Sparse Square Roots *

Manfred Cochefert[1], Jean-François Couturier[1], Petr A. Golovach[2], Dieter Kratsch[1], and Daniël Paulusma[3]

[1] Laboratoire d'Informatique Théorique et Appliquée, Université de Lorraine, 57045 Metz Cedex 01, France,
{manfred.cochefert,jean-francois.couturier,dieter.kratsch}@univ-lorraine.fr
[2] Department of Informatics, University of Bergen, PB 7803, 5020 Bergen, Norway,
petr.golovach@ii.uib.no
[3] School of Engineering and Computing Sciences, Durham University, Science Laboratories, South Road, Durham DH1 3LE, UK,
daniel.paulusma@durham.ac.uk

Abstract. We show that it can be decided in polynomial time whether a graph of maximum degree 6 has a square root; if a square root exists, then our algorithm finds one with minimum number of edges. We also show that it is FPT to decide whether a connected n-vertex graph has a square root with at most $n-1+k$ edges when this problem is parameterized by k. Finally, we give an exact exponential time algorithm for the problem of finding a square root with maximum number of edges.

1 Introduction

Squares and square roots are classical concepts in graph theory. The square G^2 of the graph $G = (V_G, E_G)$ is the graph with vertex set V_G such that any two distinct vertices $u, v \in V_G$ are adjacent in G^2 if and only if u and v are of distance at most 2 in G. A graph H is a square root of G if $G = H^2$. Note that there exist graphs with no square root and that there exist graphs with many square roots. The characterization of those graphs that have a square root, or equivalently of those graphs that are the square of a graph, has already been studied in the 1960s. Mukhopadhyay [17] characterized squares of undirected graphs in 1967, whereas Geller [8] did the same for directed graphs in 1968. Neither characterization yields a polynomial time algorithm for recognizing squares. In fact, in 1994, Motwani and Sudan [16] showed that the problem of recognizing whether a given graph has a square root is NP-complete. As we will discuss, this fundamental result triggered a lot of research on the computational complexity of recognizing squares of graphs and computing square roots under the presence of additional structural assumptions. In particular, the following two recognition questions have attracted attention; here $\mathcal{G}$ denotes some fixed graph class.

(1) How hard is it to recognize the graphs that are the square of a graph from $\mathcal{G}$?
(2) How hard is is to recognize the graphs from $\mathcal{G}$ that have a square root?

* Supported by EPSRC (EP/G043434/1), ERC (267959) and ANR project AGAPE.

Ross and Harary [18] characterized those graphs that are the square of a tree. They proved that if a connected graph G has a tree square root, then this root is unique up to isomorphism. Moreover, they gave an algorithm for determining a tree that is a square root of any graph known to be the square of some tree. Lin and Skiena [13] obtained a linear time algorithm for deciding whether a graph is the square of a tree. They also proved that it can be decided in linear time whether a planar graph has a square root, and their algorithm finds such a square root if it exists. Lau and Corneil [10] gave a polynomial time algorithm for recognizing graphs that are the square of a proper interval graph, and they showed that the following three problems are NP-complete: recognizing the graphs that are the square of a chordal graph, the graphs that are the square of a split graph, and the chordal graphs that have a square root, respectively. Lau [9] gave a polynomial time algorithm that recognizes the graphs that are the square of a bipartite graph. Le and Tuy [11] obtained structural and algorithmic results for squares of block graphs that generalize the aforementioned results for squares of trees. In a later paper [12] they presented a quadratic time algorithm for recognizing the graphs that are the square of a strongly chordal split graph. Recently, Milanic and Schaudt [14] considered two other subclasses of chordal graphs, namely trivially perfect graphs and threshold graphs, and they gave linear time algorithms for recognizing the graphs from these two classes that have a square root. Adamaszek and Adamaszek [1] proved that if a graph has a square root of girth at least 6, then this square root is unique up to isomorphism. Farzad, Lau, Le and Tuy [7] gave a polynomial time algorithm for recognizing the graphs that have a square root of girth at least 6. They also showed that this problem is NP-complete for square roots of girth 4. The latter result was improved by Farzad and Karimi [6], who established the dichotomy by showing that the problem of recognizing the graphs that have a square root of girth 5 is NP-complete.

Our Results. In the first part of our paper (Section 3) we give a polynomial time algorithm that recognizes the graphs of maximum degree 6 that have a square root. If a square root exists, then our algorithm finds one with minimum number of edges and thus solves the following optimization problem for graphs of maximum degree 6.

Minimum Square Root
Input: a graph G and a positive integer s.
Question: does there exist a graph H with at most s edges such that $G = H^2$?

It can be shown that graphs of maximum degree at most 5 that have a square root have bounded pathwidth, which leads to a linear-time recognition algorithm of such graphs. However, this is not the case for graphs of maximum degree at most 6: consider the square of a wall with subdivided edges. Our approach is to preprocess a given graph G of maximum degree at most 6 in order to obtain a graph of bounded pathwidth.

