The $k$-Leaf Spanning Tree Problem Admits a Klam Value of 39

Meirav Zehavi

Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel
meizeh@cs.technion.ac.il

Abstract. Given an undirected graph $G$ and a parameter $k$, the $k$-Leaf Spanning Tree ($k$-LST) problem asks if $G$ contains a spanning tree with at least $k$ leaves. This problem has been extensively studied over the past three decades. In 2000, Fellows et al. [FSTTCS’00] explicitly asked whether it admits a klam value of 50. A steady progress towards an affirmative answer continued until 5 years ago, when an algorithm of klam value 37 was discovered. In this paper, we present an $O^*(3.188^k)$-time parameterized algorithm for $k$-LST, which shows that the problem admits a klam value of 39. Our algorithm is based on an interesting application of the well-known bounded search trees technique, where the correctness of rules crucially depends on the history of previously applied rules in a non-standard manner.

1 Introduction

We study the well-known $k$-Leaf Spanning Tree ($k$-LST) problem. Given an undirected graph $G = (V, E)$ and a parameter $k$, it asks if $G$ contains a spanning tree with at least $k$ leaves. Due to its general nature, $k$-LST has applications in a variety of areas, including, for example, the design of ad-hoc sensor networks (see [21,22]) and computational biology (see, e.g., [18]). Furthermore, $k$-LST is tightly linked to the classic $k$-Connected Dominated Set ($k$-CDS) problem. On the one hand, given a spanning tree $T$ with at least $k$ leaves, the set of internal vertices of $T$ forms a connected dominating set of size at most $|V| - k$. On the other hand, given a connected dominating set $S$ of size at most $|V| - k$, one can construct a spanning tree $T$ with at least $k$ leaves (simply attach the vertices in $V \setminus S$ as leaves to a tree spanning the subgraph of $G$ induced by $S$).

Even in restricted settings, it has long been established that $k$-LST is NP-hard (see, e.g., [8]). Thus, over the past three decades, $k$-LST has been extensively studied in the fields of Parameterized Complexity, Exact Exponential-Time Computation and Approximation. We focus on parameterized algorithms, which attempt to solve NP-hard problems by confining the combinatorial explosion to a parameter $k$. More precisely, a problem is fixed-parameter tractable (FPT) with respect to a parameter $k$ if it can be solved in time $O^*(f(k))$ for some function $f$, where $O^*$ hides factors polynomial in the input size.
Table 1 presents a summary of FPT algorithms for $k$-LST. The *klam value* of an algorithm that runs in time $O^*((f(k)))$ is the maximal value $k$ such that $f(k) < 10^{20}$. In 2000, Fellows et al. [14] explicitly asked whether $k$-LST admits a klam value of 50. A steady progress towards an affirmative answer continued until 2010, when an algorithm of klam value 37 was discovered by Binkele-Raible and Fernau [20].

In this paper, we present a deterministic polynomial-space FPT algorithm for $k$-LST that runs in time $O^*(3.188^k)$, which shows that the problem admits a klam value of 39. Our result, like previous algorithms for this problem, is based on the bounded search trees technique (see Section 2): Essentially, when applying a branching rule, we determine the “role” of a vertex in $G$—i.e., we decide whether it should be, in the constructed tree, a leaf or an internal vertex (which, in turn, may determine roles of other vertices). Also, along with the constructed tree (to be completed to a spanning tree), we maintain a list of “floating leaves”—vertices in $G$ that are not yet attached to the constructed tree, but whose role as leaves has been already determined.

Our result makes the following interesting use of the bounded search trees technique: nodes (of a search tree) depend on the *history* of nodes that precede them in a non-standard manner—the correctness of many of our reduction and branching rules crucially relies on formerly executed branching rules, particularly on the fact that certain branches considered by them could not lead to the construction of a solution. More precisely, for certain vertices whose role is to be determined, our decision will rely on the fact that there is no solution in which their parents are leaves. Problematic vertices in whose examination we cannot rely on such a fact will be handled by a “marking” approach—we will be able to consider our treatment of them as better than it is, since we previously considered the treatment of the vertices that marked them as worse than it is.

1 The $k$-LST algorithm in [17] is incorrect (see web.stanford.edu/~rrwill/projects.html).
2 Preliminaries

**Bounded Search Trees:** Bounded search trees is a fundamental technique in the design of recursive FPT algorithms (see [11]). Roughly speaking, in applying this technique, one defines a list of rules of the form Rule X. [condition] action, where X is the number of the rule in the list. At each recursive call (i.e., a node in the search tree), the algorithm performs the action of the first rule whose condition is satisfied. If by performing an action, the algorithm recursively calls itself at least twice, the rule is a branching rule, and otherwise it is a reduction rule. We only consider polynomial time actions that increase neither the parameter nor the size of the instance, and decrease at least one of them. Observe that, at any given time, we only store the path from the current node to the root of the search tree (rather than the entire tree).

The running time of the algorithm can be bounded as follows. Suppose that the algorithm executes a branching rule where it recursively calls itself ℓ times, such that in the i-th call, the current value of the parameter decreases by \( b_i \). Then, \( (b_1, b_2, \ldots, b_\ell) \) is called the branching vector of this rule. We say that \( \alpha \) is the root of \( (b_1, b_2, \ldots, b_\ell) \) if it is the (unique) positive real root of \( x^{b^*} - b_1 + x^{b^*} - b_2 + \ldots + x^{b^*} - b_\ell \), where \( b^* = \max\{b_1, b_2, \ldots, b_\ell\} \). If \( b > 0 \) is the initial value of the parameter, and the algorithm (a) returns a result when (or before) the parameter is negative, and (b) only executes branching rules whose roots are bounded by a constant \( c \), then its running time is bounded by \( O^*(c^b) \).

### Standard Definitions and Notation:

Given a graph \( G = (V, E) \) and a vertex \( v \in V \), let \( N(v) \) denote the set of neighbors of \( v \) (in \( G \), which will be clear from context). Given subsets \( S, U \subseteq V \), let \( \text{Paths}(S, v, U) \) denote the set of paths that start from a vertex in \( S \) and end at the vertex \( v \), whose internal vertices belong to \( U \) (only). Given a rooted tree \( T = (V_T, E_T) \), let \( \text{Lea}(T), \text{Int}(T) \) and \( \text{Chi}_i(T) \) denote the leaf-set, the set of internal vertices and the set of vertices with exactly \( i \) children in \( T \), respectively. Clearly, \( \text{Lea}(T) = \text{Chi}_0(T) \). Given a vertex \( v \in V_T \), let \( \text{par}(v) \) and \( \text{Sib}(v) \) denote the parent and set of siblings of \( v \) (in \( T \), which will be clear from context), respectively.

3 The Algorithm

Clearly, we can assume WLOG that \( G \) is connected (otherwise there is no solution). Our algorithm, \( \text{Alg} \), is based on the bounded search tree technique (see Section 2), following the ideas described in the introduction. Each call to \( \text{Alg} \) is associated with an instance \( (G = (V, E), T = (V_T, E_T), L, M, F, k) \). Since \( G \) and \( k \) are always the graph and parameter given as part of the (original) input, we simplify the notation to \( (T, L, M, F) \). This corresponds to:

- A rooted subtree \( T \) of \( G \). 
- \( L \) ("fixed leaves") and \( M \) ("marked leaves") are disjoint subsets of \( \text{Lea}(T) \).

---

2 Recall that vertices \( v \) and \( u \) are siblings if they have the same parent.
Informally, \( T \) is a tree that we try to extend to a solution (a spanning tree with at least \( k \) leaves) by attaching vertices to its leaves; \( L \) contains leaves in \( T \) that should be leaves in the solution; \( M \) contains leaves in \( T \) that other vertices have “marked”, thus when their roles are decided, the measure (defined below) is decreased by a value large enough for our purpose; \( F \) contains vertices in \( G \) that are outside \( L \), but whose roles as leaves has been already determined. For the sake of clarity, we denote \( N = \text{Lea}(T) \setminus (L \cup M) \) and \( MN = M \cup N \).

Our goal is to accept the instance iff \( G \) has a spanning tree \( S = (V_S, E_S) \) that complies with \((T, L \cup F)\)—i.e., (1) \( T \) is a subtree of \( S \), (2) vertices in \( L \cup F \) are leaves in \( S \), and (3) the neighbor set of an internal vertex in \( T \) is the same as its neighbor set in \( S \). By calling Alg with \((T = (\{r\}, \emptyset), \emptyset, \emptyset, \emptyset)\) for all \( r \in V \), accepting iff at least one of the calls accepts, we clearly solve \( k\)-LST in time that is bounded by \( O^* \) of the running time of Alg.

