Finding a minimum spanning tree with a small non-terminal set

Hanaka, Tesshu
Kyushu University

Kobayashi, Yasuaki
Hokkaido University

出版情報：pp.1-17, 2023-10-09
バージョン：

Finding a Minimum Spanning Tree with a Small Non-Terminal Set

Tesshu Hanaka
Kyushu University, Japan

Yasuaki Kobayashi
Hokkaido University, Japan

Abstract

In this paper, we study the problem of finding a minimum weight spanning tree that contains each vertex in a given subset V_{NT} of vertices as an internal vertex. This problem, called Minimum Weight Non-Terminal Spanning Tree, includes s-t Hamiltonian Path as a special case, and hence it is NP-hard. In this paper, we first observe that Non-Terminal Spanning Tree, the unweighted counterpart of Minimum Weight Non-Terminal Spanning Tree, is already NP-hard on some special graph classes. Moreover, it is W[1]-hard when parameterized by clique-width. In contrast, we give a $3k$-vertex kernel and $O^*(2^k)$-time algorithm, where k is the size of non-terminal set V_{NT}. The latter algorithm can be extended to Minimum Weight Non-Terminal Spanning Tree with the restriction that each edge has a polynomially bounded integral weight. We also show that Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable parameterized by the number of edges in the subgraph induced by the non-terminal set V_{NT}, extending the fixed-parameter tractability of Minimum Weight Non-Terminal Spanning Tree to a more general case. Finally, we give several results for structural parameterization.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Spanning tree, Fixed-parameter algorithms, Parameterized complexity, Kernelization

Digital Object Identifier 10.4230/LIPIcs...

Funding *Tesshu Hanaka*: JSPS KAKENHI Grant Number JP21K17707, JP21H05852, JP22H00513, JP23H04388, and JST CRONOS Grant Number JPMJCS24K2
Yasuaki Kobayashi: JSPS KAKENHI Grant Numbers JP20H00595 and JP23H03344

1 Introduction

The notion of spanning trees plays a fundamental role in graph theory, and the minimum weight spanning tree problem is arguably one of the most well-studied combinatorial optimization problems. Numerous variants of this problem are studied in the literature (e.g. [24, 25, 40]). In this paper, we consider Minimum Weight Non-Terminal Spanning Tree, which is defined as follows. In a (spanning) tree T, leaf vertices, which are of degree 1, are called *terminals*, and other vertices are called *non-terminals*. In Minimum Weight Non-Terminal Spanning Tree, we are given an edge-weighted graph $G = (V, E, w)$ with $w\colon E \to \mathbb{R}^+$ and a set of designated non-terminals $V_{\mathrm{NT}} \subseteq V$. The goal of this problem is to find a minimum weight spanning tree T of G subject to the condition that each vertex in V_{NT} is of degree at least 2 in T. We also consider the unweighted variant of Minimum Weight Non-Terminal Spanning Tree, which we call Non-Terminal Spanning Tree: Given a graph $G = (V, E)$ and $V_{\mathrm{NT}} \subseteq V$, the goal is to determine whether G has a spanning tree T such that each vertex in V_{NT} is of degree at least 2 in T.

Minimum Weight Non-Terminal Spanning Tree was firstly introduced by Zhang and Yin [44]. Unlike the minimum weight spanning tree problem, Minimum Weight

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Non-Terminal Spanning Tree is NP-hard [44]. Nakayama and Masuyama [33] observed that Non-Terminal Spanning Tree is also NP-hard as s-t Hamiltonian Path, the problem of finding a Hamiltonian path between specified vertices s and t, is a special case of Non-Terminal Spanning Tree: When $V_{\rm NT} = V \setminus \{s, t\}$, any solution of Non-Terminal Spanning Tree is a Hamiltonian path between s and t. Nakayama and Masuyama [33, 34, 35, 36] devised polynomial-time algorithms for Non-Terminal Spanning Tree on several classes of graphs, such as cographs, outerplanar graphs, and series-parallel graphs.

1.1 Our contribution

In this paper, we study Minimum Weight Non-Terminal Spanning Tree and Non-Terminal Spanning Tree from the viewpoint of parameterized complexity.

On the positive side, we first give a $3k$-vertex kernel for Non-Terminal Spanning Tree, where k is the number of vertices in $V_{\rm NT}$. We also give polynomial kernelizations for Non-Terminal Spanning Tree with respect to vertex cover number and max leaf number. For Minimum Weight Non-Terminal Spanning Tree, we give an $O^*(2^k)$-time algorithm[1] when the weight of edges is integral and upper bounded by a polynomial in the input size. Moreover, we design an $O^*(2^\ell)$-time algorithm for Minimum Weight Non-Terminal Spanning Tree for an arbitrary edge weight, where ℓ is the number of edges in the subgraph induced by $V_{\rm NT}$. Note that this parameter ℓ is "smaller" than k in the sense that $\ell \leq \binom{k}{2}$ but k could be arbitrarily large even when $\ell = 0$. Finally, we show that the property of being a spanning tree that has all vertices in $V_{\rm NT}$ as internal vertices is expressed by an $\rm MSO_2$ formula, which implies that Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable when parameterized by treewidth.

On the negative side, we observe that Non-Terminal Spanning Tree is NP-hard even on planar bipartite graphs of maximum degree 3, strongly chordal split graphs, and chordal bipartite graphs. Moreover, we observe that Non-Terminal Spanning Tree is W[1]-hard when parameterized by cliquewidth, and it cannot be solved in time $2^{o(n)}$ unless the Exponential Time Hypothesis fails, where n is the number of vertices in the input graph. In contrast to the fact that Non-Terminal Spanning Tree admits a polynomial kernelization with respect to vertex cover number, we show that it does not admit a polynomial kernelization with respect to vertex integrity under some complexity-theoretic assumption.

1.2 Related work

The minimum weight spanning tree problem is a fundamental graph problem. There are many variants of this problem, such as Minimum Diameter Spanning Tree [25], Degree-Bounded Spanning Tree [24], Diameter-Bounded Spanning Tree [21], and Max Leaf Spanning Tree [40].

Max Internal Spanning Tree is highly related to Non-Terminal Spanning Tree. Given a graph G and an integer k, the task of Max Internal Spanning Tree is to find a spanning tree T of G that has at least k internal vertices. The difference between Non-Terminal Spanning Tree and Max Internal Spanning Tree is that the former designates internal vertices in advance whereas the latter does not. Since Max Internal

[1] The O^* notation suppresses a polynomial factor of the input size.

Spanning Tree is a generalization of Hamiltonian Path, it is NP-hard. For restricted graph classes, the problem can be solved in polynomial time, such as interval graphs [30], cacti, block graphs, cographs, and bipartite permutation graphs [41]. Binkele-Raible et al. [3] give an $O^*(3^n)$-time algorithm for Max Internal Spanning Tree and then Nederlof [37] improves the running time to $O^*(2^n)$, where n is the number of vertices of the input graph. For the fixed-parameter tractability, Fomin et al. [15] show that Max Internal Spanning Tree admits a $3k$-vertex kernel. Then Li et al. [29] improve the size of the kernel by giving a $2k$-vertex kernel. The best known deterministic $O^*(4^k)$-time algorithm is obtained by combining a $2k$-vertex kernel with an $O^*(2^n)$-time algorithm, while there is a randomized $O^*(\min\{3.455^k, 1.946^n\})$-time algorithm [4]. For approximation algorithms, Li et al. [31] propose a 3/4-approximation algorithm and Chen et al. [8] improve the approximation ratio to 13/17.

Max Leaf Spanning Tree is a "dual" problem of Max Internal Spanning Tree, where the goal is to compute a spanning tree that has leaves as many as possible. This problem is highly related to Connected Dominating Set and well-studied in the context of parameterized complexity [10, 40].

The *designated leaves* version of Max Leaf Spanning Tree is called Terminal Spanning Tree. In this problem, given a graph $G = (V, E)$ and terminal set $W \subseteq V$, the task is to find a spanning tree T such that every vertex in W is a leaf in T. Unlike Non-Terminal Spanning Tree, it can be solved in polynomial time by finding a spanning tree in the subgraph induced by $V \setminus W$ and adding vertices in W as leaves.

In the context of graph theory, Non-Terminal Spanning Tree has already discussed in [13, 26]. More precisely, the paper [13] gives a sufficient condition that for a graph G, a vertex set S, and a function $f\colon S \to \mathbb{N}$, G has a spanning tree such that the degree of each vertex $v \in S$ is at least $f(v)$ in the spanning tree. Király [26] gives the shorter proofs of the sufficient condition and proposes a polynomial-time algorithm for checking the condition.

