# Listing Induced Steiner Subgraphs as a Compact Way to Discover Steiner Trees in Graphs

**Alessio Conte**
Dipartimento di Informatica, Università di Pisa, Italy
conte@di.unipi.it

**Roberto Grossi**
Dipartimento di Informatica, Università di Pisa, Italy
grossi@di.unipi.it

**Mamadou Moustapha Kanté**
Université Clermont Auvergne, LIMOS, CNRS, Aubière, France
mamadou.kante@uca.fr

**Andrea Marino**
Dipartimento di Statistica, Informatica, Applicazioni, Università di Firenze, Italy
andrea.marino@unifi.it

**Takeaki Uno**
National Institute of Informatics, Tokyo, Japan
uno@nii.ac.jp

**Kunihiro Wasa**
National Institute of Informatics, Tokyo, Japan
wasa@nii.ac.jp

## Abstract

This paper investigates induced Steiner subgraphs as a variant of the classical Steiner trees, so as to compactly represent the (exponentially many) Steiner trees sharing the same underlying induced subgraph. We prove that the enumeration of all (inclusion-minimal) induced Steiner subgraphs is harder than the well-known Hypergraph Transversal enumeration problem if the number of terminals is not fixed. When the number of terminals is fixed, we propose a polynomial delay algorithm for listing all induced Steiner subgraphs of minimum size. We also propose a polynomial delay algorithm for listing the set of minimal induced Steiner subgraphs when the number of terminals is 3.

**2012 ACM Subject Classification** Mathematics of computing → Graph enumeration

**Keywords and phrases** Graph algorithms, enumeration, listing and counting, Steiner trees, induced subgraphs

**Digital Object Identifier** 10.4230/LIPIcs.MFCS.2019.73

**Funding** This work was partially supported by JST CREST grant number JPMJCR1401 and JPMJCR18K3 Japan, by JSPS KAKENHI Grant Number JP19K20350 Japan, and by MIUR, Italy. Mamadou M. Kanté is supported by French Agency for Research under the GraphEN project (ANR-15-CE40-0009).

## 1 Introduction

Paths and cycles are the basic components for connecting nodes in undirected graphs. It is natural to ask, for a given subset of nodes, how to connect them in a component of minimal cost, where the cost is, e.g., the total sum of the weights on the edges or the number of nodes.

Classical instances of this problem follow two main patterns [7]. Some are *edge*-centric components in (usually) weighted graphs, such as shortest paths/cycles, spanning trees, Steiner trees, $k$-edge-connected components. Others are *node*-centric components in unweighted graphs, such as induced paths, chordless cycles (holes), $k$-node-connected components.

The above is not actually a dichotomy of edge- vs. node-centric components as, for example, node-weighted Steiner trees lay at the cross of the two. Nevertheless, it seems that most of the literature has been devoted to edge-centric components.

For instance, in the rich research area on Steiner trees, the online compendium in [12] reports over 60 variations of the Steiner tree problem, but just a couple are node-centric (survivable network design problem [3] and node weighted generalized Steiner tree [11]). The interest in the Steiner tree problem and its variants is well rooted in real world applications. Some examples can be seen in the 11th DIMACS challenge [8], which aimed at identifying a benchmark for algorithms and data generators, with emphasis on network optimization, computer-aided design, phylogenetic tree reconstruction and other applications. It is also of great interest in implementation challenges of the parameterized community, *e.g.*, [16], thus we believe it is worth investigating its natural extensions in node-centric scenarios.

**Induced Steiner subgraphs.** Special node-centric components have been introduced by Telle and Villanger [17] as a generalization of the induced paths.[1] In this paper, we study these components, hereafter renamed *induced Steiner subgraphs* as they seem to us a natural extension to the scenario of Steiner trees. Given an undirected graph $G = (V(G), E(G))$ and a subset $W \subseteq V(G)$ of $t = |W|$ nodes, called *terminals*, we want a set of nodes, minimal under inclusion, that connects all nodes in $W$.

▶ **Definition 1** (Induced Steiner Subgraph). *Given $W \subseteq V(G)$, an Induced Steiner Subgraph of $G$ is a set of nodes $S \subseteq V(G) \setminus W$ such that the induced subgraph $G[S \cup W]$ is connected. We say that $S$ is a (inclusion wise)* Minimal *Induced Steiner Subgraph (*MISS*) if there is no $S' \subset S$ such that $G[S' \cup W]$ is connected. The nodes in $S$ are called Steiner points.*[2]

An example of a MISS is given in Figure 1, where we illustrate also its connection with the classical notion of Steiner tree. Similarly to how Steiner trees generalize paths between two nodes, it can be noted that MISS's generalize induced paths, which correspond to MISS's for $|W| = 2$. On the other extreme, for $W = V(G)$, Steiner trees correspond to spanning trees of the graph, while there would be a single empty MISS corresponding to the whole graph.

Definition 1 makes sense in domains where nodes are preferred to edges in components, for example in network design [11] and graph database query optimization [2, 14]. In these cases, rather than a minimal set of edges that ensures connection (i.e. a tree), a minimal set of connected nodes and their induced subgraph (i.e. a MISS) are sought for. Note that the notion of induced Steiner subgraph can be easily extended to graphs with weights on the nodes, and MISS's correspond to solutions of minimal cost (otherwise their cost could be reduced by removing nodes). Our approach focuses on unweighted graphs, where Steiner trees also have many applications [12].

**Results.** In this paper we consider the problem of *output-sensitive* enumeration of MISS's, which corresponds to bounding the cost to a polynomial of the input and output size.

- In Section 3 we show that the problem is harder than the well-known TRANS-ENUM problem [10] for an arbitrary number of terminals. To classify its output-sensitive

[1] In [17], these are called MINIMAL T-CONNECTING SETS and are motivated by the study of the 2-DISJOINT CONNECTED SUBGRAPHS problem which is the following: given a connected graph $G = (V, E)$ and two disjoint subsets of terminal vertices $Z_1, Z_2 \subseteq V$, decide whether there exists a partition $A_1, A_2$ of $V(G)$, with $Z_1 \subseteq A_1$, $Z_2 \subseteq A_2$ and $G[A_1], G[A_2]$ both connected.

[2] We abuse the terminology a bit and refer by $S$ also to the induced subgraph $G[S \cup W]$, as we enumerate all feasible $S$'s while keeping the same set $W$. This simplifies notation in the rest of the paper.

