



Subexponential Parameterized Algorithms and Kernelization on Almost Chordal Graphs

Fedor V. Fomin¹ · Petr A. Golovach¹

Received: 16 September 2020 / Accepted: 11 March 2021 / Published online: 11 April 2021
© The Author(s) 2021

Abstract

We study algorithmic properties of the graph class CHORDAL- ke , that is, graphs that can be turned into a chordal graph by adding at most k edges or, equivalently, the class of graphs of fill-in at most k . It appears that a number of fundamental intractable optimization problems being parameterized by k admit *subexponential* algorithms on graphs from CHORDAL- ke . More precisely, we identify a large class of optimization problems on CHORDAL- ke solvable in time $2^{\mathcal{O}(\sqrt{k \log k})} \cdot n^{\mathcal{O}(1)}$. Examples of the problems from this class are finding an independent set of maximum weight, finding a feedback vertex set or an odd cycle transversal of minimum weight, or the problem of finding a maximum induced planar subgraph. On the other hand, we show that for some fundamental optimization problems, like finding an optimal graph coloring or finding a maximum clique, are FPT on CHORDAL- ke when parameterized by k but do not admit subexponential in k algorithms unless ETH fails. Besides subexponential time algorithms, the class of CHORDAL- ke graphs appears to be appealing from the perspective of kernelization (with parameter k). While it is possible to show that most of the weighted variants of optimization problems do not admit polynomial in k kernels on CHORDAL- ke graphs, this does not exclude the existence of Turing kernelization and kernelization for unweighted graphs. In particular, we construct a polynomial Turing kernel for WEIGHTED CLIQUE on CHORDAL- ke graphs. For (unweighted) INDEPENDENT SET we design polynomial kernels on two interesting subclasses of CHORDAL- ke , namely, INTERVAL- ke and SPLIT- ke graphs.

Keywords Parameterized complexity · Structural parameterization · Subexponential algorithms · Kernelization · Chordal graphs · Fill-in · Independent set · Clique · Coloring

✉ Petr A. Golovach
petr.golovach@uib.no

Fedor V. Fomin
fedor.fomin@uib.no

¹ Department of Informatics, University of Bergen, PB 7803, 5020 Bergen, Norway

1 Introduction

Many NP-hard graph optimization problems are solvable in polynomial or even linear time when the problem's input is restricted to a special graph class. For example, the chromatic number, the maximum size of a clique or an independent set of a perfect graph can be computed in polynomial time by the celebrated result of Grötschel, Lovász, and Schrijver [36]. For chordal graphs, this was already pointed by Gavril [32] in 1972. From the perspective of parameterized complexity, the natural question here is how stable are these friendly algorithmic properties of graph classes subject to some perturbations. For example, if an input n -vertex graph G is not chordal but can be turned into a chordal graph by adding at most k edges, how fast can we solve INDEPENDENT SET on G ? Can we solve the problem in polynomial time for constant k ? Or maybe for $k = \log n$ or even for $k = \text{poly}(\log n)$? A word of warning is on order here. Since an algorithm for INDEPENDENT SET of running time $2^{o(n)}$ will refute the Exponential Time Hypothesis (ETH) of Impagliazzo, Paturi and Zane [39, 40], and because $k \leq \binom{n}{2}$, the existence of an algorithm of running time $2^{k^{1/2-\epsilon}} \cdot n^{\mathcal{O}(1)}$ for some $\epsilon > 0$ (which is polynomial for $k = (\log n)^{2/(1-2\epsilon)}$) is unlikely. Interestingly, as we shall see, INDEPENDENT SET (and many other problems) are solvable in time $2^{k^{1/2} \log k} \cdot n^{\mathcal{O}(1)}$.

Leizhen Cai in [12] introduced a convenient notation for “perturbed” graph classes. Let \mathcal{F} be a class of graphs, then $\mathcal{F}-ke$ (respectively $\mathcal{F}-ve$) is the class of those graphs that can be obtained from a member of \mathcal{F} by deleting at most k edges (respectively vertices). Similarly one can define classes $\mathcal{F}+ke$ and $\mathcal{F}+ve$. Then for any class \mathcal{F} and optimization problem \mathcal{P} that can be solved in polynomial time on \mathcal{F} , the natural question is whether \mathcal{P} is fixed-parameter tractable parameterized by k , the “distance” to \mathcal{F} .

In this paper, we obtain several algorithmic results on the parameterized complexity of optimization problems on $\mathcal{F}-ke$, where \mathcal{F} is the class of chordal graphs (see Sect. 2 for the formal definitions of the central notions in the area). Let us remind that a graph H is *chordal* (or *triangulated*) if every cycle of length at least four has a chord, i.e., an edge between two nonconsecutive vertices of the cycle. We denote by CHORDAL- ke the class of graphs that can be made chordal graph by adding at most k edges. While parameterized algorithms for various problems on the class of CHORDAL- ke graphs were studied (see the section on previous work), our work introduces the first subexponential parameterized algorithms on this graph class. We prove the following.

Subexponential parameterized algorithms. We discover a large class of optimization problems on graph class CHORDAL- ke that are solvable in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$. Examples of such optimization problems are: the problem of finding an induced d -colorable subgraph of maximum weight (which generalizes WEIGHTED INDEPENDENT SET for $d = 1$ and WEIGHTED BIPARTITE SUBGRAPH for $d = 2$); the problem of finding a maximum weight induced subgraph admitting a

homomorphism into a fixed graph H ; the problem of finding an induced d -degenerate subgraph of maximum weight and its variants like WEIGHTED INDUCED FOREST (or, equivalently, WEIGHTED FEEDBACK VERTEX SET), WEIGHTED INDUCED TREE, INDUCED PLANAR GRAPH, WEIGHTED INDUCED PATH (CYCLE) or WEIGHTED INDUCED CYCLE PACKING; as well as various connectivity variants of these problems like WEIGHTED CONNECTED VERTEX COVER and WEIGHTED CONNECTED FEEDBACK VERTEX SET. This implies that all these problems are solvable in polynomial time for $k = (\frac{\log n}{\log \log n})^2$. On the other hand, we refute (subject to ETH) existence of a subexponential time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ algorithms on graphs in CHORDAL- ke for COLORING and CLIQUE. Moreover, our lower bounds hold for way more restrictive graph class COMPLETE- ke , the graphs within k edges from a complete graph. We also show that both problems are fixed-parameter tractable (FPT) (parameterized by k) on CHORDAL- ke graphs.

Kernelization. It follows almost directly from the previous work of Bodlaender, Jansen, and Kratsch [8, 42] that WEIGHTED INDEPENDENT SET, WEIGHTED VERTEX COVER, WEIGHTED BIPARTITE SUBGRAPH, WEIGHTED ODD CYCLE TRANSVERSAL, WEIGHTED FEEDBACK VERTEX SET and WEIGHTED CLIQUE do not admit a polynomial in k kernel (unless coNP $\not\subseteq$ NP/poly) on COMPLETE- ke and hence on CHORDAL- ke . Interestingly, these lower bounds do not refute the possibility of polynomial Turing kernelization or kernelization for unweighted variants of the problems. Indeed, we show that WEIGHTED CLIQUE on CHORDAL- ke parameterized by k admits a Turing kernel. For unweighted INDEPENDENT SET we show that the problem admits polynomial in k kernel on graph classes INTERVAL- ke and SPLIT- ke (graphs that can be turned into an interval or split graphs, correspondingly, by adding at most k edges).

Previous work. Chordal graphs form an important subclass of perfect graphs. These graphs were also intensively studied from the algorithmic perspective. We refer to books of Brandstädt, Le, and Spinrad [9], Golumbic [35], and Vandenberghe and Andersen [62] for introduction to chordal graphs and their algorithmic properties.

The problem of determining whether a graph G belongs to CHORDAL- ke , which is checking whether G can be turned into a chordal graph by adding at most k edges, is known in the literature as the MINIMUM FILL-IN problem. The name fill-in is due to the fundamental problem arising in sparse matrix computations, which was studied intensively in the past; see, e.g., the papers of Parter [55] and Rose [58]. The survey of Heggernes [38] gives an overview of techniques and applications of minimum and minimal triangulations.

MINIMUM FILL-IN (under the name CHORDAL GRAPH COMPLETION) was one of the 12 open problems presented at the end of the first edition of Garey and Johnson's book [31] and it was proved to be NP-complete by Yannakakis [64]. Kaplan et al. [44] proved that MINIMUM FILL-IN is fixed parameter tractable by giving an algorithm of running time $16^k \cdot n^{\mathcal{O}(1)}$. There was a chain of algorithmic improvements resulting in decreasing the constant in the base of the exponents [7, 11, 45] resulting with a subexponential algorithm of running time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ given by Fomin and Villanger [29]. A significant amount of work in parameterized algorithms is devoted to recognition problems of classes $\mathcal{F}-ke$, $\mathcal{F}+ke$, $\mathcal{F}-kv$, and $\mathcal{F}+kv$ for chordal graphs and various subclasses of chordal graphs [1, 2, 4, 5, 13–15, 27, 43, 50, 63].

Parameterized algorithms, mostly for graph coloring problems, were studied on perturbed chordal graphs and subclasses of this graph class [12, 60]. Among other results, Cai [12] proved that COLORING (the problem of computing the chromatic number of a graph) is FPT (parameterized by k) on SPLIT- ke graphs. Marx [49] proved that COLORING is FPT on CHORDAL + ke and INTERVAL + ke graphs but is W[1]-hard on CHORDAL + kv and INTERVAL + kv graphs. Jansen and Kratsch [41] proved that for every fixed integer d , the problems d -COLORING and d -LIST COLORING admit polynomial kernels on the parameterized graph classes SPLIT + kv , COCHORDAL + kv , and COGRAPH + kv .

Liedloff, Montealegre, and Todinca [48] gave a general theorem establishing fixed-parameter tractability for a large class of optimization problems. Let \mathcal{C}_{poly} be a class of graphs having at most $poly(n)$ minimal separators. (Since every chordal graph has at most n minimal separators, the class of chordal graphs is a subclass of \mathcal{C}_{poly} .) Let φ be a Counting Monadic Second Order Logic (CMSO) formula, G be a graph, and $t \geq 0$ be an integer. Liedloff, Montealegre, and Todinca proved that on graph class $\mathcal{C}_{poly} + kv$, the following generic problem

$$\begin{aligned} \text{Max } & |X| \\ \text{subject to } & \text{There is a set } F \subseteq V(G) \text{ such that } X \subseteq F; \\ & \text{The treewidth of } G[F] \text{ is at most } t; \\ & (G[F], X) \models \varphi. \end{aligned}$$

is solvable in time $f(t, \varphi, k) \cdot n^{\mathcal{O}(t)}$, and thus is FPT parameterized by k for a given formula φ and a fixed integer t . The problem generalizes many classical algorithmic problems like MAXIMUM INDEPENDENT SET, LONGEST INDUCED PATH, MAXIMUM INDUCED FOREST, and various packing problems as discussed by Fomin, Todinca, and Villanger in [28].

Since the class $\mathcal{C}_{poly} + kv$ contains CHORDAL- ke , the work of Liedloff et al. [48] yields that all these problems are fixed-parameter tractable on CHORDAL- ke graphs parameterized by $k + t + |\varphi|$. However, the theorem of Liedloff et al. does not yield our results. First, this theorem provides an FPT algorithm only for finding an induced subgraph of constant treewidth, which is not the case in our situation. Second, even if F is required to be an independent set, that is, the treewidth of F is zero, their technique does not derive parameterized algorithms with subexponential running times.

Organization of the paper. The remaining part of the paper is organized as follows. In Sect. 2, we introduce notation and provide some useful auxiliary results. In Sect. 3, we discuss subexponential algorithms on CHORDAL- ke . Section 4 contains the lower bounds for COLORING and CLIQUE on CHORDAL- ke . Sects. 5–7 are devoted to kernelization. In Sect. 5, we give lower bounds and construct a polynomial Turing kernel for WEIGHTED CLIQUE on CHORDAL- ke . In Sects. 6 and 7, we construct polynomial kernels for INDEPENDENT SET on INTERVAL- ke and SPLIT- ke respectively. We conclude in Sect. 8 by some open problems.

2 Preliminaries

All graphs considered in this paper are simple: finite undirected graphs without loops or multiple edges. For each of the graph problems considered in this paper, we let $n = |V(G)|$ and $m = |E(G)|$ denote the number of vertices and edges, respectively, of the input graph G if it does not create confusion. For a set $X \subseteq V(G)$, $\binom{X}{2}$ denotes the set of pairs of distinct vertices of X . For a graph G and a subset $X \subseteq V(G)$ of vertices, we write $G[X]$ to denote the subgraph of G induced by X . We write $G - X$ to denote the subgraph of G induced by $V(G) \setminus X$, and we write $G - u$ instead of $G - \{u\}$ for a single element set. Similarly, for an edge set A , $G - A$ denotes the graph $G' = (V(G), E(G) \setminus A)$, and for a set of pairs of vertices $A \subseteq \binom{V(G)}{2}$, $G + A$ is the graph $G' = (V(G), E(G) \cup A)$. For a single-element set $\{e\}$, we use $G + e$ for $G + \{e\}$. For $A \subseteq \binom{V(G)}{2}$, $G \Delta A$ denotes the graph $G' = (V(G), E(G) \Delta A)$. For a vertex v , we denote by $N_G(v)$ the (*open*) *neighborhood* of v , i.e., the set of vertices that are adjacent to v in G . The *closed neighborhood* $N_G[v]$ is $N_G(v) \cup \{v\}$. For a set of vertices $X \subseteq V(G)$, $N_G[X] = \bigcup_{v \in X} N_G[v]$ and $N_G(X) = N_G[X] \setminus X$. The *degree* of a vertex v is $d_G(v) = |N_G(v)|$. The complement of a graph G is the graph \overline{G} with $V(\overline{G}) = V(G)$ such that two distinct vertices are adjacent in \overline{G} if and only if they are not adjacent in G . A (*proper*) ℓ -*coloring* of a graph G is an assignment $c : V(G) \rightarrow \{1, \dots, \ell\}$ of *colors* $1, \dots, \ell$ to the vertices of G in such a way that adjacent vertices get distinct colors; a graph G is ℓ -*colorable* if it has an ℓ -coloring. A graph is d -*degenerate* for a nonnegative integer d , if every induced subgraph of G has a vertex of degree at most d . An equivalent way of defining d -degenerate graph is in terms of coloring orderings was given by Erdős and Hajnal [23]. A vertex ordering of a graph is a d -*coloring* ordering, if each vertex has at most d neighbors that are after it in the ordering. Then a graph G is d -degenerate if and only if it admits an d -coloring ordering.

For a graph class \mathcal{C} and a nonnegative integer k , $\mathcal{C} - ke$ denotes the class of all graphs G such that there is a set $A \subseteq \binom{V(G)}{2} \setminus E(G)$ of size at most k such that $G + A \in \mathcal{C}$. In words, this means that $\mathcal{C} - ke$ contains graphs that can be turned to be graphs of \mathcal{C} by at most k edge additions. For a set $A \subseteq \binom{V(G)}{2} \setminus E(G)$ such that $G + A \in \mathcal{C}$, we say that A is a \mathcal{C} -*modulator*.

Graph classes. A graph G is *chordal* (or *triangulated*) if it does not contain an induced cycle of length at least four. In other words, every cycle of length at least four has a chord, i.e., an edge whose end-vertices are nonconsecutive vertices of the cycle. The intersection graph of a family of intervals of the real line is called an *interval* graph; it is also said that G is an interval graph if there is a family of intervals (called *interval model* or *representation*) such that G is isomorphic to the intersection graph of this family. A graph G is said to be *split* if its vertex set can be partition into an independent set and a clique. We refer to the books of

Brandstädt, Le, and Spinrad [9] and Golumbic [35] for a detailed introduction to these graph classes. Notice that interval and split graphs are chordal.

A *triangulation* (or a *chordal complementation*) of a graph G is a chordal supergraph H with $V(H) = V(G)$. The *size* of the triangulation is $|E(H)| - |E(G)|$. The *fill-in* of a graph G , denoted $\text{fill-in}(G)$, is the minimum integer k such that $G \in \text{CHORDAL-}ke$. In other words, fill-in is the minimum number of edges whose addition makes the graph chordal. An *interval complementation* of a graph G is an interval supergraph H with $V(H) = V(G)$. Similarly, a *split complementation* of G is a split supergraph H and a *clique complementation* is a complete supergraph with $V(H) = V(G)$. The *size of interval (split, clique) completion* is $|E(H)| - |E(G)|$ and we denote the minimum size of an interval (split, clique) completion by $\text{int-comp}(G)$ ($\text{split-comp}(G)$, $\text{c-comp}(G)$ respectively). Clearly, G has an interval (split, clique) completion of size at most k if and only if $G \in \text{INTERVAL-}ke$ ($\text{SPLIT-}ke$, $\text{COMPLETE-}ke$). It is easy to see that $\text{c-comp}(G) = \binom{|V(G)|}{2} - |E(G)|$, and it is well-known that it is NP-hard to compute $\text{fill-in}(G)$ by the result of Yannakakis [64] and $\text{int-comp}(G)$ [31], and the same holds for $\text{split-comp}(G)$ as was proved by Natanzon, Shamir, and Sharan [53].

We will make use of the following observation.