To ensure that the running time of Alg is bounded by \( O^*(3.188^k) \), we propose the following measure:

**Measure:** \[2k + \frac{1}{4}|M| - |L| + |F| + \sum_{i \geq 2}(i - 1)|\text{Chi}_i(T)|].

Clearly, the measure is initially \( 2k \). Moreover, we show below that Alg can return a correct decision when the measure drops to (or below) \( 0 \). In particular, the measure was carefully selected to ensure that the roots of the branching vectors associated with the branching rules devised for Alg are bounded by \( 3.188^0 \). We note that one can easily observe, regardless of our rules, that the measure makes sense in the following manner: (1) Marking a vertex (i.e., inserting it to \( M \)) increases the measure, so when the vertex is handled, its treatment is considered to be better than it actually is, and (2) Determining the role of a vertex as a leaf (i.e., inserting it to \( L \cup F \)) or an internal vertex with at least two children (i.e., inserting it to \( \text{Chi}_i(T) \) for some \( i \geq 2 \)) decreases the measure by a significant value (at least \( 1 \)). When determining the role of a vertex as an internal vertex with one child, we can avoid decreasing the measure, since this decision will be made either in a reduction rule or in a branching rule where the role of another vertex, which decreases the measure, is determined.

To ensure the correctness of our rules, we will need to preserve the correctness of the following dependency claim (which formally states the dependency of a node in the search tree on the nodes preceding it):

**For all** \( v \in N \): Denote \( p = \text{par}(v) \), and let \((T', L', M', F')\) be the instance associated with the (unique) ancestor node (in the search tree) in which \( p \) was inserted to \( T \) as an internal vertex. Then, there is no solution that complies with \((\tilde{T}, L' \cup F' \cup \{p\})\), where \( \tilde{T} \) is the tree \( T' \) from which we remove the descendants of \( p \). Moreover, \(|\text{Sib}(v)| \leq 1\), and the following condition is true:

- If there is \( s \in \text{Sib}(v) \), then (i) \( s \notin M \cup \bigcup_{i \geq 2}\text{Chi}_i(T) \), and (ii) \( \text{Paths}(\text{Lea}(\tilde{T}) \setminus (L' \cup F' \cup \{p\}), s, V \setminus (V_{\tilde{T}} \cup F')) \neq \emptyset \).
Informally, for a vertex \( v \in N \), the above claim states that we had to determine (at an ancestor node) that the parent of \( v \) is an internal vertex (otherwise there is no solution), and that if \( v \) has a sibling \( s \) (at most one), then (i) \( s \notin M \) has at most one child in \( T \), and (ii) \( p \) is not the only vertex from which we could reach \( s \).

Next, we give the rules corresponding to a call \( \text{Alg}(T, L, M, F) \). Each rule is followed by an explanation (including, when relevant, the proof of the preservation of the dependency claim); for a branching rule, we also show that the root is known that for any rooted tree \( T \), \( \sum_{i \geq 2} (i-1) |\text{Chi}_i(T)| \leq 0 \), we have that \( |L \cup F| + \sum_{i \geq 2} (i-1) |\text{Chi}_i(T)| \geq 2k \). Therefore, either \( |L \cup F| \geq k \) or \( \sum_{i \geq 2} (i-1) |\text{Chi}_i(T)| \geq k \). In the former case, the rule clearly applies. Thus, now assume that \( \sum_{i \geq 2} (i-1) |\text{Chi}_i(T)| \geq k \). It is known that for any rooted tree \( T' \), \( \sum_{i \geq 2} (i-1) |\text{Chi}_i(T')| = |\text{Lea}(T')| - 2 + \delta \), where \( \delta \) is 1 if the root of \( T' \) belongs to \( \text{Chi}_1(T') \) and 0 otherwise (see, e.g., [21]). Therefore, we have that \( |\text{Lea}(T)| \geq k \), and again, the rule applies.

**Reduce 3.** \( |V = V_T| \) \( \) Reject.

In this rule, \( T \) is a spanning tree, and since the previous rule was not applied, it contains less than \( k \) leaves. Therefore, it cannot be extended to a solution, and we reject.

**Reduce 4.** \( |V \in \text{Lea}(T) \cap F| \) Return \( \text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F \setminus \{v\}) \).

3 Clearly, \( \text{Alg} \) may return a decision earlier—this can only improve the running time.
We turn a floating leaf that is a leaf in $T$ into a fixed leaf. The measure does not increase.

**Reduce 5.** There is $v \in V \setminus \left(\text{Int}(T) \cup L \cup F\right)$ such that $N(v) \setminus V_T = \emptyset$. Return $\text{Alg}(T, L, M \setminus \{v\}, F \cup \{v\})$.

We turn a vertex whose role has not yet been determined, and which does not have neighbors outside $T$, into a floating leaf (since it clearly cannot be an internal vertex in a solution). The measure decreases by at least 1.

**Reduce 6.** There are $v \in V \setminus \left(\text{Int}(T) \cup L \cup F\right)$ and $u \in MN$ s.t. $(N(v) \setminus V_T) \subseteq N(u)$. Return $\text{Alg}(T, L, M \setminus \{v\}, F \cup \{u\})$.

In this rule, there is a vertex $v$ whose role is undetermined, and whose neighbors outside the tree are also neighbors of some vertex $u \in MN$. Thus, if there is a solution $S$ that contains $v$ as an internal vertex, $v$ is not an ancestor of $u$ (since $u \in MN$) and we can disconnect its children and attach them to $u$, obtaining a solution with at least as many leaves as $S$. Thus, we can safely turn $v$ into a floating leaf. The measure decreases by at least 1.

**Reduce 7.** There are $v \in V \setminus \left(\text{Int}(T) \cup L \cup F\right)$ and $u \in MN$ such that $N(u) \setminus V_T \subseteq N(v)$ and $N(u) \setminus V_T \subseteq F$. Return $\text{Alg}(T, L, M \setminus \{v\}, F \cup \{u\})$.

In this rule, there is a vertex $v$ whose role is undetermined, along with a vertex $u \in MN$, such that all the neighbors of $u$ outside the tree are floating leaves that are also neighbors of $v$. Thus, if there is a solution $S$ that contains $u$ as an internal vertex, $u$ is not an ancestor of $v$ (since $N(u) \setminus V_T \subseteq F$) and we can disconnect its children and attach them to $v$, obtaining a solution with at least as many leaves as $S$. Thus, we can safely turn $u$ into a floating leaf. The measure decreases by at least 1.

**Reduce 8.** There are $v \in MN$ and $u \in V \setminus V_T$ such that $\text{Paths}(MN \setminus \{v\}, u, V \setminus (V_T \cup F)) = \emptyset$. Let $X = N(v) \setminus V_T$.

1. If $|X| = 1$: Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, w) : w \in X\}), L, M \setminus \{v\}, F)$.
2. Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, w) : w \in X\}), L, (M \setminus \{v\}) \cup (\text{Sib}(v) \cap N) \cup X, F)$.

In this rule, there is a vertex $v \in MN$ and a vertex $u$ outside the tree such that $v$ is the only vertex in $MN$ from which we can reach $u$ (while using neither additional vertices from $T$ nor floating leaves). Therefore, if there is a solution $S$, it contains $v$ as an internal vertex. Moreover, the vertices in $X$ are not ancestors of $v$ (since $X \cap V_T = \emptyset$ and $v \in MN$); thus, we can disconnect each of them from its parent in $S$ and attach it to $v$ as a child, obtaining a solution with at least as many leaves as $S$. This implies that we can safely turn $v$ into an internal vertex such that the vertices in $X$ are its children. In the first case, the measure clearly does not increase. In the second case, the measure both decreases by at least $(|X| - 1)$ (since $v$ is inserted to $\text{Chi}_{|X|}(T)$) and increases.
by at most $\frac{1}{4}(|X| + 1)$ (since $X \cup (\text{Sib}(v) \cap \mathcal{N})$ is inserted to $M$, where by the dependency claim, $|\text{Sib}(v) \cap \mathcal{N}| \leq 1$); therefore, the measure decreases by at least $\frac{3}{4}|X| - \frac{5}{4} \geq \frac{1}{4}$.

**Reduce 9.** [There are $v \in MN$ and $\{u\} = N(u) \setminus V_T$ such that $|N(u) \setminus V_T| = 1$]

Return $\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)$.

In this rule, there is a vertex $v \in MN$ with exactly one neighbor $u$ outside $T$, where $u$ also has only one neighbor outside $T$. Then, if there is a solution $S$ where $v$ is an internal vertex, we can disconnect $u$ from $v$ and attach the subtree of $u$ to another neighbor $w \in V \setminus (\text{Int}(T) \cup L \cup F)$ of a vertex in the subtree (since the previous rule was not applied, a vertex $w$ as required exists), obtaining a solution with at least as many leaves as $S$ (since we turned $v$ into a leaf, and we turned at most two leaves in $S$ into internal vertices—if exactly two, then $u$ was also an internal vertex in $S$ that is now a leaf). Therefore, it is safe to fix $v$ as a leaf. The measure decreases by at least $1$.

Due to lack of space, we next give a rule that demonstrates the power of the dependency claim, although we suppose that this rule is applied after Rule 9.