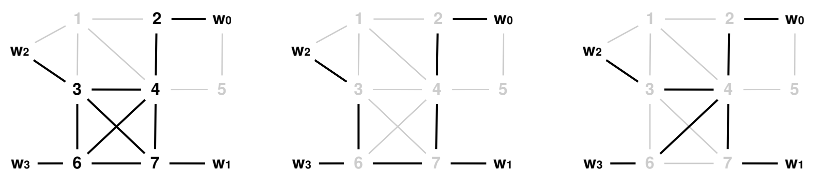

**Figure 1** A MISS (left) and two Steiner trees corresponding to the same MISS (center and right).

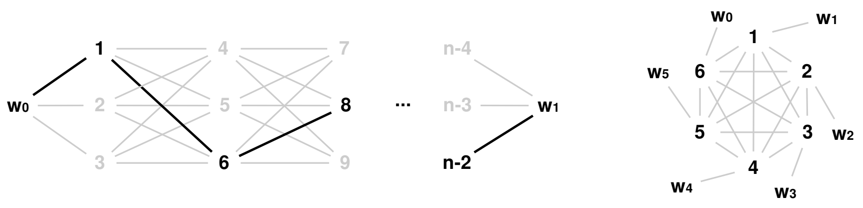

**Figure 2** Left: A graph with $\Omega(3^{n/3})$ minimum Steiner trees and MISS's. Right: A graph with 1 MISS and $n^{n-2}$ minimum Steiner trees (by Cayley's formula).

enumeration complexity, we consider the problem when the number of terminals $t = |W|$ is fixed. Concerning this, we show that the EXTENSION PROBLEM (see Section 3.2) is NP-complete already for $t \geq 2$, meaning that the sub-problems of any backtracking algorithm [10] should be carefully designed to avoid incurring instances of NP-complete problems.

- In this scenario, we propose a polynomial delay algorithm in Section 4 when the number of terminals is $t = 3$. (Note that the delay is the time required to list the "next" solution.) This is the first non-trivial case, as for $t = 1$ the problem is not defined, and for $t = 2$ it is the well-known problem of listing induced paths.
- Furthermore, we investigate in Section 2 the relationships of MISS's and Steiner trees, proving properties which allow us to use the algorithm of Dourado et al. [9] as a subroutine in order to list just the MISS's of *minimum* size with output-sensitive guarantees, for a bounded number $t$ of terminals.

**Discussion.** Given a MISS $S$ for $W$, we observe that each spanning tree in the induced subgraph $G[S \cup W]$ is a Steiner tree for $W$. Also the vice versa holds, so this could be useful for the packing edge-disjoint Steiner tree problem, which received attention in recent years [1, 4, 13] and consists of finding a largest possible set of Steiner trees that do not share edges or nodes.

If we look for an *individual* min-size Steiner subgraph for $W$ (i.e., with the least number of nodes), it can be obtained as the subgraph induced by the Steiner points of a min-size Steiner tree for $W$. A different situation occurs instead if we look for *all* min-size induced Steiner subgraphs for $W$. A number of deep results for Steiner trees could be employed: for example, [9] proposes an algorithm for listing them with linear delay, assuming that the number $t = |W|$ of terminals is $O(1)$; [15] introduces a general enumeration framework for matroids with certain properties, which can be applied to generalized Steiner trees, containing as special cases minimal Steiner trees and minimal Steiner forests. Unfortunately, this approach gives rise over and over to the same min-size Steiner subgraph in our case, as there can be exponentially many Steiner trees for the same set of Steiner points (Fig. 2(left)).

Here comes an interesting feature of a MISS, as it is a compact representation of many Steiner trees. Observing that a min-size Steiner subgraph for $W$ is a MISS for $W$ of minimum size, we can thus list the MISS's for $W$ twice: the first time to discover the minimum size, and the second time to single out the MISS's of that size (i.e. min-size induced Steiner subgraphs).

This approach can be dramatically more efficient than listing min-size Steiner trees to catch min-size induced Steiner subgraphs (see Fig. 2 (right)). On the other hand, the results in this paper (see Section 2.2) show the latter to be a viable approach if $W$ has bounded size.

As for weighted graphs, we observe that the min-edge-weight Steiner tree problem reduces to min-node-weight Steiner subgraph problem.[3] Once we can list MISS's efficiently, we can also find min-node-weight induced Steiner subgraphs and thus min-edge-weight Steiner trees.

In summary, finding MISS's subsumes finding min-node-weight, min-edge-weight, and min-size Steiner trees, and becomes a sort of "universal" problem for this family of connectivity problems on graphs that compactly represent these trees. This motivates the problem of *listing* MISS's efficiently, which is central to this paper.

Previous results on MISS enumeration were already obtained in [17] from an *input-sensitive* point of view:[4] the authors proved that the number of MISS's for $t$ terminals, in any graph of $n$ nodes, is at most $\binom{n-t}{t-2} \cdot 3^{\frac{n-t}{3}}$, and propose an *input-sensitive* algorithm running in $O^*(\binom{n-t}{t-2} \cdot 3^{\frac{n-t}{3}})$ total time, where the $O^*$ notation ignores factors polynomial in $n$. Here we consider the opposite direction, i.e., output-sensitive enumeration.

**Preliminaries.** We refer to [7] for our graph terminology. Let $G = (V(G), E(G))$ be an undirected graph, where $V(G)$ is its node set and $E(G)$ its edge set. $N(v)$ is the set of neighbors of a node $v$. When $G$ is clear from the context we will refer to node and edge sets simply as $V$ and $E$. For a set of nodes $A \subseteq V(G)$, let $E[A] \subseteq E(G)$ be the set of edges whose endpoints are both in $A$, and denote by $G[A] = (A, E[A])$ the graph thus obtained, also called the subgraph *induced* by $A$. A subgraph is a forest if it is acyclic, and a tree if it is a connected forest. A node $v \in I$ is a *leaf* if $v$ has degree 1 in $G[I]$. Removing a leaf from a tree yields a tree. Regarding the Steiner Tree/Subgraph problem, we will refer to the set of terminal nodes as $W$, and to its size $|W|$ as $t$. We recall that an induced Steiner subgraph is a set of nodes $S$ such that $G[S \cup W]$ is connected, and is minimal if inclusion-minimal. For a MISS $S$, we call *junctions* the nodes in $S \cup W$ which, in $G[S \cup W]$, have either degree $\geq 3$, or are terminals with degree $\geq 2$.