In the second part of our paper (Section 4) we take a parameterized road to square roots; up to our knowledge, this has not been done so far. A problem with input size n and a parameter k is said to be *fixed parameter tractable* (or FPT)

if it can be solved in time $f(k) \cdot n^{O(1)}$ for some function f that only depends on k. Because any square root of a connected n-vertex graph G is a connected spanning subgraph of G, every square root of G has at least $n-1$ edges. This means that $s \geq n-1$ for any yes-instance (G, s) of Minimum Square Root. As such, a natural choice for the parameter would be $k = s - (n-1)$. This leads to the following problem that we call the Tree $+k$ Edges Square Root problem: given a graph G and an integer k, has G a square root with at most $n-1+k$ edges? We show that this problem is FPT when parameterized by k.

In the third part of our paper (Section 5) we consider the Maximum Square Root problem, which is the problem of finding a square root with maximum number of edges. We present an exact exponential time algorithm for Maximum Square Root. In Section 5 we also observe that it is FPT to decide whether a square root can be obtained by at most k edge deletions.

2 Preliminaries and Structural Lemmas

We only consider finite undirected graphs without loops and multiple edges. We refer to the textbook by Diestel [4] for any undefined graph terminology. Let G be a graph. We denote the vertex set of G by V_G and the edge set by E_G. The subgraph of G induced by a subset $U \subseteq V_G$ is denoted by $G[U]$. The graph $G - U$ is the graph obtained from G by removing all vertices in U. If $U = \{u\}$, we also write $G - u$. A set S is a separator in a connected graph G if $G - S$ is disconnected. For two disjoint subsets of vertices X, Y in G, a set of vertices S is an *(X,Y)-separator*, if $G - S$ has no path connecting a vertex of X with a vertex of Y. An (X, Y)-separator S is *minimal*, if no proper subset of S is an (X, Y)-separator. The *distance* $\mathrm{dist}_G(u, v)$ between a pair of vertices u and v of G is the number of edges of a shortest path between them. The *open neighborhood* of a vertex $u \in V_G$ is defined as $N_G(u) = \{v \mid uv \in E_G\}$, and its *closed neighborhood* is defined as $N_G[u] = N_G(u) \cup \{u\}$. Two vertices u, v are said to be *true twins* if $N_G[u] = N_G[v]$, and u, v are *false twins* if $N_G(u) = N_G(v)$. A vertex u is *simplicial*, if $N_G(u)$ is a clique. The degree of a vertex $u \in V_G$ is denoted $d_G(u) = |N_G(u)|$. The maximum degree of G is $\Delta(G) = \max\{d_G(v) | v \in V_G\}$. A vertex of degree one is said to be a *pendant* vertex.

A *tree decomposition* of a graph G is a pair (X, T) where T is a tree and $X = \{X_i \mid i \in V_T\}$ is a collection of subsets (called *bags*) of V_G such that the following three conditions hold: i) $\bigcup_{i \in V_T} X_i = V_G$, ii) for each edge $xy \in E_G$, $x, y \in X_i$ for some $i \in V_T$, and iii) for each $x \in V_G$ the set $\{i \mid x \in X_i\}$ induces a connected subtree of T. The *width* of a tree decomposition $(\{X_i \mid i \in V_T\}, T)$ is $\max_{i \in V_T} \{|X_i| - 1\}$. The *treewidth* $\mathbf{tw}(G)$ of a graph G is the minimum width over all tree decompositions of G. If T restricted to be a path, then we say that (X, T) is a *path decomposition* of a graph G. The *pathwidth* $\mathbf{pw}(G)$ of G is the minimum width over all path decompositions of G.

In the remainder of this section we give some structural results about sparse square roots. We start with the following observation that we will frequently use.

Observation 1 *Let H be a square root of a connected graph G.*

i) If u is a pendant vertex of H, then u is a simplicial vertex of G.
ii) If u, v are pendant vertices of H adjacent to the same vertex, then u, v are true twins in G.
iii) If u, v are pendant vertices of H adjacent to different vertices, then u and v are not adjacent in G unless $H = K_2$.

We now state a number of lemmas, the proofs of which have been omitted due to space restrictions, although we note that the proof of Lemma 1 is straightforward. Moreover, Lemmas 1 and 2 can also be found implicitly in the paper of Ross and Harary [18]. Because Ross and Harary [18] consider tree square roots, whereas we are concerned with finding general square roots, we cannot apply their results directly, and as such we give explicit statements of these lemmas.

Lemma 1. *Let H be a square root of G. Let $\{u_1, \ldots, u_r\} \subseteq V_H$ for some $r \geq 3$ induce a star in H with central vertex u_1. Let $u_3, \ldots, u_r$ be pendant and $\{u_2\}$ be a $(\{u_1, u_3, \ldots, u_r\}, V_H \setminus \{u_1, \ldots, u_r\})$-separator. Then $\{u_1, \ldots, u_r\}$ is a clique of G, and $\{u_1, u_2\}$ is a minimal $(\{u_3, \ldots, u_r\}, V_G \setminus \{u_1, \ldots, u_r\})$-separator of G.*

