Algorithmica (2015) 71:471–495
DOI 10.1007/s00453-013-9815-y

Linear-Time Algorithms for Tree Root Problems

Maw-Shang Chang · Ming-Tat Ko · Hsueh-I Lu

Received: 29 May 2011 / Accepted: 17 July 2013 / Published online: 7 August 2013

Abstract Let T be a tree on a set V of nodes. The *p-th power* T^p of T is the graph on V such that any two nodes u and w of V are adjacent in T^p if and only if the distance of u and w in T is at most p. Given an n-node m-edge graph G and a positive integer p, the *p-th tree root problem* asks for a tree T, if any, such that $G = T^p$. Given an n-node m-edge graph G, the *tree root problem* asks for a positive integer p and a tree T, if any, such that $G = T^p$. Kearney and Corneil gave the best previously known algorithms for both problems. Their algorithm for the former (respectively, latter) problem runs in $O(n^3)$ (respectively, $O(n^4)$) time. In this paper, we give $O(n+m)$-time algorithms for both problems.

Keywords Graph power · Graph root · Tree power · Tree root · Chordal graph · Maximal clique · Minimal node separator

A preliminary version of the paper appeared in SWAT 2006 [11].

Hsueh-I. Lu also holds joint appointments in the Graduate Institute of Networking and Multimedia and the Graduate Institute of Biomedical Electronics and Bioinformatics, National Taiwan University. Most of the work was done while Maw-Shang Chang was with Department of Computer Science and Information Engineering, National Chung Cheng University.

M.-S. Chang
Department of Computer Science and Information Engineering, Hungkuang University, Shalu District, Taichung City 43302, Taiwan
e-mail: mschang@sunrise.hk.edu.tw

M.-T. Ko
Institute of Information Science, Academia Sinica, Taipei 115, Taiwan
e-mail: mtko@iis.sinica.edu.tw

H.-I Lu (✉)
Department of Computer Science and Information Engineering, National Taiwan University, 1 Roosevelt Road, Section 4, Taipei 106, Taiwan
e-mail: hil@csie.ntu.edu.tw
url: www.csie.ntu.edu.tw/~hil

1 Introduction

Let H be a graph on a set V of nodes. The *p-th power* H^p of H is the graph on V such that any two nodes u and w of V are adjacent in H^p if and only if the distance of u and w in H is at most p. If $G = H^p$, then we say that graph H is a *p-th root* of graph G or, equivalently, G is the *p-th power* of H. Graph roots and graph powers have been extensively studied in the literature [1–6, 12, 15, 16, 18–21, 23, 26, 28, 38, 39, 41, 43, 47, 51]. See [8, Sect. 10.6] for a survey. Motwani and Sudan [40] proved that recognizing squares of graphs is NP-complete. Lau [33] showed that squares of bipartite graphs can be recognized in polynomial time and proved the NP-completeness of recognizing cubes of bipartite graphs. Lau and Corneil [34] also studied the tractability of recognizing powers of proper interval, split, and chordal graphs. Lin and Skiena [35] gave a linear-time algorithm to find square roots of planar graphs.

If $G = T^p$ for some tree T and integer p, we say that G is a p-th tree power and call tree T a *p-th tree root* of G. Given a graph G and a positive integer p, the *p-th tree root problem* asks for a tree T, if any, with $G = T^p$. Given a graph G, the *tree root problem* asks for a tree T and an integer p, if any, with $G = T^p$. Various versions of tree-root problems are associated with fundamental issues in distributed computing [32, 36] and phylogeny [7, 13, 43]. Ross and Harary [45] characterized squares of trees and showed that square tree roots, when they exist, are unique up to isomorphism. Lin and Skiena [35] gave a linear-time algorithm to recognize squares of trees. Kearney and Corneil [32] gave the best previously known algorithms for the p-th tree root problem and the tree root problem. Their algorithm for the p-th tree root problem runs in $O(n^3)$ time for any n-node m-edge graph, leading to an $O(n^4)$-time algorithm for the tree root problem. Gupta and Singh [25] gave a characterization of p-th tree powers and proposed a heuristic algorithm to construct a p-th tree root. Their algorithm runs in $O(n^3)$ time, but its correctness is not proved in the paper. It was unknown whether the p-th tree root problem can be solved in $o(n^3)$ time [32, 33]. In this paper we improve Kearney and Corneil's result [32] by giving $O(m+n)$-time algorithms for the tree root problem and the p-th tree root problem for any given p. Our results lead to the first known $O(m+n)$-time obtainable information-theoretically optimal $2n + O(\log n)$-bit succinct encoding for any n-node m-edge tree power. (See, e.g., [14, 27, 31, 37, 42, 46] for results in succinct encodings.)

The rest of the paper is organized as follows. Section 2 gives the preliminaries. Section 3 gives our $O(m+n)$-time algorithm for the $2h$-th tree root problem for any given positive integer h. Section 4 gives our $O(m+n)$-time algorithm for the $(2h+1)$-st tree root problems for any given non-negative integer h. Section 5 gives our $O(m+n)$-time algorithm for the tree root problem. Section 6 concludes the paper.

2 Preliminaries

For any set S, let $|S|$ denote the cardinality of S. All graphs in this paper are undirected and simple. Let G be a graph. Let $V(G)$ (respectively, $E(G)$) consist of the nodes (respectively, edges) of G. For any subset U of $V(G)$, let $G[U]$ denote the subgraph of G induced by U. A node is *dominating* in G if it is adjacent to all other nodes in G. Let $Dom(G)$ consist of the dominating nodes of G. $Dom(G)$ can be computed from G in $O(|V(G)| + |E(G)|)$ time by computing the degrees of all nodes of G and letting $Dom(G)$ consist of the nodes with degree $|V(G)| - 1$.

A *clique* of G is a complete subgraph of G. A clique K of G is *maximal* if $K \cup \{u\}$ for any node $u \in V(G) \setminus K$ is not a clique of G. Let $\mathcal{K}_G$ consist of the maximal cliques of G. For each node u of G, let $\mathcal{K}_G(u)$ consist of the maximal cliques of G containing u. A node u is *simplicial* in G if $|\mathcal{K}_G(u)| = 1$. For any simplicial node u of G, let $\kappa_G(u)$ denote the unique maximal clique of G that contains u.

Lemma 2.1 *For any two graphs G and H on the same node set, we have that $G = H$ if and only if $\mathcal{K}_G(u) = \mathcal{K}_H(u)$ holds for each node u.*

Proof The only-if direction is straightforward. To see the other direction, suppose that $G \neq H$. There must be two nodes u and w such that edge (u, w) belongs to exactly one of G and H. Without loss of generality, we may assume that (u, w) belongs to G but does not belong to H. At least one maximal clique of G contains both u and w. No maximal clique of H contains both u and w. We have $\mathcal{K}_G(u) \neq \mathcal{K}_H(u)$ and $\mathcal{K}_G(w) \neq \mathcal{K}_H(w)$. □

A subset S of $V(G)$ is a *separator* of a connected graph G if $G[V(G) \setminus S]$ has at least two connected components. A separator S of G is *minimal* if any proper subset of S is not a separator of G. A separator S of G is a (u, w)*-separator* of G if nodes u and w are in different connected components of $G[V(G) \setminus S]$. A (u, w)-separator S of G is *minimal* if any proper subset of S is not a (u, w)-separator of G. A *minimal node separator* of G is a minimal (u, w)-separator of G for some nodes u and w of G. A minimal separator of G has to be a minimal node separator of G. However, a minimal node separator of G is not necessarily a minimal separator of G, because a minimal (u, w)-separator of G may contain a minimal (x, y)-separator of G for some other nodes x and y. Let $\mathcal{S}_G$ consist of the minimal node separators of G.

2.1 Notation for Trees

Let T be a tree. Let $Path_T(u, w)$ denote the path of T between nodes u and w. Let $dist_T(u, w)$ denote the distance of nodes u and w in T. For any node u and any integer i, let $N_{T,i}(u)$ consist of the nodes w with $dist_T(u, w) \leq i$. Let $diam(T) = \max_{u,w \in V(T)} dist_T(u, w)$. We define $T(i)$ recursively as follows: Let $T(0) = T$. For each i with $1 \leq i \leq diam(T)/2$, let $T(i)$ be the tree obtained by deleting the leaves of $T(i - 1)$. Let $Centroid(T)$ be $T(\lfloor diam(T)/2 \rfloor)$, which is either a single node or a single edge. $Centroid(T)$ can be computed from T in linear time by iteratively replacing T with $T \setminus S$, where S consists of the leaves of T, until $T \setminus S = \emptyset$.

For any positive integer h, we say that a node u is *h-extreme* in T if

$$\min_{w \in V(T(h))} dist_T(u, w) = h.$$

Each h-extreme node of T is a leaf of T. If T^{2h} is not complete, then u is simplicial in T^{2h} if and only if u is h-extreme in T. Also, if T^{2h+1} is not complete, then u is

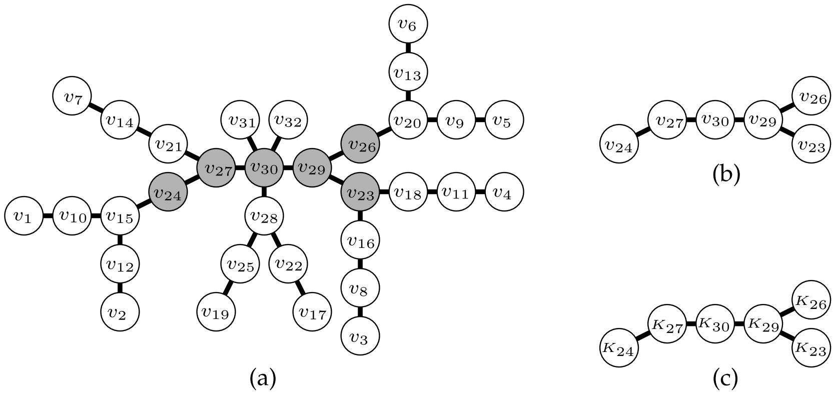

Fig. 1 (**a**) A tree T. (**b**) $T(3)$. (**c**) The clique tree of T^6, where $K_i = N_{T,3}(v_i)$

simplicial in T^{2h+1} if and only if u is $(h+1)$-extreme in T. A set U of nodes is *h-disjoint* in T if

- each node of U is h-extreme in T and
- the distance of any two distinct nodes of U in T is at least $2h$.

The empty set is h-disjoint in T for any $h \geq 1$. We use the tree T in Fig. 1(a), which appeared in [32], to illustrate the aforementioned notation: $N_{T,3}(v_{10}) = \{v_1, v_2, v_{10}, v_{12}, v_{15}, v_{24}, v_{27}\}$. $T(3)$ is the subtree of T induced by $\{v_{23}, v_{24}, v_{26}, v_{27}, v_{29}, v_{30}\}$, as shown in Fig. 1(b). All leaves of T except v_{31} and v_{32} are 3-extreme nodes of T. $\{v_3, v_4, v_7, v_{17}\}$ is a 3-disjoint node set of T. $\{v_3, v_{17}, v_{19}\}$ is a 2-disjoint node set of T but not a 3-disjoint node set of T.

2.2 Chordal Graphs

A graph G is *chordal* if it contains no induced subgraph (where nodes can be deleted but not the edges) which is a cycle of size greater than three. Chordal graphs, which can be recognized in linear time [44, 50], have been extensively studied in the literature. G is chordal if and only if each minimal node separator of G induces a clique in G [17]. Sreenivasa Kumar and Veni Madhavan [48] showed that the minimal node separators of a chordal graph can be listed in linear time, but their algorithm may list a minimal node separator more than once. We need the following lemma.