The notions for induced Steiner subgraphs and MISS given in Definition 1 can be applied to any subset of the terminals in $W$: for example, $S$ is a MISS for $X \subseteq W$ if $G[S \cup X]$ is connected and so on. Without loss of generality, we can assume that no two terminals in $W$ are connected by an edge: otherwise we can contract that edge into a single node and eliminate the resulting duplicated edges, preserving a one-to-one correspondence between solutions of the old graph and the new. In the following, given a set of nodes $S$, let $W(S)$ be the set of terminals in $W$ that have neighbors in $S$. Note that $S$ is an induced Steiner subgraph for $X = W(S)$ as long as $G[S \cup W(S)]$ is connected. Let us make the following observation before continuing.

▶ **Fact 1.** *Let $S$ be a* MISS *for a set $W$ of terminals. Then, $G[S \cup W]$ is not 2-connected. Moreover, $G[S \cup W]$ cannot contain a leaf vertex $v \in S$.*

**Proof.** Because no two terminals are adjacent, if $G[S \cup W]$ is 2-connected, then $|S| \geq 2$. By definition of 2-connectedness, for any vertex $v \in S$, we have that $G[(S \setminus \{v\}) \cup W]$ is connected, contradicting the minimality of $S$.

If $G[S \cup W]$ contains a leaf vertex $v \in S$, then $G[(S \setminus \{v\}) \cup W]$ is still connected as no path can contain $v$ as an internal vertex, contradicting the minimality of $S$. ◀

[3] It suffices to replace each each edge $\{x, y\}$ with weight $w$ by two edges $\{x, z\}$ and $\{z, y\}$ where $z$ is a new node of weight $w$ and $x$ and $y$ have weight zero [11]

[4] In exact exponential-time algorithms, a input-sensitive cost measures the running time in the input length.

## 2 Properties of Steiner Subgraphs and Listing Minimum Steiner Subgraphs

We are interested in this section in the enumeration of MISS's of minimum size, and refer to them as *minimum* MISS*'s*. In this section we prove some useful properties of MISS's in general, and some special properties of minimum MISS's, to outline the relationship between minimum Steiner trees and MISS's (Section 2.1) and to get an output-sensitive algorithm for minimum MISS's (Section 2.2).

### 2.1 Relationship with minimum Steiner trees

We first prove that, for a given set $W$ of terminals, one minimum MISS corresponds to many Steiner trees (Lemma 2). We then prove an upper bound (in terms of $t$) for the number of junctions and the degree of the nodes involved in a MISS. We highlight here that both upper bounds hold for any MISS, minimum or not. Aside of telling that MISS's have strong topological properties, these bounds allow us to obtain the converse of Lemma 2, that is how to turn MISS's into Steiner trees (Lemma 5).

**One-to-many correspondence to minimum Steiner trees.** A minimum Steiner tree $T$ for the set $W$ of terminals is clearly a spanning tree of the subgraph $G[M \cup W]$ induced by the set $M = V(T) \setminus W$ of nodes plus the terminals, and thus $M$ is a minimum MISS for $W$. On the other hand, consider any minimum MISS $M$ for $W$: each spanning tree $T$ of $G[M \cup W]$ is actually a minimum Steiner tree for $W$. For example, the two trees in Figure 1 correspond to the same MISS.

▶ **Lemma 2.** *Given a set $W$ of terminals and a minimum* MISS *$M$ for $W$, any spanning tree $T$ of the induced subgraph $G[M \cup W]$ is a minimum Steiner tree for $W$.*

**Proof.** All the terminals in $W$ are connected through $T$ by definition of spanning tree, so $T$ is a Steiner tree, not necessarily of minimum size (and the terminals are not necessarily all in its leaves). Suppose by contradiction that there exists another Steiner tree $T'$ of size smaller than $T$ (where $T'$ can use nodes or edges outside $G[M \cup W]$). Then the subgraph $G[M' \cup W]$ induced by the set $M = V(T') \setminus W$ of nodes of $T'$ would be a MISS with fewer nodes than $G[M \cup W]$, contradicting the fact that the latter is a minimum MISS. ◀

**Upper bound on the number of junctions.** Consider a MISS $S$ (not necessarily minimum), and recall that a *junction* is either a node of $S$ with degree at least 3 in the induced subgraph $G[S \cup W]$, or a terminal in $W$ with degree at least 2 in $G[S \cup W]$. Although $S$ is minimal, it can contain arbitrarily long induced paths independently of the number $t$ of terminals. Still, we can bound the number of junctions in terms of $t$.

▶ **Lemma 3.** *Given a set $W$ of terminals, any* MISS *$S$ contains at most $3t$ junctions.*

**Proof.** By induction on the number of terminals. Recalling that $t = |W|$, if $t \leq 3$ then the claim is trivially true (see also Fact 2). Assume then that $t \geq 4$.

Consider the induced subgraph $G[S \cup W]$ ignoring the rest of $G$. In order to prove our claim, we replace each junction terminal $v'$ with a new node $v$, and link $v'$ and $v$ together with an edge, so that $v'$ becomes a leaf terminal and $v$ is a junction. Now, as no junction can have degree 2, we replace each induced path $x_1, x_2, \ldots, x_\ell$ by an edge $\{x_1, x_\ell\}$ when $x_1$ and $x_\ell$ have degree $\neq 2$.

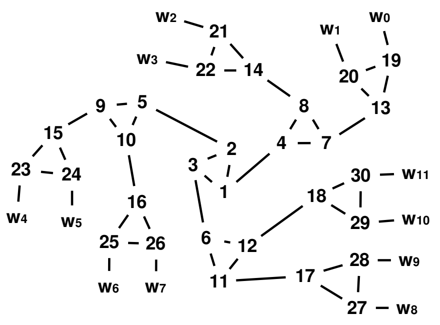

**Figure 3** A MISS with 12 terminals and 30 junctions. Suitably enlarging the structure sets the junctions/terminals ratio arbitrarily close to 3.

Let $G'$ be the resulting graph. Note that $G'$ and $G[S \cup W]$ have the same number of terminals and junctions by construction. As a result, it suffices to prove our claim using $G'$. Note also that each node $v$ in $G'$ has degree 1 if and only if it is a terminal, and at least 3 otherwise. If $G'$ is a tree, then the claim holds as the number of internal nodes of degree 3 in a tree is upper bounded by the number of leaves.