Observation 1 *For every graph G , $\text{c-comp}(G) \geq \text{int-comp}(G) \geq \text{fill-in}(G)$ and $\text{c-comp}(G) \geq \text{split-comp}(G) \geq \text{fill-in}(G)$.*

In particular, this observation implies that complexity lower bounds obtained for graph problems parameterized by the clique completion size hold when they are parameterized by the interval or split completion or by the fill-in. Similarly, the hardness for the interval or split completion parameterization implies the hardness for the fill-in parameterization.

Natanzon, Shamir, and Sharan [52] proved that fill-in admits a polyopt approximation.

Proposition 1 [52] *There is a polynomial algorithm that, given a graph G and a nonnegative integer k , either correctly reports that $\text{fill-in}(G) > k$ or returns a triangulation of G of size at most $8k^2$.*

Tree decompositions. A *tree decomposition* of a graph G is a pair $T = (T, \{X_t\}_{t \in V(T)})$, where T is a tree whose every node t is assigned a vertex subset $X_t \subseteq V(G)$, called a bag, such that the following three conditions hold:

- (T1) $\bigcup_{t \in V(T)} X_t = V(G)$. In other words, every vertex of G is in at least one bag.
- (T2) For every $uv \in E(G)$, there exists a node t of T such that bag X_t contains both u and v .
- (T3) For every $u \in V(G)$, the set $T_u = \{t \in V(T) | u \in X_t\}$, i.e., the set of nodes whose corresponding bags contain u , induces a connected subtree of T .

To distinguish between the vertices of the decomposition tree T and the vertices of the graph G , we will refer to the vertices of T as *nodes*. By the classical result due to Buneman and Gavril [10, 33], every chordal graph G has a tree decomposition such that each bag of the decomposition is a maximal clique of G . Such a tree decomposition is a *clique tree* of the chordal graph G .

It is more convenient to describe dynamic programming algorithms on tree decompositions of a nice particular form. We think of nice tree decompositions as rooted trees. That is, for a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ we distinguish one vertex r of T which will be the root of T . We say that such a rooted tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is *nice* if the following conditions are satisfied:

- $X_r = \emptyset$ and $X_\ell = \emptyset$ for every leaf ℓ of T . In other words, all the leaves as well as the root contain empty bags.
- Every non-leaf node of T is of one of the following three types:
 - **Introduce node:** a node t with exactly one child t' such that $X_t = X_{t'} \cup \{v\}$ for some vertex $v \notin X_{t'}$; we say that v is *introduced* at t .
 - **Forget node:** a node t with exactly one child t' such that $X_t = X_{t'} \setminus \{w\}$ for some vertex $w \in X_{t'}$; we say that w is *forgotten* at t .
 - **Join node:** a node t with two children t_1, t_2 such that $X_t = X_{t_1} = X_{t_2}$.

Throughout the paper, given a nice tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a graph G , we denote by V_t the union of the bags in the subtree of T rooted in a node $t \in V(T)$.

We will be using the following proposition, see e.g., [49].

Proposition 2 *Every n -vertex chordal graph G admits a nice tree decomposition $T = (T, \{X_t\}_{t \in V(T)})$ such that every bag X_t of G is a clique. This decomposition has $\mathcal{O}(n^2)$ nodes and can be constructed in polynomial time.*

Parameterized complexity and kernelization. We refer to the books of Cygan et al. [18], Downey and Fellows [22], and Fomin et al. [26] for the detailed introduction to the field. Here we only briefly review the basic notions.

Parameterized complexity is a two-dimensional framework for studying the computational complexity of a problem. One dimension is the input size n , and the other is a parameter k associated with the input.

A parameterized problem is said to be *fixed-parameter tractable* (or FPT) if it can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$ for a computable function f .

Parameterized complexity theory also provides tools for obtaining complexity lower bounds. Here we use lower bounds based on the *Exponential Time Hypothesis* (ETH) formulated by Impagliazzo, Paturi and Zane [39, 40]. For an integer $k \geq 3$, let q_k be the infimum of the real numbers c such that the k -SATISFIABILITY problem can be solved in time $\mathcal{O}(2^{cn})$, where n is the number of variables. Exponential Time Hypothesis states that $q_3 > 3$. In particular, ETH implies that k -SATISFIABILITY cannot be solved in time $2^{\mathcal{O}(n)}n^{\mathcal{O}(1)}$.

A *compression* of a parameterized problem Π_1 into a (non-parameterized) problem Π_2 is a polynomial algorithm that maps each instance (I, k) of Π_1 with the input I and the parameter k to an instance I' of Π_2 such that

- (i) (I, k) is a yes-instance of Π_1 if and only if I' is a yes-instance of Π_2 , and
- (ii) $|I'|$ is bounded by $f(k)$ for a computable function f .

The output I' is also called a *compression*. The function f is said to be the *size* of the compression. A compression is *polynomial* if f is polynomial. Kernelization can be seen as a special case of compression when the algorithm is required to map a parameterized problem to itself. Formally, a *kernelization* algorithm for a parameterized problem Π is a polynomial algorithm that maps each instance (I, k) of Π to an instance (I', k') of Π such that

- (i) (I, k) is a yes-instance of Π if and only if (I', k') is a yes-instance of Π , and
- (ii) $|I'| + k'$ is bounded by $f(k)$ for a computable function f .

Respectively, (I', k') is a *kernel* and f is its *size*. A kernel is *polynomial* if f is polynomial.

Every decidable parameterized problem is FPT if and only if it admits a kernel. However, it is unlikely that every problem in FPT has a polynomial kernel (see, e.g., [26] for the details). Still, even if a parameterized problem admits no polynomial kernel up to some complexity conjectures, sometimes we can reduce it to solving of a polynomial number of instances of the same problem such that the size of each instance is bounded by a polynomial of the parameter.

Let Π be a parameterized problem and let $f : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ be a computable function. A *Turing kernelization* or *Turing kernel* for Π of size f is an algorithm that decides whether an instance (I, k) of Π is a yes-instance in time polynomial in $|I| + k$, when given access to an oracle that decides whether (I', k') is a yes-instance of Π in a single step if $|I'| + k' \leq f(k)$.

It is typical to describe a compression or kernelization algorithm as a series of *reduction rules*. A reduction rule is a polynomial algorithm that takes as an input an instance of a problem and output another, usually reduced, instance. A reduction rule is *safe* if the input instance is a yes-instance if and only if the output instance is a yes-instance.

In our paper, we consider kernelization/compression for weighted problems. Because we are using weights, we have to compress their values as well. For this, we follow the approach proposed by Etscheid et al. [24] that is based on the algorithm for compressing numbers given by Frank and Tardos in [30]. We state the result of Frank and Tardos in the form provided in [24].

Proposition 3 [30] *There is an algorithm that, given a vector $w \in \mathbb{Q}^h$ and an integer N , in polynomial time finds a vector $\bar{w} \in \mathbb{Z}^h$ with $\|\bar{w}\|_\infty \leq 2^{4h^3} N^{h(h+2)}$ such that $\text{sign}(w \cdot b) = \text{sign}(\bar{w} \cdot b)$ for all vectors $b \in \mathbb{Z}^h$ with $\|b\|_1 \leq N - 1$.*

3 Subexponential Algorithms for Induced d -colorable Subgraphs

To construct subexponential algorithms on CHORDAL–ke, we consider tree decompositions such that each bag is “almost” a clique.

Definition 1 (*k-almost chordal graph*) Let k be a nonnegative integer. A tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of a graph G is *k-almost chordal* if for every $t \in V(T)$, $c\text{-comp}(G[X_t]) \leq k$, that is, every bag can be converted to a clique by adding at most k edges.

Note that every chordal graph has 0-almost chordal tree decomposition.

Given a k -almost chordal tree decomposition, we construct dynamic programming algorithms that are subexponential in k for various problems. The crucial property of the graphs in CHORDAL–ke is that we can build nice k -almost chordal tree decompositions for them in subexponential in k time by using the following result of Fomin and Villanger [29].

Proposition 4 [29] *Deciding whether graph G is in CHORDAL–ke can be done in time $2^{\mathcal{O}(\sqrt{k} \log k)} + \mathcal{O}(k^2 nm)$. Moreover, if $G \in \text{CHORDAL–ke}$, then the corresponding triangulation can be found in time $2^{\mathcal{O}(\sqrt{k} \log k)} + \mathcal{O}(k^2 nm)$.*

Lemma 1 *A nice k -almost chordal decomposition of a graph $G \in \text{CHORDAL–ke}$ with at most n^2 bags can be constructed in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$.*

Proof Let G be a graph. By Proposition 4, in time $2^{\mathcal{O}(\sqrt{k} \log k)} + \mathcal{O}(k^2 nm)$ we can construct a triangulation H of G such that $H = G + A$ for $A \subseteq \binom{V(G)}{2}$ of size at most k . Using Proposition 2, we construct in polynomial time a nice tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of H with at most n^2 bags such that every bag X_t is a clique of H . Clearly, \mathcal{T} is a nice tree decomposition of G and $G[X_t] = H[X_t] - A$ for $t \in V(T)$. Therefore, $c\text{-comp}(G[X_t]) \leq k$ for $t \in V(T)$, that is, \mathcal{T} is a nice k -almost chordal decomposition of G . \square

We need the following folklore observation. We prove it here for completeness.

Lemma 2 *A d -degenerate graph G has at most $2^d \cdot n$ cliques and all cliques of G can be listed in $2^d \cdot n^{\mathcal{O}(1)}$ time.*

Proof Let G be a d -degenerate graph, and let v_1, \dots, v_n be a d -coloring ordering of its vertices. Since $d_{G[v_i, \dots, v_n]}(v_i) \leq d$, $G[v_i, \dots, v_n]$ has at most 2^d cliques containing v_i for every $i \in \{1, \dots, n\}$. Therefore, G has at most $2^d \cdot n$ cliques. The cliques can be enumerated by brute force checking the subsets of the neighbors of v_i in $G[v_i, \dots, v_n]$ for every $i \in \{1, \dots, n\}$. Because a d -coloring ordering can be found in polynomial (in fact, linear) time [51], the total running time is $2^d \cdot n^{\mathcal{O}(1)}$. \square

The crux of our subexponential algorithms is in the following combinatorial lemma.

Lemma 3 Let $d \geq 1$ be an integer. Let G be a graph and let \mathcal{F} be a set of induced d -colorable subgraphs of G . Let $U \subseteq V(G)$ be a set of vertices of G such that $c\text{-comp}(G[U]) \leq k$, that is, U can be made a clique by adding at most k edges. Then

- for every $F \in \mathcal{F}$,

$$|U \cap V(F)| \leq \frac{3d + \sqrt{d^2 + 8dk}}{2},$$

and

- the size of the projection of \mathcal{F} on U , that is, the size of the family

$$\mathcal{S} = \{S \mid S = U \cap V(F) \text{ for some } F \in \mathcal{F}\}$$

is at most $(1 + 2^{(\sqrt{1+8k}-1)/2} \cdot |U|)^d$.

Moreover, there is an algorithm that in time $2^{\mathcal{O}(d\sqrt{k})} \cdot \mu^{\mathcal{O}(d)}$ outputs a family of sets $\mathcal{S}' \supseteq \mathcal{S}$ such that each set from \mathcal{S}' has at most $\frac{3d + \sqrt{d^2 + 8dk}}{2}$ vertices, the number of sets in \mathcal{S}' is $(1 + 2^{(\sqrt{1+8k}-1)/2} \cdot n)^d$ and $G[S]$ is d -colorable for $S \in \mathcal{S}'$.

Proof We partition U into sets X and Y as follows. Let X be the vertices of U that have at least one non-neighbor in U . In other words, for every $v \in X$, there is $u \in U$ that is not adjacent to v . Two observations about set X will be useful. First, because U , and hence X , can be turned into a clique by adding at most k edges, we have that $|X| \leq 2k$. Second, the remaining vertices of U , namely, $Y = U \setminus X$, form a clique. For every set $S \in \mathcal{S}$, we define $S_X = X \cap S$ and $S_Y = Y \cap S$. Note that $S = S_X \cup S_Y$.

Because Y is a clique in G , no d -colorable subgraph from \mathcal{F} can contain more than d vertices from Y . Hence, $|S_Y| \leq d$.

Let $x = |S_X|$. Because $G[S_X]$ is an induced subgraph of some d -colorable graph $F \in \mathcal{F}$, we have that $G[S_X]$ is d -colorable. On the other hand, since $c\text{-comp}(G[U]) \leq k$, $G[S_X]$ can be turned into complete graph by adding at most k edges. These two conditions are used to estimate x . Let us recall that a *Turán graph* is the complete d -partite graph on x vertices whose partition sets differ in size by at most 1. According to Turán's theorem, see e.g. the book of Diestel [21], Turán graph has the maximum possible number of edges among all d -colorable graphs. The number of edges in Turán's graph is at most $\frac{1}{2}x^2 \frac{d-1}{d}$. Thus,

$$\binom{x}{2} - k \leq |E(G[S_X])| \leq \frac{1}{2}x^2 \frac{d-1}{d}$$

and

$$k \geq \binom{x}{2} - \frac{1}{2}x^2 \frac{d-1}{d} = \frac{x^2 - dx}{2d}.$$

Therefore,

$$x \leq \frac{d + \sqrt{d^2 + 8dk}}{2}.$$

We obtain that

$$|S| = |S_X| + |S_Y| \leq x + d \leq \frac{3d + \sqrt{d^2 + 8dk}}{2},$$

which proves the first claim of the lemma.

To prove the second claim, let $H = G[U]$. Observe that \bar{H} , the complement of H , has at most k edges. Consider $Z \subseteq V(H)$. If $|Z| \leq \frac{\sqrt{1+8k+1}}{2}$, then the minimum degree $\delta(\bar{H}[Z]) \leq \frac{\sqrt{1+8k-1}}{2}$. If $|Z| > \frac{\sqrt{1+8k+1}}{2}$, then

$$\delta(\bar{H}[Z]) \leq \frac{2|E(\bar{H}[Z])|}{|Z|} \leq \frac{4k}{\sqrt{1+8k+1}} = \frac{\sqrt{8k+1}-1}{2},$$

that is, the minimum degree of every induced subgraph of \bar{H} is at most $\frac{\sqrt{8k+1}-1}{2}$. Therefore, \bar{H} is $\frac{\sqrt{1+8k-1}}{2}$ -degenerate.

An induced subgraph of H with the vertex set S is d -colorable if and only if S can be partitioned into at most d independent sets. Equivalently, $H[S]$ is d -colorable if and only if S can be partitioned into at most d cliques of \bar{H} . By Lemma 2, \bar{H} has at most $2^{(\sqrt{8k+1}-1)/2} \cdot |U|$ cliques. Then U contains at most $(1 + 2^{(\sqrt{1+8k-1})/2} \cdot |U|)^d$ subsets S such that S can be partitioned into at most d cliques of \bar{H} .

To complete the proof, observe that the cliques of \bar{H} can be listed in time $2^{(\sqrt{1+8k-1})/2} \cdot n^{\mathcal{O}(1)}$ by Lemma 2. Then in $2^{\mathcal{O}(d\sqrt{k})} \cdot n^{\mathcal{O}(d)}$ time, we construct \mathcal{S}' by considering the unions of d cliques; the cliques can be the same or empty. \square

Let G be a graph and let F be an induced d -colorable subgraph of G . Informally, Lemma 3 says that for a given a k -almost chordal tree decomposition, every bag of this tree decomposition contains roughly $\mathcal{O}(d + \sqrt{dk})$ vertices of F . This statement combined with dynamic programming over the tree decomposition could easily bring us to the algorithm computing a maximum d -colored subgraph of G in time $n^{\mathcal{O}(d+\sqrt{dk})}$. However, this is not what we are shooting for; such an algorithm is not fixed-parameter tractable with parameter k . This is where the second part of the lemma becomes extremely helpful. Let us look at the family of all d -colorable induced subgraphs of G . Then the number of different intersections of the graphs from this family with a single bag of the tree decomposition is bounded by $2^{\mathcal{O}(d\sqrt{k})} \cdot n^{\mathcal{O}(d)}$. This allows us to bound the number of “partial solutions” in the dynamic programming, which in turn brings us to a parameterized subexponential algorithm. As an example of the applicability of Lemma 3, we give an algorithm for the following generic problem.

WEIGHTED d -COLORABLE SUBGRAPH

Input: Graph G with weight function $w: V(G) \rightarrow \mathbb{R}$.
Task: Find a d -COLORABLE induced subgraph H of G of maximum weight $\sum_{v \in V(H)} w(v)$.

For $d = 1$, this is the problem of finding an independent set of maximum weight, the WEIGHTED INDEPENDENT SET problem. For $d = 2$, this is the problem of finding an induced bipartite subgraph of the maximum weight, WEIGHTED BIPARTITE SUBGRAPH.

We prove the following theorem for WEIGHTED d -COLORABLE SUBGRAPH. Note that in Theorem 1 we do not require that the input graph is in CHORDAL- ke .

Theorem 1 Let $d \geq 1$ be an integer. For a given graph G with a nice k -almost chordal tree decomposition with $n^{\mathcal{O}(1)}$ bags, the WEIGHTED d -COLORABLE SUBGRAPH problem is solvable in time $2^{\mathcal{O}(\sqrt{k} \cdot d \log d)} \cdot n^{\mathcal{O}(d)}$.

Proof Let $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a nice k -almost chordal tree decomposition of G with $|V(T)| = n^{\mathcal{O}(1)}$. We perform dynamic programming over \mathcal{T} . Let us note that the width of the decomposition can be of order of n . The proof of the correctness for this dynamic programming is very similar to the one provided normally for graphs of bounded treewidth. However, the running time analysis is based on Lemma 3.