Lemma 2.2 (Chandran and Grandoni [10]) *It takes $O(m+n)$ time to list the minimal node separators of any n-node m-edge chordal graph without redundancy.*

A *clique tree* of G is a tree $\mathcal{T}$ with $V(\mathcal{T}) = \mathcal{K}_G$ such that each $\mathcal{K}_G(u)$ with $u \in V(G)$ induces a subtree of $\mathcal{T}$. For instance, Fig. 1(c) is a clique tree of graph T^6, where T is as shown in Fig. 1(a). Gavril [22] and Buneman [9] showed that graph G is chordal if and only if G has a clique tree. It takes linear time to compute a clique tree for any chordal graph (see, e.g., [30]). A chordal graph may have more than one clique tree [29]. A chordal graph is *uniquely representable* [49] if it admits a unique clique tree.

Lemma 2.3 (Sreenivasa Kumar and Veni Madhavan [49]) *A chordal graph G is uniquely representable if and only if each minimal node separator of G is contained in exactly two maximal cliques of G.*

If (K_1, K_2) is an edge of a clique tree $\mathcal{T}$ of chordal graph G, define $I(K_1, K_2)$ to be the set consisting of the nodes u of G such that (K_1, K_2) belongs to the subtree of $\mathcal{T}$ induced by $\mathcal{K}_G(u)$. That is, $I(K_1, K_2) = \{u \in V(G) \mid (K_1, K_2) \in E(\mathcal{T}[\mathcal{K}_G(u)])\}$.

Lemma 2.4 (Ho and Lee [29]) *A subset S of the nodes of a chordal graph G is a minimal node separator if and only if $S = I(K_1, K_2)$ for some edge (K_1, K_2) of $\mathcal{T}$.*

2.3 Tree Powers

Tree powers are chordal [32, 35]. The next two lemmas display dualities among the maximal cliques and minimal node separators of $2h$-th and $(2h + 1)$-st tree powers. For example, if T is as shown in Fig. 1(a), then by Lemma 2.6(1) with $h = 3$, the maximal cliques of T^6 are $N_{T,3}(v_i)$, $i = 23, 24, 26, 27, 29, 30$. By Lemma 2.6(2) with $h = 2$, the minimal node separators of T^5 are $N_{T,2}(v_i)$ with $i \in \{23, 24, 26, 27, 29, 30\}$. The bijection $N_{T,h}$ between the nodes of tree $T(h)$ and the maximal cliques of graph T^{2h}, as ensured by Lemma 2.5(1), will be shown in Lemma 2.7 to be a isomorphism between tree $T(h)$ and the unique clique tree of graph T^{2h}. For instance, if T is as shown in Fig. 1(a), then Fig. 1(c) shows the clique tree of T^6. Observe that the clique tree is isomorphic to $T(3)$, as shown in Fig. 1(b). Our tree-root algorithms are all based on this isomorphism.

Lemma 2.5 (Gupta and Singh [25]) *Let T be a tree. Let h be a non-negative integer. The following statements hold for any subset K of $V(T)$.*

1. *K is a maximal clique of T^{2h} if and only if there is a node u of $T(h)$ with $N_{T,h}(u) = K$.*
2. *K is a maximal clique of T^{2h+1} if and only if there is an edge (u, w) of $T(h)$ with $N_{T,h}(u) \cup N_{T,h}(w) = K$.*

Lemma 2.6 *Let T be a tree. Let h be a non-negative integer. The following statements hold for any subset S of $V(T)$.*

1. *S is a minimal node separator of T^{2h} if and only if there is an edge (u, w) of $T(h)$ with $N_{T,h}(u) \cap N_{T,h}(w) = S$.*
2. *S is a minimal node separator of T^{2h+1} if and only if there is a node u of $T(h+1)$ with $N_{T,h}(u) = S$.*

Proof We first prove the following claim for any clique tree $\mathcal{T}$ of any chordal graph G. *Claim 1. S is a minimal node separator of G if and only if $S = K_1 \cap K_2$ holds for some edge (K_1, K_2) of $\mathcal{T}$.* By Lemma 2.4, the claim follows from $I(K_1, K_2) = K_1 \cap K_2$, which can be proved by verifying that, under the condition that (K_1, K_2) is an edge of $\mathcal{T}$, we have (a) (K_1, K_2) is an edge of $\mathcal{T}[\mathcal{K}_G(u)]$ if and only if $\{K_1, K_2\} \subseteq \mathcal{K}_G(u)$ and (b) $\{K_1, K_2\} \subseteq \mathcal{K}_G(u)$ if and only if $u \in K_1 \cap K_2$.

Statement 1 holds trivially when $h = 0$. We assume $h \geq 1$. To prove the only-if direction of Statement 1, let S be a minimal node separator of T^{2h}. By Claim 1, there are two maximal cliques K_1 and K_2 of T^{2h}, adjacent in $\mathcal{T}$, such that $S = K_1 \cap K_2$. By Lemma 2.5(1), there are two nodes u and w of $T(h)$ such that $K_1 = N_{T,h}(u)$ and $K_2 = N_{T,h}(w)$. To see that (u, w) is indeed an edge of $T(h)$, assume for contradiction that u and w are not adjacent in T. We have $diam(T[K_1 \cap K_2]) \leq 2h - 2$. It follows that $K_1 \cap K_2$ is not even a separator of T^{2h}, contradicting the definition of S. As for the if direction of Statement 1, let $S = N_{T,h}(u) \cap N_{T,h}(w)$ for two nodes u and w of $T(h)$ that are adjacent in T. We have $diam(T[S]) = 2h - 1$ and thus S is a separator of T^{2h}. Therefore, $S \setminus \{x\}$, for any node x in S, is not a separator of T^{2h}. That is, S is a minimal separator of T^{2h}. It follows that S is a minimal node separator of T^{2h}.

To prove Statement 2, we first show the following claim. *Claim 2. For any node u of $T(h+1)$, $N_{T,h}(u)$ is a minimal separator of T^{2h+1}.* Let $S = N_{T,h}(u)$. $F = T[V(T) \setminus S]$ has at least two connected components. For any node x in F, we have $dist_T(u, x) \geq h+1$. Let x and y be nodes in two distinct connected components of F. We have $dist_T(x, y) \geq 2h + 2$, so S is an (x, y)-separator in T^{2h+1}. Let x and y be two nodes in distinct connected components of F with $dist_T(u, x) = dist_T(u, y) = h + 1$. For any node w in S, we have $dist_T(w, x) \leq 2h + 1$ and $dist_T(w, y) \leq 2h + 1$, i.e., $T^{2h+1}[V(T) \setminus (S \setminus \{w\})]$ is connected. S is a minimal separator of T^{2h+1}.

We are ready to prove Statement 2. Since a minimal separator of T^{2h+1} has to be a minimal node separator of T^{2h+1}, Claim 2 implies the if direction of Statement 2. As for the other direction, let S be a minimal node separator of T^{2h+1}. By Claim 1 and Lemma 2.5(2), there are two edges (u_1, w_1) and (u_2, w_2) of $T(h)$ such that S is the intersection of two maximal cliques $K_1 = N_{T,h}(u_1) \cup N_{T,h}(w_1)$ and $K_2 = N_{T,h}(u_2) \cup N_{T,h}(w_2)$ of T^{2h+1}. If (u_1, w_1) and (u_2, w_2) do not have a common endpoint, then there is a node w of $T(h+1)$ with $S \subsetneq N_{T,h}(w)$, contradicting the fact, by Claim 2, that $N_{T,h}(w)$ is a minimal separator. Edges (u_1, w_1) and (u_2, w_2) have a common endpoint u. Let us say $u = u_1 = u_2$. We have $S = N_{T,h}(u)$. Since (u_1, w_1) and (u_2, w_2) are edges of $T(h)$, nodes u, w_1, and w_2 are in $T(h)$. Since u has at least two neighbors w_1 and w_2 in $T(h)$, u is not a leaf of $T(h)$. Therefore, u is a node of $T(h+1)$. □

Lemma 2.7 *For any tree T and any positive integer h, T^{2h} is uniquely representable. Moreover, $T(h)$ is isomorphic to the clique tree of T^{2h} via the isomorphism $N_{T,h}$.*

Proof By Lemma 2.6(1), $N_{T,h}$ is a bijection between $V(T(h))$ and $\mathcal{K}_{T^{2h}}$. By Lemma 2.6(1), each minimal node separator of T^{2h} has the form $N_{T,h}(u) \cap N_{T,h}(w)$ for some nodes u and w adjacent in $T(h)$. Observe that $N_{T,h}(u) \cap N_{T,h}(w) \nsubseteq N_{T,h}(x)$ for any node x of $T(h)$ other than u and w. By Lemma 2.3, T^{2h} has a unique clique tree. Let $\mathcal{T}$ be the tree with

$$V(\mathcal{T}) = \{N_{T,h}(u) \mid u \in V(T(h))\};$$
$$E(\mathcal{T}) = \{(N_{T,h}(u), N_{T,h}(w)) \mid (u, w) \in E(T(h))\}.$$

$T(h)$ is isomorphic to $\mathcal{T}$ via the bijection $N_{T,h}$. By Lemma 2.6(1), we have

$$\begin{aligned}\mathcal{K}_{T^{2h}}(u) &= \{N_{T,h}(w) \mid w \in V(T(h)), dist_T(u, w) \leq h\}\\ &= \{N_{T,h}(w) \mid w \in V(T(h)) \cap N_{T,h}(u)\}.\end{aligned}$$

Since $T[V(T(h)) \cap N_{T,h}(u)]$ is a subtree of $T(h)$, $\mathcal{T}[\mathcal{K}_{T^{2h}}(u)]$ is a subtree of $\mathcal{T}$. Therefore, $\mathcal{T}$ is the clique tree of T^{2h}. □

The following lemma shows how the isomorphism between $T(h)$ and the clique tree of T^{2h} help determining the position of each node of T with respect to $T(h)$. Specifically, the pairs (K, i) with $N_{\mathcal{T},i}(K) = \mathcal{K}_{T^{2h}}(u)$ reveal the possible positions of node u in T. Such pairs will be formally defined as coordinates of u in Sect. 3.1. For instance, let T be as shown in Fig. 1(a). Node v_{24} belongs to $T(3)$. The maximal cliques of T^{2h} that contain v_{24} are K_{24}, K_{27}, K_{30}, and K_{29}. Observe that they are the nodes of $\mathcal{T}$ whose distances to K_{24} in $\mathcal{T}$ are no more than 3. Node v_{14} does not belong to $T(3)$. Node v_{27} is the node of $T(3)$ that is closest to v_{14} in T. The maximal cliques of T^{2h} that contain v_{14} are K_{24}, K_{27}, and K_{30}. Observe that they are the nodes of $\mathcal{T}$ whose distances to K_{27} in $\mathcal{T}$ are no more than $3 - dist_T(v_{14}, v_{27}) = 1$.

Lemma 2.8 *Let T be a tree. Let h be a positive integer. Let $\mathcal{T}$ be the clique tree of T^{2h}.*

1. *If u is a node of $T(h)$, then*

$$N_{\mathcal{T},h}(K) = \mathcal{K}_{T^{2h}}(u),$$

   where $K = N_{T,h}(u)$.
2. *If u is a node of T not in $T(h)$ and w is the node of $T(h)$ that is closest to u in T, then*

$$N_{\mathcal{T},h-dist_T(u,w)}(K) = \mathcal{K}_{T^{2h}}(u),$$

   where $K = N_{T,h}(w)$.

