edges in $\widetilde{H}^*_{sc}$ which have exactly one end-point in N_i. By construction of $\mathcal{T}$, an edge uv in $\widetilde{H}^*_{sc}$, where $u \in N_i$ and $v \in N_j$ for $i \neq j$, corresponds to edge $v_{N_i}v_{N_j}$ in $\mathcal{T}$. Therefore, the number of boundary vertices in N_i is upper bounded by the number of edges incident to v_{N_i} in $\mathcal{T}$, i.e. $|B_i| \leq d_{\mathcal{T}}(v_{N_i})$. As short-circuiting vertices in L to obtain $\mathcal{S}$ leaves the degrees of vertices (of $\mathcal{T}$) apart from L unchanged, we have $|B_j| \leq d_{\mathcal{S}}(v_{N_j})$, for an arbitrary vertex v_{N_j} of $\mathcal{S}$. As the nodes are vertex disjoint, by summing we get that $\left(\sum_{i:v_{N_i}\in V(\mathcal{S})} |B_i|\right) \leq 2|E(\mathcal{S})|$, where the factor of 2 comes from each edge of $\mathcal{S}$ being counted twice, once for each of its end-points. The claim follows by using the fact that $|E(\mathcal{S})| \leq 2k$, which follows from Observation 14.

Using the above claim about the number of boundary vertices, we get

$$\begin{aligned}
\sum_{i:v_{N_i}\in V(\mathcal{S})} |V(N_i)| &\leq \sum_{i:v_{N_i}\in V(\mathcal{S})} 6 \cdot |T_i \cup B_i| \\
&\leq 6 \cdot \Bigg(\sum_{i:v_{N_i}\in V(\mathcal{S})} |T_i| + |B_i| \Bigg) \\
&= 6 \cdot \Bigg(k + \sum_{i:v_{N_i}\in V(\mathcal{S})} |B_i| \Bigg) \\
&\leq 6 \cdot (k + 4k) \\
&= 30k
\end{aligned}$$

where the first inequality follows from Lemma 3, and the first equality follows from disjointness of nodes and the fact that every terminal belongs to a node corresponding to some vertex of $\mathcal{S}$ (Observation 13).

□

Though we did not argue about the size of L, we show the following important property of nodes corresponding to vertices (of $\mathcal{T}$) in L.

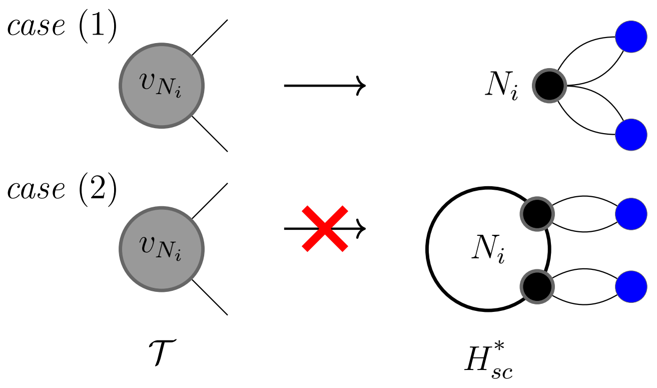
Lemma 5. *For every vertex* $v_{N_i} \in L$*, the corresponding node* N_i *consists of only* 1 *(Steiner) vertex, which is also the only boundary vertex of* N_i*.*

Proof. Consider an arbitrary vertex $v_{N_i} \in L$. By construction of $\mathcal{T}$, the number of edges incident to v_{N_i} (in $\mathcal{T}$), is same as the number of edges in $\widetilde{H}^*_{sc}$ with exactly one end-point in N_i (with the end-points being boundary vertices). As $d_{\mathcal{T}}(v_{N_i}) = 2$, we get that $0 < |B_i| \leq 2$ (see Figure 2.5), i.e., N_i either has a single boundary vertex or two boundary vertices.

Case (1): $|B_i| = 1$. Let $B_i = \{v\}$. Then, as N_i contains only Steiner vertices, if N_i contained any other vertex apart from v (which would be a non-boundary vertex), it would contradict the minimality of H^*_{sc} (Lemma 1), because all the vertices (except v) and edges belonging to N_i could be removed from H^*_{sc}, still keeping the resulting graph $(2, T)$-edge connected.

Case (2): $|B_i| = 2$. Let $B_i = \{u, v\}$. Using Lemma 2 we know that N_i is minimally $(2, \{u, v\})$-edge connected (as N_i contains no terminals). Thus, N_i consists of *exactly* two edge-disjoint paths between u and v. Using Observation 10 we know that for any non-boundary vertex s (of N_i), $d_{N_i}(s) \geq 3$, as N_i contains only Steiner vertices by definition. However, this implies that there exist $x, y \in V(N_i)$ s.t. there are parallel edges between x and y, in N_i. This leads to a contradiction as N_i is a simple graph. □

Figure 2.5: Local view of $v_{N_i} \in L$ being expanded back to node N_i in H^*_{sc}. By Lemma 5, only *case* (1) is possible, where N_i consists of a single vertex. Black vertices are Steiner vertices, and the blue vertices can be terminals or Steiner vertices.



It follows from the definition of $\mathcal{S}$ that every edge $v_{N_i}v_{N_j} \in \mathcal{S}$ corresponds to a path $v_{N_i}, \cdots, v_{N_j}$ in $\mathcal{T}$, s.t. all the internals vertices of the path are from the set L. Recalling our discussion from section 2.1.1, we also know, that every edge $v_{N_x}v_{N_y} \in \mathcal{T}$ corresponds to two (internally) vertex-disjoint u-v paths in H^*, where $u \in V(N_x)$ and $v \in V(N_y)$. Thus, using Lemma 5, it follows that every edge $v_{N_i}v_{N_j} \in \mathcal{S}$ corresponds to two *edge*-disjoint p-q paths in H^*, where $p \in V(N_i)$ and $q \in V(N_j)$.

2.3 Patching things together

Before we finally describe the randomized XP algorithm in Theorem 1, we mention algorithms in Lemma 6 and Lemma 7, which we use as sub-routines.

Lemma 6 (Min-cost flow algorithm;[Kle67]). *Given a graph G and two vertices u and v, one can find two edge-disjoint u-v paths (in G) with the minimum number of edges in poly-time.*

We can assume that the algorithm in Lemma 6 outputs "fail" if G does not have two edge-disjoint u-v paths (this can be checked in poly-time using a *max-flow* algorithm [Din06]).

The objective behind algorithm $\mathcal{A}$ (in Lemma 7) is to extract out the special case, where one is interested in computing a feasible solution to Minimum 2-ECSS, which has fewer edges than any other feasible solution with only degree 2 Steiner vertices.

Definition 9. If Z is a $(2, T')$-edge connected graph (with no isolated vertices) s.t. all the Steiner vertices (vertices apart from T') are of degree 2, then Z is called a *skeleton* $(2, T')$-edge connected graph.

Lemma 7. *There exists an algorithm $\mathcal{A}$, which given as input a simple graph $G(V, E)$ and $T' \subseteq V$ (called terminals), where $g = |T'| > 1$, behaves as follows.*

1. *$\mathcal{A}$ runs in time $g^{\mathcal{O}(g)} n^{\mathcal{O}(1)}$, where $n = |V|$.*
2. *$\mathcal{A}$ always outputs "fail" or a $(2, T')$-edge connected subgraph of G.*
3. *If there exists a skeleton $(2, T')$-edge connected subgraph of G, then with probability at least $1 - \exp(-\operatorname{poly}(n))$, $\mathcal{A}$ outputs a $(2, T')$-edge connected subgraph H s.t. $|E(H)| \leq |E(Z)|$, where Z is a skeleton $(2, T')$-edge connected subgraph with the minimum number of edges.*

Assuming G has a skeleton $(2, T')$-edge connected subgraph, $\mathcal{A}$ essentially, with high probability outputs a $(2, T')$-edge connected subgraph of G, with no more edges than *any skeleton* $(2, T')$-edge connected subgraph (of G). For now, we use algorithm $\mathcal{A}$ as a sub-routine and describe it later in section 2.3.1.

Description of algorithm 1 (for Minimum 2-ECSS). Given the input graph $G(V, E)$ and set of terminals $T \subseteq V$, we begin by guessing the tree structure[2] of $\mathcal{S}$. Let the tree guessed be denoted by S', and the vertices in S' by $v_1, v_2, \cdots, v_s$. For every vertex v_i ($i \in [s]$), we guess a subset of vertices C_i (of G) with the following constraints.