2 Preliminaries

In this paper, we use several basic concepts and terminology in parameterized complexity. We refer the reader to [10].

Graphs. Let $G = (V, E)$ be an undirected graph. We denote by $V(G)$ and $E(G)$ the vertex set and edge set of G, respectively. We use n to denote the number of vertices in G. For $X \subseteq V$, we denote by $G[X]$ the subgraph induced by X. For $v \in V$ and $X \subseteq V$, $G - v$ and $G - X$ denote $G[V \setminus \{v\}]$ and $G[V \setminus X]$, respectively. We also define $G - e = (V, E \setminus \{e\})$ and $G + e = (V, E \cup \{e\})$. For $v \in V$, we let $N_G(v)$ denote the set of neighbors of v and $N_G[v] = N_G(v) \cup \{v\}$. This notation is extended to sets: for $X \subseteq V$, we let $N_G[X] = \bigcup_{v \in X} N_G[v]$ and $N_G(X) = N_G[X] \setminus X$. When the subscript G is clear from the context, we may omit it.

A *forest* is a graph having no cycles. If a forest is connected, it is called a *tree*. In a forest F, a vertex is called a *leaf* if it is of degree 1 in F, and called an *internal* vertex otherwise.

▶ **Definition 1.** *For a vertex set $X \subseteq V$ and a forest F, we say F is* admissible *for X if each $v \in X$ is an internal vertex in F.*

In Non-Terminal Spanning Tree, given a non-terminal set V_{NT}, the goal is to determine whether there is an admissible spanning tree for V_{NT} in G. Throughout the paper, we denote $k = |V_{\mathrm{NT}}|$ and $\ell = |E(G[V_{\mathrm{NT}}])|$, and assume that $k \le n - 2$. We also define an optimization version of Non-Terminal Spanning Tree. In Minimum Weight Non-Terminal Spanning Tree, we are additionally given an edge weight function $w\colon E(G) \to \mathbb{R}$, the goal is to find an admissible spanning tree T for V_{NT} in G minimizing its total weight $w(E(T)) = \sum_{e \in E(T)} w(e)$. We also assume that the input graph G is connected as otherwise, our problems are trivially infeasible. The following easy proposition is useful to extend a forest into a spanning tree.

▶ **Proposition 2.** *Let G be a connected graph and F be a forest in G. Then there is a spanning tree that contains all edges in F.*

Graph parameters. A set S of vertices is called a *vertex cover* if every edge has at least one endpoint in S. The vertex cover number $\mathtt{vc}(G)$ of G is defined by the size of a minimum vertex cover in G. The *vertex integrity* $\mathtt{vi}(G)$ of G is the minimum integer p satisfying that there exists $S \subseteq V$ such that $|S| + \max_{H \in \mathtt{cc}(G-S)} |V(H)| \le p$, where $\mathtt{cc}(G - S)$ is the set of connected components of $G - S$. The *max leaf number* $\mathtt{ml}(G)$ of G is the maximum integer q such that there exists a spanning tree having q leaves in G. For the definition of treewidth $\mathtt{tw}(G)$, pathwidth $\mathtt{pw}(G)$, and treedepth $\mathtt{td}(G)$, we refer the reader to the books [38, 10]. It is well-known that these graph parameters have the following relationship.

▶ **Proposition 3** ([1, 14, 22])**.** *For every graph G, it holds that $\mathtt{tw}(G) \le \mathtt{pw}(G) \le \mathtt{td}(G) - 1 \le \mathtt{vi}(G) - 1 \le \mathtt{vc}(G)$ and $\mathtt{tw}(G) \le \mathtt{pw}(G) \le 2\mathtt{ml}(G)$.*

Matroids. Let U be a finite set. A pair $\mathcal{M} = (U, \mathcal{B})$ with $\mathcal{B} \subseteq 2^U$ is called a *matroid*[2] if the following axioms are satisfied:

- $\mathcal{B}$ is nonempty;
- For $X, Y \in \mathcal{B}$ with $X \neq Y$ and for $x \in X \setminus Y$, there is $y \in Y \setminus X$ such that $(X \setminus \{x\}) \cup \{y\} \in \mathcal{B}$.

It is not hard to verify that all the sets in $\mathcal{B}$ have the same cardinality.

There are many combinatorial objects that can be represented as matroids. Let $H = (V, E)$ be a graph. If $\mathcal{B}_H$ consists of all subsets of edges, each of which forms a spanning tree in H, then the pair $(E, \mathcal{B}_H)$ is a matroid, which is called a *graphic matroid*. We are also interested in another matroid. Let U be a finite set and let $\{U_1, U_2, \ldots, U_t\}$ be a partition of U: $U_i \cap U_j = \emptyset$ for $i \neq j$ and $\bigcup_{1 \le i \le t} U_i = U$. For $1 \le i \le t$, let ℓ_i and u_i be two non-negative integers with $\ell_i \le u_i$. For a non-negative integer r, we define a set $\mathcal{B}_r$ consisting of all size-r subsets $U' \subseteq U$ such that $\ell_i \le |U' \cap U_i| \le u_i$ for $1 \le i \le t$. Then, the pair $(U, \mathcal{B}_r)$ is a matroid unless $\mathcal{B}_r = \emptyset$.

▶ **Proposition 4.** *If $\mathcal{B}_r \neq \emptyset$, then $(U, \mathcal{B}_r)$ is a matroid.*

Proof. It suffices to show that $\mathcal{B}_r$ satisfies the second axiom of matroids. Let $X, Y \in \mathcal{B}_r$ with $X \neq Y$. For $1 \le i \le t$, let $x_i = |X \cap U_i|$ (resp. $y_i = |Y \cap U_i|$). Let $x \in X \setminus Y$ and assume that $x \in U_i$. Suppose first that $x_i \le y_i$. Since $x \in X \setminus Y$ and $|X \cap U_i| \le |Y \cap U_i|$, there is $y \in U_i$ such that $y \in Y \setminus X$. Then, we have $Z = (X \setminus \{x\}) \cup \{y\} \in \mathcal{B}_r$ as $|Z \cap U_j| = |X \cap U_j|$ for $1 \le j \le t$. Suppose next that $x_i > y_i$. As $\sum_{1 \le j \le k} x_j = \sum_{1 \le j \le k} y_j = r$, there is an index i' with $x_{i'} < y_{i'}$. This implies that $U_{i'}$ contains an element $y \in Y \setminus X$. Then, $Z = (X \setminus \{x\}) \cup \{y\} \in \mathcal{B}_r$ as $\ell_i \le y_i \le x_i - 1 = |Z \cap U_i|$, $|Z \cap U_{i'}| = x_{i'} + 1 \le y_{i'} \le u_{i'}$, and $|Z \cap U_j| = |X \cap U_j|$ for all $j' \neq i, i'$. ◀

[2] This definition is equivalent to that defined by the family of *independent sets* [39].

Let $\mathcal{M}_1 = (U, \mathcal{B}_1)$ and $\mathcal{M}_2 = (U, \mathcal{B}_2)$ be matroids. It is well known that there is a polynomial-time algorithm to check whether $\mathcal{B}_1 \cap \mathcal{B}_2$ is empty when set U and an *independence oracle* is given as input (e.g., [19]), where an independence oracle for a matroid $\mathcal{M} = (U, \mathcal{B})$ is a black-box procedure that given a set $U' \subseteq U$, returns true if there is a set $B \in \mathcal{B}$ that contains U'. Moreover, we can find a minimum weight common base in two matroids in polynomial time.

▶ **Theorem 5** ([19]). *Let U be a finite set and let $w \colon U \to \mathbb{R}$. Given two matroids $\mathcal{M}_1 = (U, \mathcal{B}_1)$ and $\mathcal{M}_2 = (U, \mathcal{B}_2)$, we can compute a set $X \in \mathcal{B}_1 \cap \mathcal{B}_2$ minimizing $w(X)$ in polynomial time, provided that the polynomial-time independence oracles of $\mathcal{M}_1$ and $\mathcal{M}_2$ are given as input.*

Finally, we observe that there are polynomial-time independence oracles for graphic matroids and $(U, \mathcal{B}_r)$. By Proposition 2, the independence oracle for the graphic matroid defined by a connected graph G returns true for given $F \subseteq E(G)$ if and only if F is a forest. The following lemma gives an independence oracle for $(U, \mathcal{B}_r)$.