Proof Let v be a node of $T(h)$. We have $u \in N_{T,h}(v)$ if and only if $dist_T(u, v) \le h$. (1) If $u \in T(h)$, then $u \in N_{T,h}(v)$ if and only if

$$dist_{T(h)}(u, v) = dist_T(u, v) \le h.$$

By Lemma 2.7, $T(h)$ is isomorphic to the clique tree $\mathcal{T}$ of T^{2h} via the isomorphism $N_{T,h}$. Thus, $dist_{T(h)}(u, v) = dist_{\mathcal{T}}(K, N_{T,h}(v))$. $\mathcal{K}_{T^{2h}}(u)$ consists of the maximal cliques $K' = N_{T,h}(v)$ of T^{2h} that contains node u. Since K' contains u if and only if $dist_{\mathcal{T}}(K, K') \le h$, we have $\mathcal{K}_{T^{2h}}(u) = N_{\mathcal{T},h}(K)$. Statement 1 holds. (2) If $u \notin T(h)$, then $u \in N_{T,h}(v)$ if and only if

$$dist_{T(h)}(w, v) = dist_T(w, v) \le h - dist_T(u, w).$$

By Lemma 2.7, $T(h)$ is isomorphic to the clique tree $\mathcal{T}$ of T^{2h} via the isomorphism $N_{T,h}$. Thus, $dist_{T(h)}(w, v) = dist_{\mathcal{T}}(K, N_{T,h}(v))$. $\mathcal{K}_{T^{2h}}(u)$ consists of the maximal cliques $K' = N_{T,h}(v)$ of T^{2h} that contains node u. Since K' contains u if and only if $dist_{\mathcal{T}}(K, K') \le h - dist_T(u, w)$, we have $\mathcal{K}_{T^{2h}}(u) = N_{\mathcal{T},h-dist_T(u,w)}(K)$. Statement 2 holds. □

2.4 Centers of a Tree with Respect to a Set of Leaves

Definition 2.9 Let T be a tree. Let X be a set of leaves of T. A node c of T is an *X-center* of T if

$$dist_T(c, x) \geq dist_T(c, y)$$

holds for any nodes $x \in X$ and $y \in V(T)$.

For instance, let T be the tree in Fig. 1(b). If $X = \{v_{23}, v_{24}, v_{26}\}$, then v_{30} is the only X-center of T. If $X = \{v_{23}, v_{26}\}$, then v_{24}, v_{27}, and v_{30} are the X-centers of T.

Lemma 2.10 *Given a tree T and a set X of leaves of T, it takes $O(|V(T)|)$ time to output all X-centers of T.*

Proof The lemma holds trivially if T consists of one or two nodes. If X is empty, then all nodes of T are X-centers of T. The rest of the proof focuses on the case that T has at least three nodes and X is non-empty. Let x be an arbitrary node in X. Let c^* be the node in $Centroid(T)$ with maximum $dist_T(c^*, x)$. We claim that T has X-centers if and only if c^* is an X-center of T. The if direction is straightforward. As for the only-if direction, let c be an X-center of T such that among all X-centers the depth of T rooted at c is minimum. We show $c = c^*$. Since T has at least three nodes, c cannot be a leaf of T. Let T be rooted at c. Let T_i be the i-th subtree of T rooted at the children of c, where the depths of T_i are in nonincreasing order. Let d_i be the depth of T_i. If $d_1 = d_2$, then c is the unique node of $Centroid(T)$, thereby $c = c^*$. If $d_1 > d_2$, it follows from c being an X-center of T that each node of X is a leaf of T_1. If $d_1 \geq d_2 + 2$, then the root of T_1 would be an X-center of T that contradicts the choice of c. If $d_1 = d_2 + 1$, then $c = c^*$.

By the above claim, it takes $O(|V(T)|)$ time to determine whether T admits an X-center by checking whether c^* is indeed an X-center of T. The remaining X-centers can be identified by the following easily verifiable observation: A node $y \neq c^*$ of T is an X-center of T if and only if the connected component of $T \setminus \{c^*\}$ containing y does not contain any node in X. Therefore, all X-centers of T can be computed in $O(|V(T)|)$ time. □

2.5 Global Settings

Since chordal graphs can be recognized in linear time [44, 50] and it takes linear time to compute a clique tree of a chordal graph [30], the rest of the paper assumes that the input n-node and m-edge graph G is chordal and we are given a clique tree $\mathcal{T}$ of G.

3 Solving the $2h$-th Tree Root Problem

This section shows the following theorem.

Theorem 3.1 *The $2h$-th tree root problem for any n-node m-edge graph G and positive integer h can be solved in $O(n + m)$ time.*

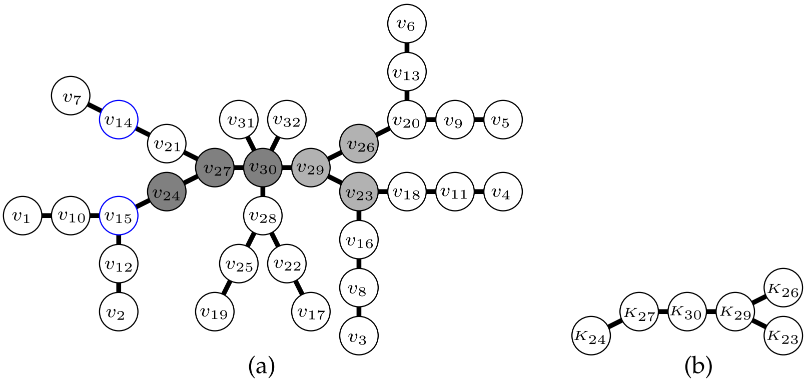

Fig. 2 (**a**) A tree T, where *gray nodes* are the nodes of $T(3)$. The nodes of $T(3)$ in $N_{T,3}(v_{14})$ are v_{24}, v_{27}, and v_{30}. The nodes of $T(3)$ in $N_{T,3}(v_{15})$ are also v_{24}, v_{27}, and v_{30}. (**b**) The clique tree of T^6

Instead of directly proving Theorem 3.1, we prove Lemma 3.2 below, which is stronger than Theorem 3.1 and is needed for the case to be investigated in Sect. 4. Theorem 3.1 follows immediately from Lemma 3.2 with $U = \emptyset$.

Lemma 3.2 *Given an n-node m-edge graph G, a positive integer h, and a subset U of the nodes of G, it takes $O(m + n)$ time to construct a $2h$-th tree root T of G, if any, such that U is h-disjoint in T.*

Section 3.1 introduces the concept of coordinates of a node. Section 3.2 shows how the problem of finding a $2h$-th tree root of G can be reduced to the problem of finding a special kind of coordinate assignment for the nodes of G. Section 3.3 gives an algorithm for finding such a coordinate assignment, if any, for G. Section 3.4 ensures that the algorithm can be implemented to run in linear time. Section 3.5 proves Lemma 3.2.

3.1 Coordinates

Let G be a chordal graph. Let $\mathcal{T}$ be a clique tree of G. Let $\Pi(K, i)$ consist of the nodes u with $N_{\mathcal{T},i}(K) = \mathcal{K}_G(u)$, i.e., the set of maximal cliques of G whose distances to maximal clique K in the clique tree $\mathcal{T}$ of G is no more than i equals the set of maximal cliques of G that contain node u of G.

Definition 3.3 (K, i) is a *coordinate* of u if $u \in \Pi(K, i)$.

s Although the definition of coordinate is not restricted to the case that G admits a $2h$-th tree root, let us use Fig. 2 to explain this crucial concept. Suppose that $G = T^6$. For each index i with $v_i \in V(T(3))$, let $K_i = N_{T,3}(v_i)$. The nodes of $T(3)$ in $N_{T,3}(v_{14})$ are v_{24}, v_{27}, and v_{30}. By Lemmas 2.7 and 2.8, we have $\mathcal{K}_G(v_{14}) = \{K_{24}, K_{27}, K_{30}\}$. Since $\mathcal{K}_G(v_{14}) = N_{\mathcal{T},1}(K_{27}) = N_{\mathcal{T},2}(K_{24})$, both $(K_{27}, 1)$ and $(K_{24}, 2)$ are coordinates of v_{14}. That is, $v_{14} \in \Pi(K_{27}, 1) \cap \Pi(K_{24}, 2)$. Similarly, we have $v_{15} \in \Pi(K_{27}, 1) \cap \Pi(K_{24}, 2)$.

Lemma 3.4 *If G admits a $2h$-th tree root T, then each h-extreme node u of T is simplicial in G and has a unique coordinate $(\kappa_G(u), 0)$.*

Proof Straightforward from Lemmas 2.6(1), and 2.7, and 2.8(2). □

Lemma 3.5 *It takes $O(m+n)$ time to compute all coordinates of all nodes in G.*

Proof Since $\mathcal{T}$ is a clique tree of G, $\mathcal{T}[\mathcal{K}_G(u)]$ is a subtree of $\mathcal{T}$ for each node u of G. Therefore, the condition $N_{\mathcal{T},i}(K) = \mathcal{K}_G(u)$ is equivalent to the condition that

$$dist_{\mathcal{T}}\big(K, K'\big) \le i \quad \text{if and only if} \quad K' \in \mathcal{K}_G(u)$$

holds for any maximal clique K' of G. Since $K' \in \mathcal{K}_G(u)$ if and only if K' is a maximal clique of G with $u \in K'$, we know that the condition $u \in \Pi(K, i)$ is equivalent to the condition that

$$dist_{\mathcal{T}}\big(K, K'\big) \le i \quad \text{if and only if} \quad u \in K' \tag{1}$$

holds for any maximal clique K' of G.

For each node u, let X_u consist of the maximal cliques $K \in \mathcal{K}_G(u)$ such that the degree of K in $\mathcal{T}[\mathcal{K}_G(u)]$ is strictly less than the degree of K in $\mathcal{T}$. For instance, for the graph $G = T^6$ where T is as shown in Fig. 2, $\mathcal{K}_G(v_{14}) = \{K_{24}, K_{27}, K_{30}\}$, in which K_{30} is the only one whose degrees in $\mathcal{T}$ and $\mathcal{T}[\mathcal{K}_G(v_{14})]$ are different. Therefore, $X_{v_{14}} = \{K_{30}\}$. For any maximal clique $K \in \mathcal{K}_G(u)$ and any integer $i \ge 0$, we claim that (K, i) is a coordinate of u if and only if K is an X_u-center of $\mathcal{T}[\mathcal{K}_G(u)]$ and

$$i = \max_{K' \in \mathcal{K}_G(u)} dist_{\mathcal{T}}\big(K, K'\big). \tag{2}$$

For instance, in Fig. 2, K_{24} and K_{27} are the $X_{v_{14}}$-centers of $\mathcal{T}[\{K_{24}, K_{27}, K_{30}\}]$ and

$$\max\big\{dist_{\mathcal{T}}(K_{24}, K_{24}), dist_{\mathcal{T}}(K_{24}, K_{27}), dist_{\mathcal{T}}(K_{24}, K_{30})\big\} = 2$$
$$\max\big\{dist_{\mathcal{T}}(K_{27}, K_{24}), dist_{\mathcal{T}}(K_{27}, K_{27}), dist_{\mathcal{T}}(K_{27}, K_{30})\big\} = 1.$$

The claim implies that $(K_{24}, 2)$ and $(K_{27}, 1)$ are the coordinates of v_{14} in $G = T^6$. To see the only-if direction of the claim, suppose that K is not an X_u-center of $\mathcal{T}[\mathcal{K}_G(u)]$. There is a maximal clique $K'' \in X_u$ with $dist_{\mathcal{T}}(K, K'') < i$. By definition of X_u, there is a maximal clique K' of G adjacent to K'' in $\mathcal{T}$ that does not belong to $\mathcal{K}_G(u)$. We have $dist_{\mathcal{T}}(K, K') \le i$. Since Condition (1) does not hold for K', (K, i) is not a coordinate of u. As for the other direction of the claim, suppose that K is an X_u-center of $\mathcal{T}[\mathcal{K}_G(u)]$. That is,

$$dist_{\mathcal{T}}\big(K, K'\big) \le dist_{\mathcal{T}}\big(K, K''\big) = i$$

holds for any $K' \in \mathcal{K}_G(u)$ and $K'' \in X_u$. By (2), we have $dist_{\mathcal{T}}(K, K') > i$ implies $u \notin K'$. By $u \in K$, we know that $u \notin K'$ implies the existence of a maximal clique K'' on $Path_{\mathcal{T}}(K, K')$ with $K'' \neq K'$ and $K'' \in X_u$. Therefore, $dist_{\mathcal{T}}(K, K') > dist_{\mathcal{T}}(K, K'') = i$. Since Condition (1) holds for any maximal clique K' of G, (K, i) is a coordinate of u.