- $T \subseteq \bigcup\limits_{i=1}^{s} C_i$.
- $C_i \cap C_j = \emptyset$ for $i \neq j$, and $C_i \neq \emptyset$ for all $i \in [s]$.
- $|\widetilde{C}| \leq 30k$, where $\widetilde{C} = \bigcup\limits_{i=1}^{s} C_i$ and $k = |T|$.

Lastly, for every edge $e = v_i v_j$ of S', we guess a pair of vertices u^e, w^e s.t. $u^e \in C_i$ and $w^e \in C_j$. We then compute the following:

1. two edge-disjoint u^e-w^e paths P_e and Q_e in G (assuming G has two edge-disjoint u^e-w^e paths), using Lemma 6, for every edge e of S'.
2. X_i, the output of algorithm $\mathcal{A}$ (Lemma 7) for input graph G and terminal set C_i, for every $i \in [s]$.

[2]possibly an isolated vertex.

If for every edge e of S', G has two edge-disjoint u^e-w^e paths, and $\mathcal{A}$ "succeeds" for all $i \in [s]$, i.e. there is **no** $i \in [s]$, for which X_i is "fail", we define H' to be the graph consisting of edges in $\{P_e, Q_e\}_{e \in E(S')}$ and X_i ($i \in [s]$). Finally, among all the possible guesses, we output H' with the least number of edges (or "fail" if H' is undefined for all guesses).

The proof of Theorem 1, follows from Lemma 8 and Lemma 9, which respectively show, that algorithm 1 runs in XP time for the parameter $k = |T|$, and *fails* to output an optimal solution with probability exponentially small in $n = |V(G)|$.

Lemma 8. *Algorithm 1 takes $n^{\mathcal{O}(k)}$ time.*

Proof. Using Observation 14, we know that $|V(\mathcal{S})| \leq 2k+1$. Thus, we can generate all the guesses for the tree structure of $\mathcal{S}$ in time $k^{\mathcal{O}(k)}$ by generating all possible trees on at most $2k+1$ vertices. For each guess S', we can guess $\widetilde{C}$ in time $n^{\mathcal{O}(k)}$ as $|\widetilde{C}| \leq 30k$. Also, given $\widetilde{C}$, we can generate all possible partitions $\{C_i\}_{i \in [s]}$ in time $k^{\mathcal{O}(k)}$. Lastly, for a given partition $\{C_i\}_{i \in [s]}$, we can generate all the possible guesses for $\{(u^e, w^e)\}_{e \in E(S')}$ by using brute force, in time $k^{\mathcal{O}(k)}$, as $|E(S')| \leq 2k$ and $|C_i| \leq 30k$ ($i \in [s]$). Using Lemma 6 (and $|E(S')| \leq 2k$) we know that computing P_e, Q_e for every edge e of S' takes time $n^{\mathcal{O}(1)}$. Finally, it follows from Lemma 7 (and again using $|C_i| \leq 30k$ and $s \leq 2k+1$) that, calling $\mathcal{A}$ with input graph G and terminal set C_i, for all $i \in [s]$, altogether takes time $k^{\mathcal{O}(k)} n^{\mathcal{O}(1)}$. As $k \leq n$, this brings the total running time to $n^{\mathcal{O}(k)}$. □

Observation 15. H' is $(2, T)$-edge connected.

Proof. We prove H' is $(2, \widetilde{C})$-edge connected, which implies H' is $(2, T)$-edge connected as $T \subseteq \widetilde{C}$. Using Theorem 3, it suffices to show that any p-q edge cut in H', for $p, q \in \widetilde{C}$, has at least 2 edges (note $|\widetilde{C}| > 1$, as $|T| > 1$). Consider an arbitrary cut in H' characterized by A s.t. $\emptyset \neq A \cap \widetilde{C} \subsetneq \widetilde{C}$.

Case (1): if there exists $i \in [s]$ s.t. $A \cap C_i \neq \emptyset$ and $\overline{A} \cap C_i \neq \emptyset$. Then, there exist at least 2 edges in the cut $\delta(A)$, as H' contains X_i, which is $(2, C_i)$-edge connected.

Case (2): for every $i \in [s]$, either $C_i \subseteq A_i$ or $C_i \subseteq \overline{A}$. Consider the cut in S' characterized by B s.t. $v_i \in B$, if $C_i \subseteq A$, else $v_i \in \overline{B}$. It follows from $A \cap \widetilde{C} \subsetneq \widetilde{C}$, that $\emptyset \neq B \subsetneq V(S')$. As S' is a tree, there exists an edge $e = v_x v_y$ (of S') in the cut $\delta(B)$, s.t. $v_x \in B$ and $v_y \in \overline{B}$. Consider the edge-disjoint paths P_e and Q_e in H'. By definition these paths are edge-disjoint u^e-w^e paths s.t. $u^e \in C_x$ and $w^e \in C_y$. But $v_x \in B$ implies $C_x \subseteq A$, and $v_y \in \overline{B}$ implies $C_y \subseteq \overline{A}$, and thus there are at least 2 edges in the cut $\delta(A)$. □

Lemma 9. *For sufficiently large n, algorithm 1 outputs an optimal solution, with failure*[3] *probability at most* $\exp(-\operatorname{poly}(n))$.

Proof. Before we prove the lemma, we define a notation that we use later in the proof. For a vertex v_{N_i} of $\mathcal{S}$, we define $\arg(v_{N_i})$ to be i, i.e. the index of the node corresponding to v_{N_i}.

Let v_{N_i} be an arbitrary vertex of $\mathcal{S}$, and N_i the node corresponding to it, in H^*_{sc}. Then, by definition, N_i is 2-edge connected (possibly a single vertex),

[3]where the algorithm outputs either "fail" or a non-optimal feasible solution.

Algorithm 1 Randomized XP algorithm for Minimum 2-ECSS.

```
Precondition: $|T| > 1$.
Precondition: $G$ is $(2, T)$-edge connected.
 1: function MIN2ECSS($G(V, E), T \subseteq V$)
 2:     $H \leftarrow \emptyset$, $OPT \leftarrow \infty$, $k \leftarrow |T|$
 3:     for each guess $S'$ for the tree structure of $\mathcal{S}$ do
 4:         Let the vertices of $S'$ be $v_1, v_2, \cdots, v_s$.
 5:         for each guess $\{C_i\}_{i \in [s]}$, s.t. $T \subseteq \bigcup_{i=1}^{s} C_i \subseteq V$, $C_i \cap C_j = \emptyset$ $(i \neq j)$,
                $C_i \neq \emptyset$ for all $i \in [s]$, and $|\widetilde{C}| \leq 30k$ where $\widetilde{C} = \bigcup_{i=1}^{s} C_i$    do
 6:             for each guess $\{(u^e, w^e)\}_{e \in E(S')}$, s.t. $u^e \in C_i, w^e \in C_j$
                    for $e = v_i v_j$    do
 7:                 for each edge $e \in E(S')$ do
 8:                     if $G$ has two edge-disjoint $u^e$-$w^e$ paths then
 9:                         Let $P_e$ and $Q_e$ be edge-disjoint $u^e$-$w^e$ paths, with the
                            minimum number of edges, computed using Lemma 6
10:                     end if
11:                 end for
12:                 for $i \in [s]$ do
13:                     if $|C_i| = 1$ then
14:                         $X_i \leftarrow C_i$; continue
15:                     end if
16:                     Let $X_i$ be the output of algorithm $\mathcal{A}$ for input graph $G$
                        and terminal set $C_i$
17:                                         ▷ The output $X_i$ may be “fail”
18:                 end for
19:                 if for all $i \in [s]$, $X_i \neq$ “fail” and $G$ has two
                    edge-disjoint $u^e$-$w^e$ paths for every edge $e \in E(S')$ then
20:                     $H' \leftarrow \bigcup_{i=1}^{s} E(X_i) \cup \bigcup_{e \in E(S')} (E(P_e) \cup E(Q_e))$
21:                     if $|E(H')| < OPT$ then
22:                         $OPT \leftarrow |E(H')|$
23:                         $H \leftarrow H'$
24:                     end if
25:                 end if
26:             end for
27:         end for
28:     end for
29:     if $H \neq \emptyset$ then return H
30:     end if
31:     return “fail”
32: end function
```

and thus by using Theorem 3 and Observation 5, H^* has a subgraph Y_i s.t. Y_i is skeleton $(2, V(N_i))$-edge connected. Also, using Lemma 5, Observation 7, Observation 8, and Observation 5, we know, that for every edge $e = v_{N_x}v_{N_y}$ of $\mathcal{S}$, there exist two edge-disjoint *a*-*b* paths A_e, B_e in H^* s.t. $a \in V(N_x)$ and $b \in V(N_y)$. Moreover, using disjointness of nodes, Observation 7, Observation 8, and Observation 5, it follows that the graphs in W are pairwise edge-disjoint, where $W := \{A_e\}_{e\in E(\mathcal{S})} \cup \{B_e\}_{e\in E(\mathcal{S})} \cup \{Y_i\}_{i:v_{N_i}\in V(\mathcal{S})}$.