Lemma 2. *Let $\{u_1, \ldots, u_r\}$, $r \geq 3$, be a clique in a connected graph G such that $\{u_1, u_2\}$ is a minimal $(\{u_3, \ldots, u_r\}, V_G \setminus \{u_1, \ldots, u_r\})$-separator. Let also $\{x_1, \ldots, x_p\} = N_G(u_1) \setminus \{u_1, \ldots, u_r\}$ and $\{y_1, \ldots, y_q\} = N_G(u_2) \setminus \{u_1, \ldots, u_r\}$. Let G be a graph having a square root.*

i) For any square root H of G, the following holds: $u_1u_2 \in E_H$ and, either $u_3u_1, .., u_ru_1 \in E_H$, $u_3u_2, \ldots, u_ru_2 \notin E_H$, $u_1x_1, \ldots, u_1x_p \notin E_H$, and $\{u_2\}$ is a minimal $(\{u_1, u_3, \ldots, u_r\}, V_H \setminus \{u_1, \ldots, u_r\})$-separator in H or, symmetrically, $u_3u_1, \ldots, u_ru_1 \notin E_H$, $u_3u_2, \ldots, u_ru_2 \in E_H$, $u_2y_1, \ldots, u_2y_q \notin E_H$ and $\{u_1\}$ is a minimal $(\{u_2, .., u_r\}, V_H \setminus \{u_1, .., u_r\})$-separator in H.
ii) If u_1, u_2 are true twins in G, then either $u_1x_1, \ldots, u_1x_p \in E_H$ or $u_2y_1, \ldots, u_2y_q \in E_H$, G is the union of two complete graphs with vertex sets $\{u_1, \ldots, u_r\}$ and $\{u_1, u_2, x_1, \ldots, x_p\}$, and G has two isomorphic square roots with edge sets $\{u_1u_2, \ldots, u_1u_r\} \cup \{u_2x_1, \ldots, u_2x_p\}$ and $\{u_2u_1, u_2u_3, \ldots, u_2u_r\} \cup \{u_1x_1, \ldots, u_1x_p\}$ respectively.
iii) If $N_G[u_2] \setminus N_G[u_1] \neq \emptyset$, then $u_2u_1, \ldots, u_ru_1 \in E_H$, $u_3u_2, \ldots, u_ru_2 \notin E_H$, $u_1x_1, \ldots, u_1x_p \notin E_H$, and G has a square root such that $\{u_1, \ldots, u_r\}$ induces the star with central vertex u_1 such that $u_3, \ldots, u_r$ are pendant in H; moreover, this square root can be obtained from any square root of G by the deletion of the edges u_iu_j for $i, j \in \{2, \ldots, r\}$, $i \neq j$.

Lemma 3. *Let H be a square root of G. Suppose that H contains the graph F shown in Fig. 1 as a subgraph, $r \geq 3$, $u_4, \ldots, u_r$ are pendant vertices of H, $d_H(u_2) = r - 1$, $u_1u_2u_3$ is an induced path in H that is not included in any cycle of length at most 6, $p, q \geq 1$, $\{x_1, \ldots, x_p\} = N_H(u_1) \setminus \{u_2\}$ and $\{y_1, \ldots, y_q\} = N_H(u_3) \setminus \{u_2\}$. Then $\{u_1, \ldots, u_r\}$ is a clique in G such that either $r = 3$ or $\{u_1, u_2, u_3\}$ is a minimal $(\{u_4, \ldots, u_r\}, V_G \setminus \{u_1, \ldots, u_r\})$ separator in G and the following holds:*

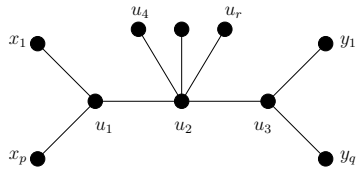

Fig. 1. The graph F.

i) $\{x_1, \ldots, x_p\} = N_G(u_1) \cap N_G(u_2) \setminus \{u_3, \ldots, u_r\}$ *and* $\{y_1, \ldots, y_q\} = N_G(u_2) \cap N_G(u_3) \setminus \{u_1, u_4, \ldots, u_r\}$*;*
ii) $d_G(u_2) = p + q + r - 1$*;*
iii) $N_G(u_1) \cap N_G(u_3) = \{u_2, u_4, u_5, \ldots, u_r\}$*;*
iv) $x_1u_3, \ldots, x_pu_3 \notin E_G$*,* $y_1u_1, \ldots, y_qu_1 \notin E_G$ *and* $x_iy_j \notin E_G$ *for* $i \in \{1, \ldots, p\}$ *and* $j \in \{1, \ldots, q\}$*.*

Lemma 4. *Let* $\{u_1, \ldots, u_r\}$*,* $r \geq 3$*, be a clique in a connected graph* G *such that either* $r = 3$ *or* $\{u_1, u_2, u_3\}$ *is a minimal* $(\{u_4, \ldots, u_r\}, V_G \setminus \{u_1, \ldots, u_r\})$*-separator. Assume also that*

i) $\{x_1, \ldots, x_p\} = N_G(u_1) \cap N_G(u_2) \setminus \{u_3, \ldots, u_r\} \neq \emptyset$ *and* $\{y_1, \ldots, y_q\} = N_G(u_2) \cap N_G(u_3) \setminus \{u_1, u_4, \ldots, u_r\} \neq \emptyset$*;*
ii) $d_G(u_2) = p + q + r - 1$*;*
iii) $N_G(u_1) \cap N_G(u_3) = \{u_2, u_4, \ldots, u_r\}$*;*
iv) $x_1u_3, \ldots, x_pu_3 \notin E_G$*,* $y_1u_1, \ldots, y_qu_1 \notin E_G$ *and* $x_iy_j \notin E_G$ *for* $i \in \{1, \ldots, p\}$ *and* $j \in \{1, \ldots, q\}$*.*