▶ **Lemma 6.** *There is a polynomial-time algorithm that given a set $U' \subseteq U$, determines whether there is a set $U'' \in \mathcal{B}_r$ with $U' \subseteq U'' \subseteq U$, that is, $|U''| = r$ and $\ell_i \le |U'' \cap U_i| \le u_i$ for all $1 \le i \le k$.*

Proof. The problem can be reduced to that of finding a degree-constrained subgraph of an auxiliary bipartite graph. To this end, we construct a bipartite graph H as follows. The graph H consists of two independent sets A and B, where each vertex a_i of A corresponds to a block U_i of the partition and each vertex b_e of B corresponds to an element e in $U \setminus U'$. For each $e \in U \setminus U'$, H contains an edge between b_e and a_i, where i is the unique index with $e \in U_i$. Then, we define functions $d^\ell \colon A \cup B \to \mathbb{N}$ and $d^u \colon A \cup B \to \mathbb{N}$ as:

$$d^\ell(v) = \begin{cases} 0 & \text{if } v \in B \\ \ell_i - |U' \cap U_i| & \text{if } v \in A \end{cases} \qquad d^u(v) = \begin{cases} 1 & \text{if } v \in B \\ u_i - |U' \cap U_i| & \text{if } v \in A \end{cases}$$

Then, we claim that there is a set U'' satisfying the condition in the statement of this lemma if and only if H has a subgraph H' such that $d^\ell(v) \le d_{H'}(v) \le d^u(v)$ for all $v \in V(H)$, where $d_{H'}(v)$ is the degree of v in H', and H' contains exactly $r - |U'|$ edges. From a feasible set $U'' \subseteq U$, we construct a subgraph H' in such a way that for each $b_e \in B$ we take the unique edge (a_i, b_e) if $e \in U'' \setminus U'$. Clearly, this subgraph H' contains $r - |U'|$ edges. Moreover, H' satisfies the degree condition as

$$d_{H'}(a_i) = |(U'' \setminus U') \cap U_i| = |U'' \cap U_i| - |U' \cap U_i| \ge \ell_i - |U' \cap U_i| = d^\ell(a_i),$$

and, analogously, we have $d_{H'}(a_i) \le d^u(a_i)$. This transformation is reversible and hence the converse direction is omitted here.

Given a graph H, an integer r', and the degree bounds d^ℓ, d^u, there is a polynomial-time algorithm that finds a subgraph H' with r' edges satisfying $d^\ell(v) \le d_{H'}(v) \le d^u(v)$ for $v \in V(H)$ [20, 42]. Hence, the lemma follows. ◀

3 Kernelization

3.1 Linear kernel for k

In this subsection, we give a linear vertex kernel of Non-Terminal Spanning Tree when parameterized by $k = |V_{\rm NT}|$. Let G be a graph with a non-terminal set $V_{\rm NT} \subseteq V(G)$. The following easy lemma is a key to our results.

▶ **Lemma 7.** *There is an admissible spanning tree for V_{NT} in G if and only if there is an admissible forest for V_{NT} in $G[N[V_{\mathrm{NT}}]]$.*

Proof. Let T be an admissible spanning tree for V_{NT} in G. We remove vertices in $V(G) \setminus N[V_{\mathrm{NT}}]$ from T. Since this does not change the degree of any vertex in V_{NT}, we have an admissible forest in $G[N[V_{\mathrm{NT}}]]$.

Conversely, suppose that there is an admissible forest F for V_{NT} in $G[N[V_{\mathrm{NT}}]]$. This forest is indeed a forest in G. By Proposition 2, there is a spanning tree of G that contains F as a subgraph. This spanning tree is admissible for V_{NT} in G as the degree of any vertex in V_{NT} is at least 2 in F. ◀

Let G' be the graph obtained from G by deleting all the vertices in $V(G)\setminus N[V_{\mathrm{NT}}]$, adding a vertex r, and connecting r to all the vertices in $N(V_{\mathrm{NT}})$. From the assumption that G is connected, G' is also connected. By Lemma 7, we can immediately obtain the following corollary.

▶ **Corollary 8.** *There is an admissible spanning tree for V_{NT} in G if and only if there is an admissible spanning tree for V_{NT} in G'.*

Due to Corollary 8, we can "safely" reduce G to G', that is, G has an admissible spanning tree for V_{NT} if and only if G' does. The graph G' may still have an arbitrary number of vertices as the size of $N_{G'}(V_{\mathrm{NT}})$ cannot be upper bounded by a function in k. To reduce this part, we apply the expansion lemma [18].

▶ **Definition 9.** *Let $H = (A \cup B, E_H)$ be a bipartite graph. For positive integer q, $M \subseteq E_H$ is called a* q-expansion *of A into B if it satisfies the following:*
- *every vertex in A is incident to exactly q edges in M.*
- *there are exactly $q|A|$ vertices in B, each of which is incident to exactly one edge in M.*

▶ **Lemma 10** (Expansion lemma [18, 43])**.** *Let q be a positive integer and $H = (A \cup B, E_H)$ be a bipartite graph such that:*
- *$|B| \geq q|A|$, and*
- *there are no isolated vertices in B.*

Then there are non-empty vertex sets $X \subseteq A$ and $Y \subseteq B$ such that:
- *$N_H(Y) \subseteq X$, and*
- *there is a q-expansion $M \subseteq E_H$ of X into Y.*

Moreover, one can find such X, Y, and M in polynomial time in the size of H.

Suppose that $N_{G'}(V_{\mathrm{NT}})$ has at least $2k$ vertices. Let $H = (V_{\mathrm{NT}} \cup N_{G'}(V_{\mathrm{NT}}), E_H)$ be the bipartite graph obtained from $G'[N_{G'}[V_{\mathrm{NT}}]]$ by deleting all edges between vertices in V_{NT} and those between vertices in $N_{G'}(V_{\mathrm{NT}})$. As $N_{G'}(V_{\mathrm{NT}})$ has no isolated vertices in H, by Lemma 10, there exists a 2-expansion $M \subseteq E_H$ of X into Y for some non-empty sets $X \subseteq V_{\mathrm{NT}}$ and $Y \subseteq N_{G'}(V_{\mathrm{NT}})$ such that $N_H(Y) \subseteq X$. We then construct a smaller instance $(\hat{G}, \hat{V}_{\mathrm{NT}})$ of Non-Terminal Spanning Tree as follows. We first delete all vertices in Y from G' and add an edge between r and x for each $x \in X$. The graph obtained in this way is denoted by $\hat{G}$. Then we set $\hat{V}_{\mathrm{NT}} = V_{\mathrm{NT}} \setminus X$. See Figure 1 for an illustration.

Observe that $\hat{G}$ is also connected. This follows from the fact that every vertex in $(N_{G'}(V_{\mathrm{NT}}) \cup X) \setminus Y$ is adjacent to r and every vertex v in $\hat{V}_{\mathrm{NT}}$ is adjacent to some vertex of $(N_{G'}(V_{\mathrm{NT}}) \cup X) \setminus Y$ in $\hat{G}$ as otherwise $v \in \hat{V}_{\mathrm{NT}}$ is adjacent to a vertex in Y, which violates the fact that $N_H(Y) \subseteq X$.

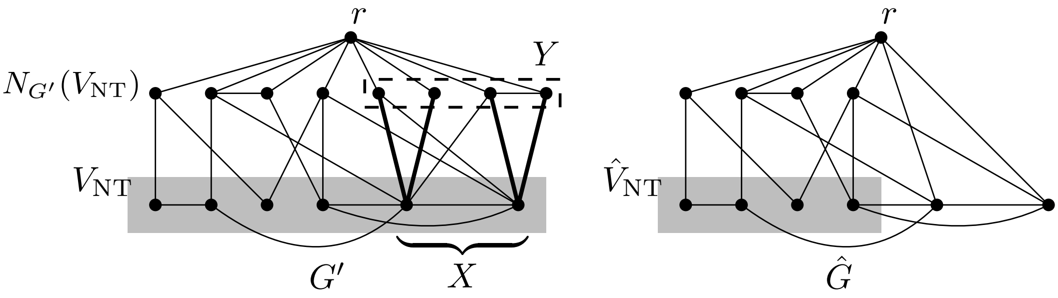

Figure 1 The construction of $\hat{G}$. The shaded areas indicate the vertices of V_{NT} and $\hat{V}_{\mathrm{NT}}$. The bold lines are edges in a 2-expansion of X into Y.