Consider the case where S' is the correct guess for the tree structure of $\mathcal{S}$ i.e. S' is isomorphic to $\mathcal{S}$, with the isomorphism effectively given by $\phi : [s] \to V(\mathcal{S})$. Using Observation 13, we know that every terminal belongs to some node corresponding to a vertex of $\mathcal{S}$, i.e. $T \subseteq \bigcup_{i:v_{N_i}\in V(\mathcal{S})} V(N_i)$. It follows from the maximality of the nodes that they are (vertex) disjoint, and by Lemma 4, the total size of all the nodes corresponding to vertices of $\mathcal{S}$ is upper bounded by $30k$. Thus, there is a case where for every $i \in [s]$, C_i is the vertex set of the node $N_{\arg(\phi(i))}$. Let e^ϕ denote the edge of $\mathcal{S}$, to which edge $e \in E(S')$ maps to. Given C_i is the correct guess for the vertex set of $N_{\arg(\phi(i))}$ (for $i \in [s]$), there is a case where u^e and w^e are correct guesses for the end-points of A_{e^ϕ}, B_{e^ϕ}, for every edge e of S'. Furthermore, using Lemma 7, Observation 14 and union bound we know that with probability at least $\delta = 1 - (2k+1)\cdot \exp(-\operatorname{poly}(n))$, $\mathcal{A}$ will output a subgraph X_i (of G), for every $i \in [s]$ s.t. $|E(X_i)| \le \left|E\left(Y_{\arg(\phi(i))}\right)\right|$. As $k \le n$, for sufficiently large n, δ is at least $1 - \exp(-\operatorname{poly}(n))$. Thus, with a probability of at least δ, H' is defined for the correct guesses made by the algorithm. Using Observation 15, we know that H' is $(2,T)$-edge connected, and the following implies it is an optimal solution:

$$\begin{aligned}
|E(H')| \quad &\le \sum_{e\in E(S')} \Big(|E(P_e)| + |E(Q_e)|\Big) + \sum_{i\in[s]} |E(X_i)| \\
&\le \sum_{e\in E(\mathcal{S})} \Big(|E(A_e)| + |E(B_e)|\Big) + \sum_{\substack{i:v_{N_i}\in\\ V(\mathcal{S})}} |E(Y_i)| \\
&\le |E(H^*)|
\end{aligned}$$

The second inequality follows from Lemma 6, Lemma 7, and ϕ being an isomorphism. The last inequality follows from graphs in W being pair-wise edge-disjoint. □

2.3.1 Proof of lemma 7

We first introduce two new problems, the Shortest Steiner cycle problem and the Shortest (a,b)-Steiner path problem. We also show in Observation 16 that Shortest (a,b)-Steiner path is reducible to Shortest Steiner cycle in polynomial time.

Problem 4 (Shortest Steiner cycle)**.** Given a simple graph $G(V,E)$ and $T \subseteq V$ (called terminals), find a cycle with the minimum number of edges that contains all the terminals.

Problem 5 (Shortest (a,b)-Steiner path)**.** Given a simple graph $G(V,E)$, $T \subseteq V$ (called terminals) and two distinct terminals $a, b \in T$ (called end-points), find an *a*-*b* path with the minimum number of edges, which contains all the terminals.

Observation 16. Shortest (a, b)-Steiner path is poly-time reducible to Shortest Steiner cycle.

Proof. Given an instance $I_1 := \Big(G, T \subseteq V(G), \{a, b\}\Big)$, of Shortest (a, b)-Steiner path, we create an instance $I_2 := \Big(G', T \cup \{x'\}\Big)$, of Shortest Steiner cycle, where G' is the graph obtained from G by adding a new vertex x', and edges ax' and bx'. It is easy to verify the following: a feasible solution for I_1 with c edges exists, *if and only if* there exists a feasible solution for I_2 with $c + 2$ edges, s.t. given a feasible solution for one instance, we can find the corresponding feasible solution for the other instance in poly-time. $\square$

Theorem 8 gives us a randomized FPT algorithm for Shortest Steiner cycle, parameterized by the number of terminals. As in the reduction outlined in the proof of Observation 16, the number of terminals increases by only one, we also get a randomized FPT algorithm (parameterized by the number of terminals) for Shortest (a, b)-Steiner path, with the same time complexity as in Theorem 8.

Theorem 8 ([BHT12])**.** There exists a $2^p n^{\mathcal{O}(1)}$ time algorithm for Shortest Steiner cycle, such that the algorithm *fails* to output a shortest cycle containing the terminals (given a cycle exists), with probability less than $\exp(-\operatorname{poly}(n))$, where p is the number of terminals and n is the number of vertices in the input graph.

It can be assumed that the algorithm in Theorem 8 always outputs either a cycle containing the terminals or "fail". We now describe algorithm $\mathcal{A}$, using the randomized FPT algorithms mentioned above, for Shortest Steiner cycle and Shortest (a, b)-Steiner path, as sub-routines.

Description of algorithm $\mathcal{A}$. Given the input graph $G(V, E)$ and set of terminals $T' \subseteq V$, we begin by guessing an integer r from $[g]$, where $g = |T'|$. We then guess for every $i \in [r]$ a subset of vertices W_i (of G), with the following constraints.

- $T' = \bigcup\limits_{i=1}^{r} W_i$.
- $W_i \cap W_j = \emptyset$ for $i \neq j$.
- $|W_1| \geq 2$ and $|W_i| \geq 1$ for all $i \in [2\,..\,r]$.

Lastly for every $i \in [2\,..\,r]$ we guess a pair of vertices a_i and b_i, s.t. $a_i, b_i \in \bigcup\limits_{j=1}^{i-1} W_j$. We then compute X_i for every $i \in [r]$, where X_i is defined as follows.

1. if $i = 1$, then X_1 is the output of the Shortest Steiner cycle sub-routine for input graph G and terminal set W_1.
2. if $i \in [2\,..\,r]$ and $a_i = b_i$, then X_i is the output of the Shortest Steiner cycle sub-routine for input graph G and terminal set $W_i \cup \{a_i\}$.
3. otherwise, X_i is the output of the Shortest Steiner (a, b)-path sub-routine for input graph G, terminal set $W_i \cup \{a_i, b_i\}$ and end-points a_i, b_i.

In case the sub-routines "succeed" for all $i \in [r]$, i.e. there is **no** i for which X_i is "fail", we define H' to be the graph consisting of edges in $\bigcup_{i=1}^{r} E(X_i)$. Finally, among all the possible guesses, we output H' which has the least number of edges, or "fail" if H' is undefined for all guesses.

Proof of Lemma 7 follows from Lemma 10, Observation 17, and Lemma 11, which respectively show that algorithm $\mathcal{A}$ satisfies the three conditions outlined in Lemma 7. We begin by proving that $\mathcal{A}$ takes time $g^{\mathcal{O}(g)}n^{\mathcal{O}(1)}$ (where $n = |V(G)|$), i.e., it runs in FPT time for the parameter g (number of terminals).

Lemma 10. *Algorithm $\mathcal{A}$ takes time $g^{\mathcal{O}(g)}n^{\mathcal{O}(1)}$.*

Proof. Trying all the possible values of r takes $\mathcal{O}(g)$ time. For a given value of $r \in [g]$, all the possible partitions $\{W_i\}_{i\in[r]}$ of T', can be generated in time $g^{\mathcal{O}(g)}$, using brute force. For a given partition $\{W_i\}_{i\in[r]}$, $\{(a_i, b_i)\}_{i\in[2..r]}$ can be guessed in time $g^{\mathcal{O}(g)}$, again using brute force. Lastly, there are at most $r \leq g$ sub-routine calls. Using Theorem 8, Observation 16 and the fact that each sub-routine is called with input graph G and a terminal set which is a subset of T', it follows that each sub-routine call takes time $2^g n^{\mathcal{O}(1)}$. Thus, the total running time of the algorithm is $g^{\mathcal{O}(g)}n^{\mathcal{O}(1)}$. □

In the following observation we show that $\mathcal{A}$ satisfies condition 2, as it always outputs either "fail" or a $(2, T')$-edge connected subgraph (of G).

Observation 17. H' is $(2, T')$-edge connected.