Then for any square root H *of* G *(if there is one), the graph* F *shown in Fig. 1 is a subgraph of* H *such that* $d_H(u_2) = r - 1$*,* $\{x_1, \ldots, x_p\} = N_H(u_1) \setminus \{u_2\}$ *and* $\{y_1, \ldots, y_q\} = N_H(u_3) \setminus \{u_2\}$*. Moreover, if* H *is a square root of* G*, then the graph obtained from* H *by deleting the edges* u_iu_j *for* $i, j \in \{4, \ldots, r\}$*,* $i \neq j$*, is a square root of* G*, where* $u_4, \ldots, u_r$ *are pendant vertices in* H*.*

Lemma 5. *Let* u, v *be true twins in a connected graph* G *with at least three vertices. Let also* G' *be the graph obtained from* G *by the deletion of* v*. If* H' *is a square root of* G'*, then the graph* H *obtained from* H' *by adding* v *with* $N_H(v) = N_{H'}(u)$ *(i.e, by adding a false twin of* u*) is a square root of* G*. If* H *is a square root of* G *such that* u, v *are false twins in* H*, then the graph* H' *obtained by the deletion of* v *is a square root of* G'*.*

3 Square Roots for Graphs of Bounded Degree

In this section we show that the Minimum Square Root problem is polynomial-time solvable for graphs of maximum degree at most 6. We start with some additional terminology and lemmas, the proofs of which have been omitted.

Let G be a connected graph, and let $u \in V_G$. We let $L_0(u), \ldots, L_{s(u)}(u)$ denote the levels in the breadth-first search (BFS) from u, that is, $L_i(u) = \{v \in V_G \mid \text{dist}_G(u, v) = i\}$ for $i = 1, \ldots, s(u)$, where $s(u)$ is the number of levels in the decomposition. Hence $L_i = \emptyset$ if $i > s(u)$.

Lemma 6. *For a connected graph G and $u \in V_G$,*

$$\mathbf{pw}(G^2) \leq \max\{|L_i(u) \cup L_{i+1}(u) \cup L_{i+2}(u)||0 \leq i \leq s(u)\} - 1.$$

We define the following auxiliary problem.

Minimum Square Root with Labels
Input: a graph G, positive integer s and sets of edges $R, B \subseteq E_G$.
Question: does there exist a graph H with at most s edges such that $G = H^2$, $R \subseteq E_H$ and $B \cap E_H = \emptyset$?

We will use the following lemma.

Lemma 7. *The* Minimum Square Root with Labels *problem can be solved in time $O(f(t)n)$ for n-vertex graphs of treewidth at most t.*

Using Lemma 6 we show that if a square root of G has no induced paths with internal vertices of degree 2 that are parts of short cycles, then G has bounded pathwidth.

Lemma 8. *Let H be a square root of a graph G with $\Delta(G) \leq 6$ such that H has no induced path xyz with $d_H(y) = 2$, $d_H(x) \geq 2$ and $d_H(z) \geq 2$ that is not included in any cycle of length at most 6 in H. Then $\mathbf{pw}(G) \leq 71$.*

The following example shows that we cannot obtain an analog of Lemma 8 for graphs of maximum degree at most 7. Let H' be a cubic graph. We construct a graph H as follows. For each vertex $u \in V_{H'}$ with $N_{H'}(u) = \{v_1, v_2, v_3\}$, u is replaced by three pairwise adjacent vertices u_1, u_2, u_3, and the edges uv_1, uv_2, uv_3 are replaced by u_1v_1, u_2v_2, u_3v_3. We observe that H is cubic and that $\Delta(H^2) = 7$. However, not only $\mathbf{pw}(H^2)$ but also $\mathbf{tw}(H^2)$ is not bounded, because the treewidth of H' can be arbitrary.

We can now prove the main theorem of this section.

Theorem 1. Minimum Square Root *can be solved in time $O(n^5)$ for n-vertex graphs of maximum degree at most 6.*

Proof. The proof is constructive. Our algorithm has two stages. At the first stage we exclude induced paths from square roots with internal vertices of degree two that are not included in short cycles in roots.

Let G be a graph of maximum degree at most 6. We use two sets of edges R and B, and we are trying to find square roots that contain edges of R but that do not contain any edge of B, that is, we are solving Minimum Square Root with Labels. Initially, $R = \emptyset$ and $B = \emptyset$. We recursively apply the following rule. Here, we say that a sequence $u_1, \ldots, u_\ell$ is *maximal* if it cannot be extended by adding new vertices in the beginning or in the end in such a way that conditions i)–v) of step 1 are fulfilled for the modified sequence.

Path reduction rule.