Let us assume that $G'$ is not a tree. By Fact 1 we know that $G[S \cup W]$ is not 2-connected and by construction $G'$ is also not 2-connected. Let $T$ be the tree of bi-connected components of $G'$ (called blocks). Because all non-terminal nodes in $G'$ are junctions, each leaf block $C$ of $T$ is an edge $\{v, w\}$ with $v$ a junction node and $w$ a terminal because all the terminal nodes are non-adjacent and a leaf block has at most one non-terminal node. Let $C$ be a block all of whose neighbors, except possibly 1, are leaf blocks. Such a block always exists, e.g., we could take a non-leaf block of maximum depth after rooting the tree arbitrarily. If except $C$, all the other blocks are leaf blocks (meaning that $T$ is a star), then we are done because $C$ is 2-connected, meaning any node removal does not disconnect it, meaning in turn any node must be connected to a leaf-terminal in order for it not to be redundant, so $C$ contains at most $t$ nodes. Otherwise, $T$ is not a star and then for all possible choices for $C$ let us take the one with biggest size. Let $w_1, \ldots, w_p$ be the terminals adjacent to junction nodes in $C$. Clearly, by the choice of $C$, $p \geq 2$ because either $C$ has size $\geq 3$, and at least two of the junction nodes are adjacent to terminals, or $C$ has size 2 and at least one of the junction nodes is adjacent to two terminals. Let $v$ be the junction node of $C$ belonging to another block that is not a leaf block ($v$ exists by the choice of $C$). Notice that $C$ has size at most $p+1$ because except possibly $v$, all the junction nodes in $C$ are adjacent to at least one terminal. Let $G'' = G' \setminus (\{w_1, \ldots, w_p\} \cup (C \setminus \{v\}))$ and consider $v$ as a terminal in $G''$. It is not hard to check that $G''$ is a MISS for $(W \setminus \{w_1, \ldots, w_p\}) \cup \{v\}$. By induction $G''$ has at most $3(t-p+1)$ junctions because $G''$ has $t-p+1$ terminals. And then $G$ has at most $3(t-p+1)+(p+1)$ junctions, which is bounded by $3t$. ◀

Furthermore, Figure 3 shows Lemma 3 to be tight (up to constant additive factors).

**Upper bound on the node degree.** Although any MISS $S$ can be arbitrarily large independently of the number $t$ of terminals, we prove that any node cannot have degree larger than $t$ in $G[S \cup W]$.

▶ **Lemma 4.** *Given a set $W$ of terminals and any MISS $S$ for $W$ and any node $v \in S \cup W$, $|N(v) \cap (S \cup W)| \leq t$.*

**Proof.** Let $G'$ be the induced subgraph $G[S \cup W]$. Let us start with the case $v \in S$. For each $w \in W$, consider one shortest path between $v$ and $w$ in $G'$, and let $S'$ be the set of nodes containing all the nodes on these $t$ shortest paths, excluding the nodes in $W$. Note that $S' \subseteq S$ by definition. As they are all shortest paths (hence induced paths with no shortcut edges), $v$ can only be adjacent to the next node in each such path, and no other: thus $|N(v) \cap (S' \cup W)| \leq t$. Since all terminals are connected to each other using these paths, clearly $S'$ is a Steiner induced subgraph. Since $S$ is minimal, the only option is $S' = S$. The case $v \in W$ follows along the same lines. ◀

**Removing edges to get a minimum Steiner trees.** Consider any MISS $S$ and select a subset of edges in the induced subgraph $G[S \cup W]$ to obtain a Steiner tree $T$. Lemma 3 and 4 imply the result that any $T$ can be obtained from $S$ by removing at most $3t^2$ edges from $G[S \cup W]$. Note that the number of edges in $G[S \cup W]$ could be much larger, but most of them are incident to degree-2 nodes of $G[S \cup W]$, and should be taken to form $T$ (otherwise some terminals in $W$ would be disconnected from the others). This property holds for the special case of minimum MISS, and hence minimum Steiner tree, immediately giving the following result.

▶ **Lemma 5.** *Given a set $W$ of terminals and any* MISS *$S$ for $W$, every Steiner tree $T$ contained in $G[S \cup W]$ can be obtained by removing at most $3t^2$ edges.*

## 2.2 Listing minimum Steiner subgraphs

The properties of Steiner subgraphs seen so far allow us to design an algorithm for listing minimum Steiner subgraphs, parameterized in $t$. Consider a minimum MISS $M$, and the subgraph $G_M = G[M \cup W]$. By Lemma 2 each spanning tree of $G_M$ is a minimum Steiner tree. By Lemma 5, $G_M$ is turned into a spanning tree by removing $\alpha$ edges, for some $\alpha \leq 3t^2$. A closer look gives a suggestion on how to do that: even if the number of nodes and edges in $G_M$ can be much larger than $3t^2$, there are at most $3t$ junctions by Lemma 3, with degree greater than 2; the rest of the (possibly many) nodes have all degree 2, and thus their incident edges must be taken to keep $G_M$ connected. Hence, the $\alpha$ edges are incident to these junctions, each of degree at most $t$ in $G_M$ by Lemma 4, thus giving at most $3t^2$ edges to select from. As there are at most $2^{3t^2}$ possible choices for the $\alpha$ edges, there are $O(2^{3t^2})$ minimum Steiner trees yielding the same minimum Steiner subgraph $M$.

We exploit this observation by using [9] to list minimum Steiner trees of $G$ with $O(n^2m + n^{t-2})$ preprocessing time and space, and $O(n)$ delay. To avoid duplications, as the same minimum MISS $M$ can be obtained $O(2^{3t^2})$ times with distinct minimum Steiner trees, we simply define a "canonical spanning tree" $T^*$ of $M$ as an arbitrary spanning tree obtained from $G_M$ with some deterministic procedure: when the algorithm in [9] yields $T$, we output its corresponding minimum MISS only if $T = T^*$, and discard it otherwise (note that this computation takes an additional $O(m)$ time per solution).

▶ **Theorem 6.** *Minimum Steiner subgraphs of $G$ for a terminal set $W$ can be listed with $O(n^2m + n^{t-2})$ preprocessing time and space, in $O(m \cdot 2^{3t^2})$ time per solution.*

## 3 Negative Results on Minimal Steiner Subgraphs

As for the classical Steiner tree problem, finding just one MISS is easy, but finding the smallest MISS, i.e., the one having minimum number of nodes, is NP-hard: as we saw in Section 2 any spanning tree of a MISS with $k$ nodes is a Steiner tree with exactly $k - 1$ edges. This means that there is a one-to-many correspondence between MISS and Steiner trees of smallest size, and that finding a smallest MISS solves the unweighted Steiner tree problem.

Concerning listing MISS's, in Section 3.1 we show that this problem is actually harder than listing minimal hypergraph transversals. In particular, this result implies that an output-polynomial algorithm for MISS's enumeration would imply an output-polynomial algorithm for minimal hypergraph transversals, which is a long-standing open problem (see the survey [10] for more information on the problem).