**Branch 10.** [There are $v, s \in \mathcal{N}$ such that $s \in \text{Sib}(v) \cap \mathcal{N}$, $|X| = |Y| = 2$, $X \cap Y = \emptyset$ and $(|X \cup Y) \cap F| \leq 1$, where $X = N(v) \setminus V_T$ and $Y = N(u) \setminus V_T$. Moreover, there is no $u \in V \setminus V_T$ such that Paths$(\mathcal{N} \setminus \{v, s\}, u, V \setminus (V_T \cup F)) = \emptyset$]

1. If $\text{Alg}(T, L \cup \{v, s\}, M, F \cup X \cup Y)$ accepts: Accept.
2. Else if $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u): u \in X\}), L \cup \{s\}, M, F)$ accepts: Accept.
3. Else if $\text{Alg}(T' = (V_T \cup Y, E_T \cup \{(s, u): u \in Y\}), L \cup \{v\}, M, F)$ accepts: Accept.
4. Return $\text{Alg}(T'' = (V_T \cup X \cup Y, E_T \cup \{(v, u): u \in X\} \cup \{(s, u): u \in Y\}), L, M, F)$.

The rule is exhaustive in the sense that we try all four options to determine the roles of $v$ and $s$. Also, recall that once a vertex is determined to be an internal vertex, we can attach each of its neighbors outside $T$ as a child (as explained in Rule 8). Thus, to prove the correctness of the rule, it suffices to show that in the first branch, inserting the vertices in $X \cup Y$ to $F$ is safe (i.e., if $(T, L \cup \{v, s\}, M, F)$ is a yes-instance, then $(T, L \cup \{v, s\}, M, F \cup X \cup Y)$ is also a yes-instance). Let $S$ be a solution to $(T, L \cup \{v, s\}, M, F)$. Suppose that there is a vertex $u \in X \cap \text{Int}(S)$. Then, we can disconnect (in $S$) the leaf $v$ from its parent and reattach it to $u$, obtaining a spanning tree $S'$ with the same number of leaves as $S$ (since $u \in \text{Int}(S)$). Next, we disconnect $s$ and reattach it (in $S'$) to another neighbor $w \in V \setminus \text{Int}(\overline{T}) \cup L' \cup F')$, where $\overline{T}$, $L'$ and $F'$ are defined as in the dependency claim (the existence of $w$ is guaranteed by the dependency claim). We thus obtain a solution $S''$ with at least as many leaves as $S'$, in which the parent of $v$ and $s$ in $T$ is a leaf. By our construction, $S''$ complies with $(\overline{T}, L' \cup F')$ (since as we progress in a certain branch, we only extend the sets Int($T$), $L$ and $F$). This contradicts the dependency claim. Thus, there is no vertex $u \in X \cap \text{Int}(S)$. Symmetrically, there is no vertex $u \in Y \cap \text{Int}(S)$. Thus, $S$ is also a solution to $(T, L \cup \{v, s\}, M, F \cup X \cup Y)$.
Next, we argue that the dependency claim holds in all branches. In the first branch, this is clearly correct. Denote \( X = \{x_1, x_2\} \) and \( Y = \{y_1, y_2\} \). Now, consider the second branch. There is no solution that complies with \((T, L \cup \{v, s\}, M, F)\), otherwise \( \text{Alg} \) would have accepted in the first branch. Moreover, \( x_1, x_2 \in \text{Lea}(T') \setminus M \) (where \( T' \) is defined in the second branch), and \( \text{Paths}(N \setminus \{v, s\}, x_1, V \setminus (V_T \cup F)), \text{Paths}(N \setminus \{v, s\}, x_2, V \setminus (V_T \cup F)) \neq \emptyset \) (this follows from the condition of the rule). Therefore, the claim holds in the second branch. Symetrically, the claim holds in the third branch. Similarly, noting that \( \text{Alg} \) did not accept in the second and third branches, the claim holds in the fourth branch.

Finally, the branching vector is at least as good as \((5, 2, 2, 2)\) since in the first branch, at least five vertices are inserted to \( L \cup F \), in each of the second and third branches, one vertex in inserted to \( \text{Chi}_2(T) \) and one vertex is inserted to \( L \), and in the fourth branch, two vertices are inserted to \( \text{Chi}_2(T) \). The root of this branching vector is at most \( 3.188^{0.5} \).

**Branch 11.** [There is \( v \in MN \) s.t. \( |X| = 2 \) and \( X \subseteq F \), where \( X = N(v) \setminus V_T \).]

Let \( Z = N(X) \setminus (\text{Int}(T) \cup L \cup F \cup \{v\}) \).

1. If \( \text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, (M \setminus \{v\}) \cup (\text{Sib}(v) \cap N), F \cup Z) \).

The rule is exhaustive in the sense that we determine that \( v \) is either a leaf or an internal vertex (in which case we can connect the neighbors of \( v \) outside \( T \) as children of \( v \)). In the second branch, we can assume that there is no solution to \((T, L \cup \{v\}, M, F)\), and thus, since Rule 8 was not applied, the correctness of inserting \( Z \) to \( F \) follows in the same manner as the correctness of the insertion of \( X \cup Y \) to \( F \) in the first branch of Rule 10. Note that the dependency claim is preserved since \( \text{Sib}(v) \cap N \) is inserted to \( M \) (in the second branch). Since Rules 7 and 8 were not applied, \( |Z| \geq 2 \). Thus, the branching vector, \((1, 1 + |Z| - \frac{1}{2} |\text{Sib}(v) \cap N|)\), is at least as good as \((1, 2 + \frac{1}{2})\), whose root is smaller than \( 3.188^{0.5} \). Observe that if \( v \in M \), the branching vector is at least as good as \((1 + \frac{1}{2}, 3)\).

We now give branching rules which determine roles of vertices in \( M \).

**Branch 12.** [There is \( v \in M \) such that \( |X| \geq 3 \), where \( X = N(v) \setminus V_T \).]

1. If \( \text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, (M \setminus \{v\}) \cup X, F) \).

In the first branch, we fix \( v \) as a leaf, while in the second branch, we turn \( v \) into an internal vertex and attach the vertices in \( X \) as its children. Recall that once a vertex is determined to be an internal vertex, we can attach each of its neighbors outside \( T \) as a child (as explained in Rule 8). Since in the second branch, the children of \( v \) in inserted to \( M \), the dependency claim remains correct.

In the first branch, the measure decreases by \( 1 + \frac{1}{2} \) (since \( v \) is moved from \( M \) to \( L \)); in the second branch, it decreases by \( 1 + \frac{1}{2}(|X| - 1) - \frac{1}{2}|X| \) (since \( v \) is moved from \( M \) to \( \text{Chi}_{|X|}(T) \), while the vertices in \( X \) are inserted to \( M \). Thus,
the branching vector is \((1\frac{1}{4}, \frac{3}{2}(|X| - 1))\), which is at least as good as \((1\frac{1}{4}, 1\frac{1}{2})\), whose root is smaller than 3.188^{0.5}.

**Branch 13.** [There is \(v \in M\) such that \(|X| = 2\), where \(X = N(v) \setminus V_T\)]

1. If \(\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)\) accepts: Accept.
2. Return \(\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \setminus \{v\}, F)\).

For correctness, follow the previous rule, noting that now the vertices in \(X\) are inserted to \(N\), while the correctness of the dependency claim holds—this is due to the fact that \(|X| = 2\), Rule 8 was not applied, and the second branch is examined only if the first branch rejected its instance. Since \(v\) is moved from \(M\) to \(L\) (first branch) or \(\text{Chi}_2(T)\) (second branch), the branching vector is \((1\frac{1}{7}, 1\frac{1}{2})\), whose root is smaller than 3.188^{0.5}.

**Branch 14.** [There is \(v \in M\) such that \(\{u\} = N(v) \setminus V_T\) and \(|X| \geq 3\), where \(X = N(u) \setminus V_T\)]

1. If \(\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)\) accepts: Accept.
2. Return \(\text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, (M \setminus \{v\}) \cup X, F)\).

This rule is similar to Rule 12 only that now, in the second branch where \(v\) is turned to an internal vertex, so does \(u\). Indeed, if there is a solution \(S\) where \(v\) is an internal vertex and \(u\) is its (only) child that is a leaf, we can disconnect \(u\) from \(v\) and attach it to another neighbor \(w \in V \setminus (\text{Int}(T) \cup L \cup F)\) (since Rule 3 was not applied, a vertex \(w\) as required exists), obtaining a solution with at least as many leaves as \(S\). Therefore, it suffices to examine (1) \(v\) as a leaf, and (2) both \(v\) and \(u\) as internal vertices. Again, the branching vector is \((1\frac{1}{4}, 1\frac{1}{2})\), whose root is smaller than 3.188^{0.5}.