1. Find a maximal sequence of vertices $u_1, \ldots, u_\ell$, $\ell \geq 3$ such that

i) u_i, u_{i+1}, u_{i+2} are pairwise adjacent for $i \in \{1, \ldots, \ell-2\}$,
ii) the sets $\{x_1, \ldots, x_p\} = N_G(u_1) \cap N_G(u_2) \setminus \{u_1, u_2, u_3\}$ and $\{y_1, \ldots, y_q\} = N_G(u_{\ell-1}) \cap N_G(u_\ell) \setminus \{u_{\ell-2}, u_{\ell-1}, u_\ell\}$ are not empty,
iii) $d_G(u_2) = p + q + 2$ if $\ell = 3$, and $d_G(u_2) = p + 3$, $d_G(u_{\ell-1}) = q + 3$, $d_G(u_i) = 4$ for $i \in \{3, \ell-2\}$ if $\ell \geq 4$,
iv) $N_G(u_i) \cap N_G(u_{i+2}) = \{u_{i+1}\}$ for $i \in \{1, \ldots, \ell-2\}$, and
v) if $\ell \leq 4$, then $x_1u_\ell, \ldots, x_pu_\ell \notin E_G$, $y_1u_1, \ldots, y_qu_1 \notin E_G$, and if $\ell = 3$, then $x_iy_j \notin E_G$ for $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$.

2. Set $R' = \{u_1u_2, u_2u_3, \ldots, u_{\ell-1}u_\ell\} \cup \{x_1u_1, \ldots, x_pu_1\} \cup \{y_1u_\ell, \ldots, y_qu_\ell\}$ and $B' = \{x_1u_2, \ldots, x_pu_2\} \cup \{y_1u_2, \ldots, y_qu_2\} \cup \{u_iu_{i+2} | 1 \leq i \leq \ell-2\} \cup \{zu_1 | z \in N_G(u_1) \setminus \{u_2, u_3, x_1, \ldots, x_p\}\} \cup \{zu_\ell | z \in N_G(u_\ell) \setminus \{u_{\ell-2}, u_{\ell-1}, y_1, \ldots, y_q\}\}$.
3. If $R \cap B' \neq \emptyset$ or $R' \cap B \neq \emptyset$, then stop and return `no`.
4. Delete vertices $u_2, u_4 \ldots, u_\ell$ from G, and also delete the edge u_1u_3 if $\ell = 3$. Set $R = (R \cup R') \cap E_G$ and $B = (B \cup B') \cap E_G$. Set $s = s - \ell + 1$.

To show that the path reduction rule is safe, consider an instance (G, R, B, s) of Minimum Square Root with Labels and assume that $u_1, \ldots, u_\ell$ is a sequence of vertices that satisfies i)–v) of step 1. By Lemma 4, for any square root H of G (if it exists), $R' \subseteq E_H$ and $B' \cap E_H = \emptyset$ for the sets R' and B' constructed at step 2. Hence, if $R \cap B' \neq \emptyset$ or $R' \cap B \neq \emptyset$, then we have a no-answer. Assume that we did not stop at step 3, and denote by $(\hat{G}, \hat{R}, \hat{B}, \hat{s})$ the instance of Minimum Square Root with Labels obtained at step 4. Let H be a solution for (G, R, B, s). Because $R' \subseteq E_H$ and $B' \cap E_H = \emptyset$ by Lemma 4, it is straightforward to check that the graph $\hat{H}$ obtained from H by the deletion of $u_2, \ldots, u_{\ell-1}$ is a solution for $(\hat{G}, \hat{R}, \hat{B}, \hat{s})$. From another side, if $\hat{H}$ is a solution for $(\hat{G}, \hat{R}, \hat{B}, \hat{s})$, then H obtained by joining u_1 and u_ℓ by a path of length $\ell - 1$ is a solution for (G, R, B, s).

We apply the path reduction rule recursively and as long as possible. Assume that we did not stop and returned `no`. To simplify notation, assume that (G, R, B, s) is the obtained instance of Minimum Square Root with Labels. Because we cannot apply the path reduction rule, by Lemma 3, we conclude that for any square root H of G, H has no induced path $u_1u_2u_3$ with $d_H(u_2) = 2$, $d_H(u_1) \geq 2$ and $d_H(u_3) \geq 2$ that is not included in any cycle of length at most 6 in H, as otherwise we could apply the rule for the maximal sequence that includes u_1, u_2, u_3. By Lemma 8, $\mathbf{pw}(G) \leq 71$ if G has a square root. Using Bodlaender's algorithm [3], we check whether $\mathbf{pw}(G) \leq 71$. If $\mathbf{pw}(G) > 71$, then we conclude that we have a no-answer. Otherwise, we solve Minimum Square Root with Labels using Lemma 7.

To conclude the proof, it remains to evaluate the complexity. Each application of the path reduction rule can be done in time $O(n^3m)$ where m is the number of edges. We can check all triples u_1, u_2, u_3 of pairwise adjacent vertices in time $O(n^3)$. Then we can construct the sets $N_G(u_1) \cap N_G(u_2) \setminus \{u_1, u_2, u_3\}$, $N_G(u_1) \cap N_G(u_2) \setminus \{u_1, u_2, u_3\}$ and check conditions i)–v) in time $O(m)$. Observe that we possibly can extend the sequence $u_1, \ldots, u_i$ for $i \geq 3$ only if $N_G(u_{i-1}) \cap N_G(u_i) \setminus \{u_{i-2}, u_{i-1}, u_i\}$ contains exactly one element. Hence, the total time needed to

obtain a maximal sequence if u_1, u_2, u_3 are given is $O(m)$. Also the checking whether we have a no-answer at step 3 and construction of the new instance can be done in linear time. As the rule is applied at most n times, we conclude that the total time for this step is $O(n^5)$. It remains to observe that the Bodlaender's algorithm [3] runs in linear time, and Square Root with Labeled Edges is also can be solved in linear time for graphs of bounded treewidth. □

We observe that by the same approach we can solve other variants of square root problems for graphs of maximum degree at most 6. We can find, for example, a square root of maximum size. Also we can count all square roots.