By the above claim and Lemma 2.10, the coordinates of u can be computed from $\mathcal{T}$ and $\mathcal{K}_G(u)$ in $O(|\mathcal{K}_G(u)|)$ time. For instance, for the node v_{14} in the tree T

in Fig. 2, we obtain $\mathcal{K}_{T6}(v_{14}) = \{K_{24}, K_{27}, K_{30}\}$ and $X_{v_{14}} = \{K_{30}\}$, and then apply Lemma 2.10 to obtain the $\{K_{30}\}$-centers K_{24} and K_{27} of the subtree of $\mathcal{T}$ induced by $\{K_{24}, K_{27}, K_{30}\}$. $|\mathcal{K}_G(u)|$ is no more than $O(1)$ plus the degree of node u in G (see, e.g., [24]). Thus, all coordinates of all nodes can be computed in time $\sum_{u \in V(G)} O(|\mathcal{K}_G(u)|) = O(n+m)$. □

3.2 A Reduction to Coordinate Assignment

A (U,h)-*assignment* is a function $\Phi : V(G) \to \mathcal{K}_G \times \{0, 1, \ldots, h\}$ that satisfies the following properties, where $\Phi_1(u)$ (respectively, $\Phi_2(u)$) is the first (respectively, second) component of $\Phi(u)$ (i.e., $\Phi(u) = (\Phi_1(u), \Phi_2(u))$):

Property 1: For each node u of G, $\Phi(u)$ is a coordinate of u.
Property 2: For each maximal clique K of G, there is a unique node u of G with $\Phi(u) = (K, h)$.
Property 3: For each node u with $\Phi_2(u) < h$, there is a node w with $\Phi(w) = (\Phi_1(u), 1 + \Phi_2(u))$.
Property 4: For each maximal clique K of G and each integer j with $1 \le j < h$, the number of nodes w of G with $\Phi(w) = (K, j)$ is no less than the number of nodes u in U with $\Phi(u) = (K, 0)$.

The following lemma shows that the problem of finding a $2h$-th tree root of G in which U is h-disjoint can be reduced to the problem of finding a coordinate assignment to the nodes of G that is a (U,h)-assignment.

Lemma 3.6 *A* $2h$*-th tree root of* G *in which* U *is* h*-disjoint can be obtained from a* (U,h)*-assignment in linear time.*

Proof Let Φ be a (U,h)-assignment. We construct a tree T with $V(T) = V(G)$ via the following three steps.

Step 1: Let C consist of the nodes u of G with $\Phi_2(u) = h$. By Properties 1 and 2 of Φ, we know that Φ provides a one-to-one mapping between C and $\mathcal{K}_G$. Let $T[C]$ be the tree isomorphic to $\mathcal{T}$ via this one-to-one mapping. Since $\mathcal{T}$ is given, this step takes linear time.
Step 2: Let $u_{t,0}$ be the t-th node in U. For each $t = 1, 2, \ldots, |U|$, let $u_{t,h}$ be the node with $\Phi(u_{t,h}) = (\Phi_1(u_{t,0}), h)$. By Property 4 of Φ, we can choose in linear time a set of $(h-1)|U|$ distinct nodes $u_{t,j}$ with $1 \le t \le |U|$ and $1 \le j \le h-1$ such that $\Phi(u_{t,j}) = (\Phi_1(u_{t,0}), j)$. Now, for each $t = 1, 2, \ldots, |U|$ and $j = 0, 1, \ldots, h-1$, we add an edge between $u_{t,j}$ and $u_{t,j+1}$. This step takes linear time.
Step 3: For each node u that is still not incident to any edge, we know $0 \le \Phi_2(u) < h$. By Property 3 of Φ, there is a node w with $\Phi(w) = (\Phi_1(u), 1 + \Phi_2(u))$. We simply add an edge between u and w. This step takes linear time.

The resulting T is a tree. U is h-disjoint in T. Each path of T attached to $T[C]$ has length no more than h. The rest of the proof ensures the correctness of the resulting tree T, i.e., $T^{2h} = G$.

We first prove $T(h) = T[C]$ by showing that each leaf of $T[C]$ is attached by a length-h path in T. By definition of T, it suffices to ensure that for each leaf K of $\mathcal{T}$, there exists a node v of G with $\Phi(v) = (K, 0)$: Let K' be the maximal clique of G with $(K, K') \in E(\mathcal{T})$. By the maximality of K, there exists a node u in $K \setminus K'$. By definition of clique tree, $\mathcal{K}_G(u)$ induces a subtree of $\mathcal{T}$. Since K is a leaf of $\mathcal{T}$, it follows that $\mathcal{K}_G(u) = \{K\}$, i.e., $(K, 0)$ is the unique coordinate of u. By Property 1 of Φ, we have $\Phi(u) = (K, 0)$.

We next show that $\Phi(v) = (K, h)$ implies $N_{T,h}(v) = K$. To show $K \subseteq N_{T,h}(v)$, let u be a node in K, where $\Phi(u) = (K', i)$. We have $dist_{\mathcal{T}}(K, K') \le i$. Let w be the node of C with $\Phi(w) = (K', h)$. Since $T[C]$ is isomorphic to $\mathcal{T}$, we know $dist_T(v, w) \le i$. By Property 1 of Φ, $u \in K'$. By the definition of T, $dist_T(u, w) = h - i$. It follows that $dist_T(v, u) = dist_T(v, w) + dist_T(w, u) \le h$. Thus, $u \in N_{T,h}(v)$. To show $N_{T,h}(v) \subseteq K$, let u be a node in $N_{T,h}(v)$, where $\Phi(u) = (K', i)$. Let w be the node with $\Phi(w) = (K', h)$. By the definition of T, $dist_T(u, v) = dist_T(u, w) + dist_T(w, v) = h - i + dist_T(w, v)$. Since $dist_T(u, v) \le h$, we have $dist_T(w, v) \le i$. Since $T[C]$ is isomorphic to $\mathcal{T}$, we have that $dist_{\mathcal{T}}(K, K') \le i$. By Property 1 of Φ, $u \in \Pi(K', i)$. Hence $\mathcal{K}_G(u) = N_{\mathcal{T},i}(K')$. By $dist_{\mathcal{T}}(K, K') \le i$, we have $u \in K$.

Since $\Phi(v) = (K, h)$ implies $N_{T,h}(v) = K$, we have $\mathcal{K}_G = \{N_{T,h}(v) \mid v \in C\}$ by Property 2 of Φ. By $T[C] = T(h)$, we know $\mathcal{K}_G = \{N_{T,h}(v) \mid v \in V(T(h))\}$. By Lemma 2.6(1), K is a maximal clique of T^{2h} if and only if there exists a node u of $T(h)$ with $N_{T,h}(u) = K$. By Lemma 2.1, we know that $\mathcal{K}_G = \mathcal{K}_{T^{2h}}$ implies $G = T^{2h}$. □

3.3 An Algorithm for Coordinate Assignment

Under the assumption that G admits a $2h$-th tree root in which a given subset U of $V(G)$ is h-disjoint, this section shows how to compute a (U, h)-assignment. To simplify the description of our algorithm, each node u of G is initially white, signifying that $\Phi(u)$ is still undefined. If $\Phi(u)$ is defined but may be changed later, then u is gray. If $\Phi(u)$ is defined and will not be changed later, then u is black. Our algorithm is as shown in Algorithm 1, whose correctness is ensured by the following lemma. Let us emphasize that the algorithm does not know T.

Lemma 3.7 *If G has a $2h$-th tree root in which U is h-disjoint, then Algorithm 1 correctly computes a (U, h)-assignment.*

Proof Let T be a $2h$-th tree root of G such that U is h-disjoint. According to Lemma 2.8(2), for any node u in $V(T) \setminus V(T(h))$, we have $u \in \Pi(K, h - dist_T(u, w))$, where w is the node of $T(h)$ that is closest to u in T and $K = N_{T,h}(w)$. Therefore, $\ell(u)$ is well defined for each node $u \in V(T) \setminus V(T(h))$. Therefore, Step 9 of algorithm main does not abort. The challenge of the proof lies in showing that the algorithm does not abort at Step 5 of algorithm main or Step 7 of subroutine hook. Let us first assume that the algorithm does not abort and show that the function Φ computed by the algorithm is a (U, h)-assignment.

- Property 1. The algorithm assigns $\Phi(u) = (K, i)$ only if $u \in \Pi(K, i)$. It is also easy to see that $\Phi(u)$ is defined for every node u of G. That is, no node remains white at the end of the algorithm. Thus, Property 1 holds.

Algorithm 1 Computing a (U, h)-assignment

Input:

- a chordal graph G;
- a clique tree $\mathcal{T}$ of G;
- a subset U of $V(G)$;
- a positive integer h.

Output:

- A (U, h)-assignment Φ.

Algorithm main

1: **for** each node u of G **do**
2: let u be white and compute the coordinates of u.
3: **end for**
4: **for** each maximal clique K of G **do**
5: choose a white node $u \in \Pi(K, h)$, if any; abort, otherwise.
6: let $\Phi(u) = (K, h)$ and let u be black.
7: **end for**
8: **for** each white node u of G **do**
9: let $\ell(u)$ be the largest integer i with $i \le h - 1$ such that $u \in \Pi(K, i)$ holds for some maximal clique K of G.
10: **end for**
11: let K^* be a maximal clique of G in $V(\mathit{Centroid}(\mathcal{T}))$.
12: **for** each white node u in $\mathit{Dom}(G)$ **do**
13: let $\Phi(u) = (K^*, h - 1)$ and let u be gray.
14: **end for**
15: **for** each node u in U **do**
16: **call** hook$(u, \ell(u))$.
17: **end for**
18: **while** there are still white nodes of G **do**
19: let u be a white node of G with the smallest $\ell(u)$.
20: **call** hook$(u, \ell(u))$.
21: **end while**

Subroutine hook(u, i)

1: let K be a maximal clique of G with $u \in \Pi(K, i)$.
2: let $\Phi(u) = (K, i)$ and let u be black.
3: **for** each white node $w \in \Pi(K, i)$ **do**
4: let $\Phi(w) = (K, i)$ and let w be gray.
5: **end for**
6: **for** $j = i + 1$ to $h - 1$ **do**
7: choose a non-black node $w \in \Pi(K, j)$, if any; abort, otherwise.
8: let $\Phi(w) = (K, j)$ and let w be black.
9: **for** each white node $w \in \Pi(K, j)$ **do**
10: let $\Phi(w) = (K, j)$ and let w be gray.
11: **end for**
12: **end for**

- Property 2. Since the algorithm does not abort at Step 5 of algorithm main, the algorithm successfully assigns coordinates for $|\mathcal{K}_G|$ nodes at Steps 4–7 of algorithm main. Since the rest of the algorithm never assigns (K, h) to any $\Phi(u)$, Property 2 holds.
- Property 3. By Property 2 of Φ, we know that Property 3 holds for any node u with $\Phi_2(u) = h - 1$. Observe that only Steps 4 and 8 of subroutine hook could make

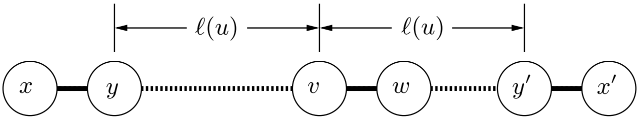

Fig. 3 An illustration for showing that u has only one coordinate

$\Phi_2(x) \le h-2$ for a node x. If so, then the next iteration of the for-loop at Step 8 assigns $\Phi_2(y) = \Phi_2(x)+1$ for some node y and turns y to black.

- Property 4. Each node u in U is an h-extreme node of T. By Lemma 3.4, u has a unique coordinate $(\kappa_G(u), 0)$. By Property 1, we have $\Phi(u) = (\kappa_G(u), 0)$. The subroutine call hook$(u, \ell(u))$ at Step 16 yields one black node w with $\Phi(w) = (\kappa_G(u), j)$ for each $j = 1, \ldots, h-1$. Therefore, Property 4 holds.