**Branch 15.** [There is \(v \in M\) such that \(\{u\} = N(v) \setminus V_T\), \(|X| = 2\) and there is no \(x \in X\) for which \(\text{Paths}(MN \setminus \{v\}, x, V \setminus (V_T \cup F \cup \{u\})) = \emptyset\), where \(X = N(u) \setminus V_T\)]

1. If \(\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)\) accepts: Accept.
2. Return \(\text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \setminus \{v\}, F)\).

This rule is similar to Rule 13 only that now, in the second branch where \(v\) is turned to an internal vertex, so does \(u\) (an action whose correctness is shown in the previous rule). Noting that there is no \(x \in X\) for which \(\text{Paths}(MN \setminus \{v\}, x, V \setminus (V_T \cup F \cup \{u\})) = \emptyset\), the dependency claim is preserved as in Rule 13. Again, the branching vector is \((1\frac{1}{4}, 1\frac{1}{2})\), whose root is smaller than 3.188^{0.5}.

**Branch 16.** [There is \(v' \in M\) such that \(\{u'\} = N(v') \setminus V_T\), \(|X'| = 2\) and (for each \(x \in X'\), one of the preceding reduction rules is applicable with \(v = x\)), where \(X' = N(u') \setminus V_T\)]

1. If \(\text{Alg}(T, L \cup \{v'\}, M \setminus \{v'\}, F)\) accepts: Accept.
This rule is similar to the previous one, where now the dependency claim is preserved since $X'$ is inserted to $M$. Since in the second branch, we next apply for each $x \in X'$ a reduction rule in a manner than decreases the measure by at least $\frac{1}{4}$ (because $x \in M$, else the measure does not necessarily decrease), the branching vector is at least as good as $(1 \frac{1}{4}, 1 \frac{1}{4})$, whose root is smaller than 3.1880.5.

Branch 17. There is $v \in M$ such that $\{u\} = N(v) \setminus V_T$ and $|X| = 2$, where $X = N(u) \setminus V_T$. Let $x$ be the vertex in $X$ such that $\text{Paths}(MN, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$ and $|Y| \geq 1$, where $Y = N(x) \setminus (V_T \cup \{u\})$. Then, $|Y| \geq 2$. Let $Y = N(x) \setminus (\text{Int}(T) \cup \{u\})$.

1. If $\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)$ accepts: Accept.
2. Else if $\text{Alg}(T' = (V_T \cup \{u\}) \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}, L \cup \{x\}, (M \cup X \setminus \{v, x\}, F \cup Y)$ accepts: Accept.
3. Return $\text{Alg}(T'' = (V_T \cup \{u\}) \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\} \cup \{(x, w) : w \in Y\}, L, (M \cup X \cup Y) \setminus \{(v, x) \cup F\}, F)$ accepts: Accept.

This rule is exhaustive in the sense that we either determine that $v$ is a leaf (branch 1), or an internal vertex (branches 2 and 3), where in the latter case, we continue and determine whether $x$ is a leaf (branch 2) or an internal vertex (branch 3). As in previous rules, upon determining that a vertex is an internal vertex, we insert its neighbors outside the tree as its children. In the second branch, we can safely insert $Y$ to $F$, since otherwise, if there is a solution to the instance in this branch excluding the requirement that $Y$ is inserted to $F$, we can construct a solution to the instance in the first branch (which contradicts the fact that $\text{Alg}$ rejected it). The dependency claim is preserved since $X \setminus \{x\}$ is inserted to $M$, where in the third branch, $Y \setminus F$ is also inserted to $M$. Observe that for the vertex in $X \setminus \{x\}$, since it is inserted to $M$ and Rule 15 was not applied, we next apply a reduction rule where the measure is decreased by at least $\frac{1}{4}$. Moreover $(X \setminus \{x\}) \cap \tilde{Y} = \emptyset$ and $\tilde{Y} \setminus F \neq \emptyset$ (since $\text{Paths}(MN, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$ and Rule 15 was not applied). Therefore, the branching vector is at least as good as $(1 \frac{1}{4}, 1 \frac{1}{4} + (1 + |Y \setminus F|, (|Y| - 1) - \frac{1}{4}|Y \setminus F|))$. The worst case is obtained when $Y \subseteq Y$, $|Y| = 2$ and $|Y \setminus F| = 1$; thus, the branching vector is at least as good as $(1 \frac{1}{4}, 1 \frac{1}{4} + (2, 2, 3)) = (1 \frac{1}{4}, 3 \frac{1}{4}, 2)$, whose root is smaller than 3.1880.5.

Branch 18. There is $v \in M$ such that $\{u\} = N(v) \setminus V_T$ and $|X| = 2$, where $X = N(u) \setminus V_T$. Let $x$ be the vertex in $X$ such that $\text{Paths}(MN, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$ and $\{y\} = N(x) \setminus (V_T \cup \{x\})$. Also, let $Z = N(y) \setminus (V_T \cup \{x\})$. Then, $|Z| \geq 3$ or there is no $z \in Z$ such that $\text{Paths}(MN, z, V \setminus (V_T \cup F \cup \{u, w, x, y\})) = \emptyset$. Let $\tilde{Z} = Z \setminus F$ if $|Z| \geq 3$, and otherwise $\tilde{Z} = \emptyset$.

1. If $\text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F)$ accepts: Accept.
2. Else if \( \text{Alg}(T') = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u) \cup \{(u, w) : w \in X\}) \cup X) \cup \{v, x\} \cup \{\{y, w) : w \in Z\}, L \cup \{x\}, (M \cup X) \setminus \{v, x\} \cup \{\{y, w) : w \in Z\}, F \cup \{y\}) \) accepts: Accept.

3. Return \( \text{Alg}(T') = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u), (x, y) \cup \{(u, w) : w \in X\}) \cup \{(y, w) : w \in Z\}, L, (M \cup X \cup Z) \setminus \{v, x\}, F \) accepts: Accept.

The correctness of this rule is similar to the previous one, except that now once we determine that \( x \) is an internal vertex (in branch 3), we also determine that \( y \) is an internal vertex (this follows as argued for the insertion of \( u \) as internal vertex once we determine that \( v \) is an internal vertex). Moreover, if \( |Z| = 2 \), we do not need to insert \( Z \) to \( M \) since then, by the condition of this rule, there is no \( z \in Z \) such that \( \text{Paths}(MN \setminus \{v\}, z, V \setminus (V_T \cup F \cup \{u, x, y\}) = \emptyset \). As in the previous rule, though noting that now \( (X \setminus \{x\}) \cup \emptyset \) is inserted to \( M \) in the third branch, the branching vector is at least as good as \((1 \frac{3}{4}, 1 \frac{1}{2} + (2, (|Z| - 1) - \frac{1}{4}|Z|)) \), which is at least as good as \((1 \frac{1}{4}, 3 \frac{1}{4}, 2 \frac{1}{2}, 1) \), whose root is smaller than 3.188

**Branch 19.** There is \( v \in M \) such that \( \{u\} = N(v) \setminus V_T \) and \( |X| = 2 \), where \( X = N(u) \setminus V_T \). Let \( x \) be the vertex in \( X \) such that \( \text{Paths}(MN, x, V \setminus (V_T \cup \{u\})) = \emptyset \) and \( \{y\} = N(x) \setminus (V_T \cup \{u\}) \). Also, let \( Z = N(y) \setminus (V_T \cup \{x\}) \).

1. If \( \text{Alg}(T, L \cup \{v\}, M \setminus \{v\}, F) \) accepts: Accept.
2. Else if \( \text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u) \cup \{(u, w) : w \in X\}) \cup X) \cup \{v, x\} \cup \{\{y, w) : w \in Z\}, L \cup \{x\}, (M \cup X) \setminus \{v, x\} \cup \{\{y, w) : w \in Z\}, F \cup \{y\}) \) accepts: Accept.
3. Return \( \text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u), (x, y) \cup \{(u, w) : w \in X\}) \cup \{(y, w) : w \in Z\}, L, (M \cup X \cup Z) \setminus \{v, x\}, F \) accepts: Accept.

The correctness of this rule is similar to the previous one. The correctness of the dependency claim is preserved since now, in the third branch, we insert \( Z \) to \( M \). Observe that since the previous rule was not applied, \( |Z| = 2 \) and there is a vertex in \( Z \) such that \( \text{Paths}(MN, z, V \setminus (V_T \cup F \cup \{u, x, y\})) = \emptyset \); then, there is a vertex in \( Z \) such that in the third branch we next apply a reduction rule that decreases the measure by at least \( \frac{1}{4} \). Thus, the branching vector is at least as good as \((1 \frac{1}{4}, 1 \frac{1}{2} + (2, (|Z| - 1) - \frac{1}{4}|Z|) + \frac{1}{4})) = (1 \frac{1}{4}, 3 \frac{1}{4}, 2), \) whose root is smaller than 3.188

For now on, since Rules 9 and 11-19 were not applied, \( M = \emptyset \). We next determine the roles of any vertex in \( N \) lacking a sibling in \( N \).

The remaining rules are given in the appendix.