4 The Tree $+k$ Edges Square Root Problem

In this section we prove the following theorem.

Theorem 2. *The* Tree $+k$ Edges Square Root *problem can be solved in time* $2^{O(k^4)} + O(n^4 m)$ *time on graphs with* n *vertices and* m *edges.*

Proof. Due the space restrictions, we only sketch the proof here.

We need the following auxiliary problem:

Tree $+k$ Edges Square Root with Labels
Input: an n-vertex graph G, a non-negative integer k and two subsets of edges $R, B \subseteq E_G$.
Parameter: k.
Question: does there exist a graph H with at most $n + k - 1$ edges such that $G = H^2$, $R \subseteq E_H$ and $B \cap E_H = \emptyset$?

In order to prove the theorem, we reduce Tree $+k$ Edges Square Root to Tree $+k$ Edges Square Root with Labels where the size of the graph in the obtained instance is bounded by a function of k. Then we solve Tree $+k$ Edges Square Root with Labels by a brute force algorithm.

Let G be a connected graph with n vertices and m edges, and let k be a positive integer. First, we check whether G has a tree square root using the algorithm by Lin and Skiena [13], and if we find one, then we stop and return a yes-answer. From now on we assume that any square root of G (if there is one) has cycles. Clearly, connected graphs that have square roots have no cut vertices. Hence, we also check whether G is 2-connected, and stop and return `no` otherwise. We introduce two sets of edges R and B. Initially $R = B = \emptyset$.

As in the algorithm of Lin and Skiena [13], we "trim" pendant edges in potential roots. Since the root we are looking for is not a tree, our trimming rule is more sophisticated and based on Lemmas 1 and 2.

Trimming rule.

1. Find a pair $S = \{u_1, u_2\}$ of two adjacent vertices such that one component of $G - S$ has a set of vertices $\{u_3, \ldots, u_r\}$ such that $\{u_1, \ldots, u_r\}$ is a clique.

2. If either $N_G[u_1] = N_G[u_2]$ or $N_G[u_1] \setminus N_G[u_2] \neq \emptyset$ and $N_G[u_2] \setminus N_G[u_1] \neq \emptyset$, then stop and return `no`.
3. If $N_G[u_1] \setminus N_G[u_2] \neq \emptyset$, then rename u_1 by u_2 and u_2 by u_1.
4. Set $R' = \{u_1u_2, \ldots, u_1u_r\}$ and $B' = \{u_iu_j | 2 \leq i < j \leq r\} \cup \{u_1x | x \in N_G(u_1) \setminus \{u_2, \ldots, u_r\}\}$.
5. If $R \cap B' \neq \emptyset$ or $R' \cap B \neq \emptyset$, then stop and return `no`. Otherwise, set $R = R \cup R'$, $B = B \cup B'$, delete $u_3, \ldots, u_r$ from G and delete the edges incident to these vertices from R and B.

We apply this rule recursively until we either stop and return `no` or else obtain an instance of Tree $+k$ Edges Square Root Labels such that we cannot apply the rule anymore. Suppose that we did not return `no`. To simplify notations, assume that (G, R, B) is the obtained instance. We need the set R constructed up to now. Let $R_0 = R$. We now apply the following rule, which is based on Lemmas 3 and 4.

Path reduction rule.

1. Find a triple $S = \{u_1, u_2, u_3\}$ of pairwise adjacent vertices such that
   i) either $N_G(u_1) \cap N_G(u_2) \cap N_G(u_3) = \emptyset$ or $N_G(u_1) \cap N_G(u_2) \cap N_G(u_3) = \{u_3, \ldots, u_r\}$ is a clique of G and S is a $(\{u_4, \ldots, u_r\}, V_G \setminus \{u_1, \ldots, u_r\})$-separator,
   ii) $\{x_1, \ldots, x_p\} = N_G(u_1) \cap N_G(u_2) \setminus \{u_1, \ldots, u_r\}$ and $\{y_1, .., y_q\} = N_G(u_3) \cap N_G(u_2) \setminus \{u_1, \ldots, u_r\}$ are not empty,
   iii) $d_G(u_2) = p + q + r - 1$,
   iv) $N_G(u_1) \cap N_G(u_3) = \{u_2, u_4, \ldots, u_r\}$, and
   v) $x_1u_3, \ldots, x_pu_3 \notin E_G$, $y_1u_1, \ldots, y_qu_1 \notin E_G$ and $x_iy_j \notin E_G$ for $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$.
2. Set $R' = \{u_2u_1, u_2u_3, \ldots, u_2u_r\}$ and $B' = \{x_1u_2, \ldots, x_pu_2\} \cup \{y_1u_2, .., y_qu_2\} \cup \{u_1u_3, \ldots, u_1u_r\} \cup \{u_3u_4, \ldots, u_3u_r\}$.
3. If $R \cap B' \neq \emptyset$ or $R' \cap B \neq \emptyset$, then stop and return `no`.
4. Delete the vertices $u_2, u_4 \ldots, u_r$ from G and delete all edges incident to these vertices from R and B. If $u_1u_3 \in B$, then delete u_1u_3 from B. Include u_1u_3 in R. Modify G by adding edges $x_1u_3, \ldots, x_pu_3$ and $y_1u_1, \ldots, y_qu_1$ in G. Put these edges in B.