The rest of the proof shows that the algorithm does not abort. By Lemma 2.8(1), each node u of $T(h)$ belongs to $\Pi(N_{T,h}(u), h)$. For each maximal clique K of G, the number of maximal cliques K' of G with $N_{\mathcal{T},h}(K') = N_{\mathcal{T},h}(K)$ is no more than $|\Pi(K,h)|$. Therefore, the algorithm does not abort at Step 5 of algorithm main.

To prove that the algorithm does not abort at Step 7 of subroutine hook, we first show that if Step 7 of subroutine hook is reached, then node u has a unique coordinate. We can focus only on the case with $1 \le \ell(u) \le h-2$, because (a) if $\ell(u) = 0$, then, by Lemma 3.4, u has a unique coordinate, and (b) if $\ell(u) \ge h-1$, then the algorithm does not enter the for-loop. By $\ell(u) \ge 1$, we know that K cannot be a leaf of $\mathcal{T}$. The reason is that if K is a leaf of $\mathcal{T}$, then T has at least one leaf whose unique coordinate is $(K, 0)$. Since the loops at Steps 15–21 of algorithm main process nodes u in non-decreasing order of $\ell(u)$, $\ell(u) \ge 1$ implies that u cannot be white at the moment hook$(u, \ell(u))$ is called. Let v be the node of $T(h)$ such that $K = N_{T,h}(v)$. Since K is not a leaf of $\mathcal{T}$, node v is not a leaf of $T(h)$. See Fig. 3 for an illustration for the proof. Let $S = N_{T(h),\ell(u)}(v)$. Since $\ell(u) \ge 1$ and $u \notin \mathit{Dom}(G)$, there is an edge (x, y) of $T(h)$ such that $y \in S \setminus \{v\}$ and $x \notin S$. Since v is not a leaf of $T(h)$, there has to be a neighbor w of v in $T(h)$ such that $\mathit{Path}_{T(h)}(w, y)$ contains v. Since $\ell(u) \le h-2$, we know $N_{T(h),\ell(u)+1}(w) \ne S$. There has to be an edge (x', y') of $T(h)$ such that $y' \in S$, $x' \notin S$, and $\mathit{Path}_{T(h)}(x', x)$ contains y, y', w, and v. By the existence of edges (x, y) and (x', y') of $T(h)$, we know that $S = N_{T(h),j}(z)$ implies $z = v$ and $j = \ell(u)$. Thus, $(K, \ell(u))$ is the unique coordinate of u.

Let u_i be the node u for the i-th subroutine call hook$(u, \ell(u))$ that Step 7 of subroutine hook is reached throughout the execution. Let K_i be maximal clique of G such that $(K_i, \ell(u_i))$ is the unique coordinate of u_i. Let v_i be the node with $K_i = N_{T,h}(v_i)$. If $u_i \in U$, then the algorithm does not abort by the h-disjointness of U in T. If $u_i \notin U$, we have $K_j \ne K_i$ for all $j < i$, since otherwise u_i cannot be white at the moment hook$(u_i, \ell(u_i))$ is called. It follows that $\mathit{Path}_T(u_i, v_i)$ and $\mathit{Path}_T(u_j, v_j)$ are disjoint. Therefore, the number of indices i such that $\mathit{Path}_T(u_i, v_i)$ contains nodes in any $\Pi(\kappa, \ell)$ is no more than the cardinality of $\Pi(\kappa, \ell)$. It follows that the algorithm does not abort at Step 7 of subroutine hook. □

By Lemmas 3.6 and 3.7, G has a $2h$-th tree root in which U is h-disjoint if and only if G admits a (U, h)-assignment. Therefore, a $2h$-th tree power can be characterized by whether it admits a $(\emptyset, h)$-assignment.

3.4 Time Complexity

The following lemma ensures that it takes linear time for Algorithm 1 to abort or finish its execution.

Lemma 3.8 *Given an n-node m-edge chordal graph G together with a clique tree $\mathcal{T}$ of G, Algorithm 1 can be implemented to run in $O(m+n)$ time. Moreover, it takes $O(m+n)$ time to verify whether the output of Algorithm 1 is indeed a (U, h)-assignment.*

Proof By Lemma 3.5, Steps 1–3 of algorithm main take $O(m+n)$ time. After obtaining all coordinates of all nodes, Steps 4–14 of algorithm main can easily be implemented to run in $O(m+n)$ time. By verifying that each node changes color $O(1)$ times throughout the execution of Algorithm 1, it is also not difficult to see that Steps 15–21 of algorithm main run in $O(m+n)$ time. Therefore, Algorithm 1 runs in $O(m+n)$ time. Since we already have all coordinates of all nodes and the coordinate assigned to each node by Φ, it takes $O(m+n)$ time to verify whether the output is a (U, h)-assignment. □

3.5 Proving Lemma 3.2

Proof If G admits a $2h$-th tree root in which U is h-disjoint, it follows from Lemmas 3.7 and 3.8 that Algorithm 1 computes a (U, h)-assignment Φ in $O(m+n)$ time. By Lemma 3.6, a $2h$-th tree root T of G in which U is h-disjoint can be computed from Φ in $O(m+n)$ time.

If G does not admit any $2h$-th tree root in which U is h-disjoint, it follows from Lemma 3.6 that G does not admit any (U, h)-assignment. By Lemma 3.8, Algorithm 1 either aborts or finishes in $O(m+n)$ time. If Algorithm 1 does not abort, by Lemma 3.8, it takes $O(m+n)$ time to detect that the resulting Φ is not a (U, h)-assignment. □

4 Solving the $(2h+1)$-st Tree Root Problem

Since the problem is trivial when $h = 0$ or the input graph is complete, this section focuses on the case that h is a positive integer and the input graph is not complete. Section 4.1 shows a linear-time reduction from the problem of recognizing $(2h+1)$-st tree powers to that of recognizing $2h$-th tree powers. Section 4.2 solves the $(2h+1)$-st tree root problem in linear time.

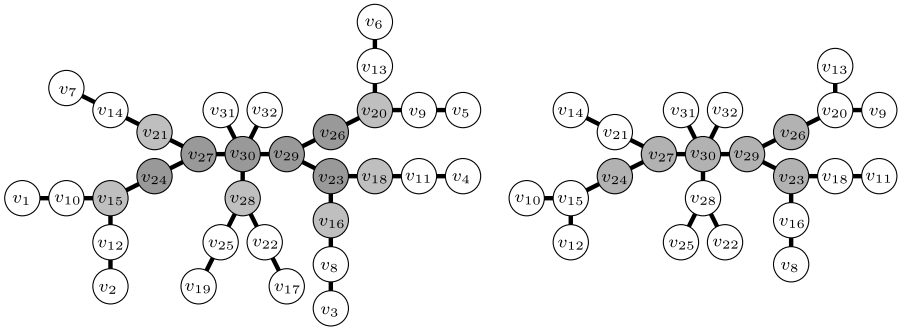

Fig. 4 (**a**) A tree T, where the nodes of $T(2)$ are in *gray*. (**b**) The 3-reduced tree R of T, where the nodes of $R(2) = T(3)$ are in *gray*

4.1 A Linear-Time Reduction

The $(h+1)$-*reduced tree* of tree T is the tree obtained from T by deleting its $(h+1)$-extreme nodes. For instance, the tree R in Fig. 4(b) is the 3-reduced tree of the tree T in Fig. 4(a).

Lemma 4.1 *Let h be a positive integer. Let T and $\hat{T}$ be trees on the same node set such that neither of T^{2h+1} and $\hat{T}^{2h+1}$ is complete. Let R be the $(h+1)$-reduced tree of T. Let $\hat{R}$ be the $(h+1)$-reduced tree of $\hat{T}$. Then, $T^{2h+1} = \hat{T}^{2h+1}$ if and only if*

1. $R^{2h} = \hat{R}^{2h}$ *and*
2. $\kappa_{T^{2h+1}}(u) = \kappa_{\hat{T}^{2h+1}}(u)$ *holds for each $(h+1)$-extreme node u of T.*

Proof We first prove the only-if direction. Suppose $T^{2h+1} = \hat{T}^{2h+1}$. By Lemma 2.6(2), the following three sets are identical: (i) the set of $(h+1)$-extreme nodes of T, (ii) the set of $(h+1)$-extreme nodes of $\hat{T}$, and (iii) the set of simplicial nodes of T^{2h+1}. Thus, Condition 2 holds trivially from Lemma 2.1. Assume for a contradiction that Condition 1 does not hold. That is, there are nodes u and w of R (and $\hat{R}$) such that $dist_{\hat{R}}(u, w) \leq 2h$ and $dist_R(u, w) > 2h$. Hence, $dist_{\hat{T}}(u, w) \leq 2h$ and $dist_T(u, w) > 2h$. By $dist_{\hat{T}}(u, w) \leq 2h$, there is a node x of $\hat{T}(h+1)$ with $\{u, w\} \subseteq N_{\hat{T},h}(x)$. By $dist_T(u, w) > 2h$, there cannot be any node y of $T(h+1)$ with $\{u, w\} \subseteq N_{T,h}(y)$. By Lemma 2.6(2), some minimal node separator of $\hat{T}^{2h+1}$ contains $\{u, w\}$, but no minimal node separator of T^{2h+1} contains $\{u, w\}$, contradicting $T^{2h+1} = \hat{T}^{2h+1}$.

As for the other direction, we show that Conditions 1 and 2 together imply the statement

$$dist_T(u, w) \leq 2h+1 \quad \text{if and only if} \quad dist_{\hat{T}}(u, w) \leq 2h+1$$

for any two nodes u and w. Suppose that one of u and w, say, u is an $(h+1)$-extreme node of T. We have (i) $dist_T(u, w) \leq 2h+1$ if and only if $w \in \kappa_{T^{2h+1}}(u)$;

and (ii) $dist_{\hat{T}}(u, w) \le 2h + 1$ if and only if $w \in \kappa_{\hat{T}^{2h+1}}(u)$. Since Condition 2 ensures $\kappa_{T^{2h+1}}(u) = \kappa_{\hat{T}^{2h+1}}(u)$, the above statement holds. It remains to prove the above statement for nodes u and w that are both in R (and $\hat{R}$). By $R^{2h} = \hat{R}^{2h}$, we have $dist_T(u, w) \le 2h$ if and only if $dist_{\hat{T}}(u, w) \le 2h$. Therefore, it suffices to show that $dist_R(u, w) = 2h + 1$ implies $dist_{\hat{R}}(u, w) \le 2h + 1$. Let (x, y) be the middle edge on the path between u and w in T. Both x and y are in $R(h)$. Assume $dist_R(u, x) = h = dist_R(w, y)$ without loss of generality. Since (x, y) is an edge of $R(h)$, it follows from Lemma 2.7 that maximal cliques $N_{R,h}(x)$ and $N_{R,h}(y)$ of R^{2h} are adjacent in the unique clique tree of $R^{2h} = \hat{R}^{2h}$. Let x' and y' be the nodes of $\hat{R}(h)$ corresponding to maximal cliques $N_{R,h}(x)$ and $N_{R,h}(y)$ via the isomorphism between $\hat{R}(h)$ and the clique tree of $\hat{R}^{2h}$ ensured by Lemma 2.7, i.e., $N_{\hat{R},h}(x') = N_{R,h}(x)$ and $N_{\hat{R},h}(y') = N_{R,h}(y)$. Thus, $dist_{\hat{R}}(x', u) \le h$ and $dist_{\hat{R}}(y', w) \le h$. Since (x', y') is an edge of $\hat{R}(h)$, we have $dist_{\hat{R}}(u, w) \le 2h + 1$. □

The following lemma shows that the above reduction can be done in linear time.