Proof. First, we observe that H' contains all the vertices in T', as $T' = \bigcup_{i=1}^{r} W_i$, and thus every vertex in T' lies on some cycle/path in H' by construction. Thus, using Theorem 3, it suffices to prove that H' is 2-edge connected. We give proof by induction. Let $I_i = \bigcup_{j=1}^{i} E(X_j)$ for $i \in [r]$, thus $I_r = H'$, where, for ease of notation we use I_i to represent both the set of edges and the corresponding graph formed by them. I_1 is 2-edge connected as X_1 is a cycle. Assuming I_i is 2-edge connected for $1 < i < r$, we prove the same for $i + 1$. By definition $I_{i+1} = I_i \cup E(X_{i+1})$. Consider an arbitrary edge cut in I_{i+1} characterized by $\emptyset \neq A \subsetneq V(I_{i+1})$.

Case (1): $V(I_i) \cap A \neq \emptyset$ and $V(I_i) \cap \overline{A} \neq \emptyset$. Then there exist at least 2 edges in the cut $\delta(A)$, as I_i is 2-edge connected by the induction hypothesis.

Case (2): $V(I_i) \subseteq A$ or $V(I_i) \subseteq \overline{A}$, w.l.o.g. we assume the former.
As $A \subsetneq V(I_{i+1})$ by assumption, there exists a vertex $w \in \overline{A}$ s.t. $w \in V(X_{i+1})$. But X_{i+1} either forms an a_{i+1}-b_{i+1} path (or a cycle containing a_{i+1}) where a_{i+1} and b_{i+1} are in $V(I_i) \subseteq A$. Thus, the cut $\delta(A)$ contains at least 2 edges. □

Before we prove Lemma 11, we make the following auxiliary observation.

Observation 18. If Z is a skeleton $(2, T')$-edge connected subgraph of G with the minimum number of edges, then Z is minimally $(2, T')$-edge connected.

Proof sketch. If we assume by contradiction that Z is not minimally $(2, T')$-edge connected, then there exists a $(2, T')$-edge connected subgraph with fewer edges than Z. However, one still needs to prove the subtle point that this implies the existence of a *skeleton* $(2, T')$-edge connected subgraph with fewer edges than Z. See Appendix A. □

Algorithm $\mathcal{A}$.

Precondition: $|T'| > 1$.

```
 1: function $\mathcal{A}(G(V,E), T' \subseteq V)$
 2:     $H \leftarrow \emptyset$, $OPT \leftarrow \infty$, $g \leftarrow |T'|$
 3:     for $r \in [g]$ do
 4:         for each guess $\{W_i\}_{i\in[r]}$ s.t. $T' = \bigcup_{i=1}^{r} W_i$, $W_i \cap W_j = \emptyset$ $(i \neq j)$,
                $|W_1| \geq 2$, and $|W_i| \geq 1$ for all $i \in [2..r]$                do
 5:             for each guess $\{(a_i, b_i)\}_{i\in[2..r]}$ s.t. $a_i, b_i \in \bigcup_{j=1}^{i-1} W_j$ do
 6:                 $X_1 \leftarrow$ SteinerCycle$(G, W_1)$
 7:                 for $i \in [2..r]$ do
 8:                     if $a_i = b_i$ then $X_i \leftarrow$ SteinerCycle$(G, W_i \cup \{a_i\})$
 9:                     else
10:                         $X_i \leftarrow$ SteinerPath$(G, W_i \cup \{a_i, b_i\}, a_i, b_i)$
11:                     end if
12:                 end for
13:                 if for all $i \in [r]$, $X_i \neq$ “fail” then
14:                     $H' \leftarrow \bigcup_{i=1}^{r} E(X_i)$
15:                     if $|E(H')| < OPT$ then
16:                         $OPT \leftarrow |E(H')|$
17:                         $H \leftarrow H'$
18:                     end if
19:                 end if
20:             end for
21:         end for
22:     end for
23:     if $H \neq \emptyset$ then
24:         return $H$
25:     end if
26:     return “fail”
27: end function
28:
29: function SteinerCycle$(G,E)$
30:     Let $C$ be the output of algorithm in Theorem 8 for input graph $G$ and
    terminal set $E$
31:     return $C$                                             ▷ $C$ can be “fail”
32: end function
33:
34: function SteinerPath$(G,E,a,b)$                              ▷ $a, b \in E$
35:     Construct $G'$ from $G$ by adding a new vertex $x'$ and edges $ax'$ and $bx'$
36:     $C \leftarrow$ SteinerCycle$(G', E \cup \{x'\})$
37:     if $C \neq$ “fail” then
38:         Let $P$ be the $a$-$b$ path obtained from $C$ by removing $x'$
39:         return $P$
40:     end if
41:     return “fail”
42: end function
```

Finally, we prove that $\mathcal{A}$ satisfies condition 3, i.e., with high probability it outputs a $(2, T^{'})$-edge connected subgraph (of G) with no more edges than any *skeleton* $(2, T')$-edge connected subgraph.

Lemma 11. *For sufficiently large n algorithm $\mathcal{A}$ satisfies condition 3 of Lemma 7.*

Proof. Assume Z is a skeleton $(2, T')$-edge connected subgraph (of G) with the minimum number of edges. Using Observation 18, we know that Z is minimally $(2, T')$-edge connected, and is thus 2-edge connected using Observation 1. Using Theorem 4, we know that there exists an ear decomposition of Z, given by $P_0, P_1, \cdots P_l$, where $l \geq 1$, as we assumed $g = |T'| > 1$. First, we show that P_1, a cycle by definition, contains at least two terminals.

Case (1): $l = 1$. In this case, Z is just a cycle given by P_1, and by our assumption of $g > 1$ contains at least two terminals.

Case (2): $l > 1$. Assume by contradiction that P_1 contains less than 2 terminals. As all the Steiner vertices in Z have degree 2, they do not serve as end-points of any ear P_i (for $i > 1$). Therefore, P_1 contains exactly one terminal u, which serves as the end-points of ear P_2, and possibly other ears. Consider the graph Z' obtained from Z by removing all the edges and vertices in P_1 (except for u). Then, Z' contains all the terminals, has only degree 2 Steiner vertices, and is 2-edge connected by Theorem 4, as it has an ear decomposition given by $u, P_2, \cdots, P_l$. But using Theorem 3, this leads to a contradiction as Z' is a skeleton $(2, T')$-edge connected subgraph of G, and $|E(Z')| < |E(Z)|$.

Similarly, we show that if $l > 1$, then for every $i \in [2\,..\,l]$, P_i has at least one internal vertex (a vertex apart from its end-points), which is a terminal. Assume by contradiction that there exists $j \in [2\,..\,l]$ s.t. P_j has no terminal as an internal vertex. Consider the graph Z' obtained by removing the ear P_j (except its end-points) from Z. As the internal vertices (if any) of P_j do not serve as end-points for any ear, Z' has an ear decomposition given by $P_0, \cdots P_{j-1}, P_{j+1}, \cdots P_l$, and is thus 2-edge connected by Theorem 4. Also, Z' contains all the terminals and has only Steiner vertices with degree 2. However, this leads to a contradiction, as using Theorem 3, Z' is a skeleton $(2, T')$-edge connected subgraph of G, with fewer edges than Z.

As P_1 contains at least two terminals, and each ear P_i (for $i > 1$) contains at least one terminal in its interior, using the definition of an ear decomposition we get that $l \leq g$. Therefore, among all the guesses for r, there is a case where $r = l$. Let T_1 denote the set of terminals in P_1, and T_i for $i \in [2\,..\,l]$, the set of terminals which are internal vertices of P_i. We know that $|T_1| \geq 2$ and $|T_i| \geq 1$ for $i \in [2\,..\,l]$, if $l > 1$. It follows from the definition of an ear decomposition that $T_i \cap T_j = \emptyset$ for $i \neq j$, and $T' = \bigcup_{i \in [l]} T_i$. Thus, for $r = l$ the algorithm will be able to guess $\{W_i\}_{i \in [r]}$ s.t. $W_i = T_i$. As Z has only Steiner vertices with degree 2, the end-points of an ear P_i ($i \in [2\,..\,l]$), are terminals from the set $\bigcup_{j=1}^{i-1} T_j$. Thus, the algorithm will also be able to correctly guess the end-points a_i, b_i of ear P_i, for every $i \in [2\,..\,l]$, given $l > 1$. For these *correct* guesses, using $l \leq g$, Theorem 8, and union bound we get that with probability at least $\delta = 1 - g \cdot \exp(-\operatorname{poly}(n))$, the algorithm computes X_i for every $i \in [r]$, s.t.

- for $i = 1$, X_1 is a shortest cycle through the terminals in T_1.
- for $i \in [2\,..\,r]$, if $a_i = b_i$, X_i is a shortest cycle through the terminals in $T_i \cup \{a_i\}$.

- for $i \in [2 \mathinner{..} r]$, if $a_i \neq b_i$, X_i is a shortest a_i-b_i path containing the terminals in $T_i \cup \{a_i, b_i\}$.