Recall that T is rooted at some node r , and for a node t of T , V_t denotes the union of all the bags present in the subtree of T rooted at t , including X_t . For vertex sets $X \subset X'$ of graph G , we say that a coloring c of $G[X]$ is *extendible* to a coloring c' of $G[X']$, if for every $x \in X$, $c(x) = c'(x)$.

For every node t , every $S \subseteq X_t$ such that $G[S]$ is d -colorable, every mapping $c : S \rightarrow \{1, \dots, d\}$ of $G[S]$, we define the following value:

$\text{cost}[t, S, c] =$ maximum possible weight of a set \hat{S} such that

$$S \subseteq \hat{S} \subseteq V_t, \hat{S} \cap X_t = S, \text{ and } c \text{ is a proper coloring of } G[S] \quad (1)$$

extendible to a proper d -coloring of $G[\hat{S}]$.

If c is not a proper coloring of $G[S]$ or if no such set \hat{S} exists, then we put $\text{cost}[t, S, c] = -\infty$. We also put $\text{cost}[t, \emptyset, c]$ be the maximum possible weight of a set \hat{S} such that $\hat{S} \subseteq V_t$, $\hat{S} \cap X_t = \emptyset$, and $G[\hat{S}]$ is d -colorable. Then $\text{cost}[r, \emptyset, c]$ is exactly the maximum weight of a d -colorable induced subgraph in G ; this is due to the fact that $V_r = V(G)$ and $X_r = \emptyset$.

Leaf node. If t is a leaf node, then we have $\text{cost}[t, \emptyset, c] = 0$. In this case, because the leaf node is an empty bag, the formula's correctness is trivial.

Introduce node. Suppose t is an introduce node with child t' such that $X_t = X_{t'} \cup \{v\}$ for some $v \notin X_{t'}$. Let S be any subset of X_t . If c is not a proper

d -coloring of $G[S]$, we put $\text{cost}[t, S, c] = -\infty$. Otherwise, we claim that the following formula holds:

$$\text{cost}[t, S, c] = \begin{cases} \text{cost}[t', S, c] & \text{if } v \notin S \\ \text{cost}[t', S \setminus \{v\}, c'] + \mathbf{w}(v) & \text{otherwise.} \end{cases} \quad (2)$$

Here c' is the coloring of $S \setminus \{v\}$ extendible to c .

Claim 1 *Formula (2) is correct.*

Proof of Claim 1 When $v \notin S$, the formula trivially holds. Suppose that $v \in S$. Let \hat{S} be a set maximizing $\text{cost}[t, S, c]$. Because coloring c' of $S \setminus \{v\}$ is extendible to coloring c of S , which in turn is extendible to a d -coloring of \hat{S} , we have that set $\hat{S} \setminus \{v\}$ is one of the sets considered in the definition of $\text{cost}[t', S \setminus \{v\}, c']$. Hence $\text{cost}[t', S \setminus \{v\}, c'] \geq \mathbf{w}(\hat{S} \setminus \{v\}) = \mathbf{w}(\hat{S}) - \mathbf{w}(v) = \text{cost}[t, S, c] - \mathbf{w}(v)$.

On the other hand, let \hat{S}' be a set for which the maximum is attained in the definition of $\text{cost}[t', S \setminus \{v\}, c']$. By the property of tree decompositions, all neighbors of v in V_t are contained in the bag X_t . This yields that set $\hat{S}' \cup \{v\}$ induces a graph whose d -coloring can be obtained by extending coloring c of S , therefore, $\text{cost}[t, S, c] \geq \mathbf{w}(\hat{S}' \cup \{v\}) = \text{cost}[t', S \setminus \{v\}, c'] + \mathbf{w}(v)$. We conclude that $\text{cost}[t, S, c] = \text{cost}[t', S \setminus \{v\}, c'] + \mathbf{w}(v)$. \square

To evaluate the running time required to compute (2), note that for fixed S and c , we have to verify whether c is a proper coloring of S , which can be done in time $\mathcal{O}(|S|^2)$ by going through all adjacencies of $G[S]$.

Forget node. Let t be a forget node with child t' such that $X_t = X_{t'} \setminus \{w\}$ for some $w \in X_{t'}$. Let S be any subset of X_t ; again we assume that c is a proper d -coloring of $G[S]$, since otherwise we put $\text{cost}[t, S, c] = -\infty$. We claim that the following formula holds:

$$\text{cost}[t, S, c] = \max \left\{ \text{cost}[t', S, c], \max_{c \text{ is extendible to } c'} \text{cost}[t', S \cup \{w\}, c'] \right\}. \quad (3)$$

Claim 2 *Formula (3) is correct.*

Proof of Claim 2 Because $V_t = V_{t'}$, we have that

$$\text{cost}[t, S, c] \geq \max \left\{ \text{cost}[t', S, c], \max_{c \text{ is extendible to } c'} \text{cost}[t', S \cup \{w\}, c'] \right\}.$$

On the other hand, let \hat{S} be a set for which the maximum is attained in the definition of $\text{cost}[t, S, c]$. If $w \notin \hat{S}$, then $\text{cost}[t, S, c] = \text{cost}[t', S, c]$. If $w \in \hat{S}$, then

$$\text{cost}[t, S, c] = \max_{c \text{ is extendible to } c'} \text{cost}[t', S \cup \{w\}, c'].$$

Thus

$$\text{cost}[t, S, c] \leq \max \left\{ \text{cost}[t', S, c], \max_{c \text{ is extendible to } c'} \text{cost}[t', S \cup \{w\}, c'] \right\}.$$

□

For running time, to compute (3), one has to go through all colorings c' of $S \cup \{w\}$ which are extensions of coloring c of S . Thus for every possible color i of w , we have to check if this color is compatible with the coloring by c the neighbors of w in S . This can be done in time $\mathcal{O}(|S|)$.

Join node. Finally, suppose that t is a join node with children t_1, t_2 such that $X_t = X_{t_1} = X_{t_2}$. Let S be any subset of X_t ; as before, we can assume that c is a proper d -coloring of $G[S]$. The claimed recursive formula is as follows:

$$\text{cost}[t, S, c] = \text{cost}[t_1, S, c] + \text{cost}[t_2, S, c] - \mathbf{w}(S). \quad (4)$$

Claim 3 *Formula (4) is correct.*

Proof of Claim 3 Let \widehat{S} be a set for which the maximum is attained in the definition of $\text{cost}[t, S, c]$ and Let $\widehat{S}_1 = \widehat{S} \cap V_{t_1}$ and $\widehat{S}_2 = \widehat{S} \cap V_{t_2}$. Because $\widehat{S} \supseteq \widehat{S}_i$, we have that $\text{cost}[t_i, S, c] \geq \mathbf{w}(\widehat{S}_i)$, $i = 1, 2$. This yields that

$$\text{cost}[t_1, S, c] + \text{cost}[t_2, S, c] \geq \mathbf{w}(\widehat{S}_1) + \mathbf{w}(\widehat{S}_2) = \mathbf{w}(\widehat{S}) - \mathbf{w}(S) = \text{cost}[t, S, c].$$

On the other hand, the union of sets maximizing the costs of $\text{cost}[t_1, S, c]$ and $\text{cost}[t_2, S, c]$, is a set whose d -coloring can be obtained from extending coloring c of S . Thus

$$\text{cost}[t_1, S, c] \geq \text{cost}[t_1, S, c] + \text{cost}[t_2, S, c] - \mathbf{w}(S).$$

□

The running time to compute (4) for fixed set S and coloring c is proportional to the time required to check whether c is a proper coloring. Again, this can be done in time $\mathcal{O}(|S|^2)$.

This concludes the description and the proof of correctness of the recursive formulas for computing the values of $\text{cost}[\cdot, \cdot, \cdot]$. The optimal subgraph of the maximum weight can be found by standard backtracking arguments. Let us now estimate the total running time. The running time of our dynamic programming algorithm, up to a multiplicative factor $\mathcal{O}(|S|^2)$, is dominated by the number of triples $[t, S, c]$. The number t is in $n^{\mathcal{O}(1)}$. Every set S should induce a d -colorable subgraph, so we can restrict our attention only to sets of the form $X_t \cap V(F)$ for some d -colorable graph F . By Lemma 3, each of these sets is of size at most $d + \frac{d+\sqrt{d^2+8dk}}{2}$ and the total number of such sets S for each bag X_t is $2^{\mathcal{O}(d\sqrt{k})} \cdot n^{\mathcal{O}(d)}$ and they can be listed in $2^{\mathcal{O}(d\sqrt{k})} \cdot n^{\mathcal{O}(d)}$ time. Thus, the number of d -colorings c of each of the sets S is $d^{\mathcal{O}(|S|)} = d^{\mathcal{O}(d+\sqrt{dk})}$. Hence the total running time of the dynamic programming is $2^{\mathcal{O}(\sqrt{k} \cdot d \log d)} \cdot n^{\mathcal{O}(d)}$. □

Combining Proposition 2, Lemma 1 and Theorem 1, we immediately obtain the following corollary. Recall that $A \subseteq \binom{V(G)}{2} \setminus E(G)$ is a CHORDAL-modulator if $G + A$ is a chordal graph.

Corollary 1 WEIGHTED d -COLORABLE SUBGRAPH on a graph $G \in \text{CHORDAL-ke}$ is solvable in time $2^{\mathcal{O}(\sqrt{k}(\log k + d \log d))} \cdot n^{\mathcal{O}(d)}$. Moreover, the problem can be solved in $2^{\mathcal{O}(\sqrt{k} \cdot d \log d)} \cdot n^{\mathcal{O}(d)}$ time if a CHORDAL-modulator of size at most k is given.

By Corollary 1, we immediately derive the following.

Corollary 2 WEIGHTED INDEPENDENT SET and WEIGHTED BIPARTITE SUBGRAPH on $G \in \text{CHORDAL-ke}$ are solvable in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$. Moreover, the problems can be solved in $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time if a CHORDAL-modulator of size at most k is given.

In the WEIGHTED VERTEX COVER, we are given a weighted graph G , and the task is to find a vertex cover of minimum weight, that is, a set of vertices X such that every edge of G has at least one endpoint in G . Similarly, in the WEIGHTED ODD CYCLE TRANSVERSAL, we are asked to find a minimum weight set of vertices such that every cycle of odd length contains at least one vertex from the set. Since the complement of every independent set is a vertex cover, and the complement of every induced bipartite subgraph is an odd cycle transversal, we have the following corollary.

Corollary 3 WEIGHTED VERTEX COVER and WEIGHTED ODD CYCLE TRANSVERSAL on CHORDAL-ke graphs are solvable in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$. Moreover, the problems can be solved in $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time if a CHORDAL-modulator of size at most k is given.

3.1 Various Extentions

We use the technique developed to prove Theorem 1 for obtaining subexponential algorithms for other problems beyond WEIGHTED d -COLORABLE SUBGRAPH. These algorithms are very similar to the one from Theorem 1 and we sketch here only a few problems and the adjustments required to apply the dynamic programming from Theorem 1 to these problems.

A homomorphism $G \rightarrow H$ from a graph G to a graph H is a mapping from the vertex set of G to that of H such that the image of every edge of G is an edge of H . In other words, a homomorphism $G \rightarrow H$ exists if and only if there is a mapping $g : V(G) \rightarrow V(H)$, such that for every edge $uv \in E(G)$, we have $g(u)g(v) \in E(H)$. There is a homomorphism from G to a complete graph K_d on d vertices if and only if G is d -colorable. Because of that, deciding whether there is a homomorphism from G to H is often referred to as the H -coloring of G . Note that if G admits an H -coloring, then G is $|V(H)|$ -colorable.

The main difference between solving WEIGHTED H -COLORABLE SUBGRAPH, the problem of finding the maximum weight induced subgraph admitting a homomorphism to H , with Theorem 1 is that the value $\text{cost}[t, S, c]$ in (1) should be

redefined. In order to find a maximum weight H -colorable induced subgraph of a graph G , we need to compute

$\text{cost}[t, S, g] = \text{maximum possible weight of a set } \hat{S} \text{ such that}$

$$\begin{aligned} S \subseteq \hat{S} \subseteq V_t, \hat{S} \cap X_t = S, \text{ and } g \text{ is a homomorphism from } G[S] \\ \text{to } H \text{ extendible to a homomorphism of } G[\hat{S}] \text{ to } H. \end{aligned}$$

The number of homomorphisms g from $G[S]$ to H does not exceed $|V(H)|^{|S|}$. The running time of operations join, introduce and forget operations in the dynamic programming is bounded by a polynomial of the number of states. Hence we can solve the problem in time $2^{\mathcal{O}(\sqrt{k}(\log k + |V(H)| \log |V(H)|))} \cdot n^{\mathcal{O}(|V(H)|)}$.

Similar running times could be derived for the variants of WEIGHTED d -COLORABLE SUBGRAPH where some additional constraints on the properties of the d -colorable induced subgraph of minimum weight are imposed by some property \mathcal{C} . For example, property \mathcal{C} could be that the required subgraph is connected, acyclic, regular, degenerate, etc. As far as the information of the partial solution required for property \mathcal{C} is characterized by set $S \subseteq V_t$ and all possible subsets of S or all permutations of S , we can solve the corresponding problem in time $2^{\mathcal{O}((d\sqrt{k}) \log(dk))} \cdot n^{\mathcal{O}(d)}$.

As a concrete example, consider WEIGHTED d -DEGENERATE SUBGRAPH, whose task is to find a maximum weight d -degenerate induced subgraph H of the input graphs G . Recall that a graph is d -degenerate for a nonnegative integer d , if every induced subgraph of G has a vertex of degree at most d . In particular, every forest is 1-degenerate. Recall also that a graph G is d -degenerate if and only if it admits a d -coloring ordering [23]. In particular, this immediately implies that every d -degenerate graph is $(d+1)$ -colorable. That is, the chromatic number bound is automatically given by the d -degeneracy property.

Let G be a graph and let π be an ordering of its vertices. Let $X \subseteq V(G)$. We say that π as an *extension* of an ordering π' of X if the vertices of X occur in π in the same order as in π' . Suppose that $X = \{x_1, \dots, x_r\}$ and the vertices of X are indexed with respect to their order in π . Denote by $\delta_\pi(G, X) = (\delta_1, \dots, \delta_r)$ the sequence of nonnegative integers such that for each $i \in \{1, \dots, r\}$, δ_i is the degree of x_i in the graph obtained from G by the deletion of the vertices that occur before x_i in π . Notice that if π is a d -coloring ordering of G , then $\delta_i \leq d$ for $i \in \{1, \dots, r\}$. We use the following lemma to construct our algorithm for WEIGHTED d -DEGENERATE SUBGRAPH.

Lemma 4 *Let G be a graph and let $V_1, V_2 \subseteq V(G)$ such that $V_1 \cup V_2 = V(G)$. Let also $X = V_1 \cap V_2$ with $r = |X|$. Then G has a d -coloring ordering if and only if there is an ordering π of X and orderings π_1 and π_2 of V_1 and V_2 respectively such that*

- (i) π_i is an extension of π for $i = 1, 2$,
- (ii) for the sequences $\delta_\pi(G[X], X) = (\delta_1, \dots, \delta_r)$, $\delta_{\pi_1}(G[V_1], X) = (\delta_1^{(1)}, \dots, \delta_r^{(1)})$ and $\delta_{\pi_2}(G[V_2], X) = (\delta_1^{(2)}, \dots, \delta_r^{(2)})$, it holds that $\delta_i^{(1)} + \delta_i^{(2)} - \delta_i \leq d$ for every $i \in \{1, \dots, r\}$.

Proof Suppose that π' is a d -coloring ordering. Then we define π , π_1 and π_2 as the orderings of X , V_1 and V_2 respectively as the ordering inherited from π , that is, every two vertices of the corresponding sets occur exactly in the same order as in π . It is straightforward to verify (i) and (ii). For the opposite direction, assume that there are π , π_1 and π_2 satisfying (i) and (ii). Let $\pi = (x_1, \dots, x_r)$. We construct the ordering π' of G as follows. First, we concatenate the suborderings of π_1 and π_2 containing the vertices that occur before x_1 in these orderings and add x_1 in the end. Then for $i = 2, \dots, r$, we consecutively add suborderings of π_1 and π_2 composed by the vertices that occur between x_{i-1} and x_i and add x_i in the end. Finally, we add suborderings of π_1 and π_2 with the vertices that are after x_r . It is easy to see that π' is a d -coloring ordering of G because of (i) and (ii). \square

Using Lemma 4, we can define the values computed by our dynamic programming algorithm for WEIGHTED d -DEGENERATE SUBGRAPH. For a non-negative integer r , we define $\Delta_r = \{(\delta_1, \dots, \delta_r) \mid 0 \leq \delta_i \leq d, 1 \leq i \leq r\}$, i.e., Δ_r is the set of all sequences of length r of nonnegative integers at most d . Similarly to (1), for every node t , every $S \subseteq X_t$ such that $G[S]$ is $(d+1)$ -colorable, every ordering π of S and every $\delta \in \Delta_{|S|}$, we define

$\text{cost}[t, S, \pi, \delta] = \text{maximum possible weight of a set } \hat{S} \text{ such that}$

$$\begin{aligned} S \subseteq \hat{S} \subseteq V_t, \hat{S} \cap X_t = S, \text{ and ordering } \pi \text{ of } S \text{ can be extended} \\ \text{to a } d\text{-coloring ordering } \pi' \text{ of } G[\hat{S}] \text{ with } \delta_{\pi'}(G[\hat{S}], S) = \delta. \end{aligned}$$

For a given $S \subseteq X_t$, there are $|S|!$ different ordering of the vertices of S and $|\Delta_{|S|}| = (d+1)^{|S|}$. By Lemma 3, we have that $|S| \leq \frac{3d + \sqrt{d^2 + 8dk}}{2}$ and, therefore, we consider $2^{\mathcal{O}((d + \sqrt{dk}) \log(dk))}$ orderings π and $2^{\mathcal{O}((d + \sqrt{dk}) \log d)}$ sequences δ . Taking into account that by Lemma 3, we consider $2^{\mathcal{O}(d\sqrt{k})} \cdot n^{\mathcal{O}(d)}$ sets S , we obtain that the table of our dynamic programming algorithm stores $2^{\mathcal{O}((d\sqrt{dk}) \log(kd))} \cdot n^{\mathcal{O}(d)}$ values. Applying Lemma 4 and the standard dynamic programming arguments, it is easy to show that we can solve WEIGHTED d -DEGENERATE SUBGRAPH using the information stored in the tables.