In Section 3.2 we investigate the reason behind the difficulty of listing MISS's, showing that classical techniques for listing algorithms which make use of the so-called *extension problem* are difficult to apply, as the extension problem for MISS's is actually NP-complete. In particular, we prove that deciding whether a partial MISS can be completed into a MISS is NP-complete. Surprisingly, our reduction shows that this result holds even if the number of terminals, i.e. $t$, is bounded. This means that a general output-polynomial algorithm for MISS's enumeration which works for any number of terminals and which is polynomial both in the size of the graph and the number of terminals should rely on different techniques.

## 3.1 Listing reduction from Hypergraph Transversal Enumeration

We show that the problem of listing MISS's is at least as hard as a well-known problem, known as HYPERGRAPH TRANSVERSAL ENUMERATION (in short TRANS-ENUM). The latter problem deals with a hypergraph $\mathcal{H} = (X, H)$, where $H \subseteq 2^X$ is the set of hyperedges. A transversal for $\mathcal{H}$ is a hitting set $S \subseteq X$ for the hyperedges in $H$, namely, $S \cap e \neq \emptyset$ for each hyperedge $e \in H$. A transversal $S$ is (inclusion-wise) minimal if no other $S' \subset S$ is a transversal. The TRANS-ENUM problem asks to list all the minimal transversals for $\mathcal{H}$, and the existence of an output-polynomial algorithm for it is a long-standing open question in the area of enumeration algorithms. Indeed, the latter would provide also the solutions for many enumeration problems arising in several diverse areas such as data-mining, integer programming, biology, databases, game theory, learning theory, and so on (see [10]).

In the following we show how to transform an instance of TRANS-ENUM, i.e. an hypergraph $\mathcal{H}$, to an instance of MISS's enumeration, i.e. a graph $G = (V, E)$ and a set $W \subseteq V$ of terminals, so for each minimal transversal of $\mathcal{H}$ there is a MISS for $W$ in $G$, and vice versa.

▶ **Lemma 7.** *There is a one-to-one reduction from* TRANS-ENUM *to* MISS*'s enumeration (even in split graphs).*

**Proof.** We refer to the bipartite formulation of TRANS-ENUM, that is we consider the bipartite graph $B$ whose vertices are partitioned into $X$ and $H$, and whose edges represent the membership of each vertex of $X$ in the hyperedges of $H$ in which it is involved, and we look at the minimal subsets of $X$ hitting all the vertices of $H$. We transform $B$ into an instance of MISS's enumeration, i.e. a graph $G$ and a set $W$, in the following way: choose $G$ as a copy of $B$, add to $G$ edges between all pairs of vertices of $X$, i.e., forming a clique, and choose as $W$ the vertices in $H$. $G$ is by construction a split graph. A minimal hitting set $S \subseteq X$ hits all the vertices in $H$ and is such that there is no $S' \subset S$ doing the same. As $G[S \cup W]$ is connected, while $G[S' \cup W]$ is not, $S$ corresponds to a MISS. Conversely, consider a MISS $S$ in $G$. Thus $G[S \cup W]$ is connected and each $S' \subset S$ is such that $G[S' \cup W]$ is not. This means that for each vertex in $W$, i.e., a vertex in $H$, one of its neighbor is in $S$, i.e., an element hitting the vertex. Moreover, as $S$ is a clique, each subset $S'$ is still connected, meaning that $G[S' \cup W]$ disconnects some terminal in $W$, that is $S'$ does not hit some vertex in $W$. Hence, $S$ corresponds to a minimal hitting set. ◀

### 3.2 Hardness of the extension problem for miss's

Providing an output-polynomial algorithm for MISS's seems to be a challenging problem, even for a small number of terminals. As the Extension problem is a core block of many classical output-sensitive algorithms such as those based on *binary partition* or *backtracking*, we argue whether we can follow this path: roughly speaking, this is the problem of deciding whether a partial solution, e.g., a subset of nodes, can be completed into a final solution. In this section we show the hardness of the Extension problem for MISS's.

▶ **Problem 1** (Extension problem for MISS's)**.** *Given a graph $G$, a terminal set $W$, and $P \subseteq V(G) \setminus W$, is there a minimal induced Steiner subgraph (*MISS*) $S$ such that $P \subseteq S$?*

Note that, if $P = \emptyset$, the answer is trivially always yes, as we can find a MISS by simply setting $S = V(G) \setminus W$, and removing nodes from it until it is possible to do so without disconnecting $W$. For $t = 1$ the problem is not well defined (or could be considered trivially true if and only if $P = \emptyset$). One may think that the hardness would be influenced by the size of $W$, or that of $P$, however, this is not the case.

For $t = 2$ and $|P| = 1$, the smallest non-trivial sizes, the problem is already hard: let $W = \{w_0, w_1\}$ and $P = \{p\}$; as MISS's for $t = 2$ are induced paths, we obtain the problem of finding an induced path between $w_0$ and $w_1$ passing through $p$. Finding such an induced path has been proved to be NP-complete in [6]. This immediately yields the following result.

▶ **Theorem 8.** *The extension problem for* MISS*'s is NP-complete, even for two terminals.*

The difficulty of the extension problem excludes the possibility of applying some well known general techniques, motivating the interest in this combinatorial structure and in ad-hoc techniques for its enumeration.

## 4 Listing Minimal Steiner Subgraphs with Three Terminals

The problem of listing MISS's seems to be a challenging one as we have shown that the extension problem, which is a core block of many output-sensitive listing algorithms based on the backtracking and binary partition techniques, is NP-complete for MISS's already when the number of terminals is 2 (see Section 3.2). Indeed, in the case of $t = 2$, if the partial solution is just an arbitrary node which is not a terminal, deciding whether there is an induced path involving that node is NP-complete [6]. However, the problem can be solved with polynomial delay [18]: since we are listing induced paths from a source to a target, this limitation can be overcome by suitably defining the partial solution; in this case, a partial solution always includes a terminal, i.e. the source. The same strategy, i.e. forcing each partial solution to include a terminal, clearly does not apply when $t > 2$, as extending that solution will lead again to the hard problem of deciding whether there is an induced path involving that terminal. We show in this section that a workaround is still possible for the special case $t = 3$.

We will characterize some special instances of the extension problem, extending what we will call a *partial Steiner subgraph* (PSS for short), showing that we can indeed find a solution in polynomial time, and that it is possible to obtain a complete binary partition algorithm by just relying on these restricted instances.