▶ **Lemma 11.** *There exists an admissible spanning tree for V_{NT} in G' if and only if there exists an admissible spanning tree for $\hat{V}_{\mathrm{NT}}$ in $\hat{G}$.*

Proof. Suppose that there exists an admissible spanning tree T for V_{NT} in G'. Since T is admissible for V_{NT}, for every $v \in V_{\mathrm{NT}} \setminus X$, T has at least two edges incident to v. These edges are contained in $\hat{G}$ as there are no edges between Y and $V_{\mathrm{NT}} \setminus X$. Thus, there is an admissible forest for $\hat{V}_{\mathrm{NT}}$ in $\hat{G}$, consisting of these two incident edges for each $v \in V_{\mathrm{NT}} \setminus X$, and by Proposition 2, $\hat{G}$ has an admissible spanning tree for $\hat{V}_{\mathrm{NT}}$.

Conversely, let $\hat{T}$ be an admissible spanning tree for $\hat{V}_{\mathrm{NT}}$ in $\hat{G}$. To construct a spanning tree of G', we select all edges of $\hat{T}$ incident to $\hat{V}_{\mathrm{NT}}$. These edges are also contained in G'. Moreover, we select all edges in the 2-expansion M of X into Y. Since there are no edges between $\hat{V}_{\mathrm{NT}}$ and Y in G', the subgraph consisting of selected edges does not have cycles. Moreover, as every vertex of V_{NT} is of degree at least 2 in the subgraph, it is an admissible forest for V_{NT} in G'. By Proposition 2, there is an admissible spanning tree for V_{NT} in G'. ◀

Our kernelization is described as follows. From a connected graph G, we first construct the graph G' by deleting the vertices in $V(G) \setminus N[V_{\mathrm{NT}}]$ and adding r adjacent to each vertex in $N(V_{\mathrm{NT}})$. By Corollary 8, (G, V_{NT}) is a yes-instance of Non-Terminal Spanning Tree if and only if so is (G', V_{NT}). If G' has at most $3k$ vertices, we are done. Suppose otherwise. As $V(G') \setminus N_{G'}[V_{\mathrm{NT}}] = \{r\}$ and $|V_{\mathrm{NT}}| = k$, we have $|N_{G'}(V_{\mathrm{NT}})| \geq 2k$. We apply the expansion lemma to G' and then obtain $\hat{G}$ and $\hat{V}_{\mathrm{NT}}$. By Lemma 11, (G, V_{NT}) is a yes-instance of Non-Terminal Spanning Tree if and only if so is $(\hat{G}, \hat{V}_{\mathrm{NT}})$. We repeatedly apply these reduction rules as long as the reduced graph $\hat{G}$ has at least $3|\hat{V}_{\mathrm{NT}}| + 1$ vertices. Therefore, we have the following theorem.

▶ **Theorem 12.** *Non-Terminal Spanning Tree admits a $3k$-vertex kernel.*

3.2 Linear kernel for vertex cover number

In this subsection, we show that Non-Terminal Spanning Tree admits a linear kernel when parameterized by vertex cover number. We first apply an analogous transformation used in the previous subsection. By Lemma 7, there is an admissible spanning tree for V_{NT} in G if and only if there is an admissible forest for V_{NT} in $G[N[V_{\mathrm{NT}}]]$. We remove all vertices of $V(G) \setminus N[V_{\mathrm{NT}}]$ and then add a vertex r that is adjacent to every vertex in $N(V_{\mathrm{NT}})$. As seen in the previous subsection, G has an admissible spanning tree for V_{NT} if and only if the obtained graph has. Thus, in the following, G consists of three vertex sets V_{NT}, $N(V_{\mathrm{NT}})$, and $\{r\}$, where r is adjacent to every vertex in $N(V_{\mathrm{NT}})$. Furthermore, we delete all the edges within $G[N(V_{\mathrm{NT}})]$, which is safe by Lemma 7. Thus, $N(V_{\mathrm{NT}})$ forms an independent set in G. Let S be a vertex cover of G and $I = V \setminus S$. As G is obtained from a subgraph of the original graph by adding a vertex r and deleting edges within $G[N(V_{\mathrm{NT}})]$, we have $|S| \leq \tau + 1$, where τ is the vertex cover number of the original graph. Suppose that $|V_{\mathrm{NT}} \cap I| \geq |S|$. Then we conclude that G has no admissible spanning tree for V_{NT}.

▷ Claim 13. If $|V_{\mathrm{NT}} \cap I| \geq |S|$, there is no admissible spanning tree for V_{NT}.

Proof. Suppose that T is an admissible spanning tree for V_{NT} in G. Then T has at least $2|V_{\mathrm{NT}} \cap I|$ edges between $V_{\mathrm{NT}} \cap I$ and S. Consider the subforest T' of T induced by $(V_{\mathrm{NT}} \cap I) \cup S$. We have

$$2|V_{\mathrm{NT}} \cap I| \leq |E(T')| \leq |V_{\mathrm{NT}} \cap I| + |S| - 1.$$

Thus, we have $|V_{\mathrm{NT}} \cap I| \leq |S| - 1$. ◀

Suppose next that $2|V_{\mathrm{NT}} \cap S| \leq |N(V_{\mathrm{NT}}) \cap I|$. We define a bipartite graph $H = ((V_{\mathrm{NT}} \cap S) \cup (N(V_{\mathrm{NT}}) \cap I), E_H)$, where E_H is the set of edges in G, each of which has an endpoint in $V_{\mathrm{NT}} \cap S$ and the other in $N(V_{\mathrm{NT}}) \cap I$. As $2|V_{\mathrm{NT}} \cap S| \leq |N(V_{\mathrm{NT}}) \cap I|$ and each vertex in $N(V_{\mathrm{NT}}) \cap I$ has a neighbor in $V_{\mathrm{NT}} \cap S$, there is a 2-expansion of $X \subseteq V_{\mathrm{NT}} \cap S$ into $Y \subseteq N(V_{\mathrm{NT}}) \cap I$ in H. By applying the same reduction rule used in Lemma 11, the obtained instance $(\hat{G}, \hat{V}_{\mathrm{NT}})$ satisfies the following claim.

▷ Claim 14. Suppose that G satisfies $2|V_{\mathrm{NT}} \cap S| \leq |N_G(V_{\mathrm{NT}}) \cap I|$. Then G has an admissible spanning tree for V_{NT} if and only if $\hat{G}$ has an admissible spanning tree for $\hat{V}_{\mathrm{NT}}$.

The proof of this claim is analogous to that in Lemma 11. Let us note that S remains a vertex cover of $\hat{G}$.

Our kernelization is formalized as follows. Let G be an input graph. Let S be a vertex cover of G with $|S| \leq 2 \cdot \mathtt{vc}(G)$ and let $I = V(G) \setminus S$. Such a vertex cover S can be computed in polynomial time by a well-known 2-approximation algorithm for the minimum vertex cover problem. Then our kernel (G', V'_{NT}) is obtained by exhaustively applying these reduction rules by Claims 13 and 14 to G. Let $S' = (V(G') \cap S) \cup \{r\}$ be a vertex cover of G' and let $I' = V(G') \setminus S'$. After these reductions, we have

$$\begin{aligned}
|V(G')| &= |S'| + |I'| \\
&\leq |S'| + |V'_{\mathrm{NT}} \cap I'| + |N(V'_{\mathrm{NT}}) \cap I'| \\
&\leq |S'| + |V'_{\mathrm{NT}} \cap I'| + 2|V_{\mathrm{NT}} \cap S'| - 1 && \text{(by Claim 14)} \\
&\leq |S'| + |S'| - 1 + 2|V_{\mathrm{NT}} \cap S'| - 1 && \text{(by Claim 13)} \\
&= 4|S'| - 2 && \text{(by } |S'| = |S| + 1\text{)} \\
&= 4|S| + 2,
\end{aligned}$$

yielding the following theorem.

▶ **Theorem 15.** *Non-Terminal Spanning Tree admits a* $(8\mathtt{vc}(G) + 2)$*-vertex kernel.*

3.3 Quadratic kernel for max leaf number

We show that Non-Terminal Spanning Tree admits a polynomial kernel when parameterized by max leaf number. Thanks to the following lemma, we can suppose that the input graph has at most $4\mathtt{ml}(G) - 2$ vertices of degree at least 3.