Many natural problems can be described by combining connectivity, degeneracy, or degree constraints. Examples of such problems are various maximization problems like WEIGHTED INDUCED TREE, WEIGHTED INDUCED PATH, WEIGHTED INDUCED CYCLE, and various packing variants of these problems like WEIGHTED INDUCED CYCLE PACKING, the problem of finding a maximum induced subgraph whose each connected component is a cycle.

We summarize these observations with the following theorem.

Theorem 2 *Let $d \geq 1$ be an integer and G be a graph from CHORDAL-ke. Then*

- WEIGHTED H -COLORABLE SUBGRAPH can be solved in $2^{\mathcal{O}(\sqrt{k}(\log k + |V(H)| \log |V(H)|))} \cdot n^{\mathcal{O}(|V(H)|)}$ time,
- WEIGHTED d -DEGENERATE SUBGRAPH is solvable in time $2^{\mathcal{O}((d\sqrt{k}) \log(dk))} \cdot n^{\mathcal{O}(d)}$,

and

- WEIGHTED INDUCED FOREST (WEIGHTED FEEDBACK VERTEX SET),
- WEIGHTED INDUCED TREE,
- WEIGHTED INDUCED PATH (CYCLE)
- WEIGHTED INDUCED CYCLE PACKING,

are solvable in $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ time.

In some cases, we can obtain a better running time if a CHORDAL-modulator of size at most k is given. For WEIGHTED H -COLORABLE SUBGRAPH, this is done in the same way as for WEIGHTED d -COLORABLE SUBGRAPH. For some other problems, like WEIGHTED INDUCED FOREST (WEIGHTED FEEDBACK VERTEX SET), this would demand using recent techniques for dynamic programming on graphs of bounded treewidth for problems with connectivity constraints (see [6, 20, 25, 56]) but this goes beyond the scope of our paper.

Another extension of Theorem 1 can be derived from the very recent results of Baste, Sau and Thilikos [3] about the \mathcal{F} -MINOR DELETION problem on graphs of bounded treewidth. Recall that a graph F is a *minor* of G if a graph isomorphic to F can be obtained from G by vertex and edge deletions and edge contractions. Respectively, G is said to be F -minor free if G does not contain F as a minor. For a family of graphs \mathcal{F} , G is \mathcal{F} -minor free if G is F -minor free for every $F \in \mathcal{F}$. For a family \mathcal{F} , the task of \mathcal{F} -MINOR DELETION is, given a graph G , to find a minimum set of vertices X such that $G - X$ is \mathcal{F} -minor free. Then \mathcal{F} -MINOR DELETION is equivalent to \mathcal{F} -MINOR FREE INDUCED SUBGRAPH, whose task is to find a maximum \mathcal{F} -minor free induced subgraph of G . A family of graphs \mathcal{F} is *connected* if every $F \in \mathcal{F}$ is a connected graph. Baste et al. [3] obtained, in particular, the following result.

Proposition 5 [3] *Let \mathcal{F} be a finite connected family of graphs. Then \mathcal{F} -MINOR DELETION can be solved in time $2^{\mathcal{O}(w \log w)} \cdot n^{\mathcal{O}(1)}$ on graphs of treewidth at most w .*¹

It is well-known (see, e.g., the book of Nesetril and de Mendez [54] for the inclusion relations between the classes of sparse graphs) that if \mathcal{F} is a finite family, then there is a positive integer d such that every \mathcal{F} -minor free graph is d -degenerate. This means that for a finite family \mathcal{F} , \mathcal{F} -minor free graphs are d -colorable for some constant d that depends on \mathcal{F} only. We use Lemma 3 and then combine our approach from Theorem 1 with the techniques of Baste et al. [3]. Using Lemma 5. Hence the following theorem.

Theorem 3 *Let \mathcal{F} be a finite connected family of graphs. Let also G be a graph from CHORDAL-ke. Then \mathcal{F} -MINOR FREE INDUCED SUBGRAPH (or, equivalently, \mathcal{F} -MINOR DELETION) can be solved in $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ time.*

¹ the constants hidden in the big- \mathcal{O} notation depend on \mathcal{F} .

For example, this framework encompasses such problems as INDUCED PLANAR SUBGRAPH or INDUCED OUTERPLANAR SUBGRAPH whose task is to find a subgraph of maximum size that is planar or outerplanar, respectively.

With a small adjustment, the dynamic programming applies to the problems with specific requirements on the complement of the maximum induced d -colored subgraph. For example, consider the following problem. A set of vertices $S \subseteq V(G)$ is a *connected vertex cover* if S is a vertex cover and $G[S]$ is connected. Then in the WEIGHTED CONNECTED VERTEX COVER problem, we are given a graph G with a weight function $w : V(G) \rightarrow \mathbb{Z}^+$, and the task is to find a connected vertex cover in G of minimum weight. Similarly, WEIGHTED CONNECTED FEEDBACK VERTEX SET is the problem of finding a connected feedback vertex set of minimum weight.

The complement of every vertex cover is an independent set, that is a 1-colorable subgraph, and the complement of every feedback vertex set is a forest, hence 2-colorable subgraph. While the connectivity constraints are not on the maximum induced subgraph but its complement, our previous arguments can be adapted to handle these problems.

Theorem 4 WEIGHTED CONNECTED VERTEX COVER and WEIGHTED CONNECTED FEEDBACK VERTEX SET are solvable in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ on CHORDAL- ke .

Proof (sketch) The proof is similar to the proof of Theorem 1, so we only briefly sketch the main idea. We also give the sketch for WEIGHTED CONNECTED VERTEX COVER only, the solution to WEIGHTED CONNECTED FEEDBACK VERTEX SET is basically the same.

Let (G, w, W) be an instance of WEIGHTED CONNECTED VERTEX COVER.

We use Lemma 1 to find a nice k -almost chordal tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of G with $\mathcal{O}(n^2)$ bags. Let r be the root of T and V_t be the union of the bags in the subtree of T rooted in t .

The following claim is crucial for our algorithm.

Claim 4 Let t be a node of T and let $S \subseteq X_t$. Then $G[S]$ has at most $\frac{3+\sqrt{1+8k}}{2}$ connected components.

To prove the claim, observe that any set of vertices constructed by picking an arbitrary vertex from each connected component of $G[S]$ is independent. Hence, the number of connected components of $G[S]$ is upper bounded by the maximum size of an independent set in X_t . By Lemma 3, any independent subset of X_t has size at most $\frac{3+\sqrt{1+8k}}{2}$ and the claim follows.

Let $t \in T$. For a set $S \subseteq X_t$, denote by $\mathcal{P}_t(S)$ the set of all partitions of S such that every two vertices $x, y \in S$ that are in the same component are in the same set of each partition; we assume that $\mathcal{P}_t(\emptyset) = \{\emptyset\}$.

Notice that $S \subseteq V(G)$ is a connected vertex cover if and only if $U = V(G) \setminus S$ is an independent set and $G[S]$ is a connected graph.

For every $t \in V(T)$, every independent set $U \subseteq X_t$ and every $P \in \mathcal{P}_t(S)$ for $S = X_t \setminus U$, our algorithm computes the value $\text{cost}[t, U, P]$ that is the minimum possible weight of a set $\hat{S} \subseteq V_t$ such that

- $\hat{S} \cap X_t = S = X_t \setminus U$,
- \hat{S} is a vertex cover of $G[V_t]$,
- for each connected component C of $G[\hat{S}]$, $V(C) \cap S \neq \emptyset$ unless $S = \emptyset$,
- every two vertices $x, y \in S$ are in the same set of P if and only if they belong to the same connected component of $G[\hat{S}]$.

We compute $\text{cost}[t, U, P]$ bottom-up for $t \in V(T)$ starting from the leaves. The computations are trivial for leaf nodes and are performed in a standard way for introduce, forget and join nodes. Finally, we compute $\text{cost}[r, \emptyset, \{\emptyset\}]$, which gives us the minimum weight of a connected vertex cover.

By Lemma 3, X_t has $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ independent subsets. By Claim 4, we have that for every $S \subseteq X_t$, $G[S]$ has at most $\frac{3+\sqrt{1+8k}}{2}$ connected components. Therefore, $|\mathcal{P}_t(S)| = 2^{\mathcal{O}(\sqrt{k} \log k)}$ and the algorithm computes $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ values of $\text{cost}[t, U, P]$ for each $t \in V(T)$. Therefore, the running time of the algorithm is $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$. \square

In this section, we discussed optimization problems but, in many cases, similar dynamic programming can be applied for counting problems. For example, we can compute the number of (inclusion) maximal independent sets, maximal bipartite subgraphs, minimal (connected) feedback vertex sets, minimal connected vertex covers in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ on CHORDAL-*ke*.

4 Beyond Induced d -colorable Subgraphs

In Sect. 3, among other algorithms, we gave a subexponential (in k) algorithm on CHORDAL-*ke* graphs computing a maximum d -colorable subgraph. In particular, this also implies that for every fixed d , deciding whether a graph from CHORDAL-*ke* is d -colorable, can be done in time subexponential in k . In this section, we show that two fundamental problems, namely, COLORING and CLIQUE, while still being FPT, but unlikely be solvable in subexponential parameterized time.

4.1 Coloring CHORDAL-*ke* Graphs

First, we consider COLORING whose task is, given a graph G and a positive integer ℓ , decide whether the chromatic number of G is at most ℓ , that is, if G is ℓ -colorable. Note that ℓ here is not a fixed constant as in Sect. 3 and may be arbitrarily large.

Cai [12] proved that COLORING is FPT (parameterized by k) on SPLIT-*ke* graphs. We generalize his result by showing that COLORING is FPT on CHORDAL-*ke*. Our approach is based on the dynamic programming which is similar to the one we used in Sect. 3.

We need the following property of graph colorings. Let G be a graph. It is well-known that an ℓ -coloring of a graph G can be seen as a partition \mathcal{X} of $V(G)$ into at most ℓ independent sets formed by the vertices of the same color that are called *color classes*. We call \mathcal{X} an ℓ -coloring partition of G . We also say that a partition \mathcal{X} of $V(G)$ into independent sets is a *coloring partition* of G . Let $\mathcal{X} = \{X_1, \dots, X_\ell\}$ be a coloring partition of G and let $U \subseteq V(G)$. We say that the partition $\mathcal{X}|_U$ of U formed by nonempty sets $U \cap X_1, \dots, U \cap X_\ell$ is a *projection* of \mathcal{X} on U ; we assume that \mathcal{X}_\emptyset contains the unique element \emptyset .

Lemma 5 *Let $G \in \text{CHORDAL-ke}$ and let $A \subseteq \binom{V(G)}{2}$ such that $|A| \leq k$ and $G' = G + A$ is a chordal graph. Let also C be a clique of G' . Then there are at most $(2k)^{2k}$ partitions of C into independent in G sets and these partitions can be enumerated in $2^{\mathcal{O}(k \log k)} \cdot n$ time.*

Proof Let S be the set of end-vertices of the elements of A in C . Then $C' = C \setminus S$ is a clique of G such that each vertex v of C is adjacent to every vertex of C' distinct from v . Therefore, for every coloring partition \mathcal{X} of G , the vertices of C' form single-element sets of the projection of \mathcal{X} on C . Therefore, only the vertices of S may be included in nontrivial sets of the partitions. Since $|S| \leq 2k$, there are at most $(2k)^{2k}$ partitions of S into independent sets. Hence, there are at most $(2k)^{2k}$ partitions of C . To enumerate all the partitions, we brute-force through all partitions of S and verify for each partition whether this is a partition of S into independent sets. Since $|C| \leq n$, the enumeration can be done in $2^{\mathcal{O}(k \log k)} \cdot n$ time. \square

Lemma 5 implies that there are at most $(2k)^{2k}$ projections of the coloring partitions of G on C and these projections can be enumerated in $2^{\mathcal{O}(k \log k)} \cdot n$ time. This allows to construct a dynamic programming algorithm for COLORING.

Theorem 5 *COLORING can be solved in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ on CHORDAL-ke.*

Proof The algorithm follows the same routine as the algorithms from Sect. 3. We only briefly sketch the main idea.

Let (G, ℓ) be an instance of COLORING. By Lemma 1, one can construct a nice k -almost chordal tree decomposition $T = (T, \{X_t\}_{t \in V(T)})$ of G with $\mathcal{O}(n^2)$ bags. Let r be the root of T and recall that V_t denotes the union of the bags in the subtree of T rooted in t . For $t \in V(T)$, let \mathcal{Y}_t be the family of all partitions of X_t into independent sets. We put $\mathcal{Y}_t = \{\emptyset\}$ if $X_t = \emptyset$.

For every $\mathcal{X} \in \mathcal{Y}_t$, we define

$$\text{Col}_t(\mathcal{X}) = \begin{cases} \text{true} & \text{if there is an } \ell\text{-coloring partition } \mathcal{X}' \text{ of } G[V_t] \text{ s.t.} \\ & \mathcal{X}'|_{X_t} = \mathcal{X}, \\ \text{false} & \text{otherwise.} \end{cases}$$

We compute the table of values of $\text{Col}_t(\mathcal{X})$ bottom-up for $t \in V(V)$ starting from the leaves. Clearly, G is ℓ -colorable if and only if $\text{Col}_t(\emptyset) = \text{true}$.

Leaf node. Computing the tables for leaves is trivial as the bags are empty, and we assume that the empty graph is ℓ -colorable.

From now on, we assume that a node $t \in V(T)$ has children, and the tables are already constructed for them.

Introduce node. Let t be an introduce node of T . Denote by t' its child and assume that $X_t = X_{t'} \cup \{v\}$ for $v \notin X_{t'}$. For $\mathcal{X} \in \mathcal{Y}_t$, we define $\mathcal{X} - v$ as the partition of $X_{t'}$ obtained from \mathcal{X} either by the deletion of $\{v\}$ if $\{v\}$ is an element of \mathcal{X} or by the deletion of v from a nontrivial set of \mathcal{X} containing v . For every $\mathcal{X} \in \mathcal{Y}_t$, we set

$$\text{Col}_t(\mathcal{X}) := (|\mathcal{X}| \leq \ell) \wedge \text{Col}_{t'}(\mathcal{X} - v).$$

Forget node. Let t be a forget node of T . Denote by t' its child and assume that $X_t = X_{t'} \setminus \{v\}$ for $v \in X_{t'}$. For $\mathcal{X} \in \mathcal{Y}_t$, we define $\mathcal{X} + v \subseteq \mathcal{Y}_{t'}$ to be the sets of all partitions \mathcal{X}' of $X_{t'}$ into independent sets such that $\mathcal{X} = \mathcal{X}' - v$, that is, every $\mathcal{X}' \in \mathcal{X} + v$ is either obtained by adding the single-element set $\{v\}$ or by including v into one of the independent set. For every $\mathcal{X} \in \mathcal{Y}_t$, we set

$$\text{Col}_t(\mathcal{X}) := \bigvee_{\mathcal{X}' \in \mathcal{X} + v} \text{Col}_{t'}(\mathcal{X}').$$

Join node. Let t be a join node of T with children t_1 and t_2 . Then $\mathcal{Y}_t = \mathcal{Y}_{t_1} = \mathcal{Y}_{t_2}$. For every $\mathcal{X} \in \mathcal{Y}_t$,

$$\text{Col}_t(\mathcal{X}) := \text{Col}_{t_1}(\mathcal{X}) \wedge \text{Col}_{t_2}(\mathcal{X}).$$

The correctness of the computation of the tables of values of $\text{Col}_t(\mathcal{X})$ for $t \in V(T)$ follows by standard arguments. By Lemma 5, we have that each table has at most $(2k)^{2k} = 2^{\mathcal{O}(k \log k)}$ elements. By the same lemma, together with the description of the computing the tables for the leave and the introduce, forget and join nodes, implies that the computation of the table for every $t \in V(T)$ can be done in time $2^{\mathcal{O}(k \log k)} n$. Therefore, the total running time is $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$. \square

Now we show that it is unlikely that COLORING is solvable in subexponential in k time. For this, we establish the complexity lower bound based on ETH. We use the result of Komusiewicz and Uhlmann [46] for the auxiliary TRIANGLE COVER problem that, given a graph G with $n = 3p$ vertices, asks whether $V(G)$ can be covered by p disjoint triangles, that is, by p copies of K_3 that are subgraphs of G .