To characterize a partial Steiner subgraph, let us define a removable node:

▶ **Definition 9.** *Let $S$ be a Steiner subgraph for $W(S)$, the set of terminals in $W$ that have neighbors in $S$. Then $v \in S$ is a* removable node *for $S$ if $G[S \cup W(S) \setminus \{v\}]$ is connected.*

Note that if $v \in S$ is a removable node then $W(S) = W(S \setminus \{v\})$ (i.e., $S \setminus \{v\}$ is still a Steiner subgraph for $W(S)$).

As an example, consider Figure 1 (left), with $W = \{w_0, w_1, w_2, w_3\}$: $S = \{2, 3, 4, 6, 7\}$ has no removable node, as $W(S) = W$ and each removal of a node of $S$ would disconnect at least one of the terminals from the others. On the other hand, $S' = \{3, 4, 6, 7\}$ has 4 as removable node, since $W(S) = \{w_1, w_2, w_3\}$.

We can now define a PSS:

▶ **Definition 10** (Partial Steiner Subgraph). *$S \neq \emptyset$ is a partial Steiner subgraph (PSS) if*

- *$W(S) \neq \emptyset$,*
- *$S$ is a Steiner subgraph for $W(S)$,*
- *there is* at most *1 removable node in $S$.*

Note that a MISS is also a partial Steiner subgraph, as it has no removable nodes. We can now detail the listing procedure in Algorithm 1:

**Algorithm 1** List all MISS's of $G$ with $t = 3$.

```
Input  : A graph $G = (V(G), E(G))$ and a terminal set $W \subseteq V(G)$ with $t = |W| = 3$
Output : All MISS's of $G$ and $W$
1  Call 3-ENUM$(G, W, \emptyset)$
2  Function 3-ENUM$(G, W, S)$
3      if $W(S) = W$ then
4          output $S$    // $S \neq \emptyset$
5      else
6          $X \leftarrow \emptyset$
7          foreach $v \in V(G) \setminus (S \cup W)$ : $S \cup \{v\}$ is a PSS do
8              if $\exists S' \supseteq (S \cup \{v\})$ : $S'$ is a MISS in $G$ then
9                  3-ENUM$(G \setminus X, W, S \cup \{v\})$
10             $X \leftarrow X \cup \{v\}$
```

In essence, every recursive node of Algorithm 1 considers a partial Steiner subgraph $S$: at each step, it identifies all nodes $v$ for which $S \cup \{v\}$ is also a partial Steiner subgraph, and check whether it is possible to eventually extend it into a solution, if so, it recurs adding $v$ to $S$; afterwards, $v$ is removed from the graph in subsequent recursive calls to avoid duplication (this is modeled by the set $X$).

To prove that the algorithm is correct, we will use the following fact and lemma.

▶ **Fact 2.** *Let $S$ be a MISS for $W$ with $t = |W| = 3$. Then $G[S \cup W]$ is either (1) a subdivision of the claw, (2) a triangle with* 3 *pending paths or (3) an induced path.*

**Proof.** Assume that $G[S \cup W]$ is neither an induced path nor a subdivision of the claw and let $P$ be an induced path in $G[S \cup W]$ from two terminals $w_a$ and $w_b$. Let $P'$ be the shortest path in $G[S \cup W]$ from $w_c$ to $P$, and let $v$ be the penultimate vertex of $P'$ not in $P$. Because $G[P \cup P' \cup W]$ is connected, we can conclude that $S = P \cup P'$, and no vertex of $P'$, but $v$, is adjacent to a vertex in $P$. If $v$ is adjacent to two non-adjacent vertices $x$ and $x'$ of $P'$, then any vertex in $P$ between $x$ and $x'$ can be removed without disconnecting the rest, contradicting the minimality of $S$. Therefore, $S$ is as stated. ◀

▶ **Lemma 11.** *Given a graph $G$ and a set of terminals $W \subseteq V(G)$ with $t = |W| = 3$, let $S$ be a* PSS *and $S'$ a* MISS *such that $S \subset S'$. Then there exists $v \in S' \setminus S$ such that $S \cup \{v\}$ is a* PSS.

**Proof.** Since $S \cup W(S)$ and $S' \cup W$ are connected subgraphs, and recalling from the preliminaries that no two terminals are adjacent, there must exist at least a node $x \in S' \setminus S$ such that $S \cup \{x\} \cup W(S)$ is connected. Let $P$ be the set of all such nodes, and let us consider the possible cases. To complete the proof, we need to show that for some node $v \in P$ we have that $S \cup \{v\}$ has at most one removable node, meaning it meets all conditions of a PSS. We separately prove 3 comprehensive cases:

- **S has no removable node**. Then it must be that $W(S) = 2$ and $S$ is exactly an induced path between two terminals $w_a$ and $w_b$.[5] Consider a shortest path from the third terminal $w_c$ to $S \cup \{w_a, w_b\}$ in $G[S' \cup W]$, and let $v$ be the last vertex of the path not belonging to $S \cup \{w_a, w_b\}$. It clearly belongs to $P$. Consider now $S \cup \{v\}$: either $v$ is connected to $w_c$, so $S \cup \{v\}$ is a MISS, i.e., $S' = S \cup \{v\}$, or $v$ is removable and $S' \neq S \cup \{v\}$. If $v$ is removable in $S \cup \{v\}$ and makes another vertex $x$ removable in $S \cup \{v\}$, then $x$ is in a path between two non-adjacent neighbors of $v$ in $S \cup \{w_a, w_b\}$. But then $S' \setminus \{x\}$ would be a Steiner subgraph of $W$, a contradiction. Thus $S \cup \{v\}$ is a PSS.
- $|\mathbf{P}| = \mathbf{1}$. Then all other nodes of $S'$ may reach $S$ only via the single node $v \in P$. This means that if any node of $S \cup \{v\}$ other than $v$ is removable, i.e., does not disconnect $v$ from the rest of $S$ and $W(S)$, then it is also removable in $S'$, contradicting the minimality of $S'$. Thus the only possible removable node in $S \cup \{v\}$ is $v$ and $S \cup \{v\}$ is a PSS.
- $|\mathbf{P}| \geq \mathbf{2}$ **and S has a removable node** $b$. We first note that $|W(S)| = 1$. Indeed, if $|W(S)| = 2$, then in any case of Fact 2, a connected subset $S$ connecting two terminals cannot be extended into a connected subset with more than one vertex from $S' \setminus S$. Therefore, we can conclude that $S$ is a path from a vertex $b$ to a terminal $w$, with $b$ the removable node of $S$. Because $|W(S)| = 1$ and $b \in S'$, then again by Fact 2 $b$ should have a neighbor $v$ not adjacent to any vertex in $S \setminus \{b\}$. Therefore, $S \cup \{v\}$ is a path included in $S'$, and is then a PSS. ◀

Furthermore, let us prove that the extension problem for PSSs is solvable in polynomial time when $t = 3$.