▶ **Lemma 16** ([28, 14, 6]). *Every graph G with max leaf number $\mathtt{ml}(G)$ is a subdivision of a graph with at most $4\mathtt{ml}(G) - 2$ vertices. In particular, G has at most $4\mathtt{ml}(G) - 2$ vertices of degree at least 3.*

Thus, we only have to reduce the number of vertices of degree at most 2. Let $V_i \subseteq V$ be the set of vertices of degree i and $V_{\geq i} = \bigcup_{j \geq i} V_j$. In the following, we say that a reduction rule is *safe* if the instance (G', V'_{NT}) obtained by applying the reduction rule is equivalent to the original instance (G, V_{NT}): G has an admissible spanning tree for V_{NT} if and only if G' has an admissible spanning tree for V'_{NT}.

In the following, we assume that each vertex in V_{NT} has degree at least 2, as otherwise the instance is clearly infeasible. Let v be a vertex of degree 1. By the assumption, we have $v \notin V_{\mathrm{NT}}$. If v has a neighbor that is not in V_{NT}, we can safely delete v. Otherwise, v has a neighbor u, which belongs to V_{NT}. Then we delete v and set $V_{\mathrm{NT}} := V_{\mathrm{NT}} \setminus \{u\}$. This is safe because the degree of u is at least 2 in any spanning tree of G.

Suppose that G has two adjacent vertices $u, v \in V_2$ of degree 2. Let x (resp. y) be the other neighbor of u (resp. v). If both u and v are contained in V_{NT}, we contract edge $e = \{u, v\}$ and include the corresponding vertex uv in V_{NT}. Since $\{x, u\}, \{u, v\}, \{v, y\}$ must be contained in any admissible spanning tree for V_{NT}, this reduction is also safe. Suppose that neither of u and v is contained in V_{NT}. If $\{u, v\}$ is a bridge in G, both $\{x, u\}$ and $\{v, y\}$ are also bridges in G, meaning that every admissible spanning tree for V_{NT} in G contains all of these bridges. Thus, we can contract edge $e = \{u, v\}$, which is safe. Otherwise, $G - e$ remains connected. If G has an admissible spanning tree T for V_{NT} containing e, forest $T - e$ is admissible for V_{NT}. Then, by Proposition 2, there is an admissible spanning tree for V_{NT} in $G - e$. Thus, we can safely remove edge e from G. In $G - e$, the degrees of u and v are exactly one. Therefore, we can further remove u, v safely.

From the above reductions, we can assume that G does not have consecutive vertices u, v of degree 2, both of which are in V_{NT} or not in V_{NT}. Let p, q, r, s be four consecutive vertices of degree 2 in G. Let x (resp. y) be the other neighbor of p (resp. s). Without loss of generality, we assume that $p, r \in V_{\mathrm{NT}}$ and $q, s \notin V_{\mathrm{NT}}$. Let G' be the graph obtained from G by contracting edge $\{p, q\}$ to a vertex pq and $V'_{NT} = V_{\mathrm{NT}} \setminus \{p, q\} \cup \{pq\}$. Since any admissible spanning tree for V_{NT} in G contains all the edges $\{x, p\}$, $\{p, q\}$, and $\{q, r\}$, we can reduce the instance (G, V_{NT}) to (G', V'_{NT}) safely.

Hence, when the above reductions are applied exhaustively, G has neither a vertex of degree 1 nor consecutive four vertices of degree 2.

Now we show that the reduced graph G' has $O(\mathtt{ml}(G)^2)$ vertices. As our reduction rules either delete a vertex of degree 1, delete an edge between vertices of degree 2, or contract an edge between vertices of degree 2, they do not newly introduce a vertex of degree at least 3. This implies that the set of vertices of degree at least 3 in G' is a subset of $V_{\geq 3}$. Moreover, as G' has no vertices of degree 1, we conclude that G' is a subdivision of a graph H with at most $4\mathtt{ml}(G) - 2$ vertices. Since this subdivision is obtained from H by subdividing each edge with at most three times, G' contains $O(\mathtt{ml}(G)^2)$ vertices.

▶ **Theorem 17.** *Non-Terminal Spanning Tree admits a quadratic vertex-kernel when parameterized by max leaf number.*

3.4 Kernel lower bound for vertex integrity

In this subsection, we show that Non-Terminal Spanning Tree does not admit a polynomial kernel when parameterized by vertex integrity unless NP $\subseteq$ coNP/poly.

▶ **Theorem 18.** *Non-Terminal Spanning Tree does not admit a polynomial kernel when parameterized by vertex integrity unless NP ⊆ coNP/poly.*

Proof. We construct an AND-cross-composition [5, 12] from s-t Hamiltonian Path. For q instances $(G_i = (V_i, E_i), s_i, t_i)$ of s-t Hamiltonian Path and a vertex r, we connect r to s_i by an edge $\{r, s_i\}$ for $1 \le i \le q$. Let G be the constructed graph and let $V_{\rm NT} = \bigcup_{1 \le i \le q} V_i \setminus \{t_i\}$. Then it is easy to see that every G_i has an s_i-t_i Hamiltonian path if and only if G has an admissible spanning tree for $V_{\rm NT}$. Since the vertex integrity of G is at most $\max_{1 \le i \le q} |V(G_i)| + 1$, the theorem holds. ◀

We remark that Non-Terminal Spanning Tree is significantly different from s-t Hamiltonian Path with respect to kernelization complexity.

▶ Remark 19. s-t Hamiltonian Path admits a polynomial kernel when parameterized by vertex integrity while it does not admit a polynomial kernel when parameterized by treedepth unless NP ⊆ coNP/poly.

Proof. We first show that s-t Hamiltonian Path admits a polynomial kernel when parameterized by vertex integrity. Let $\mathtt{vi}(G)$ be the vertex integrity of G. By a polynomial-time $O(\log \mathtt{vi}(G))$ approximation algorithm [23], we can obtain a vertex set S such that $|S| + \max_{H \in \mathtt{cc}(G-S)} |V(H)| = O(\mathtt{vi}(G) \log \mathtt{vi}(G))$ in polynomial time. If the number of connected components of $G - S$ is at least $|S| + 2$, we immediately conclude that the instance is infeasible because any Hamiltonian path has to go through at least one vertex in S from a connected component to another one. Otherwise, the number of connected components is at most $|S| + 1$. Since $|S| + \max_{H \in \mathtt{cc}(G-S)} |V(H)| = O(\mathtt{vi}(G) \log \mathtt{vi}(G))$, the number of vertices in G is at most $O(\mathtt{vi}(G)^2 \log^2 \mathtt{vi}(G))$.

To complement this positive result, we then show that the problem does not admit a polynomial kernel when parameterized by treedepth by showing an AND-cross-composition from s-t Hamiltonian Path. For q instances $(G_i = (V_i, E_i), s_i, t_i)$ of s-t Hamiltonian Path, we connect in series G_i and G_{i+1} by identifying t_i as s_{i+1} for $1 \le i \le q-1$. Let G be the constructed graph. By taking the center vertex (i.e., $s_{\lceil q/2 \rceil + 1}$) of G as a separator recursively, we can observe that the treedepth of G is at most $\lceil \log_2 q \rceil + \max_{1 \le i \le q} |V(G_i)|$. It is to see that G has an s_1-t_q Hamiltonian path if and only if every G_i has an s_i-t_i Hamiltonian path. Thus, the theorem holds. ◀

4 Fixed-Parameter Algorithms

4.1 Parameterization by k

By Theorem 12, Non-Terminal Spanning Tree is fixed-parameter tractable parameterized by $k = |V_{\rm NT}|$. A trivial brute force algorithm on a $3k$-vertex kernel yields a running time bound $2^{O(k^2)} + n^{O(1)}$, where n is the number of vertices in the input graph $G = (V, E)$.[3] However, this cannot be applied to the weighted case, namely Minimum Weight Non-Terminal Spanning Tree. In this subsection, we give an $O^*(2^k)$-time algorithm for Minimum Weight Non-Terminal Spanning Tree, provided that the weight function w is (positive) integral with $\max_{v \in V} w(v) = n^{O(1)}$. The algorithm runs in (pseudo-)polynomial space. Our algorithm is based on the Inclusion-Exclusion principle, which is quite useful to

[3] A slightly non-trivial dynamic programming algorithm yields a better running time bound $O^*(8^k)$.

design exact exponential algorithms [17, 10], and counts the number of admissible spanning trees for V_{NT} in G.