Therefore, with a probability of at least δ, H' is defined. As $g \leq n$, δ is at least $1 - \exp(-\operatorname{poly}(n))$ for sufficiently large n. Using Observation 17, we know that H' is $(2, T')$-edge connected. Finally, H' has no more edges than Z as

$$|E(H')| \ \leq \ \sum_{i=1}^{r} |E(X_i)| \ \leq \ \sum_{i=1}^{l} |E(P_i)| = |E(Z)|$$

The second inequality follows from the fact that P_1 is a cycle containing terminals in T_1, and for $l > 1$, P_i is a cycle containing terminals $T_i \cup \{a_i\}$, if $a_i = b_i$, and otherwise an a_i-b_i path containing terminals in $T_i \cup \{a_i, b_i\}$, for $i \in [2 \mathinner{..} l]$. The equality follows from the definition of an ear decomposition. $\square$

3. Another approach

Let us assume that the number of high-degree (degree at least 3) Steiner vertices in an optimal solution H^*, to Minimum 2-ECSS, can be (linearly) bounded in terms of k, the number of terminals. We could then guess the high-degree Steiner vertices in H^* in time $n^{\mathcal{O}(k)}$, where n is the number of vertices in the input graph. Using Observation 1 and Theorem 4, we know that H^* has an ear decomposition. Thus, we could also guess for each ear the terminals and high-degree Steiner vertices in its interior, along with its end-points (which can only be high-degree vertices). Finally, using Theorem 8 as a sub-routine, we could find a "cheap substitute" for every ear (this was essentially the idea behind algorithm $\mathcal{A}$). This would give us a (randomized) $n^{\mathcal{O}(k)}$ time algorithm for Minimum 2-ECSS. However, the number of high-degree Steiner vertices in an optimal solution to Minimum 2-ECSS need not be bounded in terms of k (see Figure A.1). Thus, it was necessary to first break H^* into "smaller sub-parts" for the algorithm discussed in chapter 2.

Although, it can be proved that the number of high-degree Steiner vertices in an optimal solution to Minimum 2-VCSS can be upper bounded by twice the number of terminals. Recall that in Minimum 2-VCSS, the aim is to find a subgraph (of the input graph) which has the least number of edges and is $(2, T)$-vertex connected i.e., for each pair of terminals $s, t \in T$ there are two (internally) vertex disjoint s-t paths in the subgraph.

Lemma 12 ([BCGI22]). *For any optimal solution J^* to Minimum 2-VCSS, the number of Steiner vertices s, s.t. $d_{J^*}(s) \geq 3$, is bounded from above by $2k$ (where k is the number of terminals).*

We note that analogous to Observation 1, one can show that an optimal solution to Minimum 2-VCSS is 2-vertex connected. Using Theorem 5, we also know that a 2-vertex connected graph has an open ear decomposition. Thus, using the abovementioned idea, one can obtain a (randomized) XP algorithm for Minimum 2-VCSS, as the high-degree Steiner vertices can be guessed in time, using Lemma 12.

Theorem 9 ([BCGI22]). Minimum 2-VCSS has a randomized XP algorithm[1] with a runtime of $n^{\mathcal{O}(k)}$, where k is the number of terminals and n is the number of vertices in the input graph.

3.1 Reduction to Minimum 2-VCSS

Taking into account Theorem 9, we give an alternative proof of Theorem 1 by giving a reduction from Minimum 2-ECSS to Minimum 2-VCSS. Given input $\Big(G(V, E), T \subseteq V\Big)$ to Minimum 2-ECSS, first we create a weighted graph G' with binary weights (see Figure 3.1). We *start with* G and alter it to obtain G' as follows.

[1]as defined in Theorem 1.

1. For every vertex v in G, we expand v to a complete graph K^v on $d_G(v)$ vertices. We call the vertices of K^v copies of v. In a one-to-one fashion, we then make the edges incident to v in G, incident to copies of v. We declare all the vertices of K^v as Steiner vertices and give a weight of 0 to all the edges in K^v.

2. For every terminal t in G, we create a terminal t in G', along with edges $a^t t$ and $b^t t$ where a^t, b^t are two distinct adjacent copies of t in K^t. Recall, we assumed that G is $(2, T)$-edge connected. Therefore, for every terminal t in G, $d_G(t) \geq 2$ and thus K^t has at least two distinct vertices. We give a weight of 0 to the edges $a^t t, b^t t$.

3. We give a weight of 1 to rest of the edges in G'.

We denote by $f(e) = u^e w^e$ the edge in G' that the edge $e = uw$ in G gets mapped to, where u^e, w^e are copies of u and w respectively.

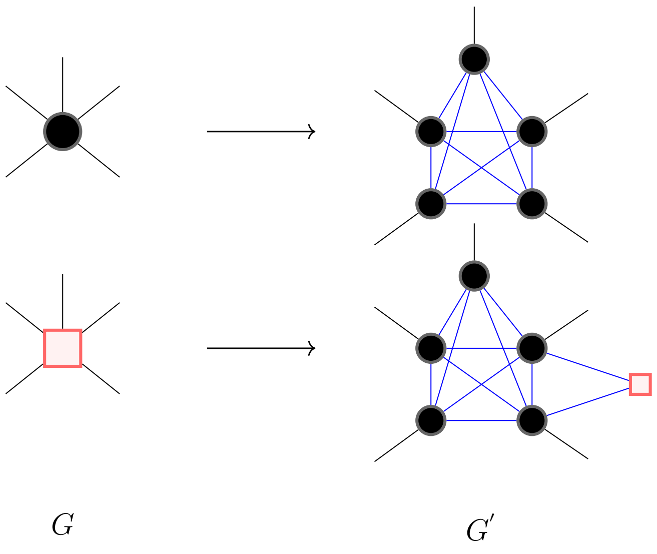


Figure 3.1: Construction of G' from G, where all the blue edges have weight 0, and the rest have a weight 1. Red (square) vertices are terminals, and black vertices are Steiner vertices.

Next, we create the graph G'' by modifying the weights of edges in G' as follows: we change the weight 0 to 1, and weight 1 to M, where $M = |E(G')| + 2 = |E(G'')| + 2$. Finally, we create the graph G''' from G'' as follows: for each edge with weight M in G'', we sub-divide it into M edges each with a weight of 1, s.t. all the new vertices introduced by sub-dividing an edge are Steiner vertices (see Figure 3.2). We denote the extended edge (recall Definition 3) obtained by sub-diving an edge $uv \in E(G'')$, by $\widetilde{uv}_{ex}$.

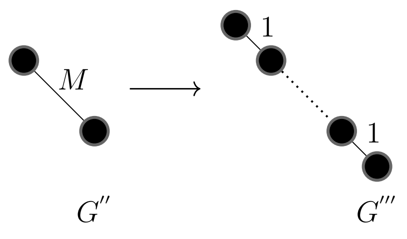


Figure 3.2: Construction of G''' from G'', where the black vertices are Steiner vertices. Note that by construction, an edge of weight M in G'' is always between two Steiner vertices.

As G', G'', and G''' have the same number of terminals as G, we also identify their terminal set by T.

3.1.1 Correctness of the reduction

It is easy to check that G''' can be constructed from G in poly-time. Also, Observation 19, Observation 20, and Observation 21 together show that given an optimal solution to Minimum 2-VCSS for the input instance (G''', T), one can compute in poly-time an optimal solution to Minimum 2-ECSS for the input instance (G, T). Consequently, using Theorem 9 we get a randomized XP algorithm for Minimum 2-ECSS parameterized by the number of terminals k, with a runtime of $n^{\mathcal{O}(k)}$, where $n = |V(G)|$.

We denote the cost (sum of the weights of edges) of a graph H by $cost(H)$.

Observation 19. G has a $(2, T)$-edge connected subgraph H with c edges *if and only if* G' has a $(2, T)$-vertex connected subgraph H' with $cost(H') = c$. Furthermore, given H', one can obtain H in poly-time.