Proposition 6 [46] *Unless ETH is false, TRIANGLE COVER cannot be solved in time $2^{o(n+m)} \cdot (n+m)^{\mathcal{O}(1)}$ even when the input restricted to the graphs without induced K_4 .*

Now we rule out the existence of a subexponential algorithm for COLORING on CHORDAL- ke parameterized by k . We show a stronger claim.

Theorem 6 COLORING cannot be solved in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ on graphs in COMPLETE–ke unless ETH fails.

Proof A set of cliques $\{C_1, \dots, C_\ell\}$ of a graph G is a *clique cover* if the cliques are pairwise disjoint and $V(G) = \bigcup_{i=1}^\ell C_i$. It is well-known that G is ℓ -colorable if and only if its complement \overline{G} has a clique cover of size ℓ . Let G be a graph with $n = 3\ell$ vertices that has no induced K_4 . Observe that G has a clique cover of size ℓ if and only if $V(G)$ can be covered by ℓ cliques of size 3, that is, by triangles. Consider a graph G with $n = 3\ell$ vertices such that \overline{G} has no induced K_4 . We obtain that (G, ℓ) is a yes-instance of COLORING if and only if \overline{G} is a yes-instance of TRIANGLE COVER. Observe that $G \in \text{COMPLETE–ke}$ if and only if \overline{G} has at most k edges. Then the existence of an algorithm for COLORING running in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ on graphs in COMPLETE–ke would imply that TRIANGLE COVER can be solved in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ on graphs without induced K_4 with at most k edges. This is impossible unless ETH is false by Proposition 6. \square

4.2 Clique in CHORDAL–ke Graphs

Now we consider the CLIQUE problem that asks, given a graph G and a positive integer ℓ , whether G has a clique of size at least ℓ . We show that CLIQUE is FPT on CHORDAL–ke when parameterized by k even for the weighted variant of the problem in Sect. 5 by demonstrating that the problem admits a Turing kernel. Here, we prove the following lower bound.

Theorem 7 CLIQUE cannot be solved in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ on graphs in COMPLETE–ke unless ETH fails.

Proof Observe that a graph G has a clique of size at least ℓ if and only if \overline{G} has an independent set of size at least ℓ . Recall that $G \in \text{COMPLETE–ke}$ if and only if \overline{G} has at most k edges. It was shown by Impagliazzo, Paturi and Zane [40] that INDEPENDENT SET cannot be solved in time $2^{o(n+m)}n^{\mathcal{O}(1)}$ on graphs with n vertices and m edges unless ETH fails. These observations immediately imply the claim of the theorem. \square

We proved that COLORING and CLIQUE do not admit subexponential algorithms on COMPLETE–ke, when parameterized by k , unless ETH fails. Note that by Observation 1, this means that subexponential algorithms cannot solve these problems on CHORDAL–ke as well unless ETH fails.

5 Kernelization on CHORDAL–ke

In this section we discuss kernelization of the problems considered in the previous section.

Jansen and Bodlaender in [42] and Bodlaender, Jansen, and Kratsch in [8] proved that WEIGHTED INDEPENDENT SET, WEIGHTED VERTEX COVER, WEIGHTED BIPARTITE SUBGRAPH, WEIGHTED ODD CYCLE TRANSVERSAL, WEIGHTED FEEDBACK VERTEX SET and CLIQUE do not admit a polynomial kernel parameterized by the size of the minimum vertex cover of the graph unless $\text{coNP} \subseteq \text{NP/poly}$.

We use the following observation.

Observation 2 *If G has a vertex cover of size at most k , then $\text{split-comp}(G) \leq \binom{k}{2}$.*

Proof Let G be a graph with a vertex cover X of size at most k . Let $A = \binom{X}{2} \setminus E(G[X])$. Clearly, $G + A$ is a split graph. Therefore, $\text{split-comp}(G) \leq \binom{k}{2}$. \square

Observation 2 and the results of [8, 42] yield the following proposition.

Proposition 7 WEIGHTED INDEPENDENT SET, WEIGHTED VERTEX COVER, WEIGHTED BIPARTITE SUBGRAPH, WEIGHTED ODD CYCLE TRANSVERSAL, WEIGHTED FEEDBACK VERTEX SET and CLIQUE do not admit a polynomial in k kernel on SPLIT-ke graphs unless $\text{coNP} \subseteq \text{NP/poly}$.

By Observation 1, these problems parameterized by k have no polynomial kernel on CHORDAL-ke as well unless $\text{coNP} \subseteq \text{NP/poly}$.

These results do not refute the existence of polynomial Turing kernels. We show that WEIGHTED CLIQUE has such a kernel. The input of WEIGHTED CLIQUE contains a graph G together with a weight function $w : V(G) \rightarrow \mathbb{Z}^+$ and a nonnegative integer W , and the task is to decide whether G has a clique C of weight at least W .

Our kernelization algorithm uses the following well-known property of chordal graphs.

Proposition 8 [34, 61] *An n -vertex chordal graph has at most n inclusion-maximal cliques, and they can be listed in linear time.*

Theorem 8 WEIGHTED CLIQUE on CHORDAL-ke parameterized by k admits a Turing kernel with at most $16k^2$ vertices with size $\mathcal{O}(k^8)$.

Proof Let (G, w, W) be an instance of WEIGHTED CLIQUE.

We apply Proposition 1 to approximate the fill-in of G . If the algorithm reports that $\text{fill-in}(G) > k$, we report that $G \notin \text{CHORDAL-ke}$ and stop. Assume that this is not the case. Then the algorithm returns a set $A \subseteq \binom{V(G)}{2}$ of size at most $8k^2$ such that $G' = G + A$ is a chordal graph. Let X be the set of vertices that are the end-vertices of the edges of A . Note that $|X| \leq 16k^2$. Then we use Proposition 8 to list all the inclusion maximal cliques C_1, \dots, C_r of G' . Let $X_i = X \cap C_i$ for $i \in \{1, \dots, r\}$. Observe that if C is a clique of G of weight at least W , then $C \subseteq C_i$ for some

$i \in \{1, \dots, r\}$. Observe also that each $G[C_i]$ contains a clique of weight at least W if and only if $G_i = G[X_i]$ contains a clique of weight at least $W_i = \max\{0, W - w(C_i \setminus X_i)\}$. Trivially, $\text{fill-in}(G_i) \leq \text{fill-in}(G)$ for $i \in \{1, \dots, r\}$, that is, each $G_i \in \text{CHORDAL-ke}$ if $G \in \text{CHORDAL-ke}$.

Observe that (G, w, W) is a yes-instance of **WEIGHTED CLIQUE** if and only if (G_i, w, W_i) is a yes-instance for at least one $i \in \{1, \dots, r\}$. Then Turing kernelization algorithm solves **WEIGHTED CLIQUE** by calling the oracle for each instance (G_i, w, W_i) with the parameter k for $i \in \{1, \dots, r\}$.

Since $|V(G_i)| = |X_i| \leq |X| \leq 16k^2$ for $i \in \{1, \dots, r\}$, we solve instances with the input graphs of bounded size. To call the oracle that solves the instances of bounded size, we have to compress the weights. For this, we apply Proposition 3.

Let $i \in \{1, \dots, r\}$ and consider the instance (G_i, w, W_i) . Let v_1, \dots, v_p be the vertices of G_i . Using the notation of Proposition 3, let $h = p + 1$ and $N = p + 2$. Consider vector $w = (w(v_1), \dots, w(v_p), W_i)^\top \in \mathbb{Z}^h$. The algorithm of Frank and Tardos finds a vector $\bar{w} = (\bar{w}_1, \dots, \bar{w}_p, \bar{W}_i)^\top$ with $\|\bar{w}\|_\infty \leq 2^{4h^3} N^{h(h+2)}$ such that $\text{sign}(w \cdot b) = \text{sign}(\bar{w} \cdot b)$ for all vectors $b \in \mathbb{Z}^h$ with $\|b\|_1 \leq N - 1$. We define $\bar{w}(v_j) = \bar{w}_j$ for $j \in \{1, \dots, p\}$ and consider the instance $(G_i, \bar{w}, \bar{W}_i)$ of **WEIGHTED INDEPENDENT SET**.

Since $\text{sign}(w \cdot b) = \text{sign}(\bar{w} \cdot b)$ for all vectors $b \in \mathbb{Z}^h$ with $\|b\|_1 \leq N - 1$, the equality holds for every b whose elements are in $\{-1, 0, 1\}$. This implies that the weights $\bar{w}(v_j)$ are positive and \bar{W}_i is nonnegative. Also we have that for every set of indices $I \subseteq \{1, \dots, p\}$, $\sum_{j \in I} w(v_j) \leq W_i$ if and only if $\sum_{j \in I} \bar{w}(v_j) \leq \bar{W}_i$. This proves that the instances (G_i, w, W_i) and $(G_i, \bar{w}, \bar{W}_i)$ are equivalent.

Since $|V(G_i)| \leq 16k^2$, we obtain that the weights in the instance $(G_i, \bar{w}, \bar{W}_i)$ can be encoded by strings of length $\mathcal{O}(k^6)$ for $i \in \{1, \dots, r\}$. Hence, the size of each instance is $\mathcal{O}(k^8)$.

Propositions 1, 8 and 3 immediately imply that the kernelization algorithm runs in polynomial time. \square

6 Independent Set on INTERVAL-ke

In this section, we show that **INDEPENDENT SET** parameterized by the size of the interval completion admits a polynomial compression into the **WEIGHTED INDEPENDENT SET** problem. We state **WEIGHTED INDEPENDENT SET** as a decision problem, whose input contains a graph G with a weight function $w : V(G) \rightarrow \mathbb{Z}^+$ and a nonnegative integer W , and the task is to decide whether G has an independent set S with $w(S) \geq W$.

More formally, we show the following theorem.

Theorem 9 **INDEPENDENT SET** on $G \in \text{INTERVAL-ke}$ admits a compression of size $\mathcal{O}(k^{56})$ into **WEIGHTED INDEPENDENT SET**.



Fig. 1 Structure of a maximum independent set in G

Since WEIGHTED INDEPENDENT SET is in NP and, consecutively, has a polynomial reduction to INDEPENDENT SET that is NP-complete [31], the theorem immediately gives the following result.

Corollary 4 INDEPENDENT SET on $G \in \text{INTERVAL-ke}$ admits a polynomial kernel when parameterized by k .

The proof of Theorem 9 is the most technical part of our paper. Before giving the formal proof, we explain technical issues and briefly sketch the compression algorithm's main ideas. We start with an approach that does not work and then explain how to fix it.

Let G be a graph and let $A \subseteq \binom{V(G)}{2} \setminus E(G)$ be a set of pairs of nonadjacent vertices such that the graph G' obtained from G by adding the edges from A becomes interval. Denote by X the set of end-vertices of the edges of A in G' .

Consider an interval model of G' . For each vertex $v \in V(G')$, let ℓ_v and r_v be, respectively, the left and right endpoint of the interval representing v . For each $v \in V(G')$, denote by G_v^ℓ and G_v^r the subgraphs of G' induced by the sets of vertices $U_v^\ell = \{u \in V(G') \mid r_u < \ell_v\}$ and $U_v^r = \{u \in V(G') \mid r_v < \ell_u\}$ respectively, and for every two distinct $u, v \in V(G')$, let G_{uv} be the subgraph induced by $U_{uv} = \{w \in V(G') \mid r_u < \ell_w \leq r_w < \ell_v\}$ (see Figure 1). For a graph H , denote by $I(H)$ a maximum independent set of H . Suppose that I is a maximum independent set of G and let $I \cap X = \{x_1, \dots, x_s\}$ with $r_{v_{i-1}} < \ell_{v_i}$ for $i \in 2, \dots, s$. Then it is possible to prove that

$$I' = I(G_{x_1}^\ell) \cup \left(\bigcup_{i=2}^s I(G_{x_{i-1}x_i}) \right) \cup I(G_{x_s}^r)$$

is a maximum independent set of G .

This allows us to create the following compression of the initial problem to an instance of WEIGHTED INDEPENDENT SET. Let \mathcal{F} be the set of all induced subgraphs G_v^ℓ , G_v^r and G_{uv} for all $u, v \in X$. Consider the graph \mathcal{G} with the set of vertices $X \cup \mathcal{F}$ with the following adjacencies: for distinct $u, v \in V(\mathcal{G})$, u and v are adjacent if and only if one of the following holds:

- $u, v \in X$ and $xy \in E(G)$,
- $u \in X, v \in \mathcal{F}$ and u is adjacent to a vertex of v in G ,
- $u, v \in \mathcal{F}$ and the subgraphs u and v have either common or adjacent vertices in G .

We define the weight $w(v)$ for $v \in V(\mathcal{G})$ be one if $v \in X$ and set $w(v) = |I(v)|$ for $v \in \mathcal{F}$. It can be shown that G has an independent set of size at least W if and only if \mathcal{G} has an independent set of weight at least W . Then we can use the technique of Frank and Tardos [30] to compress the weights.

Unfortunately, the above arguments *do not work* for the following reason. We based our construction on the assumption that we know G' and the resulting interval model. But computing an optimal interval completion is an NP-hard task. Of course, it would suffice even if we had a poly(OPT) approximation algorithm for interval completion. That is, an algorithm producing in polynomial time an edge set A of size polynomial in k , and whose addition turns the input graph G into an interval graph. However, the existence of such an approximation is a long-standing open problem. The best-known result is the $\mathcal{O}(\log n)$ approximation algorithm of Rao and Richa [57] for the minimum number of edges of an interval supergraph of an n -vertex graphs. While we were able to implement the above idea and obtain the required compression, the absence of a good approximation makes the proof more complicated.

Given a graph G , we construct a vertex set X and a set of induced subgraphs \mathcal{F} of $G - X$ such that the graph \mathcal{G} defined above have the desired property: G has an independent set of size at least W if and only if \mathcal{G} has an independent set of weight at least W . We start the construction of X using the algorithm of Natanzon, Shamir, and Sharan [52] to approximate $\text{fill-in}(G) \leq \text{int-comp}(G)$. Initially, we set X be the set of vertices in the pairs of nonadjacent vertices returned by the algorithm. Then we apply a series of reduction rules that either solve the problem or enhance X by adding vertices or delete vertices of the graph. The reduction rules use the forbidden induced subgraph characterization of interval graphs given by Lekkerkerker and Boland [47]. This way, we construct X of size $\mathcal{O}(k^3)$. Then we construct \mathcal{F} of size $\mathcal{O}(k^{14})$ and define \mathcal{G} . Here again, we use the technique of Frank and Tardos [30] to compress the weights.

The remaining part of the section contains the proof of Theorem 9. In Sect. 6.1, we introduce additional notions and state some auxiliary results. Then, in Sect. 6.2, we give the compression itself.

6.1 Technical Lemmata

An interval graph has been defined as an intersection graph of a family of intervals of the real line. For our compression algorithm, we need the characterization of interval graphs in terms of forbidden induced subgraphs.

Three pairwise nonadjacent vertices of a graph form an *asteroidal triple* (AT) if there is a path between every two of them that avoids the closed neighborhood of the third. For an asteroidal triple T of a graph G , a T -AT-witness is an inclusion minimal induced subgraph F of G such that T is an asteroidal triple of F . The vertices of T are called *terminals* of F . Clearly, F is induced by the vertices of induced paths between every two vertices of T that avoid the closed neighborhood of the third. Therefore, the existence of an asteroidal triple T can be checked in polynomial time, and then the construction of a T -AT-witness can be done in

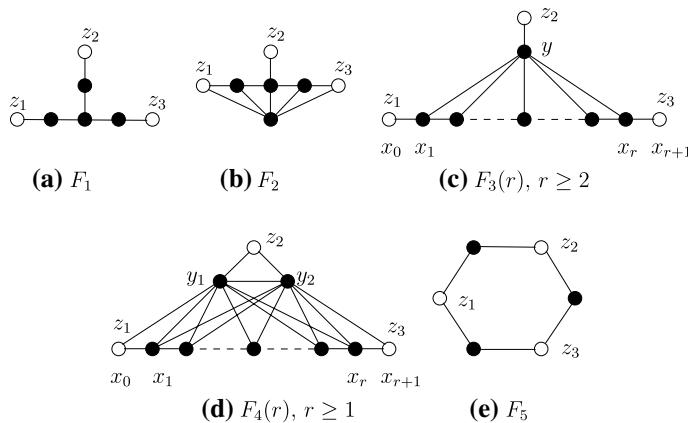


Fig. 2 Minimal asteroidal witnesses; asteroidal triples (terminals) are shown by white bullets

polynomial time using the self-reducibility technique. An *asteroidal witness* is an inclusion minimal induced subgraph that contains an asteroidal triple, and we call the vertices of an asteroidal triple *terminals* of the witness (note that the choice of terminals is not unique).

A graph G is *AT-free* if G has no asteroidal triple. We use the following classical result of Lekkerkerker and Boland [47].

Proposition 9 [47] *A graph G is an interval graph if and only if G is chordal and AT-free.*

The main result of Lekkerkerker and Boland [47] is the characterization of interval graphs by forbidden induced subgraphs. We use this characterization in the following form tailored for our purposes.

Lemma 6 [47] *If a chordal graph G contains an asteroidal triple T , then it contains a minimal asteroidal witness F isomorphic to one of the graphs $F_1, F_2, F_3(r)$ for $r \geq 2$ or $F_4(r)$ for $r \geq 1$ that are shown in Figure 2 (a)–(d) and for every nonterminal vertex v of F , it holds that $v \in V(G) \setminus T$. Moreover, such a witness F can be found in polynomial time.*

The lemma's last statement means that the vertices of an asteroidal triple of G may be only terminal vertices of a minimal witness.