▶ **Theorem 20** (Inclusion-Exclusion principle). *Let U be a finite set and let $A_1, \dots, A_t \subseteq U$. Then, the following holds:*

$$|\bigcap_{i \in \{1,\dots,t\}} A_i| = \sum_{X \subseteq \{1,\dots,t\}} (-1)^{|X|} |\bigcap_{i \in X} \overline{A_i}|,$$

where $\overline{A_i} = U \setminus A_i$ and $\bigcap_{i \in \emptyset} \overline{A_i} = U$.

Let $G = (V, E)$ be a graph with $w\colon E \to \mathbb{N}$ and let $W = \max_{v \in V} w(v)$. For $0 \le q \le (n-1)W$, let $\mathcal{T}_G^q$ be the set of all spanning trees of G with weight exactly q. By a weighted counterpart of Kirchhoff's matrix tree theorem [27], we can count the number of spanning trees in $\mathcal{T}_G^q$ efficiently.

▶ **Theorem 21** ([7]). *There is an algorithm that, given an edge-weighted graph $G = (V, E)$ with $w\colon E \to \{1, 2, \dots, W\}$ for some $W \in \mathbb{N}$ and an integer q, computes the number of spanning trees T of G with $w(T) = q$. Moreover, this algorithm runs in time $(n+W)^{O(1)}$ with space $(n+W)^{O(1)}$, where $n = |V|$.*

For $v \in V_{\mathrm{NT}}$, let $\mathcal{A}_v^q \subseteq \mathcal{T}_G^q$ be the set of spanning trees of G with weight q that have v as an internal node. Clearly, the number of admissible spanning trees for V_{NT} with weight q is $|\bigcap_{v \in V_{\mathrm{NT}}} \mathcal{A}_v^q|$. Thus, we can solve Minimum Weight Non-Terminal Spanning Tree by checking if $|\bigcap_{v \in V_{\mathrm{NT}}} \mathcal{A}_v^q| > 0$ for each q. Due to Theorem 20, it suffices to compute $|\bigcap_{v \in X} \overline{\mathcal{A}_v^q}|$ for each $X \subseteq V_{\mathrm{NT}}$, where $\overline{\mathcal{A}_v^q} = \mathcal{T}_G^q \setminus \mathcal{A}_v^q$. Here, $|\bigcap_{v \in X} \overline{\mathcal{A}_v^q}|$ is equal to the number of spanning trees $T \in \mathcal{T}_G^q$ such that every vertex in X is a leaf of T.

▶ **Lemma 22.** *Given $X \subseteq V$, we can compute $|\bigcap_{v \in X} \overline{\mathcal{A}_v^q}|$ in time $(n+W)^{O(1)}$.*

Proof. Assume that $|V| \ge 3$. Then, every leaf $w \in X$ of a spanning tree T in $\bigcap_{v \in X} \overline{\mathcal{A}_v^q}$ has exactly one neighbor in $V \setminus X$. Moreover, the subtree of T obtained by removing all vertices in X is a spanning tree of $G[V \setminus X]$. Thus, the following equality holds:

$$\left|\bigcap_{v \in X} \overline{\mathcal{A}_v^q}\right| = \sum_{0 \le q' \le q} |\mathcal{T}_{G[V \setminus X]}^{q'}| \cdot |\mathcal{M}^{q-q'}(X, V \setminus X)|.$$

In the above equality, for $Y \subseteq X$ and $0 \le j \le q$, we denote by $\mathcal{M}^j(Y, V \setminus X)$ the collection of edge subsets $M \subseteq E \cap (Y \times (V \setminus X))$ with $w(M) = j$ such that each vertex in Y is incident to exactly one edge in M. By Theorem 21, we can compute $|\mathcal{T}_{G[V \setminus X]}^{q'}|$ in $(n+W)^{O(1)}$ time. Thus, it suffices to compute $|\mathcal{M}^{q-q'}(X, V \setminus X)|$ in time $(n+W)^{O(1)}$ as well. This can be done by dynamic programming described as follows. Let $X = \{x_1, x_2, \dots, x_p\}$. For $0 \le i \le p$ and $0 \le j \le q - q'$, we define $m^j(i) = |\mathcal{M}^j(\{x_1, \dots, x_i\}, V \setminus X)|$. Clearly, $m^0(0) = 1$, $m^j(0) = 0$ for $j > 0$, and $m^j(p) = |\mathcal{M}^j(X, V \setminus X)|$ for $j \ge 0$. For $i \ge 1$, it is easy to verify that

$$m^j(i) = \sum_{e \in E \cap (\{x_i\} \times (V \setminus X))} m^{j - w(e)}(i-1),$$

where we define $m^{j'}(i-1) = 0$ for negative integer j'. We can evaluate $m^j(i)$ in time $(n+W)^{O(1)}$ by dynamic programming, and hence the lemma follows. ◀

▶ **Theorem 23.** *Minimum Weight Non-Terminal Spanning Tree is solvable in time $O^*(2^k)$ and polynomial space when the edge weight function w is integral with $\max_{v \in V} w(v) = n^{O(1)}$.*

4.2 Parameterization by ℓ

In this subsection, we give a fixed-parameter algorithm for Minimum Weight Non-Terminal Spanning Tree with respect to the number of edges ℓ in $G[V_{\rm NT}]$. Note that ℓ is a "smaller" parameter than k in the sense that $\ell \le \binom{k}{2}$, while k can be arbitrary large even if $\ell = 0$. Note also that the algorithm described in this subsection works for Minimum Weight Non-Terminal Spanning Tree with an arbitrary edge weight function, whereas the algorithm in the previous subsection works only for bounded integral weight functions.

We first consider the case where $\ell = 0$. To this end, we construct a partition of $E(G)$ as follows. Let $V_{\rm NT} = \{v_1, v_2, \ldots, v_k\}$. For each $v \in V_{\rm NT}$, we let E_v be the set of edges that are incident to v in G and let $R = E(G) \setminus (E_{v_1}, E_{v_2}, \ldots, E_{v_k})$. Since $G[V_{\rm NT}]$ is edge-less, $\{E_{v_1}, E_{v_2}, \ldots, E_{v_k}, R\}$ is a partition of $E(G)$. Clearly, a spanning tree T of G is admissible for $V_{\rm NT}$ if and only if T contains at least two edges from E_v for every $v \in V_{\rm NT}$. This condition can be represented by the intersection of the following two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$. $\mathcal{M}_1 = (E, \mathcal{B}_g)$ is just a graphic matroid of G and $\mathcal{M}_2 = (E, \mathcal{B}_{n-1})$ is defined as follows: $\mathcal{B}_{n-1}$ consists of all edge subsets $F \subseteq E(G)$ with $|F| = n-1$ such that for $v \in V_{\rm NT}$, it holds that $|F \cap E_v| \ge \ell_v := 2$. Thus, a set of edges F forms an admissible spanning tree for $V_{\rm NT}$ if and only if $F \in \mathcal{B}_g \cap \mathcal{B}_{n-1}$. By Proposition 4, $\mathcal{M}_2$ is a matroid. By Theorem 5 and Lemma 6, we can find a minimum weight admissible spanning tree of G for $V_{\rm NT}$ in polynomial time.

This polynomial-time algorithm can be extended to an $O^*(2^\ell)$-time algorithm for general $\ell \ge 0$. For each $F \subseteq E(G[V_{\rm NT}])$, we compute a minimum weight admissible spanning tree T for $V_{\rm NT}$ such that $E(T) \cap E(G[V_{\rm NT}]) = F$. To this end, we first check whether F has no cycle. If F has a cycle, there is no such an admissible spanning tree for $V_{\rm NT}$. Otherwise, we modify the matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ as follows. Let G' be the multigraph obtained from G by deleting edges in $E(G[V_{\rm NT}]) \setminus F$ and then contracting edges in F. Note that the edge set of G' corresponds to $E \setminus E(G[V_{\rm NT}])$ and hence $\mathcal{E} = \{E_{v_1} \setminus E(G[V_{\rm NT}]), \ldots, E_{v_k} \setminus E(G[V_{\rm NT}]), R \setminus E(G[V_{\rm NT}])\}$ is a partition of $E(G')$. Moreover, it is easy to see that every spanning tree of G can be modified to a spanning tree of G' by contracting all edges in F and vice-versa. Now, we define $\mathcal{M}'_1 = (E(G'), \mathcal{B}_{g'})$ as the graphic matroid of G'. For $v \in V_{\rm NT}$, we let $\ell'_v = \max(0, 2 - d_F(v))$, where $d_F(v)$ is the number of edges incident to v in F. We denote by $\mathcal{M}'_2$ a pair $(E(G'), \mathcal{B}'_{n-|F|-1})$, where $\mathcal{B}'_{n-|F|-1}$ is the family of edge sets $F' \subseteq E(G')$ with $|F'| = n - |F| - 1$ such that for $v \in V_{\rm NT}$, it holds that $|F' \cap (E_v \setminus E(G[V_{\rm NT}]))| \ge \ell'_v$. This pair is indeed a matroid as $\mathcal{E}$ is a partition of $E(G')$. Then, every spanning tree T satisfying $E(T) \cap E(G[V_{\rm NT}]) = F$ is admissible for $V_{\rm NT}$ if and only if the edge set of T belongs to the intersection of the following modified matroids $\mathcal{M}'_1$ and $\mathcal{M}'_2$.