**References**

1. Binkele-Raible, D., Fernau, H.: A parameterized measure-and-conquer analysis for finding a k-leaf spanning tree in an undirected graph. Discrete Mathematics & Theoretical Computer Science 16(1), 179–200 (2014)

6 Observe that \(|Z| \geq 2\), else Rule 16 was applied.
2. Blum, J., Ding, M., Thaeler, A., Cheng, X.: Connected dominating set in sensor networks and MANETs. Handbook of Combinatorial Optimization B, 329–369 (2005)
3. Bodlaender, H.L.: On linear time minor tests and depth-first search. In: WADS. pp. 577–590 (1989)
4. Bonsma, P.S., Brueggemann, T., J, W.: A faster FPT algorithm for finding spanning trees with many leaves. In: MFCS. pp. 259–268 (2003)
5. Bonsma, P.S., Zickfeld, F.: Spanning trees with many leaves in graphs without diamonds and blossoms. In: LATIN. pp. 531–543 (2008)
6. Daligault, J., Gutin, G., Kim, E.J., Yeo, A.: FPT algorithms and kernels for the directed $k$-leaf problem. CoRR abs/0810.4946 (2008)
7. Daligault, J., Gutin, G., Kim, E.J., Yeo, A.: FPT algorithms and kernels for the directed $k$-leaf problem. Journal of Computer and System Sciences 76(2), 144–152 (2010)
8. Douglas, R.J.: NP-completeness and degree restricted spanning trees. Discrete Mathematics 105(1–3), 41–47 (1992)
9. Downey, R.G., Fellows, M.R.: Parameterized computational feasibility. Feasible Mathematics II pp. 219–244 (1995)
10. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, New York (1999)
11. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer (2013)
12. Estivill-Castro, V., Fellows, M.R., Langston, M., Rosamond, F.A.: FPT is P-time extremal structure I. ACiD pp. 1–41 (2005)
13. Fellows, M.R., Langston, M.: On well-partial-order theory and its applications to combinatorial problems of VLSI design. Technical Report CS-88-188, Department of Computer Science, Washington State University (1988)
14. Fellows, M.R., McCartin, C., Rosamond, F.A., Stege, U.: Coordinatized kernels and catalytic reductions: an improved FPT algorithm for max leaf spanning tree and other problems. In: FSTTCS. pp. 240–251 (2000)
15. Kneis, J., Langer, A., Rossmanith, P.: A new algorithm for finding trees with many leaves. In: ISAAC. pp. 270–281 (2008)
16. Kneis, J., Langer, A., Rossmanith, P.: A new algorithm for finding trees with many leaves. Algorithmica 61, 882–897 (2011)
17. Koutis, I., Williams, R.: Limits and applications of group algebras for parameterized problems. In: ICALP. pp. 653–664 (2009)
18. Milenković, T., Memišević, V., Bonato, A., Pržulj, N.: Dominating biological networks. PLOSone 6(8) (2011)
19. Rai, M., Verma, S., Tapaswi, S.: A power aware minimum connected dominating set for wireless sensor networks. Journal of Networks 4(6) (2009)
20. Raible, D., Fernau, H.: An amortized search tree analysis for $k$-leaf spanning tree. In: SOFSEM. pp. 672–684 (2010)
21. Raible, D., Fernau, H., Gaspers, D., Liedloff, M.: Exact and parameterized algorithms for max internal spanning tree. Algorithmica 65(1), 95–128 (2013)
22. Thai, M.T., Wang, F., Liu, D., Zhu, S., Du, D.Z.: Connected dominating sets in wireless networks different transmission ranges. IEEE Transactions on Mobile Computing 6, 721–730 (2007)
A Omitted Rules

Branch 20. [There is $v \in N$ s.t. $\text{Sib}(v) \cap N = \emptyset$ and $|X| \geq 3$, where $X = N(v) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup X)$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup (X \setminus F), F)$.

The rule is exhaustive in the sense that we determine that either $v$ is a leaf or $v$ is an internal vertex (where in the latter case, we have already established that we can connect the neighbors of $v$ outside $T$ as children of $v$—see Rule 8). Thus, it only remains to show that in the first branch, we can insert $X$ to $F$. Recall that by the dependency claim, $|\text{Sib}(v)| \leq 1$. First, suppose that either $\text{Sib}(v) = \emptyset$ or there exists $s \in \text{Sib}(v)$ such that $s \in L$. Then, it is clear that we can insert $X$ to $F$ by considering the explanation given for the first branch of Rule 10 if there is a solution where $v$ is a leaf and some vertex in $X$ is not a leaf, we can disconnect $v$ from $\text{par}(v)$ and reattach it to this vertex, disconnect the sibling of $v$ (if one exists) from $\text{par}(v)$ and reattach it to another vertex in $V \setminus (\text{Int}(T) \cup L' \cup F')$, overall obtaining a solution that contradicts the correctness of the dependency claim (in particular, $\text{par}(v)$ is a leaf in this solution).

If the supposition is not true, then by the dependency claim, we have that there exists $s \in \text{Sib}(v)$ such that $s \in \text{Chi}_1(T)$. Then, let $S$ be a solution to $(T, L \cup \{v\}, M, F)$. Assume that there is a vertex $w \in X$ that is not a leaf in $S$. We start by disconnecting the leaf $v$ from its parent and attaching it to $w$, obtaining a solution $S'$ with the same number of leaves as $S$. Next, we disconnect $s$ from $\text{par}(T)$ and reattach a vertex $q$ in its subtree to a vertex $p \in V \setminus (\text{Int}(T) \cup L' \cup F')$ (whose existence is guaranteed by the dependency claim). We thus obtain a spanning tree $S''$, where all the leaves in $S'$ are leaves in $S''$, excluding possibly $q$ and $p$. The vertex $\text{par}(v)$ is a new leaf in $S''$, and if $p$ was a leaf in $S'$, then $s$ is a new leaf in $S''$. We have that $S''$ has at least as many leaves as $S$. Thus, we obtain a solution, $S''$, where the parent of $v$ and $s$ in $T$ is a leaf. By our construction, $S''$ complies with $(\tilde{T}, L' \cup F')$. This contradicts the dependency claim. Therefore, we showed that in the first branch, it is safe to insert $X$ to $F$.

Since in the second branch the vertices in $X \setminus F$ are inserted to $M$, the correctness of the dependency claim is preserved. The branching vector is $(1 + |X \setminus F|, (|X| - 1) - \frac{1}{2}|X \setminus F|)$. At worst, $|X| = 3$ and $X \subseteq F$, which results in the branching vector $(1, 2)$, whose root is smaller than $3.188^{0.5}$.

Branch 21. [There is $v \in N$ s.t. $\text{Sib}(v) \cap N = \emptyset$ and $|X|=2$, where $X = N(v) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup X)$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M, F)$.

Again, this rule is exhaustive in the sense that we determine that either $v$ is a leaf or $v$ is an internal vertex (in which case we can connect the neighbors of $v$ outside $T$ as children of $v$). In the first branch, as in the first branch of Rule 20, we can insert $X$ to $F$. The dependency claim holds in the second branch since
Alg rejected the instance in the first branch and Rule 8 was not applied. Since Rule 11 was not applied, $X \setminus F \neq \emptyset$. Thus, the branching vector, $(1 + |X \setminus F|, 1)$, is at least as good as $(2, 1)$, whose root is smaller than $3.188^{0.5}$.

Next, note that as previous reduction rules were not applied, if $v \in N$ such that $\text{sib}(v) \cap N = \emptyset$ and $\{u\} = N(v) \setminus V_T$, then $u$ is outside $F$ and has at least two neighbors outside the tree.

**Branch 22.** [There is $v \in N$ such that $\text{sib}(v) \cap N = \emptyset$, $\{u\} = N(v) \setminus V_T$ and $|X| \geq 3$, where $X = N(u) \setminus V_T$]

1. If Alg$(T, L \cup \{v\}, M, F \cup \{u\})$ accepts: Accept.
2. Return Alg$(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \cup X, F)$.

Again, this rule is exhaustive in the sense that we determine that either $v$ is a leaf (in which case we can insert $u$ to $F$) or $v$ is an internal vertex (in which case we can connect the neighbors of $u$ outside $T$ as children of $u$). Thus, it remains to argue that in the second branch, where $v$ is an internal vertex, we can also determine that $u$ is an internal vertex. This follows from the fact that if there is a solution to $(T' = (V_T \cup \{v\}, E_T \cup \{(v, u)\}), L \cup \{u\}, M, F)$, we can disconnect $u$ from $v$ and attach it to some other neighbor in $V \setminus (\text{Int}(T) \cup L \cup F)$ (this is possible, else Rule 8 was applied), and obtain a solution for the instance in the first branch—a contradiction. The dependency claim holds in the second branch since $X$ is inserted to $M$. The branching vector, $(2, (|X| - 1 - \frac{4}{3}|X|)) = (2, \frac{2}{3}|X| - 1)$, is at least as good as $(2, \frac{5}{4})$, whose root is smaller than $3.188^{0.5}$.