We say that vertex set $X \subseteq V(G)$ is a *chordal-complementing* set for G if there is $A \subseteq \binom{X}{2} \setminus E(G[X])$ such that $G + A$ is chordal. If $G + A$ is chordal, then the set of end-vertices of the edges of A is chordal-complementing. Observe also that every superset of a chordal-complementing set is chordal-complementing, and for each $U \subseteq V(G)$, $X \setminus U$ is a chordal-complementing set of $G - U$.

We say that a triple of vertices T of G is an X -touching AT if T is an asteroidal triple of $G - E(G[X])$ that has a T -AT-witness F such that either $|V(F) \cap X| \leq 1$ or $V(F) \cap X \subseteq T$. We say that F is *associated* with T .

Our compression algorithm uses the properties of chordal-complementing sets and X -touching ATs with associated witnesses given in the following two lemmas.

Lemma 7 *Let $X \subseteq V(G)$ be a chordal-complementing set for a graph G and let T be an X -touching AT triple of G . Then G has an X -touching AT T' with an associated witness F' isomorphic to one of the graphs $F_1, F_2, F_3(r)$ for $r \geq 2$, $F_4(r)$ for $r \geq 1$ or F_5 that are shown in Figure 2 (a)–(e). Moreover, T' and an associated witness F' can be constructed in polynomial time.*

Proof Suppose that $T = \{u_1, u_2, u_3\}$ is an X -touching AT. We find an associated witness F . Note that this can be done in polynomial time using the self-reducibility technique.

Assume that F is not a chordal graph. Then it contains an induced cycle C with at least four vertices. Notice that since X is a chordal-complementing set, C has at least two nonadjacent vertices from T . We consider four cases depending on the length of C .

Let C be of length 4. By symmetry, we assume without loss of generality that $u_1, u_2 \in V(C)$. Then $C = u_1v_1u_2v_2u_1$ for some $v_1, v_2 \in V(F) \setminus T$. Recall that F is induced by the vertices of induced paths between every two vertices of T that avoid the closed neighborhood of the third. The vertices v_1 and v_2 cannot belong to any induced (u_1, u_3) or (u_2, u_3) -path that avoids $N_F[u_2]$ and $N_F[u_1]$. Hence, v_1 and v_2 are vertices of an induced (u_1, u_2) -path that avoids $N_F[u_3]$. Clearly, $u_1v_2u_2$ is an induced (u_1, u_2) -path. We obtain that $F - v_2$ is T -AT-witness but this contradicts the minimality of F . We conclude that C has length at least 5.

Suppose that C has length 5. Again, we can assume without loss of generality that $u_1, u_2 \in V(C)$. Then $C = u_1v_1u_2v_2v_3u_1$ for some $v_1, v_2, v_3 \in V(F) \setminus T$. Then for every $A \subseteq \binom{X}{2} \setminus E(G[X])$, $G'' = G + A$ contains a cycle of length at least four: if $u_1u_2 \in A$, then $u_1u_2v_2v_3v_4v_1$ is such a cycle and if $u_1u_2 \notin A$, then C is a cycle of G'' . This contradicts the condition that X is a chordal-complementing set. Hence, C has length at least 6.

Assume that C has length 6. Suppose that $|V(C) \cap T| = 2$. Then we can assume that $u_1, u_2 \in V(C)$ and either $C = u_1v_1u_2v_2v_3v_4u_1$ or $C = u_1v_1v_2u_2v_3v_4u_1$ for some $v_1, v_2, v_3, v_4 \in V(F) \setminus T$. In both cases, we obtain a contradiction with the condition that X is a chordal-complementing set in the same way as in the previous case, because for every $A \subseteq \binom{X}{2} \setminus E(G[X])$, $G'' = G + A$ contains a cycle of length at least four. Therefore, $T \subseteq V(C)$ and $C = u_1v_1u_2v_3u_3v_3u_1$ for some $v_1, v_2, v_3 \in V(F) \setminus T$. We obtain that $C = F = F_5$ is a T -AT-witness as required by the lemma.

Finally, let C be of length at least 7. Then for two vertices of T , say, u_1 and u_2 , C contains an induced (u_1, u_2) -path P of length at least 3. Then for every

$A \subseteq \binom{X}{2} \setminus E(G[X])$, $G'' = G + A$ contains a cycle of length at least four that contains P as a segment. This contradicts the condition that X is a chordal-complementing set.

Assume now that F is a chordal graph. Then the claim of the lemma is a direct corollary of Lemma 6. \square

Lemma 8 *Let $X \subseteq V(G)$ be a chordal-complementing set for a graph G . Suppose that G has an X -touching AT T with an associated witness F isomorphic to one of the graphs $F_1, F_2, F_3(r)$ for $r \geq 2$, $F_4(r)$ for $r \geq 1$ or F_5 that are shown in Figure 2 (a)–(e). Then for every interval complementation H of G , H has an edge $uv \notin E(G)$ such that*

- (i) $u, v \in V(F)$,
- (ii) either $u \notin X$ or $v \notin X$.

Proof Suppose that T is an X -touching AT with an associated witness isomorphic to one of the graphs $F_1, F_2, F_3(r)$ for $r \geq 2$, $F_4(r)$ for $r \geq 1$ or F_5 . Let H be an interval complementation of G . If $|V(F) \cap X| \leq 1$, then the claim immediately follows from Proposition 9 as H is AT-free. Assume that $|V(F) \cap X| \geq 2$. Then $V(F) \cap X \subseteq T$.

Assume that $|V(F) \cap T| = 2$. If the vertices of $V(F) \cap T$ are nonadjacent in H , then the existence of $uv \notin E(G)$ satisfying (i) and (ii) follows from Proposition 9. Let these vertices be adjacent. Suppose that z_1, z_2 of F (see Figure 2) are in T . Observe that $F + z_1 z_2$ contains an induced cycle of length at least four as it is shown in Figure 3 (a)–(e). Since H is chordal, we obtain that there is $uv \in E(H) \setminus E(G)$ satisfying (i) and (ii). The case $z_2, z_3 \in T$ is symmetric. Assume that $z_1, z_3 \in T$. By symmetry, it is sufficient to consider the cases $F = F_2$, $F = F_3(r)$ and $F = F_4(r)$. Again, we observe that $F + z_1 z_3$ contains an induced cycle of length at least four as it is shown in Figure 3 (f)–(h) and the claim follows.

Let $|V(F) \cap T| = 3$, that is $T = \{z_1, z_2, z_3\}$. If the vertices of $V(F) \cap T$ are pairwise nonadjacent in H , then the existence of $uv \in E(H) \setminus E(G)$ satisfying (i) and (ii) follows from Proposition 9. If $H[T]$ contains an edge, then we apply the same arguments as above for the cases $F = F_2$, $F = F_3(r)$ and $F = F_4(r)$ and obtain that $F + E(H[T])$ contains an induced cycle of length at least four. This implies that there is $uv \in E(H) \setminus E(G)$ satisfying (i) and (ii). Let $F = F_5$. If $|E(H[T])| = 1$ or $|E(H[T])| = 2$, we again have that $F + E(H[T])$ contains an induced cycle of length at least four (see Figure 3 (e) and (i)) and the claim follows. Let $|E(H[T])| = 3$. Then $F + \{z_1 z_2, z_2 z_3, z_1 z_3\}$ coincides with $F_4(2)$. Since H has no induced subgraph isomorphic to $F_4(2)$, we have that there is $uv \in E(H) \setminus E(G)$ satisfying (i) and (ii). \square

In our compression algorithm, we have to compute a maximum independent set for chordal graphs. It was already observed by Gavril [32] in 1972 that this in polynomial (linear) time on chordal graphs.

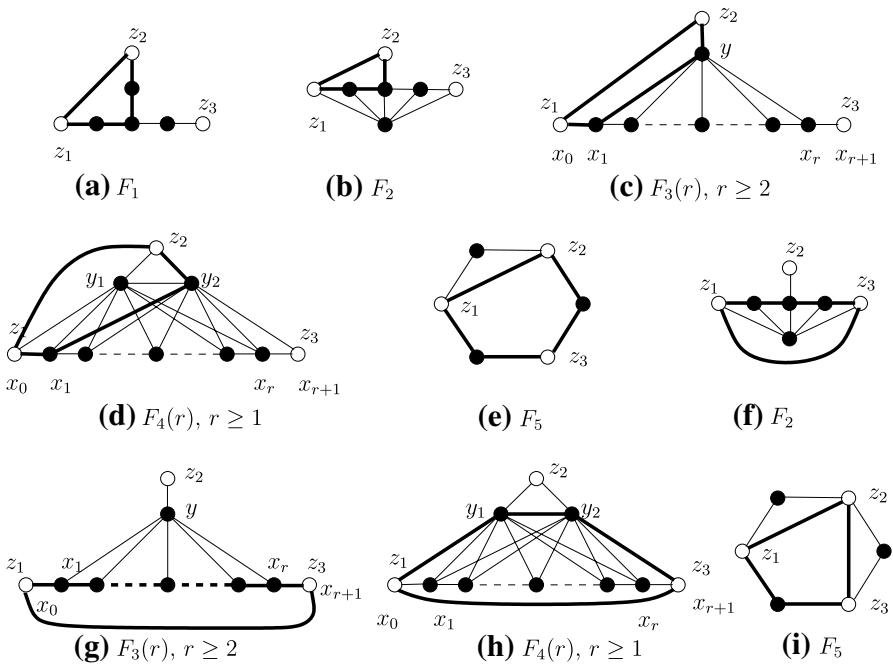


Fig. 3 Induced cycles in F shown by thick lines

Proposition 10 [32, 59] INDEPENDENT SET can be solved in time $\mathcal{O}(n+m)$ on chordal graphs.

6.2 Compression

In this section, we give a compression of INDEPENDENT SET on $G \in \text{INTERVAL-}ke$ parameterized by k . Let (G, ℓ) be an instance of INDEPENDENT SET and let a nonnegative integer k be the parameter.

First, we apply the algorithm of Natanzon, Shamir, and Sharan [52] (see Proposition 1) to approximate the fill-in of G . If the algorithm reports that $\text{fill-in}(G) > k$, we immediately stop as, clearly, $G \notin \text{INTERVAL-}ke$. Assume that this is not the case. Then the algorithm returns a set $A \subseteq \binom{V(G)}{2}$ of size at most $8k^2$ such that $G + A$ is a chordal graphs. We define X to be the set of vertices that are the end-vertices of the edges of A . Note that X is a chordal-complementing set. We apply a series of reduction rules for the instance of INDEPENDENT SET considered together with X , that is, for the triple (G, ℓ, X) .

We apply the following reduction rule to enhance X .

Reduction Rule 1 If G has an X -touching AT T with an associated witness $F = F_1$, then set $X := X \cup V(F)$.

We apply the rule exhaustively but at most $k + 1$ times. Lemma 8 guarantees that if we find an X -touching AT T with an associated witness $F = F_1$, then every interval complementation of G contains an edge uv such that u and v are nonadjacent vertices of F and at most one of them is in X . Hence the following rule is safe.

Reduction Rule 2 *If Reduction Rule 1 has been applied $k + 1$ times, then report that $G \notin \text{INTERVAL-ke}$ and stop.*

Assume that the algorithm did not stop. Since $|V(F_1)| = 7$, we obtain that $|X| \leq 8k^2 + 7k$ after this step.

In the next step, we find and delete some irrelevant vertices of G . For this, set $p = 8k^2 + 7k + 2$.

Reduction Rule 3 *If for some vertex $x \in V(G)$, the subgraph $G[N_G(x) \setminus X]$ has an independent set of size at least $p + 1$, then set $G := G - x$ and $X := X \setminus \{x\}$.*

Lemma 9 *Reduction Rule 3 is safe.*

Proof Denote by G' and X' the graph and the vertex set obtained from G and X respectively by applying the rule for some $x \in V(G)$. Note that $X' = X$ if $x \notin X$. We show that G is an independent set of size ℓ if and only G' has an independent set of the same size. It is trivial that if G' has an independent set of size ℓ , then the same set is an independent set of G . Assume that G has an independent set I of size ℓ . We prove that G' has an independent set of size at least ℓ . The claim is straightforward if $x \notin I$. Suppose that $x \in I$. Let S be an independent set of size at least $p + 1$ in $G[N_G(x) \setminus X]$. Clearly, $S \cap I = \emptyset$.

If there is $S' \subset S$ of size $p - 1$ that has no vertex adjacent to a vertex of $I \setminus (X \cup \{x\})$, then consider $I' = (I \setminus (X \cup \{x\})) \cup S'$. Observe that I' is an independent set of G' . Since $|X| \leq p - 2$, $|I'| \geq |I| \geq \ell$ and we have that I' is a required independent set of size at least ℓ in G' .

Assume from now that every $S' \subset S$ of size $p - 1$ has a vertex with a neighbor in $I \setminus (X \cup \{x\})$. Because $|S| \geq p + 1$, there are three distinct vertices $u_1, u_2, u_3 \in S$ that have neighbors in $I \setminus (X \cup \{x\})$. Denote these neighbors of u_1, u_2 and u_3 by v_1, v_2 and v_3 respectively. Note that u_1, u_2, u_3 are pairwise nonadjacent and they are adjacent to x . Notice also that $u_1, u_2, u_3, v_1, v_2, v_3 \notin X$. If $v_i = v_j$ for some distinct $i, j \in \{1, 2, 3\}$, we have that $xu_iv_iu_jx$ is an induced cycle but this contradicts the property that X is a chordal-complementing set. Hence, v_1, v_2, v_3 are pairwise distinct and v_i is not adjacent to u_j for distinct $i, j \in \{1, 2, 3\}$. Because $v_1, v_2, v_3 \in I$, these vertices are pairwise nonadjacent. But then $G[\{x, u_1, u_2, u_3, v_1, v_2, v_3\}]$ is isomorphic to F_1 (see Figure 2 (a)). This contradict the assumption that Reduction Rule 1 was applied exhaustively and the algorithm was not stopped by Reduction Rule 2. This competes the safeness proof. \square

We apply Reduction Rule 3 exhaustively. The crucial property we achieve by this rule is the following.

Observation 3 If G has an X -touching AT T with an associated witness $F = F_3(r)$ or $F = F_4(r)$ (see Figure 2 (c) and (d)), then $r \leq 2p - 1 = 16k^2 + 14k + 1$.

Proof Assume that G has an X -touching AT T with an associated witness $F = F_3(r)$. Note that only the terminals of F could be in X . Then $\{x_1, x_3, \dots, x_{2\lceil r/2 \rceil - 1}\}$ is an independent set in $G[N_G(y) \setminus X]$ (see Figure 2 (c)). Since Reduction Rule 3 cannot be applied, $r \leq 2p - 1$. If $F = F_4(r)$, the arguments are the same and the only difference is that we consider y_1 instead of y (see Figure 2 (d)). \square

Now we proceed with enhancing X .

Reduction Rule 4 If G has an X -touching AT T with an associated witness F isomorphic to one of the graphs $F_1, F_2, F_3(r)$ for $r \geq 2, F_4(r)$ for $r \geq 1$ or F_5 that are shown in Figure 2 (a)–(e), then set $X := X \cup V(F)$.

We apply the rule exhaustively but at most $k + 1$ times, because in the same way as for Reduction Rule 2, we can apply Lemma 8. Hence, the next rule is safe.

Reduction Rule 5 If Reduction Rule 4 has been applied $k + 1$ times, then report that $G \notin \text{INTERVAL-ke}$ and stop.

Assume that the algorithm did not stop. By Observation 3, each F , whose vertices have been added to X by Reduction Rule 4, has at most $2p + 4 = 32k^2 + 2k + 4$ vertices. Then

$$|X| \leq 8k^2 + 7k + k(32k^2 + 2k + 4) = 32k^3 + 10k^2 + 11k = \mathcal{O}(k^3). \quad (5)$$

By applying Reduction Rule 4, we achieve an essential property of G and X . When the rule cannot be applied anymore, by Lemma 7, the following holds.

Observation 4 The graph $G' = G - E(G[X])$ has no asteroidal triple $T \subseteq X$ with a T -AT-witness F such that $V(F) \cap X = T$.

Next, we simplify the instance (G, ℓ) of INDEPENDENT SET by the removal chordal components. It is straightforward to see that the following rule is safe.

Reduction Rule 6 If G has a connected component H that is a chordal graph, then compute the size $\alpha(H)$ of a maximum independent set of H and set $G := G - V(H)$, $X := X \setminus V(H)$ and $\ell = \ell - \alpha(H)$. If $\ell \leq 0$, then return a trivial yes-instance of WEIGHTED INDEPENDENT SET and stop.

The rule is applied exhaustively. We assume that the algorithm did not stop.

For a set $Y \subseteq X$ of size at most two, let \mathcal{C}_Y be the set of connected components of the graph $G - (X \cup N_G(Y))$ and define

$$\mathcal{C} = \bigcup_{Y \subseteq X, |Y| \leq 2} \mathcal{C}_Y.$$

Notice that \mathcal{C} is a set of connected induced subgraphs of $G - X$ and distinct subgraphs in the set can have common or adjacent vertices. For each $C \in \mathcal{C}$, let $I(C)$ be a maximum independent set of C . We use the following crucial property of \mathcal{C} .