$\mathcal{M}'_2$ is a matroid consisting a pair $(E(G) \setminus E(G[V_{\rm NT}]), \mathcal{B}'_{n-|F|-1})$, where $\mathcal{B}'_{n-|F|-1}$ is Thus, every common base in $\mathcal{B}'_g \cap \mathcal{B}'_{n-|F|-1}$ corresponds to an edge set $F' \subseteq E(G) \setminus E(G[V_{\rm NT}])$ such that $F' \cup F$ forms an admissible spanning tree for $V_{\rm NT}$ in G. Again, by Theorem 5 and Lemma 6, we can compute a minimum weight admissible spanning tree T for $V_{\rm NT}$ in G with $E(T) \cap E(G[V_{\rm NT}]) = F$ in polynomial time, which yields the following theorem.

▶ **Theorem 24.** *Minimum Weight Non-Terminal Spanning Tree can be solved in time $O^*(2^\ell)$ and polynomial space.*

As a straightforward consequence of the above theorem, Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable parameterized by the number k of non-terminals.

▶ **Corollary 25.** *Minimum Weight Non-Terminal Spanning Tree can be solved in time $O^*(2^{\binom{k}{2}})$ and polynomial space.*

4.3 Parameterization by tree-width

Nakayama and Masuyama [34, 36] propose polynomial-time algorithms for Non-Terminal Spanning Tree on several subclasses of bounded tree-width graphs, such as outerplanar graphs and series-parallel graphs. In this section, we show that Non-Terminal Spanning Tree can be solved in linear time on bounded tree-width graphs.

The property of being an admissible spanning tree for $V_{\rm NT}$ can be expressed by a formula in Monadic Second Order Logic, which will be discussed below. By the celebrated work of Courcelle [9] and its optimization version [2], Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable when parameterized by tree-width.

▶ **Theorem 26.** *Minimum Weight Non-Terminal Spanning Tree can be solved in linear time on bounded tree-width graphs.*

Proof. Let $\texttt{NTST}(F)$ be an MSO_2 formula that is true if and only if $F \subseteq E(G)$ forms an admissible spanning tree for $V_{\rm NT}$. $\texttt{NTST}(F)$ can be expressed as follows:

$$\begin{aligned}\texttt{NTST}(F) &:= \texttt{connE}(F) \wedge \texttt{acyclic}(F) \wedge_{\forall v \in V, \exists e \in F} \texttt{inc}(v,e)\\ &\qquad \wedge_{\forall v \in V_{\rm NT}, \exists e_1, e_2 \in F} ((e_1 \neq e_2) \wedge \texttt{inc}(v,e_1) \wedge \texttt{inc}(v,e_2))\\ \texttt{acyclic}(F) &:= {}_{\forall F' \subseteq F}(F' \neq \emptyset \implies {}_{\exists v \in V}\texttt{deg1}(v,F')).\end{aligned}$$

$\texttt{acyclic}(F)$ is an auxiliary formula that is true if and only if F is acyclic in G. Here, $\texttt{connE}(F)$ means that the subgraph induced by an edge set F is connected and $\texttt{deg1}(v,F')$ means that the degree of vertex v is exactly 1 in the subgraph induced by an edge set F', which can be formulated in MSO_2 [10]. By the optimization version of Courcelle's theorem [2], Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable when parameterized by tree-width. ◀

5 Hardness results

In this section, we observe that s-t Hamiltonian Path is NP-hard even on several restricted classes of graphs, which immediately implies the NP-hardness of Non-Terminal Spanning Tree as well. The results in this section follow from the following observation. Let G be a graph and let v be an arbitrary vertex in G. Let G' be a graph obtained from G by adding a new vertex v' and an edge between v' and w for each $w \in N(v)$. In other words, v and v' are false twins in G'. Then, the following proposition is straightforward.

▶ **Proposition 27.** *Suppose that G has at least three vertices. Then G has a Hamiltonian cycle if and only if G' has a Hamiltonian path between v and v'.*

Proof. Let C be a Hamiltonian cycle of G and let w be one of the two vertices adjacent to v in C. As w is adjacent to v' in G', $C - \{v,w\} + \{v',w\}$ is a Hamiltonian path between v and v' in G'. Conversely, let P be a Hamiltonian path between v and v' in G'. Let w and w' be the vertices of G' that are adjacent to v and v' in P, respectively. As P is a Hamiltonian path of length at least 3, w and w' must be distinct. Moreover, both vertices are adjacent to v in G, implying that $P - \{v',w'\} + \{v,w'\}$ is a Hamiltonian cycle of G. ◀

This proposition leads to polynomial-time reductions from Hamiltonian Cycle to s-t Hamiltonian Path on several classes of graphs. As Hamiltonian Cycle is NP-hard even on strongly chordal split graphs and chordal bipartite graphs [32], the following corollary follows.

▶ **Corollary 28.** *s-t Hamiltonian Path is NP-hard even on strongly chordal split graphs and chordal bipartite graphs.*

Furthermore, since v and v' are twins, the clique-width of G' is the same as the clique-width of G. As Hamiltonian Cycle is W[1]-hard when parameterized by clique-width [16], we obtain the following corollary.

▶ **Corollary 29.** *s-t Hamiltonian Path is W[1]-hard when parameterized by clique-width.*

In [11], de Melo, de Figueiredo, and Souza show that s-t Hamiltonian Path is NP-hard even on planar graphs of maximum degree 3. The following theorem summarizes the above facts.

▶ **Theorem 30.** *Non-Terminal Spanning Tree is NP-hard even on planar bipartite graphs of maximum degree 3, strongly chordal split graphs, and chordal bipartite graphs. Furthermore, it is W[1]-hard when parameterized by clique-width.*

Furthermore, it is known that Hamiltonian Cycle cannot be solved in time $O^*(2^{o(n)})$ unless Exponential Time Hypothesis (ETH) fails [10]. Thus, we immediately obtain the following theorem.

▶ **Theorem 31.** *Non-Terminal Spanning Tree cannot be solved in time $O^*(2^{o(n)})$ unless ETH fails.*

6 Conclusion

In this paper, we studied Non-Terminal Spanning Tree and Minimum Weight Non-Terminal Spanning Tree from the viewpoint of parameterized complexity. We showed that Non-Terminal Spanning Tree admits a linear vertex kernel with respect to the number of non-terminal vertices k, as well as polynomial kernels with respect to vertex cover number and max leaf number. For the weighted counterpart, namely Minimum Weight Non-Terminal Spanning Tree, we give an $O^*(2^k)$-time algorithm for graphs with polynomially-bounded integral edge weight and $O^*(2^\ell)$-time algorithm for graphs with arbitrary edge weight, where ℓ is the number of edges in the subgraph induced by non-terminals. We proved that Minimum Weight Non-Terminal Spanning Tree is fixed-parameter tractable when parameterized by tree-width whereas it is W[1]-hard when parameterized by clique-width.

As future work, we are interested in whether Non-Terminal Spanning Tree can be solved in time $O^*((2-\epsilon)^k)$ or $O^*((2-\epsilon)^\ell)$ for some $\epsilon > 0$. Also, it would be worth considering other structural parameterizations, such as cluster deletion number or modular-width. Another possible direction is to consider the problem of finding a spanning tree that maximizes the number of internal vertices in V_{NT}. This problem simultaneously generalizes our problem and Max Internal Spanning Tree by setting $V = V_{\mathrm{NT}}$. It would be interesting to explore the (parameterized) approximability of this problem.

References

1 The graph parameter hierarchy. https://manyu.pro/assets/parameter-hierarchy.pdf. (Accessed on 27/06/2023).

2 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991. doi:10.1016/0196-6774(91)90006-K.