**Branch 23.** [There is $v \in N$ s.t. $\text{sib}(v) \cap N = \emptyset$, $\{u\} = N(v) \setminus V_T$, and (for all $x \in X$, $\text{Paths}(N, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$), where $X = N(u) \setminus V_T$]

1. If Alg$(T, L \cup \{v\}, M, F \cup \{u\})$ accepts: Accept.
2. Return Alg$(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M, F)$.

The correctness follows similarly as in the previous rule. Since the previous rule was not applied, $|X| = 2$, and by the condition of the rule, (for all $x \in X$, $\text{Paths}(N, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$); therefore, by the order of the branches, the dependency claim is preserved. The branching vector, $(2, 1)$, has a root smaller than $3.188^{0.5}$.

**Reduce 24.** [There is $v \in N$ s.t. $\text{sib}(v) \cap N = \emptyset$ and $\{u\} = N(v) \setminus V_T$. Let $X = N(u) \setminus V_T$. Return Alg$(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \cup X, F)$.

Since the two previous rules were not applied, $|X| = 2$, and there is $x \in X$ such that $\text{Paths}(N, x, V \setminus (V_T \cup F \cup \{u\})) = \emptyset$; therefore, if there is a solution, it contains $u$ as an internal vertex. Thus, if a solution contains $v$ as a leaf, we can disconnect $v$ and reattach it to $u$, while also disconnecting the sibling of $v$ (if one exists) and reattaching its subtree to a vertex in $V \setminus (\text{Int}(T) \cup L' \cup F')$, obtaining a solution that contradicts the dependency claim (as in Rule 20). We can therefore safely determine that $v$ and $u$ are internal vertices. The measure decreases by $\frac{1}{2}$ (since $u$ is inserted to Chi$_2(T)$ and $X$ is inserted to $M$).

14
Overall, from now on, $M = \emptyset$, and for $v \in N$, we have that $|\text{Sib}(v) \cap N| = 1$. Also, the condition in Rule 10 is excluded, and any vertex in $N$ with exactly two neighbors outside $T$, has a neighbor outside $F$.

**Branch 25.** [There are $v \in N$ and $s \in \text{Sib}(v) \cap N$ such that $|N(s) \setminus V_T| = 1$ and $|X| \geq 3$, where $X = N(v) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup X)$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup (X \setminus F) \cup \{s\}, F)$.

The correctness follows by arguments similar to those in Rule 20. Similarly, noting that $s$ is inserted to $M$ (in the second branch), the preservation of the dependency claim follows. The branching vector is $(1 + |X \setminus F|, (|X| - 1) - \frac{1}{4}|X \setminus F| - \frac{1}{4})$. At worse, $|X| = 3$ and $X \subseteq F$. Thus, the vector is at least as good as $(1, \frac{13}{4})$, whose root is smaller than 3.1880.5.

**Branch 26.** [There are $v \in N$ and $s \in \text{Sib}(v) \cap N$ such that $|N(s) \setminus V_T| = 1$ and $|X| = 2$, where $X = N(v) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup X)$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup \{s\}, F)$.

The correctness follows by arguments similar to those in Rule 21. Similarly, noting that $s$ is inserted to $M$, the preservation of the dependency claim follows. The branching vector is $(1 + |X \setminus F|, 1 - \frac{1}{4})$, which is at least as good as $(2, \frac{3}{2})$, whose root is smaller than 3.1880.5.

**Branch 27.** [There are $v \in N$ and $s \in \text{Sib}(v) \cap N$ such that $|N(s) \setminus V_T| = 1$, \{u\} = N(v) \setminus V_T and $|X| \geq 3$, where $X = N(u) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup \{u\})$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \cup X \cup \{s\}, F)$.

The correctness follows by arguments similar to those in Rule 22. Similarly, noting that $s$ is inserted to $M$, the preservation of the dependency claim follows. The branching vector is $(2, (|X| - 1) - \frac{1}{4}|X| - \frac{1}{4}) = (2, \frac{3}{4}|X| - 1\frac{3}{4})$, which is at least as good as $(2, 1)$, whose root is smaller than 3.1880.5.

**Branch 28.** [There is $v \in N$ and $s \in \text{Sib}(v) \cap N$ such that $|N(s) \setminus V_T| = 1$, \{u\} = N(v) \setminus V_T and (for all $x \in X$, Paths$(N, x, V \setminus (V_T \cup F \cup \{u\})) \neq \emptyset$, where $X = N(u) \setminus V_T$]

1. If $\text{Alg}(T, L \cup \{v\}, M, F \cup \{u\})$ accepts: Accept.
2. Return $\text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \cup \{s\}, F)$.

The correctness follows by arguments similar to those in Rule 23. Similarly, noting that $s$ is inserted to $M$, the preservation of the dependency claim follows. The branching vector is $(2, 1 - \frac{1}{4}) = (2, \frac{3}{2})$, whose root is smaller than 3.1880.5.
Reduce 29. [There are \( v \in N \) and \( s \in \text{Sib}(v) \cap N \) such that \( |N(s) \setminus V_T| = 1 \) and \( \{u\} = N(v) \setminus V_T \) Let \( X = N(u) \setminus V_T \). Return \( \text{Alg}(T' = (V_T \cup \{u\} \cup X, E_T \cup \{(v, u)\} \cup \{(u, w) : w \in X\}), L, M \cup X \cup \{s\}, F) \).]

The correctness follows by arguments similar to those in Rule 24. Similarly, noting that \( s \) is inserted to \( M \), the preservation of the dependency claim follows. The measure decreases by at least \( 1 - \frac{3}{4} = \frac{1}{4} \).

For now on, if \( v \in N \), both \( v \) and its sibling have (each) at least two neighbors outside the tree (if exactly two, not both in \( F \)).

Branch 30. [There are \( v \in N \), \( s \in \text{Sib}(v) \cap N \) and \( u \in V \setminus V_T \) such that \( \text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F)) = \emptyset \) and \( |X| \geq 3 \), where \( X = N(v) \setminus V_T \) Let \( Y = N(s) \setminus V_T \).

1. If \( \text{Alg}(T' = (V_T \cup Y, E_T \cup \{(s, u) : u \in Y\}), L \cup \{v\}, M \cup Y, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup X \cup \{s\}, F) \).

This rule is exhaustive in the sense that we determine that either \( v \) is a leaf or an internal vertex, where upon determining the a vertex is an internal vertex, we insert its neighbors outside \( T \) as its children. In the first branch we determine that \( s \) is internal, since otherwise we cannot reach the vertex \( u \). Clearly, the dependency claim holds (since in the first branch, we insert \( Y \) to \( M \), and in the second branch, we insert \( X \cup \{s\} \) to \( M \)). The branching vector is \( (1 + (|Y| - 1) - \frac{1}{4}|Y|, (|X| - 1) - \frac{1}{4}(|X| + 1)) = \left( \frac{3}{4}|Y|, \frac{3}{4}|X| - \frac{5}{4} \right) \), which is at least as good as \((1.5, 1)\) whose root is smaller than 3.188056.

Branch 31. [There are \( v \in N \), \( s \in \text{Sib}(v) \cap N \) and \( u \in V \setminus V_T \) such that \( \text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F)) = \emptyset \) Let \( X = N(v) \setminus V_T \) and \( Y = N(s) \setminus V_T \).

1. If \( \text{Alg}(T' = (V_T \cup Y, E_T \cup \{(s, u) : u \in Y\}), L \cup \{v\}, M \cup Y, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup \{s\}, F) \).

For correctness of the rule and preservation of the dependency claim, follow the explanation given for the previous rule (though note that now, since \( |X| = 2 \), we do not need to insert \( X \) to \( M \) in the third branch). Since the previous rule was not applied, \( |X| = |Y| = 2 \). Now, observe that when we have a vertex in \( M \), we terminate the execution (by applying Rule 24 or 25), or apply a reduction rule where the measure decreases by at least 0.25 (note that this statement is true because we have a vertex in \( M \), otherwise some of our reduction rules may not decreases the measure), or apply a branching rule (one of the Rules \([11,12] \) whose branching vector is at worst \((1.25, 1.25)\) or a combination of \((1.25, 1.25)\) and vector whose root is smaller than 3.188056. In the first branch, we insert two vertices, \( y_1, y_2 \in Y \), to \( M \). If for the first one examined among them, \( y_1 \), we apply a branching vector whose root is \((1.25, 1.25)\), then, by the order and conditions of our rules, there is now a vertex (a neighbor of \( y_1 \)) that is reachable only from \( y_2 \), and thus we need to determine (in the first branch where \( y_1 \) is determined to be a leaf) that \( y_2 \) is an internal vertex—correspondingly, the measure decreases
by 0.25. We therefore obtain that at worst, the branching vector of this rule is 
\( (2 - 0.25 - 0.25 + t, 0.75 + (1.25 + 1.25)) \), where \( t = (1.25 + 0.25, 1.25 + 1.25) \).
That is, at worst, the branching vector of this rule is \( (3, 4, 4, 2, 2) \), whose root is smaller than \( 3.188^{0.5} \).