Proof. Let H be a $(2, T)$-edge connected subgraph of G. Then, consider the subgraph H' of G', consisting of edges in

$$\bigcup_{e \in E(H)} \{f(e)\} \;\cup\; \bigcup_{v \in V(H)} E(K^v) \;\cup\; \bigcup_{t \in T} \{a^t t, b^t t\}$$

First, we note $cost(H') = |E(H)|$, as the edges in the complete graph K^v have weight 0, and so do the edges $a^t t, b^t t$. We prove H' is $(2, T)$-vertex connected. Fix two distinct arbitrary terminals p, t. As H is $(2, T)$-edge connected it has two edge-disjoint p-t paths P_1, P_2. We consider two p-t paths Q_1, Q_2 in H', where we obtain Q_1 from P_1 (and similarly Q_2 from P_2). Let P_1 be given by $e_1, e_2, \cdots, e_l$ s.t. $e_i \in E(H)$, $e_i = u_i u_{i+1}$ and $u_1 = p, u_{l+1} = t$; and similarly let P_2 be given by $h_1, h_2, \cdots, h_y$ s.t. $h_i = w_i w_{i+1}$. Then, consider Q_1 and Q_2 respectively, given by

$$Q_1^{start}, f(e_1), \cdots\cdots, u_i^{e_{i-1}} u_i^{e_i}, f(e_i), u_{i+1}^{e_i} u_{i+1}^{e_{i+1}}, \cdots\cdots, f(e_l), Q_1^{end}$$

and

$$Q_2^{start}, f(h_1), \cdots\cdots, w_i^{h_{i-1}} w_i^{h_i}, f(h_i), w_{i+1}^{h_i} w_{i+1}^{h_{i+1}}, \cdots\cdots, f(h_y), Q_2^{end}$$

Here, Q_1^{start} and Q_2^{start} are (internally) vertex-disjoint p^{e_1}-p and p^{h_1}-p paths[2] respectively, consisting of only edges from $E(K^p) \cup \{a^p p, b^p p\}$. Similarly Q_1^{end} and Q_2^{end} are (internally) vertex-disjoint t^{e_l}-t and t^{h_y}-t paths, consisting of only edges from $E(K^t) \cup \{a^t t, b^t t\}$. It is not hard to see that Q_1, Q_2 are (internally) vertex-disjoint p-t paths; with the idea being that if P_1, P_2 intersect at some internal vertex u, then the edges in Q_1, Q_2 "corresponding" to the ones incident to u in P_1, P_2, are incident to different copies of u (where the copies induce a complete graph).

For the other direction, let H' be a $(2,T)$-vertex connected subgraph of G' such that $cost(H') = c$. Then, consider the graph H obtained from H' in the following manner: for a Steiner vertex $s \in G$, we contract all copies of s (in H') to a single vertex and label it s (removing the edges between contracted vertices), and similarly, for a terminal t we contract all its copies into t. It is not hard to check that H is a subgraph of G (as, in a nutshell, we just reversed the steps taken to create G' from G), which contains all the terminals in T. Also, $|E(H)| = c$, as while contracting we removed exactly the edges with weight 0 in H'. Now assume by contradiction that H is not $(2,T)$-edge connected. Then, by Theorem 3 there exists a p-t edge cut in H, characterized by $\emptyset \neq A \subsetneq V(H)$, where p, t are distinct terminals (note we assume $|T| > 1$), such that $p \in A, t \in \overline{A}$ and there are less than 2 edges in the cut $\delta(A)$. Consider the p-t cut in H' characterized by B, where if vertex $v \in A$ $(\overline{A})$, then we put all the vertices that were contracted in H' to obtain v, in B $(\overline{B})$. This leads to a contradiction as the cut $\delta(B)$ has less than 2 edges, but H' is $(2,T)$-vertex connected and hence $(2,T)$-edge connected. □

Thus, it follows that given a $(2,T)$-vertex connected subgraph H' of G', with the least cost, one can find an optimal solution to Minimum 2-ECSS for the input instance (G,T) in poly-time.

Observation 20. Given a $(2,T)$-vertex connected subgraph H'' of G'' with the least cost, one can obtain a $(2,T)$-vertex connected subgraph H' of G' with the least cost in poly-time.

Proof. Let H'' contain a edges with weight 1 and b edges with weight M. Consider the subgraph H' of G' containing the same[3] edges as H''. Then clearly H' is $(2,T)$-vertex connected and $cost(H') = b$. Assume by contradiction that there exists a subgraph H of G' which is $(2,T)$-vertex connected and $cost(H) = c < b$. Let H contain d edges with weight 0. Consider the subgraph $\widetilde{H}$ of G'' with the same edges as H. Then, $\widetilde{H}$ is $(2,T)$-vertex connected. But this leads to a contradiction as

$$\begin{aligned} cost(H'') &= a + b \cdot M \\ &= a + M + (b-1) \cdot M \\ &\geq a + M + c \cdot M \\ &> d + c \cdot M \\ &= cost(\widetilde{H}) \end{aligned}$$

[2]recall, p^{e_1} and p^{h_1} are copies of p, to which the edges $f(e_1)$ and $f(h_1)$ are respectively incident.

[3]recall G' and G'' have the same edge set and differ only in the weights given to the edges.

The first inequality follows from c and b being integers, and the last follows from the fact that $d \leq |E(G')| < M$ and $a \geq 0$. $\square$

Observation 21. Given a $(2,T)$-vertex connected subgraph H''' of G''' with the least cost, one can obtain a $(2,T)$-vertex connected subgraph H'' of G'' with the least cost in poly-time.

Proof. First, we observe that for any extended edge $\widetilde{uv}_{ex}$ in G''', H''' either contains $\widetilde{uv}_{ex}$ or it does not contain *any* edge from $\widetilde{uv}_{ex}$ (recall $\widetilde{uv}_{ex}$ is the extended edge obtained by sub-diving the edge uv of G''). Otherwise, the edges (and internal vertices) of $\widetilde{uv}_{ex}$ in H''' could be removed, still keeping the graph $(2,T)$-vertex connected, as all the internal vertices of $\widetilde{uv}_{ex}$ are Steiner vertices of degree 2. However, this would contradict the optimality of H'''. Now, consider the graph H'' obtained by replacing back every extended edge $\widetilde{uv}_{ex}$ in H''', by edge uv of weight M. Trivially, $cost(H'') = cost(H''')$. We argue that H'' is $(2,T)$-vertex connected. Consider two arbitrary terminals p and t. As H''' is $(2,T)$-vertex connected, it contains two (internally) vertex-disjoint p-t paths P_1 and P_2. Again using the fact that for any arbitrary extended edge $\widetilde{uv}_{ex}$ (in H'''), all its internal vertices are degree 2 Steiner vertices, it follows that either P_1, P_2 do not contain any edge from $\widetilde{uv}_{ex}$ or $\widetilde{uv}_{ex}$ belongs to exactly one of the paths from P_1 and P_2. Thus, the paths obtained by replacing every extended edge $\widetilde{uv}_{ex}$ in P_1, P_2 by edge uv, are (internally) vertex-disjoint p-t paths in H''. Lastly, it is not hard to check that if there existed a $(2,T)$-vertex connected subgraph of G'' with cost strictly less than $cost(H'')$, one could replace each edge uv of weight M in such a subgraph by $\widetilde{uv}_{ex}$, obtaining a $(2,T)$-vertex connected subgraph of G''' with cost strictly less than $cost(H''')$, leading to a contradiction. $\square$

Lastly, we note that finding a $(2,T)$-vertex connected subgraph of G''' with the least cost is equivalent to finding an optimal solution to the Minimum 2-VCSS problem for input graph G''' and terminal set T (as all the edges of G''' have unit weight).

4. Multiple copies of edges being allowed

In this chapter, we prove Theorem 2, improving over the FPT algorithm for BI-SCSS by Chitnis et al. [CFM17] in terms of the runtime.

Theorem 10 ([CFM17])**.** There is a $2^{\mathcal{O}(k^2)}n^{\mathcal{O}(1)}$ time FPT algorithm (with parameter k) for BI-SCSS, where k is the number of terminals and n is the number of vertices in the input graph.

First, we prove that BI-SCSS and weighted Minimum 2-ECSM are polynomially equivalent.

Observation 22. Weighted Minimum 2-ECSM and BI-SCSS are polynomially reducible to each other.

Proof. Let $\left(\widetilde{G}, T \subseteq V(G)\right)$ be the input to weighted Minimum 2-ECSM, and (G, T) the input to BI-SCSS, where G is the directed graph obtained from $\widetilde{G}$, by having directed edges $\overrightarrow{uv}$ and $\overrightarrow{vu}$ for every edge uv of $\widetilde{G}$, s.t. $\overrightarrow{uv}$ and $\overrightarrow{vu}$ have the same weight as uv.

Let $\widetilde{H}$ be an optimal solution (without any isolated vertices)[1] to weighted Minimum 2-ECSM, with cost c. Then, $\widetilde{H}$ is minimally $(2, T)$-edge connected, and using Observation 3 we know that there is no edge (of $\widetilde{G}$) in $\widetilde{H}$ with more than two copies. Also, it follows from Observation 1, Theorem 4, and Theorem 6 that we can find an ear decomposition of $\widetilde{H}$ in poly-time. Consider the directed graph H obtained from $\widetilde{H}$ by orienting all the edges of an ear in one direction[2]. Then, it is not hard to check that H is strongly connected, contains all the terminals, and is a solution to the BI-SCSS instance with cost c.