▶ **Lemma 12.** *Given a graph $G$, terminal set $W \subseteq V(G)$ with $t = |W| = 3$ and a* PSS *$S$, the question "is there a* MISS *$S'$ of $G$ such that $S \subseteq S'$?" can be answered in $O(mn)$ time.*

**Proof.** Let us consider the three possible cases based on $W(S)$:

- $|\mathbf{W(S)}| = \mathbf{1}$. Let $W(S) = \{w_a\}$; as $S \neq \emptyset$, $S$ consists of a path from $w_a$ to the only removable node $b$ of $S$. Let $R$ be the set of nodes not in $S$ or $W$ that are neighbors of either $w_a$ or some node in $S \setminus \{b\}$, i.e., $R = \{v \in V(G) \setminus (S \cup W) : N(v) \cap (W(S) \cup S \setminus \{b\}) \neq \emptyset\}$. Note that connecting terminals via nodes of $R$ makes $b$ redundant, as such nodes can reach $w_a$ without using $b$; thus we must connect at least one terminal to $w_a$ without using nodes of $R$, i.e., we may use at most one node of $R$. Consider $G \setminus R$: note that $b$ is the only node in $S \cup \{w_a\}$ with neighbors outside the solution. How many terminals distinct from $w_a$ are in the same connected component of $G \setminus R$ as $b$? If *zero*, we can answer **no**, as we will need to connect the remaining two terminals via $R$, making $b$ redundant (thus the result not minimal). If *two*, we can answer **yes**, as the three terminals are in the same connected component, and any MISS of $G \setminus R$ will fully contain $S$. Finally, if *one*,

[5] If $W(S) = 1$, $S$ is an induced path from a terminal, and the node opposite to it is always removable.

let us call it $w_b$ and let $C_1$ be the connected component of $G \setminus R$ containing $w_a$, $b$ and $w_b$, and $C_2$ the one containing the final terminal $w_c$; recall that we may use one node $c \in R$, and this node must connect $S$ to $C_2$. We try all possible choices for $c$, noting that $c$ cannot make any node of $S$ removable other than $b$ (otherwise it would remain removable in $S'$), and when testing $c$ we must remove all its neighbors in $C_1 \setminus S \cup W$ from the graph (otherwise we may use them to bypass $b$, making $b$ redundant). If any such suitable $c$ exists, we get a MISS $S'$ by taking any induced path from $c$ to $w_c$ in $C_2$ and any induced path from $b$ to $w_b$ in $C_1 \setminus (N(c) \setminus (S \cup W))$, and adding them to $S$, and thus we can answer **yes**. Otherwise, there is no other possibility of minimally connecting $w_b$ and $w_c$ to $w_a$ without making $b$ or some other node redundant, so we answer **no**.

- $|\mathbf{W}(\mathbf{S})| = \mathbf{2}$. Let $W(S) = \{w_a, w_b\}$. If there is a removable node $b$ in $S$, define as before $R = \{v \in V(G) \setminus (S \cup W) : N(v) \cap (W(S) \cup S \setminus \{b\}) \neq \emptyset\}$. In this case, we may not use any node of $R$ to connect the remaining terminal $w_c$, as this would allow us to bypass $b$, making it redundant. Thus if $w_c$ is in the same connected component as $b$ in $G \setminus R$, we can answer **yes**, as any induced path between $b$ and $w_c$ in this graph yields a MISS when added to $S$. Otherwise, there is no connection without using some node of $R$ and making $b$ redundant, so we can answer **no**. Otherwise, if there is no removable node in $S$, define $R' = \{v \in V(G) \setminus (S \cup W) : N(v) \cap (W(S) \cup S) \neq \emptyset\}$: clearly any MISS extending $S$ must contain at least one (actually, exactly one) node from $R'$, and for any $v \in R'$, $v$ is removable in $S \cup \{v\}$, so we can test all possible nodes in $R'$ as above, since $S \cup \{v\}$ has a removable node (trivially accounting for the special cases where $v$ directly connects to the third terminal resulting directly in a MISS, or where $v$ makes some other node in $S$ removable and thus cannot be added to $S$).
- $|\mathbf{W}(\mathbf{S})| = \mathbf{3}$. Then either $S$ is a MISS and the answer is **yes**, or it has removable nodes and the answer is **no**.

As for the time complexity, removing a set of nodes and computing the connected components of a graph can be done in $O(m)$ time. The cost is dominated by cases $|\mathbf{W}(\mathbf{S})| = \mathbf{1}$ and $|\mathbf{W}(\mathbf{S})| = \mathbf{2}$, where we need to perform such a test for each node in $R$ or $R'$, respectively. As these sets have size $O(n)$ we can answer the question in $O(mn)$ time. ◀

We can finally claim correctness and complexity of the algorithm:

▶ **Theorem 13.** *Given $G$ and a set of terminals $W \subseteq V(G)$ with $t = |W| = 3$, Algorithm 1 lists all MISS's of $G$ for $W$ in $O(mn^3)$ time per solution.*

**Proof.** Firstly, the initial call with $S = \emptyset$ clearly does not trigger the output in Line 4 as $W(S) = \emptyset$. On all other recursive calls, $S$ is a PSS such that there exist some MISS $S'$ including $S$ (due to Line 8 in the parent call). Whenever $W(S) = W$, it follows that $S' = S$ because $S'$ is minimal and cannot contain other (redundant) nodes. So the algorithm only outputs MISS's.

Furthermore, let $S^*$ be an arbitrary MISS, we show that it is found: let $X$ be any recursion node of Algorithm 1 such that no node of $S^*$ has been removed from $G$, and the current $S$, called here $S_X$, is fully contained in $S^*$ ($X$ exists as both conditions are met in the beginning, when no node has been removed from $G$ and $S = \emptyset$). Let $x$ be the first node of $S^* \setminus S_X$ considered on Line 7 such that $S_X \cup \{x\}$ is a PSS ($x$ exists by Lemma 11): when $x$ is considered, clearly no node of $S^*$ has yet been removed from $G$; furthermore, the condition in Line 8 is true by the existence of $S^*$, so a child recursive call is generated with $S_Y = S_X \cup \{x\}$ and $G$ still fully containing $S^*$. By induction, the algorithm will, in some path of the recursion tree, at each step add one more node of $S^*$ without removing any from $G$, until eventually $S^*$ is found.