Lemma 10 *There is a set $S \subseteq X$ and a family $\mathcal{C}^* \subseteq \mathcal{C}$ of pairwise disjoint graphs without adjacent (in G) vertices in distinct subgraphs such that*

$$I^* = S \cup \bigcup_{C \in \mathcal{C}^*} I(C)$$

is a maximum independent set of G .

Proof Let I be a maximum independent set of G . Let $S = I \cap X$ and denote by C_1, \dots, C_s the connected components of $G - (X \cup N_G(S))$. Clearly, $I \setminus S \subseteq V(C_1) \cup \dots \cup V(C_s)$. Moreover, if I_i is an arbitrary maximum independent set of C_i for $i \in \{1, \dots, s\}$, then $I' = S \cup I_1 \cup \dots \cup I_s$ is a maximum independent set of G . We claim that for each $i \in \{1, \dots, s\}$, there is a set $Y \subseteq S$ of size at most two such that $C_i \in \mathcal{C}_Y$.

To obtain a contradiction, assume that for some $i \in \{1, \dots, s\}$, $C_i \notin \mathcal{C}_Y$ for every $Y \subseteq S$ of size at most two. Then $|S| \geq 3$ and there are distinct vertices $u_1, u_2, u_3 \in S$ such that for every $j \in \{1, 2, 3\}$, there is $v_j \in N_G(u_j) \setminus X$ such that

- (i) $v_j \notin N_G(u_h)$ for $h \in \{1, 2, 3\} \setminus \{j\}$, and
- (ii) v_j is adjacent to some vertex w_j of C_i .

Consider any two distinct vertices u_j and u_h for $j, h \in \{1, 2, 3\}$. Let t be the unique element of $\{1, 2, 3\} \setminus \{j, h\}$. Since C_i is connected, there is a (w_j, w_h) -path P in C_i . Notice that P avoids $N_G[u_t]$. Let $P' = u_j v_j P v_h u_h$. We have that P' is an (u_j, u_h) -path in $G' = G - E(G[X])$ avoiding the neighborhood of u_t . Since this holds for any choice of j and h , we obtain that $T = \{u_1, u_2, u_3\} \subseteq X$ is an asteroidal triple T in G' with a T -AT-witness F such that $V(F) \cap X = T$ but this contradicts Observation 4. This proves that for each $i \in \{1, \dots, s\}$, there is a set $Y \subseteq S$ of size at most two such that $C_i \in \mathcal{C}_Y$.

We obtain that $C_1, \dots, C_s \in \mathcal{C}$ are pairwise disjoint graphs without adjacent vertices in distinct subgraphs and

$$I^* = S \cup I(C_1) \cup \dots \cup I(C_s)$$

is a maximum independent set of G . □

We show that the size of \mathcal{C} is bounded by a polynomial of the parameter. First, we prove an auxiliary claim.

Lemma 11 *Let $x \in Y \subseteq X$, where $|Y| \leq 2$, and let $\mathcal{C}' \subseteq \mathcal{C}_Y$ be the set of graphs in \mathcal{C}_Y that have at least one neighbor in $N_G(x) \setminus X$. Then $|\mathcal{C}'| \leq 3p$ for $p = 8k^2 + 7k + 2$.*

Proof To obtain a contradiction, assume that $|\mathcal{C}'| > 3p$, that is, there are at least $3p + 1$ connected components of $G' = G - (N_G(Y) \cup X)$ that have neighbors in $N_G(x) \setminus X$. Let $y \in N_G(x) \setminus X$. If y is adjacent to at least $p + 1$ connected components of G' , then $N_G(y) \setminus X$ contains an independent set of size at least $p + 1$. In this case we would be able to apply Reduction Rule 3; which is a contradiction. Therefore, each $y \in N_G(x) \setminus X$ has neighbors in at most p connected components of G' . Since $\mathcal{C}' \geq 3p + 1$, we conclude that there are three distinct vertices $y_1, y_2, y_3 \in N_G(x) \setminus X$ such that there are three distinct connected components $C_1, C_2, C_3 \in \mathcal{C}'$ with the property that for every $i \in \{1, 2, 3\}$, y_i has a neighbor $z_i \in V(C_i)$ and y_i has no neighbor in C_j for $j \in \{1, 2, 3\} \setminus \{i\}$. Consider $H = G[\{x, y_1, y_2, y_3, z_1, z_2, z_3\}]$. It is easy to see that z_1, z_2, z_3 is an asteroidal triple in this graph and H contains an induced subgraph isomorphic to F_1 , $F_3(2)$ or $F_3(3)$ (see Figure 2 (a) and (c)) depending on the adjacencies between y_1 , y_2 and y_3 . This means that we would be able to apply Reduction Rule 4 contradicting the assumption that the rule was applied exhaustively. This proves that $|\mathcal{C}'| \leq 3p$. \square

$$|\mathcal{C}| = \mathcal{O}(k^{14}).$$

Lemma 12

Proof By the definition, $\mathcal{C} = \bigcup_{Y \subseteq X, |Y| \leq 2} \mathcal{C}_Y$. We upper bound $|\mathcal{C}_Y|$ for $Y \subseteq X$ of size at most two.

Let $Y = \emptyset$. Observe that each connected component of G contains a vertex of X , because G has no connected component that is a chordal graph as Reduction Rule 6 cannot be applied. Note that if a vertex x has neighbors in at least $p + 1$ connected components of $G - X$ for $p = 8k^2 + 7k + 2$, then $N_G(x) \setminus X$ contains an independent set of size at least $p + 1$ and we would be able to apply Reduction Rule 3. Therefore, each vertex of X has neighbors in at most p components of $G - X$. Hence, $|\mathcal{C}_\emptyset| \leq p|X| = \mathcal{O}(k^5)$ by the definition of p and (5).

Suppose that $Y = \{x\}$ for $x \in X$. Denote by $\mathcal{C}^{(1)}$ the set of graphs in \mathcal{C}_Y that have vertices adjacent to $N_G(x) \setminus X$ and let $\mathcal{C}^{(2)}$ be the set of graphs in \mathcal{C}_Y that have no vertex adjacent to $N_G(x) \setminus X$. We have that $\mathcal{C}_Y = \mathcal{C}^{(1)} \cup \mathcal{C}^{(2)}$. By exactly the same arguments as for $Y = \emptyset$, we obtain that $|\mathcal{C}^{(2)}| \leq p(|X| - 1)$. By Lemma 11, $|\mathcal{C}^{(1)}| \leq 3p$. We obtain that

$$|\mathcal{C}_Y| = |\mathcal{C}^{(1)} \cup \mathcal{C}^{(2)}| \leq |\mathcal{C}^{(1)}| + |\mathcal{C}^{(2)}| \leq 3p + p(|X| - 1)$$

and, therefore, $|\mathcal{C}_Y| = \mathcal{O}(k^5)$ by the definition of p and (5).

Suppose now that $Y = \{x_1, x_2\}$ for distinct $x_1, x_2 \in X$. Denote by $\mathcal{C}^{(i)}$ the set of graphs in \mathcal{C}_Y that have vertices adjacent to some vertices of $N_G(x_i) \setminus X$ for $i \in \{1, 2\}$ and let $\mathcal{C}^{(3)}$ be the set of graphs in \mathcal{C}_Y that have no vertex adjacent to $N_G(Y) \setminus X$. We have that $\mathcal{C}_Y = \mathcal{C}^{(1)} \cup \mathcal{C}^{(2)} \cup \mathcal{C}^{(3)}$. By exactly the same arguments as for $Y = \emptyset$, we obtain that $|\mathcal{C}^{(3)}| \leq p(|X| - 2)$. By Lemma 11, $|\mathcal{C}^{(1)}| \leq 3p$ for $i \in \{1, 2\}$. Then

$$|\mathcal{C}_Y| = |\mathcal{C}^{(1)} \cup \mathcal{C}^{(2)} \cup \mathcal{C}^{(3)}| \leq |\mathcal{C}^{(1)}| + |\mathcal{C}^{(2)}| + |\mathcal{C}^{(3)}| \leq 6p + p(|X - 2|)$$

and, therefore, $|\mathcal{C}_Y| = \mathcal{O}(k^5)$ by the definition of p and (5).

Since there are $|X|$ single-element subsets $Y \subseteq X$ and $\binom{|X|}{2}$ two-element subsets $Y \subseteq X$, we have that $|\mathcal{C}| = \mathcal{O}(k^{14})$ by (5). \square

Construction of the instance of WEIGHTED INDEPENDENT SET. At the next step of our compression algorithm we construct the instance of WEIGHTED INDEPENDENT SET as follows.

- Construct the graph G^* with the vertex set $X \cup \mathcal{C}$ by making every two distinct vertices u and v either adjacent or nonadjacent by the following rule:
 - if $u, v \in X$, then u and v are adjacent in G^* if and only if they are adjacent in G ,
 - if $u \in X$ and $v \in \mathcal{C}$, then u and v are adjacent if and only if u is adjacent to a vertex of the subgraph v in G ,
 - if $u, v \in \mathcal{C}$, then u and v are adjacent if and only if the subgraph u and v of G have either common vertices or adjacent vertices in G .
- For $v \in V(G^*)$, set the weight $w(v) = 1$ if $v \in X$ and set $w(v)$ be the size of a maximum independent set of the subgraph v of G .

Lemma 13 *The instance (G, ℓ) is a yes-instance of INDEPENDENT SET if and only if (G^*, w, ℓ) is a yes-instance of WEIGHTED INDEPENDENT SET.*

Proof Let (G, ℓ) be a yes-instance of INDEPENDENT SET. By Lemma 10, there is a set $S \subseteq X$ and a family $\mathcal{C}^* \subseteq \mathcal{C}$ of pairwise disjoint graphs without adjacent vertices in distinct subgraphs such that

$$I = S \cup \bigcup_{C \in \mathcal{C}^*} I(C)$$

is a maximum independent set of G , where $I(C)$ is a maximum independent set of C . By the definition of G^* , we have that $S \cup \mathcal{C}^*$ is an independent set of G^* of weight $|S| + \sum_{C \in \mathcal{C}^*} |I(C)| = |I|$. Hence, (G^*, w, ℓ) is a yes-instance of WEIGHTED INDEPENDENT SET.

Assume now that (G^*, w, ℓ) is a yes-instance of WEIGHTED INDEPENDENT SET. Consider an independent set I^* of weight at least ℓ in G^* . Let $S = I^* \cap X$ and $\mathcal{C}^* = I^* \cap \mathcal{C}$. By the definition of G^* , we have that $S \subseteq X$ is an independent set of G , and the graphs of \mathcal{C}^* are disjoint induced subgraphs of G that have no vertices adjacent to S and there are no two adjacent vertices that are in distinct graphs of \mathcal{C}^* . Every graph $C \in \mathcal{C}^*$ has an independent set $I(C)$ of size $w(C)$ by the definition of the weights. This means that $I = S \cup \bigcup_{C \in \mathcal{C}^*} I(C)$ is an independent set in G of size at least ℓ . Therefore, (G, ℓ) is a yes-instance of INDEPENDENT SET. \square

By Lemma 12, G^* has $\mathcal{O}(k^{14})$ vertices, that is, the size of G^* is bounded by a polynomial of the parameter. To complete the construction of the compressed instance, it remains to reduce the weights of vertices. We do it by making use of Proposition 3. Let v_1, \dots, v_s be the vertices of G^* . Following the notation of Proposition 3, let $h = s + 1$ and $N = s + 2$. Consider the vector $w = (w(v_1), \dots, w(v_s), \ell)^T \in \mathbb{Z}^h$. The algorithm of Frank and Tardos finds a vector $\bar{w} = (w_1, \dots, w_s, W)$ with $\|\bar{w}\|_\infty \leq 2^{4h^3} N^{h(h+2)}$ such that $\text{sign}(w \cdot b) = \text{sign}(\bar{w} \cdot b)$ for all vectors $b \in \mathbb{Z}^h$ with $\|b\|_1 \leq N - 1$. We define $w^*(v_i) = w_i$ for $i \in \{1, \dots, s\}$ and consider the instance (G^*, w^*, W) of WEIGHTED INDEPENDENT SET. This completes the construction of the compression. The properties of the obtained instance of WEIGHTED INDEPENDENT SET are summarized in the following lemma.

Lemma 14 *The instance (G^*, w, ℓ) is a yes-instance of WEIGHTED INDEPENDENT SET if and only if (G^*, w^*, W) is a yes-instance. The size of (G^*, w^*, W) is $\mathcal{O}(k^{56})$.*

Proof Notice that, in particular, the equality $\text{sign}(w \cdot b) = \text{sign}(\bar{w} \cdot b)$ holds for all vectors $b \in \mathbb{Z}^h$ such that each element of b is $-1, 0$ or 1 . This implies that the elements of \bar{w} are positive and for every $J \subseteq \{1, \dots, s\}$, $\sum_{i \in J} w(v_i) \geq \ell$ if and only if $\sum_{i \in J} w_i \geq W$. Clearly, for every set of vertices $S \subseteq V(G^*)$, $\sum_{v \in S} w(v) \geq \ell$ if and only if $\sum_{v \in S} w^*(v) \geq W$. This means that (G^*, w, ℓ) is a yes-instance of WEIGHTED INDEPENDENT SET if and only if (G^*, w^*, W) is a yes-instance.

Since $\|\bar{w}\|_\infty \leq 2^{4h^3} N^{h(h+2)}$, we have that $w(v) \leq 2^{4(s+1)^3} (s+2)^{(s+1)(s+3)}$ for every $v \in V(G^*)$ and the same upper bound holds for W . This implies that the weights of the vertices and W can be encoded by a string of length $\mathcal{O}(k^{42})$, because by Lemma 12, $|V(G^*)| = \mathcal{O}(k^{14})$. Because G^* has $\mathcal{O}(k^{14})$ vertices and $\mathcal{O}(k^{28})$ edges, the size of (G^*, w^*, W) is $\mathcal{O}(k^{56})$. \square

Running time. Finally, we have to show that the compression algorithm is polynomial. The construction of the initial set X can be done in polynomial time by Proposition 1. Then we apply Reduction Rules 1–6. It is straightforward to see that Reduction Rule 1 can be applied in polynomial time as we are looking in it for an induced subgraph of bounded size. Reduction Rules 2 and 5 are trivial. Reduction Rules 3 and 6 are polynomial by Proposition 10. Reduction Rule 4 is polynomial by Lemma 7. Since Reduction Rules 1 and 4 are applied at most $k + 1$ times and Reduction Rules 3 and 6 are applied at most $|V(G)|$ times, we conclude all the rules can be applied in polynomial time. In the next step, we construct the instance (G^*, w, ℓ) and the step is polynomial due to Proposition 10. Finally, we reduce the weight, and this can be done in polynomial time by Proposition 3.

The proof of Theorem 9 is complete.

7 Independent Set on SPLIT-ke

In this section, we show that INDEPENDENT SET admits a polynomial kernel when parameterized by the split completion size.

Computing $\text{split-comp}(G)$ is NP-hard [53]. Interestingly, if we allow not only edge additions but also edge deletions, then the problem becomes polynomial-time solvable [37].

Proposition 11 [37] *There is an algorithm that, given a graph G , in polynomial time finds a set $A \subseteq \binom{V(G)}{2}$ of minimum size such that $G \triangle A$ is a split graph.*

Theorem 10 INDEPENDENT SET on SPLIT–ke admits a polynomial kernel with at most $2k^2(k+2)$ vertices when parameterized by k .

Proof Let (G, ℓ) be an instance of INDEPENDENT SET and let a nonnegative integer k be the parameter.

We use Proposition 11 and find a set $A \subseteq \binom{V(G)}{2}$ of minimum size such that $G' = G \triangle A$ is a split graph. If $|A| > k$, we conclude that $G \notin \text{SPLIT–ke}$ and stop. Assume that is not the case. Let $D = A \cap E(G)$ and $C = A \setminus E(G)$, that is, D is the set of deleted edges and C is the set of added edges. We find a partition of $V(G')$ into a clique K and an independent set I . By the minimality of A , the edges of D have their end-vertices in I , and the edges of C have their end-vertices in K . Let X be the set of end-vertices of C and set $Y = K \setminus X$. Note that $|X| \leq 2k$ and every vertex of X is adjacent to each vertex of Y .

We apply a series of reduction rules for (G, ℓ) together with the sets I, X, Y, D .

Reduction Rule 7 *If $D \neq \emptyset$, then for $uv \in D$ do the following.*

- *If each of u and v is not adjacent to at least $k+2$ vertices of Y in G , then report that $G \notin \text{SPLIT–ke}$ and stop.*
- *Otherwise, pick a vertex of $\{u, v\}$ with the minimum number of nonneighbors in Y , say u , and*
 - *set $D := D - uv$,*
 - *set $I := I \setminus \{u\}$,*
 - *set $X := X \cup \{u\} \cup (Y \setminus N_G(u))$,*
 - *set $Y := Y \cap N_G(u)$.*

We apply the rule exhaustively until D becomes empty.

Claim 5 *If the algorithm stops while executing Reduction Rule 7, then $\text{split-comp}(G) > k$.*

Proof of Claim 5 The algorithm stops if there is $uv \in D$ such that $|Y \setminus N_G(u)| \geq k+2$ and $|Y \setminus N_G(v)| \geq k+2$. Let $R \subseteq \binom{V(G)}{2} \setminus E(G)$ be a set of pairs of vertices of minimum size such that $\hat{G} = G + R$ is a split graph. Let (S, Z) be a partition of $V(\hat{G})$ into an independent set S and a clique Z . Notice that either $u \in Z$ or $v \in Z$. By symmetry, assume without loss of generality that $u \in Z$. Observe that Y is a clique of G .