Lemma 4.2 *Given an n-node m-edge chordal graph G, it takes $O(m + n)$ time to either* (1) *ensure that G does not admit* $(2h + 1)$*-st tree roots for any positive integer h or* (2) *obtain a graph G' such that if $G = T^{2h+1}$ holds for some tree T and some positive integer h, then $G' = R^{2h}$, where R is the* $(h + 1)$*-reduced tree of T.*

Proof Let $\mathcal{S}$ be the of minimal node separators of G. Let G' be the union of the cliques on S for all minimal node separators $S \in \mathcal{S}$. By Lemmas 2.6(1) and 2.6(2), if R is the $(h + 1)$-reduced tree of T, then the set consisting of the minimal node separators of T^{2h+1} equals the set consisting of the maximal cliques of R^{2h}. Thus, G' has the required property. For instance, the set $\mathcal{S}$ of the minimal node separators of $G = T^5$, where T is as shown in Fig. 4(a), equals the set of maximal cliques of R^4, where R is as shown in Fig. 4(b). That is, $\mathcal{S} = \{S_{23}, S_{24}, S_{26}, S_{27}, S_{29}, S_{30}\}$, where

$$
\begin{aligned}
S_{23} &= \{v_8, v_{11}, v_{16}, v_{18}, v_{23}, v_{26}, v_{29}, v_{30}\}\\
S_{24} &= \{v_{10}, v_{12}, v_{15}, v_{21}, v_{24}, v_{27}, v_{30}\}\\
S_{26} &= \{v_9, v_{13}, v_{20}, v_{23}, v_{26}, v_{29}, v_{30}\}\\
S_{27} &= \{v_{14}, v_{15}, v_{21}, v_{24}, v_{27}, v_{28}, v_{29}, v_{30}, v_{31}, v_{32}\}\\
S_{29} &= \{v_{16}, v_{18}, v_{20}, v_{23}, v_{26}, v_{27}, v_{28}, v_{29}, v_{30}, v_{31}, v_{32}\}\\
S_{30} &= \{v_{21}, v_{22}, v_{23}, v_{24}, v_{25}, v_{26}, v_{27}, v_{28}, v_{29}, v_{30}, v_{31}, v_{32}\}.
\end{aligned}
$$

By Lemma 2.2, it takes $O(m + n)$ time to obtain $\mathcal{S}$ without redundancy from G. We have $\sum_{S \in \mathcal{S}} |S| = O(m + n)$, but $\sum_{S \in \mathcal{S}} |S|^2$ is not necessarily $O(m + n)$. Hence, it may take superlinear time to obtain G' from $\mathcal{S}$ naively. We prove the lemma by showing that Algorithm 2 correctly outputs G' in $O(m + n)$ time. Let us show two iterations of the while-loop to illustrate how Algorithm 2 works. Suppose that v_8 and $S_{23} = \{v_8, v_{11}, v_{16}, v_{18}, v_{23}, v_{26}, v_{29}, v_{30}\}$ are chosen by Steps 6–7 in the first iteration. Steps 8–10 add edges (v_8, v_j) for all $j \in \{11, 16, 18, 23, 26, 29, 30\}$ to the output graph. Step 11 deletes v_8 from S_{23}. Since the current S_{23} is the only set of $\mathcal{S}$ containing v_{11}, we have $count(v_{11}) = 1$, implying that Steps 13–16 are skipped

Algorithm 2 A linear-time algorithm that obtains R^{2h} from T^{2h+1} without knowing T and h, where R is the $(h+1)$-reduced tree of T

Input: The set $\mathcal{S}$ of minimal node separators of the input chordal graph G.

Output: The union graph G' of the cliques on node sets S for all $S \in \mathcal{S}$.

Algorithm reduce

1: let graph G' be initialized as empty.
2: **for** each node u of G **do**
3:     let $count(u)$ be the number of node sets S in $\mathcal{S}$ with $u \in S$.
4: **end for**
5: **while** $\mathcal{S}$ is not empty **do**
6:     choose a node u with $count(u) = 1$, if any; abort, otherwise.
7:     let S be the set of $\mathcal{S}$ that contains u.
8:     **for** each node w in S other than u **do**
9:         add edge (u, w) to the output graph G'.
10:     **end for**
11:     delete u from S.
12:     **if** $\min_{w \in S} count(w) = 2$ **then**
13:         **for** each node w in S **do**
14:             decrease $count(w)$ by one.
15:         **end for**
16:         delete S from $\mathcal{S}$.
17:     **end if**
18: **end while**
19: output G'.

in the first iteration. Suppose that v_{11} and $S_{23} = \{v_{11}, v_{16}, v_{18}, v_{23}, v_{26}, v_{29}, v_{30}\}$ are chosen by Steps 6–7 in the second iteration. Steps 8–10 add edges (v_{11}, v_j) for all $j \in \{16, 18, 23, 26, 29, 30\}$ to the output graph. Step 11 deletes v_{11} from S_{23}. Since each node of the current $S_{23} = \{v_{16}, v_{18}, v_{23}, v_{29}, v_{30}\}$ belongs to at least two sets of the current $\mathcal{S}$, Steps 13–15 decrease $count(v_j)$ by one for each $j \in \{16, 18, 23, 26, 29, 30\}$ and Step 16 deletes S_{23} from $\mathcal{S}$, leading to $\mathcal{S} = \{S_{24}, S_{26}, S_{27}, S_{29}, S_{30}\}$. Observe that Algorithm 2 does not need to know T or h.

Since Step 11 deletes a node from a set S in $\mathcal{S}$, the while-loop ends or aborts in at most n iterations. Since G is chordal, each $S \in \mathcal{S}$ is a clique of G. Thus, each edge (u, w) added to the output graph by Step 9 belongs to G. Since Step 11 deletes node u right after edges (u, w) are added in Step 9, Step 9 runs for at most m times throughout the execution of Algorithm 2. By $count(w) > 0$ for each node w of G throughout the execution of Algorithm 2, Step 14 runs for at most $\sum_{S \in \mathcal{S}} |S| = O(m + n)$ time. Hence, Algorithm 2 finishes or aborts in $O(m + n)$ time. The rest of the proof shows that if $G = T^{2h+1}$, then Algorithm 2 correctly outputs G'. Each edge of the output graph is added by Step 9. If edge (u, w) is added by Step 9, then nodes u and w belong to the set $S \in \mathcal{S}$ chosen by Step 7, implying that (u, w) is an edge of G'. It remains to ensure that each edge of G' is added to the output graph by Step 9.

According to Steps 6–7, 11, 13–16, at the beginning of each iteration of the while-loop, for each existing node x, $count(x)$ correctly stores the number of sets in the current $\mathcal{S}$ that contain node x. A set S of $\mathcal{S}$ can be deleted only at Step 16, which can be reached only if $\min_{w \in S} count(w) = 2$. Thus, after executing Step 16 to delete S from $\mathcal{S}$, each node of S is still in some set of $\mathcal{S}$. Step 11 is the only step that can remove a node from the union of all sets in $\mathcal{S}$. Let u_i be the node u chosen in the i-th iteration of the while-loop. That is, u_i is the i-th node deleted from the union of all sets in $\mathcal{S}$. The rest of the proof shows that if (u_i, u_j) with $i < j$ is an edge of $G' = R^{2h}$, then (u_i, u_j) is added to the output graph by Step 9 before u_i is deleted from S in the same iteration.

Let R_i be the subgraph of R induced by $\{u_i, u_{i+1}, \ldots\}$. $R_1 = R$ is a tree. By induction on i, we show that R_i for each $i \geq 1$ is a tree: Since $count(u_i) = 1$ holds at the beginning of the i-th iteration of the while-loop, node u_i is simplicial in R_i^{2h}. By Lemma 2.6(1), u_i is a leaf of R_i, implying that $R_{i+1} = R_i \setminus \{u_i\}$ is a tree. Now, let $\mathcal{T}_i$ be the clique tree of R_i^{2h}. Let $\mathcal{S}_i$ be the set $\mathcal{S}$ at the beginning of the i-th iteration. Let S_i be the set S chosen by Step 7 in the i-th iteration of the while-loop. To show that edge (u_i, u_j) of R^{2h} with $i < j$ is added to the output graph by Step 9 in the i-th iteration, it suffices to show $u_j \in S_i$ as follows.

$\mathcal{S}_1$ consists of the maximal cliques of R_1^{2h}. By induction on i, we show that $\mathcal{S}_i$ consists of the maximal cliques of R_i^{2h}: If Steps 13–15 are not reached in the i-th iteration, then the induction step holds trivially by Lemma 2.6(1). If Steps 13–15 are reached in the i-th iteration, then S_i is a leaf of $\mathcal{T}_i$ and u_i the only simplicial node of R_i^{2h} that belongs to S_i. Therefore, $S_i \setminus \{u_i\}$ is a subset of some set in $\mathcal{S}_{i+1}$. It also follows from Lemma 2.6(1) that $\mathcal{S}_{i+1}$ consists of the maximal cliques of R_{i+1}^{2h}. Since $\mathcal{S}_i$ consists of the maximal cliques of R_i^{2h} and R_i^{2h} contains at least one simplicial node, there must be a node u_i with $count(u_i) = 1$. Therefore, Step 6 does not abort throughout the execution. If Step 6 aborts, then G does not admit $(2h+1)$-st tree roots for any positive integer h. To show $u_j \in S_i$ for each edge (u_i, u_j) of R^{2h} with $i < j$, let k be the largest index such that $\mathcal{S}_k$ contains a set that contains $\{u_i, u_j\}$. Since u_i is deleted in the i-th iteration, $k \leq i$. We have shown above that $S_k \setminus \{u_k\}$ has to be a subset of a set in $\mathcal{S}_{k+1}$. Assume $k < i$ for contradiction. By $k < i < j$, we have $\{u_i, u_j\} \subseteq S_k \setminus \{u_k\}$, implying that some set in $\mathcal{S}_{k+1}$ contains $\{u_i, u_j\}$, contradicting the definition of k. Therefore, $k = i$, implying $u_j \in S_i$. The lemma is proved. □

4.2 A Linear-Time Algorithm for the $(2h+1)$-st Tree Root Problem

Theorem 4.3 *The $(2h+1)$-st tree root problem for any given n-node m-edge graph G and any given non-negative integer h can be solved in $O(n+m)$ time.*

Proof The proof assumes $h \geq 1$. By Lemma 2.2, we obtain the set $\mathcal{S}$ of the minimal node separators of G without redundancy in $O(m+n)$ time. We compute the set $\mathcal{K}$ of maximal cliques of G without redundancy from the given clique tree of G and obtain the simplicial nodes of G in $O(m+n)$ time. Let $\mathcal{K}^*$ consist of the sets $K \in \mathcal{K}$ that contains simplicial nodes of G. For each $K \in \mathcal{K}^*$, if there is a node of K that belongs to exactly one set in $\mathcal{S}$, then let u_K be an arbitrary one of such nodes; otherwise, output that G does not admit any $(2h+1)$-st tree roots. To see the reason why the non-existence of u_K implies that the $(2h+1)$-st tree root problem has no solution, suppose that $G = T^{2h+1}$ and R is the $(h+1)$-reduced tree of T. By Lemma 2.6(2), $S \in \mathcal{S}$ if and only if there is a node u of $T(h+1) = R(h)$ with $N_{T,h}(u) = N_{R,h}(u) = S$. By Lemma 2.6(2), $K \in \mathcal{K}$ if and only if there is an edge (u, w) of $T(h)$ with $N_{T,h}(u) \cup N_{T,h}(w) = K$. Therefore, each simplicial node of G

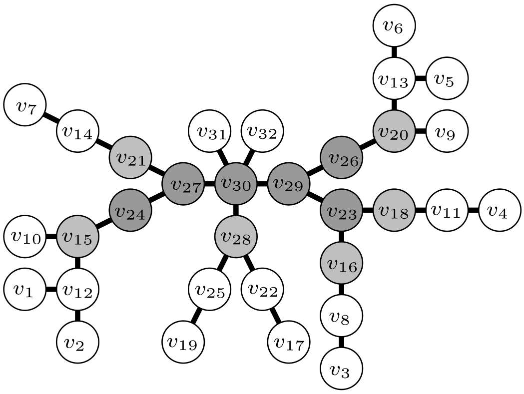

Fig. 5 A 5-th tree root of T^5, where T is the tree in Fig. 4(a)

is an h-extreme node of T. If $K \in \mathcal{K}^*$, there is an edge (u, w) of $T(h)$ with $N_{T,h}(u) \cup N_{T,h}(w) = K$ and $v \in N_{T,h}(w) \setminus N_{T,h}(u)$. That is, u is a leaf of $T(h+1) = R(h)$ and w is a leaf of $T(h)$. Let y be the unique neighbor of x in T. We have $y \in K$ and that $N_{T,h}(u)$ is the unique set $S \in \mathcal{S}$ with $y \in S$. For instance, suppose that T is as shown in Fig. 4(a). The 3-reduced tree R of T is as shown in Fig. 4(b). $x = v_6$ is a simplicial node of $G = T^5$. $K = N_{T,2}(v_{26}) \cup N_{T,2}(v_{20})$ is the unique set in $\mathcal{K}$ with $v_6 \in K$. $y = v_{13}$ is the unique neighbor of v_6 in T. $v_{13} \in K$. $S = N_{T,2}(v_{26})$ is the unique set in $\mathcal{S}$ with $v_{13} \in S$.