For now on, if \( v \in N \) and \( s \in \text{Sib}(v) \cap N \), there is no \( u \in V \setminus V_T \) such that \( \text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F)) = \emptyset \).

**Branch 32.** [There are \( v \in N \), \( s \in \text{Sib}(v) \cap N \) and \( u \in (X \cap Y) \setminus F \), where \( X = N(v) \setminus V_T \) and \( Y = N(s) \setminus V_T \) If \( |X| \geq 3 \), let \( \tilde{X} = X \setminus F \), and else \( \tilde{X} = \emptyset \). Symmetrically, denote \( \tilde{Y} \).

1. If \( \text{Alg}(T' = (V_T \cup Y, E_T \cup \{(s, u) : u \in Y\}), L \cup \{v\}, M \cup \tilde{Y}, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup \tilde{X} \cup \{s\}, F) \).

This rule is exhaustive in the sense that we determine that either \( v \) is a leaf or an internal vertex, where upon determining that a vertex is an internal vertex, we insert its neighbors outside \( T \) as its children. We need to argue that in the first branch, it is safe to determine that \( s \) is an internal vertex. To this end, it suffices to show that if there is a solution \( S \) to \( (T, L \cup \{v, s\}, M, F) \), then we reach a contradiction. Indeed, we can disconnect the leaves \( v \) and \( s \), reattaching them to \( u \) (in \( S \)), obtaining a solution with at least as many leaves as \( S \) where, in particular, \( \text{par}(v) \) is a leaf—this contradicts the dependency claim. Moreover, since we have already established that there is no \( w \in V \setminus V_T \) such that \( \text{Paths}(N \setminus \{v, s\}, w, V \setminus (V_T \cup F)) = \emptyset \), and because \( s \) is inserted to \( M \) in the second branch, the correctness of the dependency claim is preserved. The branching vector is \( (1 + (|Y| - 1) - \frac{1}{4}|\tilde{Y}|, (|X| - 1) - \frac{1}{4}|\tilde{X}| - \frac{1}{4}) \). At worse, \( |X| = |Y| = 2 \), and we obtain the branching vector \( (2, 0.75) \), whose root is smaller than \( 3.188^{0.5} \).

For now on, if \( v \in N \) and \( s \in \text{Sib}(v) \cap N \), we have that \( (N(v) \cap N(s)) \setminus V_T \subseteq F \).

**Branch 33.** [There are \( v \in N \), \( s \in \text{Sib}(v) \cap N \) such that \( |X| \geq 3 \) and \( |X \cap F| \geq 2 \), where \( X = N(v) \setminus V_T \)]

1. If \( \text{Alg}(T, L \cup \{v\}, M, F) \) accepts: Accept.
2. Return \( \text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L, M \cup (X \setminus F) \cup \{s\}, F) \).

This rule is exhaustive in the sense that we determine that either \( v \) is a leaf or an internal vertex, where upon determining that \( v \) is an internal vertex, we insert its neighbors outside \( T \) as its children. Also, since in the second branch we insert \( (X \setminus F) \cup \{s\} \) to \( M \), the dependency claim is preserved. The branching vector is \( (1, (|X| - 1) - \frac{1}{4}(X \setminus F) \cup \{s\}) = (1, |X| - \frac{1}{4}|X \setminus F| - \frac{5}{4}) \). Since \( |X| \geq 3 \) and \( |X \cap F| \geq 2 \), this branching vector is at least as good as \((1, 1.5)\), whose root is smaller than \( 3.188^{0.5} \).

**Branch 34.** [There are \( v \in N \) and \( s \in \text{Sib}(v) \cap N \) such that \( |X| \geq 3 \), where \( X = N(v) \setminus V_T \)] Let \( Y = N(s) \setminus V_T \), \( Z = X \setminus Y \) and \( \tilde{X} = X \setminus F \). If \( |Y| \geq 3 \), let \( \tilde{Y} = Y \setminus F \), and else \( \tilde{Y} = \emptyset \). Symmetrically, denote \( \tilde{Z} \).
1. If $\text{Alg}(T, L \cup \{v, s\}, M, F \cup X \cup Y)$ accepts: Accept.
2. Else if $\text{Alg}(T' = (V_T \cup X, E_T \cup \{(v, u) : u \in X\}), L \cup \{s\}, M \cup X, F)$ accepts: Accept.
3. Else if $\text{Alg}(T' = (V_T \cup Y, E_T \cup \{(s, u) : u \in Y\}), L \cup \{v\}, M \cup Y, F)$ accepts: Accept.
4. Return $\text{Alg}(T' = (V_T \cup X \cup Y, E_T \cup \{(v, u) : u \in Z\} \cup \{(s, u) : u \in Y\}), L, M \cup Z \cup Y, F)$.

The rule is exhaustive in the sense that we try all four options to determine the roles of $v$ and $s$, where once a vertex is determined to be an internal vertex, we can attach its neighbors outside $T$ as its children. Also, to see that in the first branch we can insert $\{a, b\} \rightarrow F$, follow the explanation given for the first branch in Rule 10. To obtain a good enough branching vector (explained below), in the fourth branch we attach the common neighbors outside $T$ to $s$ (thus, if $|X| = 3$, $|Y| = 2$ and there is one common neighbor, in the fourth branch, both $v$ and $s$ have two children—this implies that we do not need to insert their children to $M$). Also, since we have established that there is no $u \in V \setminus V_T$ such that Paths($N \setminus \{v, s\}, u, V \setminus (V_T \cup F)$) = $\emptyset$, the dependency claim is preserved in all branches—observe that once we set vertex with at least three children, we insert those outside $F$ to $M$.

The branching vector is $((2 + |X \cup Y|) - 1) + 1 + (|X| - 1) - \frac{1}{4}|X| - (|Y| - 1) - \frac{1}{4}|Y| = 2$ (also note that $|X| \geq 3$). Therefore, we obtain a branching vector that is at least as good as the one of Rule 10.

For now on, if $v \in N$, we have that $|N(v) \setminus V_T| = 2$.

Branch 35. [There are $v \in N, s \in \text{Sib}(v) \cap N, u \in F$ and $a, b$ s.t. $\{a, u\} = N(v) \setminus V_T$, $\{u, b\} = N(s) \setminus V_T, |X \cup Y| \geq 4, |X| \geq 2$ and $|Z| \geq 1$, where $X = N(a) \setminus (V_T \cup \{v, s, a\}), Y = N(b) \setminus (V_T \cup \{s, u, a\})$ and $Z = Y \setminus X$. Moreover, there is no $w \in Z$ s.t. Paths($N \setminus \{v, s\}, (X \setminus F), w, V \setminus (V_T \cup F \cup \{v, s, b\}) = \emptyset$] If $|Z| \geq 3$, let $Z = Z \setminus F$, and else $Z = \emptyset$.

1. If $\text{Alg}(T, L \cup \{v, s\}, M, F \cup \{a, b\})$ accepts: Accept.
2. Else if $\text{Alg}(T' = (V_T \cup \{a, u\}, E_T \cup \{(v, a), (v, u)\}), L \cup \{s\}, M, F)$ accepts: Accept.
3. Else if $\text{Alg}(T' = (V_T \cup \{u, b\}, E_T \cup \{(s, u), (s, b)\}), L \cup \{v\}, M, F)$ accepts: Accept.
4. Return $\text{Alg}(T' = (V_T \cup \{a, u, b\} \cup X \cup Y, E_T \cup \{(v, a), (v, u), (s, b)\} \cup \{(a, w) : w \in X\} \cup \{(b, w) : w \in Z\}), L, (M \cup X \cup Z) \setminus F, F)$.
branch we can insert \{a, b\} to \(F\), follow the explanation given for the first branch in Rule 10. Thus, it is enough to show that in the fourth branch, once determining that \(v\) and \(s\) are internal vertices, we can also determine that \(a\) and \(b\) are internal vertices. To this end, suppose that \(\text{Alg}\) did not accept in any of the branches, but there is a solution \(S\) to \((T' = (V_T \cup \{a, u, b\}, E_T \cup \{(v, a), (v, u), (s, b)\}), L, M, F)\). Since \(\text{Alg}\) did not accept in the fourth branch, at least one among \(a\) and \(b\) is a leaf in \(S\). Suppose that \(a\) is a leaf. Then, we can disconnect \(a\) from \(v\) and attach it to another neighbor in \(V \setminus V_T\) (the existence of a neighbor as required is guaranteed since Rule 8), while disconnecting \(u\) and attaching it to \(s\), obtaining a solution \(S'\) to the instance in third branch—a contradiction. Similarly, if \(b\) is a leaf in \(S\), we get there is a solution to the instance in the second branch—a contradiction. Therefore, it is safe to determine (in the fourth branch) that \(a\) and \(b\) are internal vertices.