For the other direction, let H be an optimal solution to BI-SCSS with cost c. Then consider the undirected graph (possibly multi-graph) $\widetilde{H}$ obtained from H by removing edge directions. Trivially, $\widetilde{H}$ has cost c, contains all the terminals in T, and is connected. We show that $\widetilde{H}$ is 2-edge connected, and hence $(2, T)$-edge connected using Theorem 3. If any edge uv is removed from $\widetilde{H}$, then there still exists a u-v path q_{uv} in $\widetilde{H}\backslash\{uv\}$, and thus $\widetilde{H}\backslash\{uv\}$ is connected, because: if the corresponding directed edge $\overrightarrow{uv}$ (or $\overrightarrow{vu}$) is removed from H, there still exists a directed path from v to u (or u to v) as H is strongly connected, and q_{uv} can be taken to be the corresponding undirected path in $\widetilde{H}$. Thus, $\widetilde{H}$ is a solution of cost c to weighted Minimum 2-ECSM for the input instance $(\widetilde{G}, T)$. ☐

We note that both the instances $(\widetilde{G}, T)$ and (G, T) in Observation 22 have the same number of terminals. Thus, using Theorem 10 and Observation 22, we get an FPT algorithm (parameterized by the number of terminals) for weighted Minimum 2-ECSM, with the same runtime as in Theorem 10.

We now piece together previous results to show that weighted Minimum 2-ECSM admits an FPT algorithm with a runtime of $2^{\mathcal{O}(k \log k)}n^{\mathcal{O}(1)}$, where k is the

[1]recall, we are always interested in the case $|T| > 1$.

[2]we note that this idea of orienting all the edges of an ear in one direction is a known way of proving Robbins' theorem [Rob39b], which states that the graphs that have strong orientations are exactly the 2-edge connected graphs.

number of terminals and n is the number of vertices in the input graph. This, along with the reduction outlined in Observation 22 will complete the proof of Theorem 2. We start by making the following observation, which helps us take the metric closure of the input graph without loss of generality.

Observation 23. For weighted Minimum 2-ECSM, one can take the metric closure of the input graph without loss of generality[3].

Proof. Let $\big(G, T \subseteq V(G)\big)$ be the original input instance, and $\widetilde{G}$ the metric closure of G. Consider a solution H, of cost c, for the input instance (G, T). Then, $\widetilde{H}$ obtained by considering the same edges as H, but weights according to $\widetilde{G}$ is a solution for the input $(\widetilde{G}, T)$, of cost at most c.

Now, consider $\widetilde{H}$, an arbitrary solution of cost c, for input $(\widetilde{G}, T)$. Consider the graph H obtained in the following manner: for each edge uv in $\widetilde{H}$, we replace it with a shortest u-v path in G, s.t. every time we use an edge e of G, we use a new copy of e. Trivially, H is $(2, T)$-edge connected (as $\widetilde{H}$ is), has cost c, and is a solution for the original input instance. $\square$

The following theorem lets us upper-bound the size (number of edges) of an optimal solution to weighted Minimum 2-ECSM, in terms of the number of terminals, given the input graph defines a metric.

Theorem 11 ([Jor03])**.** Suppose the input graph G is a complete graph with edge weights according to a metric. Then, there exists an optimal solution to weighted Minimum 2-ECSM s.t. the number of edges in the optimal solution is upper bounded by $4k$, where k is the number of terminals in G.

Thus, using Observation 23 and Theorem 11, we can assume w.l.o.g. that there always exists an optimal solution to weighted Minimum 2-ECSM, whose size is upper bounded by $4k$.

Lastly, we use that weighted Minimum 2-ECSM admits an FPT algorithm for the parameter 'solution size' (number of edges in the solution).

Theorem 12 ([FMvL21])**.** There exists a $2^{\mathcal{O}(l \log l)} n^{\mathcal{O}(1)}$ FPT algorithm for EC-SNDP, where l is the number of edges in the solution, and n is the number of vertices in the input graph.

Using Theorem 12 along with the bound of $4k$ on the size of an optimal solution, yields an FPT algorithm for weighted Minimum 2-ECSM with a runtime of $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$. What is left to argue is that weighted Minimum 2-ECSM is a special case of EC-SNDP (recall Problem 3). Let O be an *arbitrary* optimal solution (possibly with parallel edges) to weighted Minimum 2-ECSM for input graph G and terminal set T. It follows from the optimality of O that it is minimally $(2, T)$-edge connected. Using Observation 3, O cannot have more than two copies of any edge of G. Thus, the weighted Minimum 2-ECSM problem can be seen as a special case of EC-SNDP, where the input graph is $\widetilde{G}$ and all the demands $d_{s,t}$ for $s, t \in T$ are set to two. Here, $\widetilde{G}$ is a multi-graph that contains exactly two copies of every edge in G.

[3]the proof can easily be generalized to show that the observation holds for weighted Minimum k-ECSM, for any $k \geq 1$.

Conclusion

In chapter 2 and chapter 3, we described randomized XP algorithms with parameter k (number of terminals) for finding an optimal solution to the Minimum 2-ECSS problem, s.t. the algorithms failed with a small probability. This still leaves an open question, can the approximation factor of 2 for Minimum 2-ECSS (due to [Jai01]) be improved by a poly-time or an FPT algorithm?

Open question 1. *Does there exist a poly-time or an* FPT *(with* ***parameter k****) algorithm for (weighted) Minimum* 2-ECSS *with approximation ratio better than* 2*?*

A comparatively less studied parameter we did not explore in this work is p, the number of Steiner vertices in the solution (where parameterization by p is as defined by Dvořák et al. [DFK+21]), for which we have the following open question analogous to 1:

Open question 2. *Does there exist an* FPT *(or even* XP*) algorithm with* ***parameter p****, for (weighted) Minimum* 2-ECSS*, with approximation ratio better than* 2*?*

In chapter 4, we discussed a $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ FPT algorithm for weighted Minimum 2-ECSM. It would be interesting to see if there exists an FPT algorithm (parameter k) with a better runtime.

Open question 3. *Does there exist a* $2^{\mathcal{O}(k)} n^{\mathcal{O}(1)}$ FPT *algorithm for weighted Minimum* 2-ECSM*?*

Bibliography

[B2́2] Miklós Bóna. *A Walk Through Combinatorics.* World Scientific Publishing, 2022.

[BCGI22] Ishan Bansal, Joe Cheriyan, Logan Grout, and Sharat Ibrahimpur. Algorithms for 2-connected network design and flexible Steiner trees with a constant number of terminals, 2022. arXiv:2206.11807.

[BCGZ05] André Berger, Artur Czumaj, Michelangelo Grigni, and Hairong Zhao. Approximation Schemes for Minimum 2-Connected Spanning Subgraphs in Weighted Planar Graphs. In *Algorithms - ESA 2005*, pages 472–483. Springer Berlin Heidelberg, 2005.

[BG07] André Berger and Michelangelo Grigni. Minimum Weight 2-Edge Connected Spanning Subgraphs in Planar Graphs. In *Automata, Languages and Programming*, pages 90–101. Springer Berlin Heidelberg, 2007.

[BGRS13] Jarosław Byrka, Fabrizio Grandoni, Thomas Rothvoss, and Laura Sanità. Steiner Tree Approximation via Iterative Randomized Rounding. *Journal of the ACM*, 60(1):1–33, 2013.

[BHKK07] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets möbius: fast subset convolution. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing - STOC '07*. ACM Press, 2007.

[BHT12] Andreas Björklund, Thore Husfeldt, and Nina Taslaman. Shortest Cycle Through Specified Elements. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1747–1753. Society for Industrial and Applied Mathematics, 2012.

[BK16] Glencora Borradaile and Philip Klein. The Two-Edge Connectivity Survivable-Network Design Problem in Planar Graphs. *ACM Transactions on Algorithms*, 12(3):1–29, 2016.

[Bor26] Otakar Borůvka. O jistém problému minimálním. *Práce Mor. Přírodověd. Spol. V Brně III*, pages 37–58, 1926.

[CC02] Miroslav Chlebík and Janka Chlebíková. Approximation Hardness of the Steiner Tree Problem on Graphs. In *Algorithm Theory - SWAT 2002*, pages 170–179. Springer Berlin Heidelberg, 2002.

[Ç19] Ali Çivril. A New Approximation Algorithm for the Minimum 2-Edge-Connected Spanning Subgraph Problem, 2019. arXiv:1911.07232.

[CFK$^+$15] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

[CFM17] Rajesh Chitnis, Andreas Emil Feldmann, and Pasin Manurangsi. Parameterized Approximation Algorithms for Bidirected Steiner Network Problems, 2017. arXiv:1707.06499.