Let U consist of these nodes u_K, one for each K in $\mathcal{K}^*$. By Lemma 4.2, we spend $O(m+n)$ time to either (i) ensure that G does not admit any $(2h+1)$-st tree roots or (ii) obtain a graph G' from G such that if $G = T^{2h+1}$ for a tree T, then $G' = R^{2h}$ where R is the $(h+1)$-reduced tree of T. If $G = T^{2h+1}$, then U is h-disjoint in R. For instance, let T be as shown in Fig. 4(a). $T(2)$ has six leaves: v_{15}, v_{16}, v_{18}, v_{20}, v_{21}, and v_{28}. Thus, $G = T^{2h+1}$ has six maximal cliques that contain simplicial nodes of G:

$$N_{T,2}(v_{15}) \cup N_{T,2}(v_{24}), \qquad N_{T,2}(v_{16}) \cup N_{T,2}(v_{23}), \qquad N_{T,2}(v_{18}) \cup N_{T,2}(v_{23}),$$
$$N_{T,2}(v_{20}) \cup N_{T,2}(v_{26}), \qquad N_{T,2}(v_{21}) \cup N_{T,2}(v_{27}), \qquad N_{T,2}(v_{28}) \cup N_{T,2}(v_{30}).$$

A possible choice of U is $\{v_{12}, v_8, v_{11}, v_{13}, v_{14}, v_{22}\}$, where $N_{T,2}(v_{24})$ (respectively, $N_{T,2}(v_{23})$, $N_{T,2}(v_{23})$, $N_{T,2}(v_{26})$, $N_{T,2}(v_{27})$, and $N_{T,2}(v_{30})$) is the unique set in $\mathcal{S}$ that contains v_{12} (respectively, v_8, v_{11}, v_{13}, v_{14}, and v_{22}). U is 2-disjoint in R.

We apply Lemma 3.2 to obtain in $O(m+n)$ time a $2h$-th tree root T' of G' in which U is h-disjoint. The output tree of our $O(m+n)$-time algorithm is obtained from T' by attaching each simplicial node x to the node u_K in U with $x \in K$. For instance, if $U = \{v_{12}, v_8, v_{11}, v_{13}, v_{14}, v_{22}\}$ and T' is the tree R in Fig. 4(b), then the output tree of our algorithm is as shown in Fig. 5.

We show that if $G = \hat{T}^{2h+1}$ holds for some unknown tree $\hat{T}$, then the algorithm correctly outputs a tree T with $G = T^{2h+1}$. Let $\hat{R}$ be the $(h+1)$-reduced tree of $\hat{T}$. By Lemma 4.2, we know $G' = \hat{R}^{2h}$. Since each node u_K of U belongs to exactly one minimal node separator of G, u_K has to be an h-extreme node of $\hat{R}$. Moreover, if u and u' are two distinct nodes in U, then $dist_{\hat{T}}(u, u') \geq 2h$, since otherwise u and u' would belong to the same maximal clique of G, contradicting the definition of U.

Therefore, $\hat{T}$ is a $2h$-th tree root of G' such that U is h-disjoint in $\hat{T}$. It follows from Lemma 3.2 that the tree R obtained by our algorithm is a $2h$-th tree root of G' such that U is h-disjoint in T. We have $R^{2h} = \hat{R}^{2h}$. To show that $\kappa_{\hat{T}^{2h+1}}(x) = \kappa_{T^{2h+1}}(x)$ holds for each simplicial node x of G, let u_K be the node of U to which x is attached in T. Let y be the node of $T(h+1)$ that is closest to x in T. Let $\hat{y}$ be the node of $\hat{T}(h+1)$ that is closest to x in $\hat{T}$. $\kappa_{T^{2h+1}}(x)$ consists of the nodes in $N_{T,h}(y)$ plus the nodes attached to u_K in T. Since $N_{T,h}(y)$ is the maximal clique of R^{2h} that contains u_K and $N_{\hat{T},h}(\hat{y})$ is the maximal clique of $\hat{R}^{2h}$ that contains u_K, by Lemma 2.6(2) we have $N_{T,h}(y) = N_{\hat{T},h}(\hat{y})$. A node w belongs to $K \setminus N_{\hat{T},h}(\hat{y})$ if and only if w is an $(h+1)$-extreme node of $\hat{T}$ with $dist_{\hat{T}}(w,z) = h$, where z is the node of $\hat{T}(h)$ that is closest to u_K in $\hat{T}$. Therefore, these nodes w are the simplicial nodes of G that belong to K. We have $\kappa_{\hat{T}^{2h+1}}(x) = K = \kappa_{T^{2h+1}}(x)$. By Lemma 4.1, we know $T^{2h+1} = \hat{T}^{2h+1} = G$. □

5 Solving the Tree Root Problem

Let K^* be a maximal clique of G in $V(\mathit{Centroid}(\mathcal{T}))$. Let J consist of the positive integers j such that $\Pi(K^*, j)$ is non-empty. Define

$$h_{\min}(G) = \begin{cases} \max J & \text{if } |\mathit{Dom}(G)| = 0; \\ (\mathit{diam}(\mathcal{T}) + |\mathit{Dom}(G)| - 1)/2 & \text{if } 1 \le |\mathit{Dom}(G)| \le 2; \\ \lceil \mathit{diam}(\mathcal{T})/2 \rceil + 1 & \text{if } |\mathit{Dom}(G)| \ge 3. \end{cases}$$

Lemma 5.1 *For any subset U of $V(G)$, $h_{\min}(G)$ is the smallest positive integer h, if any, such that G admits a $2h$-th tree root in which U is h-disjoint.*

Proof Let H consist of the positive integers h such that G admits a $2h$-th tree root in which U is h-disjoint. We show that if H is non-empty, then $h_{\min}(G) = \min H$. For brevity of proof, we regard $\mathcal{T}$ as being rooted at K^*: For any maximal clique K of G other than K^*, let the *parent* $\pi(K)$ of K in $\mathcal{T}$ be the maximal clique K' of G such that (K, K') is an edge of $\mathit{Path}_{\mathcal{T}}(K, K^*)$. We say that K is a *child* of $\pi(K)$ in $\mathcal{T}$.
Case 1: $|\mathit{Dom}(G)| = 0$ By Property 2 of any (U,h)-assignment, we have $H \subseteq J$. We show

$$H = \{\max J\}$$

by proving that $j \le h$ holds for any positive integers $j \in J$ and $h \in H$. Assume for a contradiction that $j > h$ holds for some positive integers $j \in J$ and $h \in H$. Let T be a $2h$-th tree root of G in which U is h-disjoint. Since $\mathit{Dom}(G) = \emptyset$, we know $\mathit{diam}(T) > 4h$. Therefore, $\mathit{diam}(\mathcal{T}) = \mathit{diam}(T(h)) > 2h$. It follows from $\mathit{diam}(\mathcal{T}) > 2h$ and $j > h$ that

$$N_{\mathcal{T},h}(K^*) \subsetneq N_{\mathcal{T},j}(K^*).$$

Let w be a node in $\Pi(K^*, j)$, implying $\mathcal{K}_G(w) = N_{\mathcal{T},j}(K^*)$. Let u^* be the node with $K^* = N_{T,h}(u^*)$. We know $u^* \in \Pi(K^*, h)$, implying $\mathcal{K}_G(u^*) = N_{\mathcal{T},h}(K^*)$. It follows that

$$\mathcal{K}_G(u^*) \subsetneq \mathcal{K}_G(w),$$

thereby $u^* \neq w$. Since K^* belongs to $Centroid(\mathcal{T})$, it follows from the isomorphism between $\mathcal{T}$ and $T(h)$ that u^* belongs to $Centroid(T(h))$. By $diam(T(h)) > 2h$ and $u^* \neq w$, there is a node u in $V(T(h))$ such that $dist_T(u, u^*) = h$ and $dist_T(u, w) > h$. Let $K = N_{T,h}(u)$. We have $u^* \in V(K)$ and $w \notin V(K)$. It follows that K is a maximal clique in $\mathcal{K}_G(u^*) \setminus \mathcal{K}_G(w)$, contradicting $\mathcal{K}_G(u^*) \subsetneq \mathcal{K}_G(w)$.
Case 2: $1 \leq |Dom(G)| \leq 2$ We show

$$H = \{(diam(\mathcal{T}) + |Dom(G)| - 1)/2\}.$$

Let h be an integer in H. Let T be a $2h$-th tree root of G in which U is h-disjoint. By $1 \leq |Dom(G)| \leq 2$, it is not hard to see $Dom(G) = V(Centroid(T))$. If $|Dom(G)| = 1$, then $diam(T) = 4h$. It follows that $diam(T(h)) = diam(\mathcal{T}) = 2h$. If $|Dom(G)| = 2$, then $diam(T) = 4h - 1$. It follows that $diam(T(h)) = diam(\mathcal{T}) = 2h - 1$. For either case, we have $2h = diam(\mathcal{T}) + |Dom(G)| - 1$. Since G has exactly one clique tree $\mathcal{T}$, the diameter of $\mathcal{T}$ is fixed. We have $|H| = 1$.
Case 3: $|Dom(G)| \geq 3$ We show

$$\min H = \lceil diam(\mathcal{T})/2 \rceil + 1.$$

Let h be an integer in H. Let T be a $2h$-th tree root of G in which U is h-disjoint. By $|Dom(G)| \geq 3$, we have $diam(T) \leq 4h - 2$. Therefore, $diam(\mathcal{T}) = diam(T(h)) \leq 2h - 2$, implying that $\min H \geq diam(\mathcal{T})/2 + 1$. Observe that $\lceil diam(\mathcal{T})/2 \rceil + 1$ is the smallest integer that is no less than $diam(\mathcal{T})/2 + 1$. We prove the equality by showing that if $2h \geq diam(\mathcal{T}) + 4$, then G also admits a $(2h - 2)$-nd tree root in which U is $(h - 1)$-disjoint.

The rest of the proof assumes $diam(\mathcal{T}) \leq 2h - 4$, which directly implies that

$$N_{\mathcal{T},h-2}(K^*) = \mathcal{K}_G. \tag{3}$$

For any maximal clique K of G other than K^*, observe that $diam(\mathcal{T}) \leq 2h - 4$ also implies

$$\Pi(K, h) \subseteq \Pi(\pi(K), h - 1); \tag{4}$$

$$\Pi(K, h - 1) \subseteq \Pi(\pi(K), h - 2). \tag{5}$$

Let Φ be a (U, h)-assignment. Let Φ' be obtained from Φ by the following steps.