Since in the fourth branch we insert \((X \setminus F) \cup \tilde{Z}\) to \(M\), and by the condition, there is no \(w \in Y \setminus X\) s.t. \(\text{Paths}((N \setminus \{v, s\}) \cup (X \setminus F), w, V \setminus (V_T \cup F \cup \{v, s, b\})) = 0\), the dependency claim is preserved. Now we analyze the branching vector. The branching vector is \((4, 2, 2, 1 + |(X \setminus Y) - 2| - \frac{1}{2}|X \setminus F| - \frac{1}{2}|Z|)\). Since \(|X \cup Y| \geq 4, |X| \geq 2\) and \(|Z| \geq 1\), this branching vector is at least as good as \((4, 2, 2, 2.25)\), whose root is smaller than 3.188.5.

**Branch 36.** [The same condition as in Rule 34 except that there is \(w \in Z\) s.t. \(\text{Paths}((N \setminus \{v, s\}) \cup (X \setminus F), w, V \setminus (V_T \cup F \cup \{v, s, b\})) = 0\).

1. If \(\text{Alg}(T' = (V_T \cup \{a, u\}, E_T \cup \{(v, a), (v, u)\}), L \cup \{s\}, M, F)\) accepts: Accept.
2. Else if \(\text{Alg}(T' = (V_T \cup \{a, b\}, E_T \cup \{(s, u), (s, b)\}), L \cup \{v\}, M, F)\) accepts: Accept.
3. Return \(\text{Alg}(T' = (V_T \cup \{a, u, b\} \cup X \cup Y, E_T \cup \{(v, a), (v, u), (s, b)\} \cup \{(a, w) : w \in X\} \cup \{(b, w) : w \in Z\}, L, (M \cup X \cup Y) \setminus F, F)\).

For correctness, we need to show that in this rule, unlike the previous one, we can skip examining the instance in the first branch. Indeed, since there is \(w \in Y \setminus X\) such that \(\text{Paths}((N \setminus \{v, s\}) \cup (X \setminus F), w, V \setminus (V_T \cup F \cup \{v, s, b\})) = 0\), once we determine that \(v\) and \(s\) are leaves, inserting \(\{a, b\}\) to \(F\) (see the first branch of the previous rule), we necessarily get a no-instance (since we cannot connect \(w\) to the constructed tree). Observe that, although now there is \(w \in Y \setminus X\) such that \(\text{Paths}((N \setminus \{v, s\}) \cup (X \setminus F), w, V \setminus (V_T \cup F \cup \{v, s, b\})) = 0\), the dependency claim is still preserved since in the fourth branch, we also insert \(Y \setminus F\) to \(M\) (rather than only \(X \setminus F\)). Since \(|X \cup Y| \geq 4\), the branching vector is at least as good as \((2, 2, 2)\), whose root is smaller than 3.188.5.

**Branch 37.** [There are \(v \in N, s \in \text{Sib}(v) \cap N, u \in F\) and \(a, b\) s.t. \(\{a, u\} = N(v) \setminus V_T, \{u, b\} = N(s) \setminus V_T\), and \(|X \cup Y| \leq 3\) or \(|X| \leq 1\) or \(Y \subseteq X\), where \(X = N(a) \setminus (V_T \cup \{v, u, b\})\) and \(Y = N(b) \setminus (V_T \cup \{s, u, a\})\).

1. If \(\text{Alg}(T, L \cup \{v, s\}, M, F \cup \{a, b\})\) accepts: Accept.
2. Else if \(\text{Alg}(T' = (V_T \cup \{a, u\}, E_T \cup \{(v, a), (v, u)\}), L \cup \{s\}, M, F)\) accepts: Accept.
3. Return $\text{Alg}(T' = (V_T \cup \{a, b\}, E_T \cup \{(s, u), (s, b)\}), L \cup \{v\}, M, F)$.

This rule is similar to Rule 35 except that now we do not examine its fourth branch. Recall that we established in Rule 35 that if $v$ and $s$ are internal vertices, so are $a$ and $b$ (where the children of $b$ do not include neighbors of $a$)—this is not possible if $Y \subseteq X$. Since $|X| \leq 1$ or $|X \cup Y| \leq 3$, we must have that one of $a$ or $b$ is a leaf or a vertex with only one child. Then, we obtain a contradiction in the same manner as in Rule 35—although now $a$, for example, might be vertex of one child rather than a leaf, the proof is similar (we possibly need to reattach a vertex in the subtree of $a$ rather than $a$). The dependency claim is clearly preserved (in particular, recall again that we have already established (after Rule 31) that there is no $w \in V \setminus V_T$ such that $\text{Paths}(N \setminus \{v, s\}, w, V \setminus (V_T \cup F)) = \emptyset$). The branching vector is $(4, 2, 2)$, whose root is smaller than $3.188^{0.5}$.

Finally, we are only left with instances where there are $v \in N$, $s \in \text{Sib}(v) \cap N$, $\{a, b\} = N(v) \setminus V_T$ and $\{c, d\} = N(s) \setminus V_T$ (a, $b$, $c$, $d$ are distinct vertices), such that $a, c \in F$ and $b, d \notin F$. Also, recall that there is no $w \in V \setminus V_T$ such that $\text{Paths}(N \setminus \{v, s\}, w, V \setminus (V_T \cup F)) = \emptyset$. These instances are handled in the two following rules.

**Branch 38.** [There are $v, s, a, b, c, d$ as described in the remark above. Moreover, there is no $u \in V \setminus V_T$ such that $\text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F \cup \{b\})) = \emptyset$ or $\text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F \cup \{d\})) = \emptyset$.]

1. If $\text{Alg}(T, L \cup \{v, s\}, M, F \cup \{b, d\})$ accepts: Accept.
2. Else if $\text{Alg}(T' = (V_T \cup \{a, b\}, E_T \cup \{(v, a), (v, b)\}), L \cup \{s, a\}, M, F \setminus \{a\})$ accepts: Accept.
3. Else if $\text{Alg}(T' = (V_T \cup \{c, d\}, E_T \cup \{(s, c), (s, d)\}), L \cup \{v, c\}, M, F \setminus \{c\})$ accepts: Accept.
4. Return $\text{Alg}(T' = (V_T \cup \{a, b, c, d\}, E_T \cup \{(v, a), (v, b), (s, c), (s, d)\}), L, M, F)$.

As in previous rules of the same form, this rule is exhaustive in the sense that we try all four options to determine the roles of $v$ and $s$, set the neighbors outside $T$ of an internal vertex as its children, and in the first branch (as in the first branch, e.g., of Rule 10), insert the neighbors outside $T$ of $v$ and $s$ to $F$. Also, as in Rule 10 the dependency claim is preserved in all branches. In the second branch, we have an instance where the only applicable next rule is one among Rules 1, 3, 5, 7, 9, 11 and 20, 24 (in particular, Rule 8 is skipped because, by the condition of the rule, there is no $u \in V \setminus V_T$ such that $\text{Paths}(N \setminus \{v, s\}, u, V \setminus (V_T \cup F \cup \{b\})) = \emptyset$). Thus, if the algorithm does not return a decision, we either apply a reduction rule where the measure decreases by at least 0.5, or a branching rule whose branching vector is at least as good as $(1, 2)$. The same claim applies for the instance in the third branch. Therefore, the branching vector is at least as good as $(4, 2 + (1, 2), 2 + (1, 2), 2) = (4, 4, 4, 3, 2)$, whose root is smaller than $3.188^{0.5}$.

**Branch 39.** [Remaining case. There are $a, b, c, d$ as described in the remark preceding the previous rule.]
1. If $\text{Alg}(T') = (V_T \cup \{a, b\}, E_T \cup \{(v, a), (v, b)\}, L \cup \{s, a\}, M, F \setminus \{a\})$ accepts: Accept.
2. Else if $\text{Alg}(T') = (V_T \cup \{c, d\}, E_T \cup \{(s, c), (s, d)\}, L \cup \{v, c\}, M, F \setminus \{c\})$ accepts: Accept.
3. Return $\text{Alg}(T' = (V_T \cup \{a, b, c, d\}, E_T \cup \{(v, a), (v, b), (s, c), (s, d)\}, L, M, F)$.

This rule is similar to the previous one, except that we do not consider its first branch. However, there is no solution to the first branch (in the previous rule), since once we determine that $v, s, b, d$ are all leaves, we cannot extend the constructed tree to a spanning tree (since now there exists $u \in V \setminus V_T$ such that $\text{Paths}(N \setminus \{v, s, b\}, u, V \setminus (V_T \cup F \cup \{b\})) = \emptyset$ or $\text{Paths}(N \setminus \{v, s, d\}, u, V \setminus (V_T \cup F \cup \{d\})) = \emptyset$). As in the previous rule, the dependency claim is preserved. The branching vector is $(2, 2, 2)$, whose root is smaller than $3.188^{0.5}$. 

21