[CL99] Artur Czumaj and Andrzej Lingas. On Approximability of the Minimum-Cost k-Connected Spanning Subgraph Problem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '99, page 281–290, USA, 1999. Society for Industrial and Applied Mathematics.

[CSS01] J. Cheriyan, A. Sebo, and Z. Szigeti. Improving on the 1.5-Approximation of a Smallest 2-Edge Connected Spanning Subgraph. *SIAM Journal on Discrete Mathematics*, 14(2):170–180, 2001.

[Deo04] Narsingh Deo. *Graph theory with applications to engineering and computer science*. PHI Learning, 2004.

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer London, 2013.

[DFK+21] Pavel Dvoř ák, Andreas E. Feldmann, Dušan Knop, Tomáš Masařík, Tomáš Toufar, and Pavel Veselý. Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices. *SIAM Journal on Discrete Mathematics*, 35(1):546–574, 2021.

[Din06] Yefim Dinitz. Dinitz' Algorithm: The Original Version and Even's Version. In *Lecture Notes in Computer Science*, pages 218–240. Springer Berlin Heidelberg, 2006.

[DW71] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[FJ82] Greg N. Frederickson and Joseph Jájá. On the relationship between the biconnectivity augmentation and travelling salesman problems. *Theoretical Computer Science*, 19(2):189–201, 1982.

[FKLM20] Andreas Emil Feldmann, CS Karthik, Euiwoong Lee, and Pasin Manurangsi. A survey on approximation in parameterized complexity: Hardness and algorithms. *Algorithms*, 13(6):146, 2020.

[FKM+07] B. Fuchs, W. Kern, D. Molle, S. Richter, P. Rossmanith, and X. Wang. Dynamic Programming for Minimum Steiner Trees. *Theory of Computing Systems*, 41(3):493–500, 2007.

[FMvL21] Andreas Emil Feldmann, Anish Mukherjee, and Erik Jan van Leeuwen. The Parameterized Complexity of the Survivable Network Design Problem, 2021. arXiv:2111.02295.

[GGTW08] Harold N. Gabow, Michel X. Goemans, Éva Tardos, and David P. Williamson. Approximating the smallest k-edge connected spanning subgraph by LP-rounding. *Networks*, 53(4):345–357, 2008.

[GJ78] M. R. Garey and David S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, 1978.

[HHL00] D. Frank Hsu, Xiao-Dong Hu, and Guo-Hui Lin. On Minimum-Weight k-Edge Connected Steiner Networks on Metric Spaces. *Graphs Comb.*, 16(3):275–284, 2000.

[HV03] Olivier Hudry and Vijay V. Vazirani. Approximation algorithms. *Mathématiques et sciences humaines*, (161), 2003.

[HVV19] 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.

[Jai01] Kamal Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica*, 21(1):39–60, 2001.

[Jor03] Tibor Jordán. On minimally k-edge-connected graphs and shortest k-edge-connected Steiner networks. *Discrete Applied Mathematics*, 131(2):421–432, 2003.

[Kar72] Richard M. Karp. Reducibility Among Combinatorial Problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[KK01] Piotr Krysta and V. S. Anil Kumar. Approximation Algorithms for Minimum Size 2-Connectivity Problems. In *STACS 2001*, pages 431–442. Springer Berlin Heidelberg, 2001.

[KKGZ21] Anna R. Karlin, Nathan Klein, Shayan Oveis Gharan, and Xinzhi Zhang. An Improved Approximation Algorithm for the Minimum k-Edge Connected Multi-Subgraph Problem, 2021. arXiv:2101.05921.

[Kle67] Morton Klein. A Primal Method for Minimal Cost Flows with Applications to the Assignment and Transportation Problems. *Management Science*, 14(3):205–220, 1967.

[Kle08] Philip N. Klein. A Linear-Time Approximation Scheme for TSP in Undirected Planar Graphs with Edge-Weights. *SIAM Journal on Computing*, 37(6):1926–1952, 2008.

[KM05] Hervé Kerivin and A. Ridha Mahjoub. Design of Survivable Networks: A survey. In *In Networks*, pages 1–21, 2005.

[KV94] Samir Khuller and Uzi Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214–235, 1994.

[LPRS17] Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 224–237. ACM, 2017.

[Men27] Karl Menger. Zur allgemeinen Kurventheorie. *Fundamenta Mathematicae*, 10(1):96–115, 1927.

[MP00] A.R. Mahjoub and P. Pesneau. On Steiner 2-edge connected polytopes. *Electronic Notes in Discrete Mathematics*, 5:210–213, 2000.

[Nar16] Vishnu V. Narayan. A 17/12-Approximation Algorithm for 2-Vertex-Connected Spanning Subgraphs on Graphs with Minimum Degree At Least 3, 2016. arXiv:1612.04790.

[RE06] M. Resende and P. Pardalos (Eds.). *Handbook of Optimization in Telecommunications.* Springer, 2006.

[Rob39a] H. E. Robbins. A Theorem on Graphs, with an Application to a Problem of Traffic Control. *The American Mathematical Monthly*, 46(5):281, 1939.

[Rob39b] H. E. Robbins. A Theorem on Graphs, with an Application to a Problem of Traffic Control. *The American Mathematical Monthly*, 46(5):281–283, 1939.

[Sch12] Jens M. Schmidt. A Simple Test on 2-vertex and 2-edge Connectivity, 2012. arXiv:1209.0700.

[SV14] 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. *Combinatorica*, 34:597–629, 2014.

[Whi32] Hassler Whitney. Non-separable and planar graphs. *Transactions of the American Mathematical Society*, 34(2):339–362, 1932.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, 2011.

List of Figures

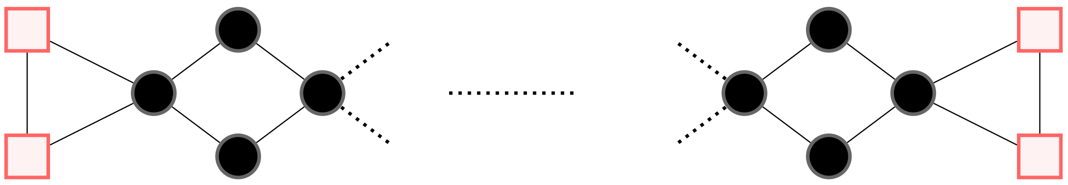
A. Missing proofs and figures

Proof sketch for Observation 1. Let us assume by contradiction that $G(V, E)$ is not 2-edge connected. Then, there exists an edge cut characterized by $\emptyset \neq A \subsetneq V$, such that there is exactly 1 edge in the cut $\delta(A)$; note G is connected (follows from minimality of G). Also, both A and $\overline{A}$ have at least one terminal[1]. Otherwise, w.l.o.g. say A contains no terminals, then A (along with the cut edge) could be removed, still leaving the graph $(2, T)$-edge connected, which contradicts the minimality of G. Thus there exist $x, y \in T$ s.t. $x \in A$ and $y \in \overline{A}$. As G is $(2, T)$-edge connected, using Theorem 3 there should be at least 2 edges in the cut $\delta(A)$, which leads to a contradiction. $\square$

Proof of Observation 2. If G is a multi-graph and contains 2 copies of an edge uv, then the observation trivially holds for both copies of uv. Let us assume G has a single copy of edge uv. As G is 2-edge connected, using Theorem 3 there exist (at least) two edge-disjoint paths between u, v. Let these paths be p_1 and p_2. If none of these paths contain edge uv, then uv and p_2 together form a cycle. Otherwise, w.l.o.g. let p_1 contain uv; as we are looking at simple u-v paths it implies $p_1 = uv$. Thus, again uv and p_2 together form a cycle (as p_1, p_2 are edge-disjoint). $\square$

Proof sketch for Observation 18. Z by definition is $(2, T')$-edge connected; assume by contradiction it is not minimal. Then there exists an edge e in Z s.t. $Z' := Z \backslash e$ is $(2, T')$-edge connected[2]. Let C be the connected component of Z' containing the terminals. Let Z'' be the graph obtained from C by repeatedly removing any degree 1 Steiner vertices. Then it is not hard to check that Z'' is a skeleton $(2, T')$-edge connected subgraph of G. But this leads to a contradiction as $|E(Z'')| < |E(Z)|$. $\square$

Figure A.1: An example showing that in a minimally $(2, T)$-edge connected graph, the number of Steiner vertices with a degree of at least 3 need not be bounded in terms of the number of terminals. Red (square) vertices are terminals, and black vertices are Steiner vertices.



[1]recall, we assume throughout that $|T| > 1$.
[2]note Z has no isolated vertices by Definition 9.