Finally, duplication is impossible by the binary partition argument, as all solutions generated from a recursive call on Line 9 will contain $v$, and all subsequent recursive calls will remove $v$ from $G$ (see Line 10), thus will find different solutions.

As for the cost per solution, since all leaves of the recursion tree output solutions, it is bound by the cost of the recursion nodes of a root-to-leaf path in the recursion tree, which are $O(n)$ as each node adds a node to $S$ when recurring. In each node, we must identify all $v$ such that $S \cup \{v\}$ is a PSS: some trivial case analysis (using Fact 2) can reveal that this is true only when $v$ is a neighbor only of the removable node of $S$, or when $S$ has no removable node and $v$ has at most two neighbors in $S$, adjacent to each other, thus this takes $O(|N(v)|)$, for a total of $O(\sum_{v \in V(G)} |N(v)|) = O(m)$ time. Furthermore, the test in Line 8 takes $O(mn)$ time by Lemma 12 and is performed $O(n)$ times, for a total cost per recursion node of $(mn^2)$. The cost per solution of Algorithm 1 is thus $O(mn^3)$. ◀

## 5 Conclusions

We considered minimal Steiner induced subgraphs, a natural variant of Steiner trees, shifting our focus on a node-centric view. We studied these combinatorial objects providing an output-sensitive algorithm for minimum MISS's. On the other hand, getting all MISS's with an output-sensitive algorithm seems to be more challenging for unbounded number of terminals, as we have shown that this problem is actually harder than listing all the minimal hypergraph transversals. Moreover, the fact that the extension problem is actually NP-complete, even with bounded number of terminals, makes us thinking that an *ad hoc* strategy should be thought for different sizes of $W$. For $t = 2$, the problem can be solved by known listing algorithm for induced paths. In this paper we have provided an output-sensitive algorithm for the case $t = 3$.

As a final remark, we observe that for many problems, like, for instance, listing maximum size and maximal independent sets [5], listing minimum/maximum solutions is harder than listing minimal/maximal solutions. In contrast, in the case of MISS's with a bounded number of terminals, listing minimum size MISS's seems to be easier than getting all MISS's, as we could efficiently solve the former case for $t = O(1)$, while the latter only for $t \leq 3$.

## References

1 A. Aazami, J. Cheriyan, and K. R. Jampani. Approximation Algorithms and Hardness Results for Packing Element-Disjoint Steiner Trees in Planar Graphs. *Algorithmica*, 63(1–2):425–456, June 2012.

2 Lijun Chang, Xuemin Lin, Lu Qin, Jeffrey Xu Yu, and Wenjie Zhang. Index-based Optimal Algorithms for Computing Steiner Components with Maximum Connectivity. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 459–474, New York, NY, USA, 2015. ACM. `doi:10.1145/2723372.2746486`.

3 Chandra Chekuri, Alina Ene, and Ali Vakilian. Node-Weighted Network Design in Planar and Minor-Closed Families of Graphs. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 206–217. Springer, 2012.

4 Joseph Cheriyan and Mohammad R. Salavatipour. Packing element-disjoint Steiner trees. *ACM Transactions on Algorithms*, 3(4):47:1–47:10, November 2007. `doi:10.1145/1290672.1290684`.

5 Alessio Conte, Roberto Grossi, Andrea Marino, Takeaki Uno, and Luca Versari. Listing Maximal Independent Sets with Minimal Space and Bounded Delay. In *String Processing and*

*Information Retrieval - 24th International Symposium, SPIRE 2017, Palermo, Italy, September 26-29, 2017, Proceedings*, pages 144–160, 2017. doi:10.1007/978-3-319-67428-5_13.

6 Nicolas Derhy and Christophe Picouleau. Finding induced trees. *Discrete Applied Mathematics*, 157(17):3552–3557, 2009.

7 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

8 DIMACS. 11th DIMACS Implementation Challenge. http://dimacs11.zib.de/, 2014 (page accessed April 2019).

9 Mitre C. Dourado, Rodolfo A. Oliveira, and Fábio Protti. Algorithmic aspects of Steiner convexity and enumeration of Steiner trees. *Annals of Operations Research*, 223(1):155–171, December 2014. doi:10.1007/s10479-014-1607-5.

10 Thomas Eiter, Kazuhisa Makino, and Georg Gottlob. Computational Aspects of Monotone Dualization: A Brief Survey. *Discrete Appl. Math.*, 156(11):2035–2049, June 2008.

11 Sudipto Guha and Samir Khuller. Improved Methods for Approximating Node Weighted Steiner Trees and Connected Dominating Sets. *Information and Computation*, 150(1):57–74, 1999. doi:10.1006/inco.1998.2754.

12 Mathias Hauptmann and Marek Karpinski. A Compendium on Steiner Tree Problems. http://theory.cs.uni-bonn.de/info5/steinerkompendium/, Accessed February 2018.

13 Daiki Hoshika and Eiji Miyano. Approximation Algorithms for Packing Element-Disjoint Steiner Trees on Bounded Terminal Nodes. *IEICE Transactions*, 99-A(6):1059–1066, 2016. URL: http://search.ieice.org/bin/summary.php?id=e99-a_6_1059.

14 Jiafeng Hu, Xiaowei Wu, Reynold Cheng, Siqiang Luo, and Yixiang Fang. On Minimal Steiner Maximum-Connected Subgraph Queries. *IEEE Trans. Knowl. Data Eng*, 29(11):2455–2469, 2017. doi:10.1109/TKDE.2017.2730873.

15 L. Khachiyan, E. Boros, K. Elbassioni, V. Gurvich, and K. Makino. On the Complexity of Some Enumeration Problems for Matroids. *SIAM Journal on Discrete Mathematics*, 19(4):966–984, January 2005. doi:10.1137/S0895480103428338.

16 PACE. The parametrized Algorithms and Computational Experiments Challenge. https://pacechallenge.wordpress.com/pace-2018/, 2018.

17 Jan Arne Telle and Yngve Villanger. Connecting Terminals and 2-Disjoint Connected Subgraphs. In *Graph-Theoretic Concepts in Computer Science*, pages 418–428. Springer Berlin Heidelberg, 2013.

18 Takeaki Uno. An Output Linear Time Algorithm for Enumerating Chordless Cycles. *IPSJ SIG Notes*, 2003(110):47–53, November 2003.