3 Daniel Binkele-Raible, Henning Fernau, Serge Gaspers, and Mathieu Liedloff. Exact and Parameterized Algorithms for Max Internal Spanning Tree. *Algorithmica*, 65(1):95–128, 2013. doi:10.1007/s00453-011-9575-5.

4 Andreas Björklund, Vikram Kamat, Łukasz Kowalik, and Meirav Zehavi. Spotting Trees with Few Leaves. *SIAM Journal on Discrete Mathematics*, 31(2):687–713, 2017. doi:10.1137/15M1048975.

5 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM J. Discret. Math.*, 28(1):277–305, 2014. doi:10.1137/120880240.

6 Hans L. Bodlaender, Bart M.P. Jansen, and Stefan Kratsch. Kernel bounds for path and cycle problems. *Theoretical Computer Science*, 511:117–136, 2013.

7 Andrei Z. Broder and Ernst W. Mayr. Counting minimum weight spanning trees. *J. Algorithms*, 24(1):171–176, 1997. doi:10.1006/jagm.1996.0851.

8 Zhi-Zhong Chen, Youta Harada, Fei Guo, and Lusheng Wang. An approximation algorithm for maximum internal spanning tree. *Journal of Combinatorial Optimization*, 35(3):955–979, 2018. doi:10.1007/s10878-017-0245-7.

9 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

10 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

11 Alexsander Andrade de Melo, Celina M. H. de Figueiredo, and Uéverton S. Souza. On the terminal connection problem. In Tomás Bures, Riccardo Dondi, Johann Gamper, Giovanna Guerrini, Tomasz Jurdzinski, Claus Pahl, Florian Sikora, and Prudence W. H. Wong, editors, *SOFSEM 2021: Theory and Practice of Computer Science - 47th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2021, Bolzano-Bozen, Italy, January 25-29, 2021, Proceedings*, volume 12607 of *Lecture Notes in Computer Science*, pages 278–292. Springer, 2021. doi:10.1007/978-3-030-67731-2_20.

12 Andrew Drucker. New limits to classical and quantum instance compression. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 609–618. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.71.

13 Yoshimi Egawa and Kenta Ozeki. Spanning trees with vertices having large degrees. *J. Graph Theory*, 79(3):213–221, 2015. doi:10.1002/jgt.21824.

14 Michael Fellows, Daniel Lokshtanov, Neeldhara Misra, Matthias Mnich, Frances Rosamond, and Saket Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory of Computing Systems*, 45(4):822–848, 2009.

15 Fedor V. Fomin, Serge Gaspers, Saket Saurabh, and Stéphan Thomassé. A linear vertex kernel for maximum internal spanning tree. *Journal of Computer and System Sciences*, 79(1):1–6, 2013. doi:10.1016/j.jcss.2012.03.004.

16 Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Intractability of clique-width parameterizations. *SIAM J. Comput.*, 39(5):1941–1956, 2010. doi:10.1137/080742270.

17 Fedor V. Fomin and Dieter Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2010. doi:10.1007/978-3-642-16533-7.

18 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM J. Discret. Math.*, 30(1):383–410, 2016.

19 András Frank. A weighted matroid intersection algorithm. *J. Algorithms*, 2(4):328–336, 1981. doi:10.1016/0196-6774(81)90032-8.

20 Harold N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 448–456, 1983. doi:10.1145/800061.808776.

21 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

22 Tatsuya Gima, Tesshu Hanaka, Masashi Kiyomi, Yasuaki Kobayashi, and Yota Otachi. Exploring the gap between treedepth and vertex cover through vertex integrity. *Theor. Comput. Sci.*, 918:60–76, 2022. doi:10.1016/j.tcs.2022.03.021.

23 Tatsuya Gima, Tesshu Hanaka, Yasuaki Kobayashi, Ryota Murai, Hirotaka Ono, and Yota Otachi. Structural parameterizations of vertex integrity. In Ryuhei Uehara, Katsuhisa Yamanaka, and Hsu-Chun Yen, editors, *WALCOM: Algorithms and Computation - 18th International Conference and Workshops on Algorithms and Computation, WALCOM 2024, Kanazawa, Japan, March 18-20, 2024, Proceedings*, volume 14549 of *Lecture Notes in Computer Science*, pages 406–420. Springer, 2024. doi:10.1007/978-981-97-0566-5_29.

24 Michel X. Goemans. Minimum bounded degree spanning trees. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 273–282, 2006. doi:10.1109/FOCS.2006.48.

25 Refael Hassin and Arie Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53(2):109–111, 1995.

26 Zoltán Király. Spanning tree with lower bound on the degrees. *Discret. Appl. Math.*, 242:82–88, 2018. doi:10.1016/j.dam.2017.12.005.

27 G. Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.18471481202, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.18471481202, doi:https://doi.org/10.1002/andp.18471481202.

28 Daniel J. Kleitman and Douglas B. West. Spanning trees with many leaves. *SIAM Journal on Discrete Mathematics*, 4(1):99–106, 1991.

29 Wenjun Li, Yixin Cao, Jianer Chen, and Jianxin Wang. Deeper local search for parameterized and approximation algorithms for maximum internal spanning tree. *Information and Computation*, 252:187–200, 2017. doi:10.1016/j.ic.2016.11.003.

30 Xingfu Li, Haodi Feng, Haotao Jiang, and Binhai Zhu. Solving the maximum internal spanning tree problem on interval graphs in polynomial time. *Theoretical Computer Science*, 734:32–37, 2018. Selected papers from the The Tenth International Frontiers of Algorithmics Workshop (FAW 2016). doi:10.1016/j.tcs.2017.09.017.

31 Xingfu Li, Daming Zhu, and Lusheng Wang. A $\frac{4}{3}$-approximation algorithm for the maximum internal spanning tree problem. *Journal of Computer and System Sciences*, 118:131–140, 2021. doi:10.1016/j.jcss.2021.01.001.

32 Haiko Müller. Hamiltonian circuits in chordal bipartite graphs. *Discrete Mathematics*, 156(1):291–298, 1996. doi:https://doi.org/10.1016/0012-365X(95)00057-4.

33 Shin-ichi Nakayama and Shigeru Masuyama. A linear time algorithm for finding a spanning tree with non-terminal set V_{NT} on cographs. *IEICE Trans. Inf. Syst.*, 99-D(10):2574–2584, 2016. doi:10.1587/transinf.2016EDP7021.

34 Shin-ichi Nakayama and Shigeru Masuyama. A linear time algorithm for finding a minimum spanning tree with non-terminal set V_{NT} on outerplanar graphs. *IEICE Trans. Inf. Syst.*, 100-D(3):434–443, 2017. doi:10.1587/transinf.2016FCP0010.

35 Shin-ichi Nakayama and Shigeru Masuyama. A linear-time algorithm for finding a spanning tree with non-terminal set V_{NT} on interval graphs. *IEICE Trans. Inf. Syst.*, 101-D(9):2235–2246, 2018. doi:10.1587/transinf.2018EDP7047.

36 Shin-ichi Nakayama and Shigeru Masuyama. A linear time algorithm for finding a minimum spanning tree with non-terminal set V_{NT} on series-parallel graphs. *IEICE Trans. Inf. Syst.*, 102-D(4):826–835, 2019. doi:10.1587/transinf.2018EDP7232.

37 Jesper Nederlof. Fast polynomial-space algorithms using möbius inversion: Improving on steiner tree and related problems. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009,* *Proceedings, Part I*, volume 5555 of *Lecture Notes in Computer Science*, pages 713–725. Springer, 2009. doi:10.1007/978-3-642-02927-1_59.

38 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

39 James G. Oxley. *Matroid Theory (Oxford Graduate Texts in Mathematics)*. Oxford University Press, Inc., USA, 2006.

40 Frances Rosamond. *Max Leaf Spanning Tree*, pages 1211–1215. Springer New York, New York, NY, 2016. doi:10.1007/978-1-4939-2864-4_228.

41 Gopika Sharma, Arti Pandey, and Michael C. Wigal. Algorithms for maximum internal spanning tree problem for some graph classes. *Journal of Combinatorial Optimization*, 44(5):3419–3445, 2022.

42 Yossi Shiloach. Another look at the degree constrained subgraph problem. *Inf. Process. Lett.*, 12(2):89–92, 1981. doi:10.1016/0020-0190(81)90009-0.

43 Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2):32:1–32:8, 2010. doi:10.1145/1721837.1721848.

44 Tongquan Zhang and Ying Yin. The minimum spanning tree problem with non-terminal set. *Information Processing Letters*, 112(17):688–690, 2012. doi:10.1016/j.ipl.2012.06.012.