We apply the rule recursively and as long as possible. Suppose that we did not stop and return `no`. As before, assume that (G, R, B) is the obtained instance. Recall that R_0 is the set of vertices placed in R by the trimming rule. Let $R_1 = R_0 \cap R$ and $R_2 = R \setminus R_1$. Now we are ready to describe the final reduction rule based on Observation 1 and Lemma 5.

Simplicial vertex reduction rule.

1. Find the set S of all simplicial vertices v of G such that v is not incident to the edges of R_2 and if v is incident to an edge of R_1, then all other edges incident to v are in B.
2. If $|V_G \setminus S| > 15k - 14$, then stop and return `no`.

3. Construct the partition $S_1, \ldots, S_t$ of S such that any two vertices in each S_i are true twins, and vertices from S_i and S_j are not adjacent if $i \neq j$. Let $X_1, \ldots, X_t$ be the sets of vertices incident to the edges of R_1 in $S_1, \ldots, S_t$ respectively.
4. If $t > 15k - 14$, then stop and return `no`.
5. If for some $i \in \{1, \ldots, t\}$, all the edges of R_1 incident to the vertices of X_i have no common end-point, then stop and return `no`.
6. For each $i \in \{1, \ldots, t\}$, if $|X_i| > 1$, then delete arbitrary $|X_i| - 1$ vertices of X_i from G and S_i, and delete the edges of R, B incident to these vertices.
7. For each $i \in \{1, \ldots, t\}$, if $|S_i| > 15k-13$, then delete arbitrary $|S_i|-15k+13$ vertices of $S_i \setminus X_i$ from G.

For these rules, we prove that if we stop while executing them, then the problem has no solution. If $(\hat{G}, \hat{R}, \hat{B})$ is the instance obtained by one application of the rules, then $\hat{G}$ is connected, Tree $+k$ Edges Square Root with Labels has a yes-answer for $(\hat{G}, \hat{R}, \hat{B})$ if and only if Tree $+k$ Edges Square Root with Labels has a yes-answer for (G, R, B), and $\hat{G}$ has at most $(15k-14)(15k-12)$ vertices.

To complete the proof of Theorem 2, it remains to solve the obtained reduced instance $(\hat{G}, \hat{R}, \hat{B})$ and evaluate running time. As the obtained graph has at most $(15k - 14)(15k - 12)$ vertices, it has at most $(15k - 14)(15k - 12)((15k - 14)(15k - 12) - 1)/2$ edges. Therefore, we can solve Tree $+k$ Edges Square Root with Labels for the obtained instance in time $2^{O(k^4)}$ by brute force checking all edge subsets of size at most $|V_{\hat{H}}| + k - 1$. Now we observe that the trimming and path reduction rules are applied at most n times to construct $(\hat{G}, \hat{R}, \hat{B})$. Each application of the trimming rule can be done in time $O(n^2m)$ and each application of the path reduction rule takes $O(n^3m)$. Finally, the simplicial vertex reduction rule can be done in $O(nm)$. Hence, the total running time is $2^{O(k^4)} + O(n^4m)$. □

Note we reduced Tree $+k$ Edges Square Root to Tree $+k$ Edges Square Root with Labels, i.e., we did not obtain a polynomial kernel. In fact, a polynomial kernel for Tree $+k$ Edges Square Root can be obtained by similar reduction rules, but the obtained graph would have more than $(15k - 14)(15k - 12)$ vertices.

5 Conclusions

We proved that Tree $+k$ Edges Square Root is FPT when parameterized by k. We also showed that Minimum Square Root can be solved in polynomial time for graphs of maximum degree at most 6. It would be interesting to know whether this degree restriction is tight. Is it possible to solve the problem in polynomial time for graphs of maximum degree at most Δ for some fixed $\Delta \geq 7$? Is there a fixed Δ such that Minimum Square Root is NP-complete for graphs of maximum degree at most Δ? This question is open even if we ask about the existence of any (not necessarily minimum) square root. Another interesting

direction of research is to consider square roots of bounded degree. It is trivial to check whether a graph has a square root of maximum degree at most two. Can the existence of a subcubic square root be tested in polynomial time?

Is it possible to construct an exact algorithm for Minimum Square Root that is better than the trivial exact algorithm for this problem? It can be noted that if we consider Maximum Square Root (i.e. we ask about a square root of maximum size), then such an algorithm exists. This algorithm is based on the simple observation that to construct a square root H from a given graph G, for every pair of adjacent edges not belonging to a triangle we have to delete at least one of these edges. Since the structure of the paths in G is crucial, the following auxiliary graph $\mathcal{P}(G)$ with vertex set E_G is useful: for any distinct edges $e_1 = xy$ and $e_2 = yz$ with a common end-point such that $xz \notin E_G$, e_1e_2 is an edge of $\mathcal{P}(G)$. Clearly, for a given graph G, $\mathcal{P}(G)$ is a subgraph of the line graph of G. This leads to the following lemma, the proof of which has been omitted.