1. For each node u with $\Phi_2(u) \leq h - 2$, we simply let $\Phi'(u) = \Phi(u)$.
2. For the node u with $\Phi(u) = (K^*, h)$, i.e., $K^* = N_{T,h}(u)$, let $\Phi'(u) = (K^*, h - 1)$. For each node u of G with $\Phi(u) = (K^*, h - 1)$, let $\Phi'(u) = (K^*, h - 2)$.
3. For each maximal clique $K \neq K^*$ of G that is not a leaf of $\mathcal{T}$, we do the following:
   (a) Choose an arbitrary child K' of K in $\mathcal{T}$ and let $\Phi'(u') = (K, h - 1)$, where u' is the node with $K' = N_{T,h}(u')$.
   (b) For each child K'' of K in $\mathcal{T}$ other than K', let $\Phi'(u'') = (\pi(K), h - 2)$, where u'' is the node with $K'' = N_{T,h}(u'')$.
   (c) For each node u of G with $\Phi(u) = (K, h - 1)$, let $\Phi'(u) = (\pi(K), h - 2)$.
4. For each maximal clique $K \neq K^*$ of G that is a leaf of $\mathcal{T}$, we do the following:
   (a) Choose an arbitrary node u with $\Phi(u) = (K, h - 1)$ and let $\Phi'(u) = (K, h - 1)$.
   (b) For each node $w \neq u$ with $\Phi(w) = (K, h - 1)$, let $\Phi'(w) = (\pi(K), h - 2)$.

5. For each node w with $\Phi(w) = (K, h)$ and $\pi(K) = K^*$, let $\Phi'(w) = (K^*, h - 2)$.

We show that Φ' is a $(U, h - 1)$-assignment.

- Property 1: $\Phi'(u)$ is well defined for each node u of G. Moreover, by Property 1 of Φ and (3), (4), and (5), we know that $\Phi'(u)$ is a coordinate of u.
- Property 2: By Property 2 of Φ and Steps 2, 3(a), and 4(a), one can see that for each maximal clique K of G, there exists a unique node u of G with $\Phi'(u) = (K, h-1)$.
- Property 3: Property 2 of Φ' ensures that if u is a node with $\Phi'_2(u) = h - 2$, then there is a node w with $\Phi'_1(w) = \Phi'_1(u)$ and $\Phi'_2(w) = \Phi'_2(u) + 1$. Property 3 of Φ and Step 1 ensures that if u is a node with $\Phi'_2(u) < h - 2$, then there is a node w with $\Phi'_1(w) = \Phi'_1(u)$ and $\Phi'_2(w) = \Phi'_2(u) + 1$.
- Property 4: It follows immediately from Step 1.

The lemma is proved. □

Theorem 5.2 *Given an n-node m-edge graph G, it takes $O(n+m)$ time to determine the smallest positive integer h, if any, such that G admits a $2h$-th tree root.*

Proof Observe that $h_{\min}(G)$ can be computed from G and $\mathcal{T}$ in linear time, even if $h_{\min}(G)$ is undefined (e.g., when $J = Dom(G) = \emptyset$). Therefore, the theorem follows from Lemma 5.1 with $U = \emptyset$. □

Theorem 5.3 *Given an n-node m-edge graph G, it takes $O(n+m)$ time to determine the smallest positive integer h, if any, such that G admits a $(2h + 1)$-st tree root.*

Proof Observe that the algorithm described by Lemma 4.2 does not require the knowledge of h. Also, the definition of $h_{\min}(G)$ has nothing to do with the choice of U. Therefore, we can apply Lemma 4.2 to obtain $G' = R^{2h}$ from $G = T^{2h+1}$, and then spend linear time to compute $h_{\min}(G')$. By Lemma 4.1, $h_{\min}(G')$ is the smallest positive integer h, if any, such that G admits a $(2h + 1)$-st tree root. □

6 Conclusion

The combination of Theorems 3.1 and 4.3 yields a linear-time algorithm for the p-th tree root problem for any given integer p. Moreover, we show how to compute in linear time the smallest even integer p (Theorem 5.2) and odd integer p (Theorem 5.3), if any, such that a graph admits a p-th tree root. Combining Theorems 3.1, 4.3, 5.2, and 5.3, we have a linear-time algorithm for the tree-root problem. It would be of interest to see if our techniques can be extended to work for larger graph classes.

Acknowledgements Maw-Shang Chang thanks the Institute of Information Science of Academia Sinica of Taiwan for their hospitality and support where part of this research took place.

References

1. Agnarsson, G., Halldórsson, M.M.: Coloring powers of planar graphs. SIAM J. Discrete Math. **16**(4), 651–662 (2003)

2. Agnarsson, G., Halldórsson, M.M.: On colorings of squares of outerplanar graphs. In: Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 244–253 (2004)
3. Agnarsson, G., Halldórsson, M.M.: Strongly simplicial vertices of powers of trees. Discrete Math. **307**(21), 2647–2652 (2007)
4. Alon, N., Lubetzky, E.: Graph powers, Delsarte, Hoffman, Ramsey, and Shannon. SIAM J. Discrete Math. **21**(2), 329–348 (2007)
5. Alon, N., Lubetzky, E.: Independent sets in tensor graph powers. J. Graph Theory **54**(1), 73–87 (2007)
6. Alon, N., Mohar, B.: The chromatic number of graph powers. Comb. Probab. Comput. **11**(1), 1–10 (2002)
7. Brandstädt, A., Hundt, C., Mancini, F., Wagner, P.: Rooted directed path graphs are leaf powers. Discrete Math. **310**(4), 897–910 (2010)
8. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph Classes: a Survey. SIAM, Philadelphia (1999)
9. Buneman, P.: Characterization of rigid circuit graphs. Discrete Math. **9**, 205–212 (1974)
10. Chandran, L.S., Grandoni, F.: A linear time algorithm to list the minimal separators of chordal graphs. Discrete Math. **306**, 351–358 (2006)
11. Chang, M.-S., Ko, M.-T., Lu, H.-I.: Linear-time algorithms for tree root problems. In: Proceedings of the 10th Scandinavian Workshop on Algorithm Theory, pp. 411–422 (2006)
12. Chebikin, D.: Graph powers and k-ordered Hamiltonicity. Discrete Math. **308**(15), 3220–3229 (2008)
13. Chen, Z.-Z., Jiang, T., Lin, G.: Computing phylogenetic roots with bounded degrees and errors. SIAM J. Comput. **32**(4), 864–879 (2003)
14. Chiang, Y.-T., Lin, C.-C., Lu, H.-I.: Orderly spanning trees with applications. SIAM J. Comput. **34**(4), 924–945 (2005)
15. DeVos, M., McDonald, J., Scheide, D.: Average degree in graph powers. J. Graph Theory **72**(1), 7–18 (2013)
16. Dinur, I., Friedgut, E., Regev, O.: Independent sets in graph powers are almost contained in juntas. Geom. Funct. Anal. **18**(1), 77–97 (2008)
17. Dirac, G.A.: On rigid circuit graphs. Abh. Math. Semin. Univ. Hamb. **25**, 71–76 (1961)
18. Dom, M., Guo, J., Hüffner, F., Niedermeier, R.: Closest 4-leaf power is fixed-parameter tractable. Discrete Appl. Math. **156**(18), 3345–3361 (2008)
19. Dvořák, Z., Král', D., Nejedlý, P., Škrekovski, R.: Coloring squares of planar graphs with girth six. Eur. J. Comb. **29**(4), 838–849 (2008)
20. Escalante, F., Montejano, L., Rojano, T.: Characterization of n-path graphs and of graphs having nth root. J. Comb. Theory, Ser. B **16**, 282–289 (1974)
21. Flotow, C.: Graphs whose powers are chordal and graphs whose powers are interval graphs. J. Graph Theory **24**(4), 323–330 (1997)
22. Gavril, F.: The intersection graphs of subtrees in trees are exactly chordal graphs. J. Comb. Theory, Ser. B **16**, 47–56 (1974)
23. Geller, D.P.: The square root of a digraph. J. Comb. Theory **5**, 320–321 (1968)
24. Golumbic, M.: Algorithmic Graph Theory and Perfect Graphs, 2nd edn. North-Holland, Amsterdam (2004)
25. Gupta, S.K., Singh, A.: On tree roots of graphs. Int. J. Comput. Math. **73**, 157–166 (1999)
26. Harary, F., Karp, R.M., Tutte, W.T.: A criterion for planarity of the square of a graph. J. Comb. Theory **2**, 395–405 (1967)
27. He, X., Kao, M.-Y., Lu, H.-I.: A fast general methodology for information-theoretically optimal encodings of graphs. SIAM J. Comput. **30**(3), 838–846 (2000)
28. Hell, P., Yu, X., Zhou, H.S.: Independence ratios of graph powers. Discrete Math. **127**(1–3), 213–220 (1994)
29. Ho, C.-W., Lee, R.C.T.: Counting clique trees and computing perfect elimination schemes in parallel. Inf. Process. Lett. **31**, 61–68 (1989)
30. Hsu, W.-L., Ma, T.-H.: Fast and simple algorithms for recognizing chordal comparability graphs and interval graphs. SIAM J. Comput. **28**, 1004–1020 (1999)
31. Jacobson, G.: Space-efficient static trees and graphs. In: Proceedings of the 30th Annual Symposium on Foundations of Computer Science, pp. 549–554 (1989)
32. Kearney, P.E., Corneil, D.G.: Tree powers. J. Algorithms **29**, 111–131 (1998)
33. Lau, L.C.: Bipartite roots of graphs. ACM Trans. Algorithms **2**(2), 178–208 (2006)
34. Lau, L.C., Corneil, D.G.: Recognizing powers of proper interval, split, and chordal graphs. SIAM J. Comput. **18**(1), 83–102 (2004)
35. Lin, Y.L., Skiena, S.: Algorithms for square roots of graphs. SIAM J. Discrete Math. **8**, 99–118 (1995)

36. Linial, N.: Locality in distributed graph algorithms. SIAM J. Comput. **21**, 193–201 (1992)
37. Markenzon, L., da Costa Pereira, P.R.: A compact representation for chordal graphs. In: Seventh Cologne Twente Workshop on Graphs and Combinatorial Optimization, pp. 174–176 (2008)
38. Mohar, B., Vodopivec, A.: The genus of Petersen powers. J. Graph Theory **67**(1), 1–8 (2011)
39. Molloy, M., Salavatipour, M.R.: A bound on the chromatic number of the square of a planar graph. J. Comb. Theory, Ser. B **94**(2), 189–213 (2005)
40. Motwani, R., Sudan, M.: Computing roots of graphs is hard. Discrete Appl. Math. **54**, 81–88 (1994)
41. Mukhopadhyay, A.: The square root of a graph. J. Comb. Theory **2**, 290–295 (1967)
42. Munro, J.I., Raman, V.: Succinct representation of balanced parentheses, static trees and planar graphs. SIAM J. Comput. **31**(3), 762–776 (2001)
43. Nishimura, N., Ragde, P., Thilikos, D.M.: On graph powers for leaf-labeled trees. J. Algorithms **42**(1), 69–108 (2002)
44. Rose, D., Tarjan, R., Lueker, G.: Algorithmic aspects of vertex elimination of graph. SIAM J. Comput. **5**(2), 266–283 (1976)
45. Ross, I.C., Harary, F.: The squares of a tree. Bell Syst. Tech. J. **39**, 641–647 (1960)
46. Sadakane, K., Navarro, G.: Fully-functional succinct trees. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 134–149 (2010)
47. Simonyi, G.: Asymptotic values of the Hall-ratio for graph powers. Discrete Math. **306**(19–20), 2593–2601 (2006)
48. Sreenivasa Kumar, P., Veni Madhavan, C.E.: Minimal vertex separators of chordal graphs. Discrete Appl. Math. **89**, 155–168 (1998)
49. Sreenivasa Kumar, P., Veni Madhavan, C.E.: Clique tree generalization and new subclasses of chordal graphs. Discrete Appl. Math. **117**, 109–131 (2002)
50. Tarjan, R., Yannakakis, M.: Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs and selectively reduce acyclic hypergraphs. SIAM J. Comput. **13**(3), 566–576 (1984)
51. van den Heuvel, J., McGuinness, S.: Coloring the square of a planar graph. J. Graph Theory **42**(2), 110–124 (2003)