Lemma 9. *Let H be a spanning subgraph of G. Then H is a square root of the graph G if and only if E_H is an independent set of $\mathcal{P}(G)$ and for all adjacent vertices u, v in G, u and v are at distance at most 2 in H.*

By using Lemma 9 we obtain an exact exponential time algorithm for Maximum Square Root.

Theorem 3. Maximum Square Root *can be solved by an exact exponential time algorithm of running time $O^*(3^{m/3})$, where m denotes the number of edges of the input graph.*

Proof. Let G be a graph. We compute the graph $\mathcal{P}(G)$, enumerate all maximal independent sets I of $\mathcal{P}(G)$, and verify for each $I \subseteq E$ whether G is the square of the graph $H_I = (V_G, I)$. Out of those graphs H_I that are square roots of G, return the one with maximum number edges; if no such graph H_I has been found, then G has no square roots. Correctness follows from Lemma 9. The graph $\mathcal{P}(G)$ can be computed in time $O(m^2)$. All the maximal independent sets of the m-vertex graph $\mathcal{P}(G)$ can be enumerated in time $O^*(3^{m/3})$ using the polynomial delay algorithm of Tsukiyama et al. [19], since $\mathcal{P}(G)$ has at most $3^{m/3}$ maximal independent sets [15]. Finally, for each maximal independent set I, we can check in time $O(nm)$ whether $(H_I)^2 = G$. Hence the overall running time of our algorithm is $O^*(3^{m/3})$. □

Lemma 9 also implies that it can be decided in time $O^*(2^k)$ whether a square root of a graph G can be obtained by deleting at most k edges. It is sufficient to check whether $\mathcal{P}(G)$ has a vertex cover C of size at most k such that $H = (V_G, E_G \setminus C)$ is a square root of G. All vertex covers of size at most k of a graph can be enumerated by adapting the standard $O^*(2^k)$ branching algorithm for the vertex cover problem (see e.g. [5]).

Aingworth, Motwani and Harary [2] proved that if H is a square root of a connected n-vertex graph $G \neq K_n$, then $|E_G \setminus E_H| \geq n-2$. Trivially, a complete graph is its own square root. Hence, we conclude the paper with the following

question: is it FPT to decide whether a connected n-vertex graph $G \neq K_n$ has a square root with at least $|E_G| - |V_G| - k + 2$ edges when parameterized by k? In particular, can it be decided in polynomial time whether a connected graph G has a square root with *exactly* $|E_G| - |V_G| + 2$ edges?

References

1. Adamaszek, A., Adamaszek, M.: Uniqueness of graph square roots of girth six. Electr. J. Comb. 18(1) (2011)
2. Aingworth, D., Motwani, R., Harary, F.: The difference between a graph and its square. Util. Math. 54, 223–228 (1998)
3. Bodlaender, H.L.: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM J. Comput. 25(6), 1305–1317 (1996)
4. Diestel, R.: Graph theory, Graduate Texts in Mathematics, vol. 173. Springer, Heidelberg, fourth edn. (2010)
5. Downey, R.G., Fellows, M.R.: Parameterized complexity. Monographs in Computer Science, Springer-Verlag, New York (1999)
6. Farzad, B., Karimi, M.: Square-root finding problem in graphs, a complete dichotomy theorem. CoRR abs/1210.7684 (2012)
7. Farzad, B., Lau, L.C., Le, V.B., Tuy, N.N.: Complexity of finding graph roots with girth conditions. Algorithmica 62(1-2), 38–53 (2012)
8. Geller, D.P.: The square root of a digraph. J. Combinatorial Theory 5, 320–321 (1968)
9. Lau, L.C.: Bipartite roots of graphs. ACM Transactions on Algorithms 2(2), 178–208 (2006)
10. Lau, L.C., Corneil, D.G.: Recognizing powers of proper interval, split, and chordal graph. SIAM J. Discrete Math. 18(1), 83–102 (2004)
11. Le, V.B., Tuy, N.N.: The square of a block graph. Discrete Mathematics 310(4), 734–741 (2010)
12. Le, V.B., Tuy, N.N.: A good characterization of squares of strongly chordal split graphs. Inf. Process. Lett. 111(3), 120–123 (2011)
13. Lin, Y.L., Skiena, S.: Algorithms for square roots of graphs. SIAM J. Discrete Math. 8(1), 99–118 (1995)
14. Milanic, M., Schaudt, O.: Computing square roots of trivially perfect and threshold graphs. Discrete Applied Mathematics, in press
15. Moon, J.W., Moser, L.: On cliques in graphs. Israel J. Math. 3, 23–28 (1965)
16. Motwani, R., Sudan, M.: Computing roots of graphs is hard. Discrete Applied Mathematics 54(1), 81–88 (1994)
17. Mukhopadhyay, A.: The square root of a graph. J. Combinatorial Theory 2, 290–295 (1967)
18. Ross, I.C., Harary, F.: The square of a tree. Bell System Tech. J 39, 641–647 (1960)
19. Tsukiyama, S., Ide, M., Ariyoshi, H., Shirakawa, I.: A new algorithm for generating all the maximal independent sets. SIAM J. Comput. 6(3), 505–517 (1977)