Hence, $|Y \setminus Z| \leq 1$. This implies that R contains at least $k + 1$ edges incident to u in \hat{G} whose other end-vertices are in Y . Therefore, $\text{split-comp}(G) = |R| \geq k + 1$. \square

Claim 5 guarantees that if we stop by Reduction Rule 5, then $G \notin \text{SPLIT-ke}$. Assume that the algorithm did not stop. Then we obtain that the constructed sets X , Y , and I have the properties summarized in the following claim.

Claim 6 *The sets X , Y , I form a partition of the vertices of G such that*

- (i) *I is an independent set in G ,*
- (ii) *Y is a clique in G ,*
- (iii) *for every $v \in X$, $Y \subseteq N_G(v)$,*
- (iv) *$|X| \leq (k + 2)k$,*
- (v) *for every independent set $S \subseteq X$, $|S| \leq 2k$.*

Proof of Claim 6 It is straightforward that (X, Y, I) is a partition of $V(G)$.

To see (i), it is sufficient to observe that only the edges of D had both their end-vertices in I in the initial I , and we exclude at least one end-vertex of every edge of D from I by Reduction Rule 7.

The property (ii) is trivial as Y was a clique before we started to apply Reduction Rule 7 and we only delete vertices from Y by the rule.

For (iii), observe that if v is a vertex of the initial set X , then $Y \subseteq N_G(v)$ by the definition of X and Y . Then, if we add a vertex $u \in I$ to Y by Reduction Rule 7, then we delete the vertices of $Y \setminus N_G(u)$ from Y . Note that these vertices are in X and, since Y is a clique, they are adjacent to all remaining vertices of Y . Hence, $Y \subseteq N_G(v)$ for every $v \in X$.

To show (iv), notice that initially $|X| \leq 2|C|$. Then, whenever we apply Reduction Rule 7, we add to X at most $k + 2$ vertices. The rule is applied at most $|D|$ times. We obtain that $|X| \leq 2|C| + (k + 2)|D| \leq (k + 2)|A| \leq (k + 2)k$.

Finally, to prove (v), observe that initially $|X| \leq 2|C|$ and, therefore, every independent set with its vertices in the initial set X has size at most $2|C|$. By each application of Reduction Rule 7, we put a vertex $u \in I$ in X and add a clique $Y \setminus N_G(u)$. Hence we can increase the maximum size of an independent subset of X by at most two. The rule is applied at most $|D|$ times. We conclude that the maximum size of an independent subset of X is at most $2|C| + 2|D| = 2|A| \leq 2k$. \square

These properties allow us to apply the following two rules.

Reduction Rule 8 Set $Y := Y \setminus N_G(I)$.

Reduction Rule 9 If $|Y| \geq 2$, then delete $|Y| - 1$ arbitrary vertices of Y and set $I := I \cup Y$.

Claim 7 *Reduction Rules 8 and 9 are safe and the set I constructed by Reduction Rule 9 is independent.*

Proof of Claim 7 Let G' be the graph obtained from G by the application of Reduction Rules 8 and 9 and denote by I' the set obtained from I . For the safeness proof, it is sufficient to show that if G has an independent set S with $|S| \geq \ell$, then G' has an independent set of size at least ℓ . If $S \cap Y = \emptyset$, $S \subseteq V(G')$, that is, S is an independent set of G' . Suppose that there is $v \in S \cap Y$. Since Y is a clique, v is the unique vertex of S in Y . Since $Y \subseteq N_G(u)$ for every $u \in X$, $X \cap S = \emptyset$. This means that $S \setminus \{v\} \subseteq I$.

Suppose that there is $u \in I$ such that $v \in N_G(u)$. Consider $S' = (S \setminus \{v\}) \cup \{u\}$. We obtain that S' is an independent set, since $S' \subseteq I$. Clearly, $|S'| = |S| \geq \ell$.

Assume now that $v \in Y \setminus N_G(I)$. Note that the vertices of $Y \setminus N_G(I)$ are true twins in G , that is, for every $x, y \in Y \setminus N_G(I)$, $N_G[x] = N_G[y]$. Hence, we can assume without loss of generality that u was not deleted by Reduction Rule 7 and $S \subseteq V(G')$.

To see that I' constructed by Reduction Rule 9 is independent, it is sufficient to observe that we include in I a unique vertex of Y that is not adjacent to other vertices of I . \square

Observe that after applying Reduction Rules 8 and 9, we have that (X, I) is a partition of $V(G)$, where I is an independent set.

Reduction Rule 10 *If there is a vertex $u \in X$ such that $|N_G(u)| \geq 2k$, then set $X := X \setminus \{u\}$.*

Claim 8 *Reduction Rule 10 is safe.*

Proof of Claim 8 Denote by G' the graph obtained from G by the application of Reduction Rule 10 for a vertex $u \in X$. It is sufficient to show that if G has an independent set S with $|S| \geq \ell$, then G' has an independent set of size at least ℓ . If $u \notin S$, then S is an independent set of G' . Assume that $u \in S$. Consider the set I . Observe that $|S \cap I| \leq |I| - 2k$. By Claim 6 (v), $|S \cap X| \leq 2k$. We obtain that $|I| \geq |S|$. Since I is an independent set in G' , the claim follows. \square

We apply the rule exhaustively. We obtain G with the property that $|N_G(u) \cap I| \leq 2k - 1$ for every $u \in X$.

Finally, we exhaustively apply the following reduction rule.

Reduction Rule 11 *If there is an isolated vertex u , then set $G := G - u$, $X := X \setminus \{u\}$, $I := I \setminus \{u\}$ and $\ell := \ell - 1$.*

It is straightforward to see that the rule is safe.

Now we can show that the obtained graph G has a bounded size.

Claim 9 *The graph G has at most $2k^2(k + 2)$ vertices.*

Proof of Claim 9 After the exhaustive application of Reduction Rule 11, G has no isolated vertices. Recall that by Claim 6 (iv), $|X| \leq (k+2)k$. Since each vertex of X is adjacent to at most $2k-1$ vertices in I , $|I| \leq (2k-1)(k+2)k$. Then $|V(G)| \leq |X| + |I| \leq (k+2)k + (2k-1)(k+2)k = 2k^2(k+2)$. \square

This completes the construction of the kernel.

The initial construction of X , Y , D and I is polynomial by Proposition 11. It is straightforward to see that Reduction Rules 7–11 runs in polynomial time. Hence, our kernelization algorithm is polynomial. \square

8 Conclusion

In this paper, we initiated the study of parameterized subexponential and kernelization algorithms on CHORDAL- ke graphs. The existence of such algorithms makes this graph class a very interesting object for studies. For other structural parameters, like treewidth or vertex cover, we have quite good understanding about the complexity of various optimization problems derived from general meta-theorems like Courcelle’s or Pilipczuk’s theorems [17, 56] and advanced algorithmic techniques [19, 20, 25]. We believe that further exploration of the complexity landscape of fill-in parameterization is an interesting research direction. If an optimization problem is NP-complete on chordal graphs, like DOMINATING SET, then on CHORDAL- ke this problem is in Para – NP. On the other hand, even if a problem is solvable in polynomial time on chordal graphs, in theory, there is nothing preventing it from being Para – NP on CHORDAL- ke . Is there a natural graph problem with this property? For many problems that are solvable in polynomial time on chordal graphs, we also established FPT algorithms on CHORDAL- ke class. This does not exclude a possibility that there are problems that are not FPT parameterized by k but solvable in polynomial time for every fixed k . We do not know any such problem (in other words, the problem in class XP) yet. It will be interesting to see, if there is any natural graph problem of such complexity. In addition, we proved that there are problems that are FPT on CHORDAL- ke when parameterized by k and which cannot be solved in subexponential time unless ETH fails. We believe it would be exciting to obtain a logical characterization of problems that can be solved in subexponential time on CHORDAL- ke when parameterized by k , similar to the classical Courcelle’s theorem [17].

Some concrete open problems. Observe that for our subexponential dynamic programming algorithms, we only need a k -almost chordal tree decomposition of the input graph, that is, a decomposition where each bag can be made a clique by adding at most k edges. (Recall Definition 1.) The maximum of numbers $c\text{-comp}(G[X_t]) \leq k$ can be significantly smaller than the minimum fill-in of a graph. For graphs in CHORDAL- ke , we can find fill-in in a subexponential in k time by the algorithm of Fomin and Villanger [29]. However, we do not know if it is FPT in k to decide, whether a graph admits a k -almost chordal tree decomposition. And if yes, can it be done in subexponential time?

The only reason why many of the algorithms introduced in our work run in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ and not $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ is because we do not know how to find a fill-in of size k in time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$. The best known lower bound given by Cao and Sandeep [16] rules out algorithms of time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ and better algorithms for fill-in would imply better algorithms for various optimization problems on CHORDAL- ke graphs. Moreover, to get rid of the logarithm in the exponent, we do not need an exact algorithm. By the results of Natanzon, Shamir and Sharan [52] (Proposition 1), fill-in(G) can be approximated in polynomial time within a polyopt factor $8 \cdot \text{fill-in}(G)$. Deciding whether $\text{fill-in}(G) \leq k$ can be done in time $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ by the results of Fomin and Villanger [29] (Proposition 4). Is there an FPT constant-factor approximation algorithm with running time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$? The existence of such an algorithm would improve running times of the algorithms for many problems. For example, pipelined with our results, it would imply that WEIGHTED INDEPENDENT SET is solvable in $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time on CHORDAL- ke .

Finally, we proved that INDEPENDENT SET on INTERVAL- ke and SPLIT- ke admit polynomial kernels when parameterized by k . We leave open the question whether or not this problem has a polynomial (Turing) kernel on CHORDAL- ke .

Acknowledgements We thank Torstein Strømme, Daniel Lokshtanov, and Pranabendu Misra for fruitful discussions on the topic of this paper. We also grateful to Saket Saurabh for helpful suggestions that allowed us to improve our results. A preliminary version of this paper appeared in the proceedings of ESA 2020. The research leading to these results has been supported by the Research Council of Norway via the project “MULTIVAL” (grant no. 263317).

Funding Open access funding provided by University of Bergen (incl Haukeland University Hospital).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Agrawal, A., Lokshtanov, D., Misra, P., Saurabh, S., Zehavi, M.: Feedback vertex set inspired kernel for chordal vertex deletion. ACM Trans. Algorithms **15**(1), 11:1–11:28 (2019). <https://doi.org/10.1145/3284356>
2. Agrawal, A., Misra, P., Saurabh, S., Zehavi, M.: Interval vertex deletion admits a polynomial kernel. In: Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1711–1730. SIAM (2019). <https://doi.org/10.1137/1.9781611975482.103>
3. Baste, J., Sau, I., Thilikos, D.M.: A complexity dichotomy for hitting connected minors on bounded treewidth graphs: the chair and the banner draw the boundary. In: Proceedings of the 31th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 951–970. SIAM (2020)
4. Bessy, S., Perez, A.: Polynomial kernels for Proper Interval Completion and related problems. Information and Computation **231**, 89–108 (2013). <https://doi.org/10.1016/j.ic.2013.08.006>

5. Bliznets, I., Fomin, F.V., Pilipczuk, M., Pilipczuk, M.: Subexponential parameterized algorithm for interval completion. ACM Trans. Algorithms **14**(3), 35:1–35:62 (2018). <https://doi.org/10.1145/3186896>
6. Bodlaender, H.L., Cygan, M., Kratsch, S., Nederlof, J.: Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. Inf. Comput. **243**, 86–111 (2015). <https://doi.org/10.1016/j.ic.2014.12.008>
7. Bodlaender, H.L., Heggernes, P., Villanger, Y.: Faster parameterized algorithms for minimum fill-in. Algorithmica **61**(4), 817–838 (2011). <https://doi.org/10.1007/s00453-010-9421-1>
8. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. **28**(1), 277–305 (2014). <https://doi.org/10.1137/120880240>
9. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph classes: a survey. SIAM Monographs on Discrete Mathematics and Applications. SIAM (1999). <https://doi.org/10.1137/1.9780898719796>
10. Buneman, P.: A characterization of rigid circuit graphs. Discr. Math. **9**, 205–212 (1974)
11. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Inf. Process. Lett. **58**(4), 171–176 (1996)
12. Cai, L.: Parameterized complexity of vertex colouring. Discrete Appl. Math. **127**(3), 415–429 (2003). [https://doi.org/10.1016/S0166-218X\(02\)00242-1](https://doi.org/10.1016/S0166-218X(02)00242-1)
13. Cao, Y.: Linear recognition of almost interval graphs. In: Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1096–1115. SIAM (2016)
14. Cao, Y.: Unit interval editing is fixed-parameter tractable. Inf. Comput. **253**, 109–126 (2017). <https://doi.org/10.1016/j.ic.2017.01.008>
15. Cao, Y., Marx, D.: Interval deletion is fixed-parameter tractable. ACM Trans. Algorithms **11**(3), 21:1–21:35 (2015). <https://doi.org/10.1145/2629595>
16. Cao, Y., Sandeep, R.B.: Minimum fill-in: Inapproximability and almost tight lower bounds. Inf. Comput. **271**, 104514 (2020). <https://doi.org/10.1016/j.ic.2020.104514>
17. Courcelle, B.: The monadic second-order logic of graphs I: recognizable sets of finite graphs. Inf. Comput. **85**, 12–75 (1990)
18. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer, Berlin (2015). <https://doi.org/10.1007/978-3-319-21275-3>
19. Cygan, M., Kratsch, S., Nederlof, J.: Fast hamiltonicity checking via bases of perfect matchings. In: Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC), pp. 301–310. ACM (2013)
20. Cygan, M., Nederlof, J., Pilipczuk, M., Pilipczuk, M., van Rooij, J.M.M., Wojtaszczyk, J.O.: Solving connectivity problems parameterized by treewidth in single exponential time. In: Proceedings of the 52nd Annual Symposium on Foundations of Computer Science (FOCS), pp. 150–159. IEEE (2011)
21. Diestel, R.: Graph Theory, 4th Edition, Graduate texts in mathematics, vol. 173. Springer, Berlin (2012)
22. Downey, R.G., Fellows, M.R.: Fundamentals of parameterized complexity. Texts Comput. Sci. (2013). <https://doi.org/10.1007/978-1-4471-5559-1>
23. Erdős, P., Hajnal, A.: On chromatic number of graphs and set-systems. Acta Math. Acad. Sci. Hungar **17**, 61–99 (1966). <https://doi.org/10.1007/BF02020444>
24. Etscheid, M., Kratsch, S., Mnich, M., Rögl, H.: Polynomial kernels for weighted problems. J. Comput. Syst. Sci. **84**, 1–10 (2017). <https://doi.org/10.1016/j.jcss.2016.06.004>
25. Fomin, F.V., Lokshtanov, D., Panolan, F., Saurabh, S.: Efficient computation of representative families with applications in parameterized and exact algorithms. J. ACM **63**(4), 29:1–29:60 (2016). <https://doi.org/10.1145/2886094>
26. Fomin, F.V., Lokshtanov, D., Saurabh, S., Zehavi, M.: Kernelization. Cambridge University Press, Cambridge (2019). Theory of parameterized preprocessing
27. Fomin, F.V., Saurabh, S., Villanger, Y.: A polynomial kernel for proper interval vertex deletion. SIAM J. Discrete Math. **27**(4), 1964–1976 (2013). <https://doi.org/10.1137/12089051X>
28. Fomin, F.V., Todinca, I., Villanger, Y.: Large induced subgraphs via triangulations and CMSO. SIAM J. Comput. **44**(1), 54–87 (2015). <https://doi.org/10.1137/140964801>
29. Fomin, F.V., Villanger, Y.: Subexponential parameterized algorithm for minimum fill-in. SIAM J. Comput. **42**(6), 2197–2216 (2013)
30. Frank, A., Tardos, É.: An application of simultaneous diophantine approximation in combinatorial optimization. Combinatorica **7**(1), 49–65 (1987). <https://doi.org/10.1007/BF02579200>

31. Garey, M.R., Johnson, D.S.: Computers and Intractability. A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)
32. Gavril, F.: Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM J. Comput. **1**(2), 180–187 (1972). <https://doi.org/10.1137/0201013>
33. Gavril, F.: Algorithms on circular-arc graphs. Networks **4**(4), 357–369 (1974)
34. Gavril, F.: The intersection graphs of subtrees in trees are exactly the chordal graphs. J. Combin. Theory Ser. B **16**, 47–56 (1974). [https://doi.org/10.1016/0095-8956\(74\)90094-x](https://doi.org/10.1016/0095-8956(74)90094-x)
35. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York (1980)
36. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization, *Algorithms and Combinatorics*, vol. 2, second edn. Springer-Verlag, Berlin (1993). <https://doi.org/10.1007/978-3-642-78240-4>
37. Hammer, P.L., Simeone, B.: The splittance of a graph. Combinatorica **1**(3), 275–284 (1981). <https://doi.org/10.1007/BF02579333>
38. Heggernes, P.: Minimal triangulations of graphs: a survey. Discrete Math. **306**(3), 297–317 (2006). <https://doi.org/10.1016/j.disc.2005.12.003>
39. Impagliazzo, R., Paturi, R.: Complexity of k-sat. In: Proceedings of the 14th Annual IEEE Conference on Computational Complexity (CCC), pp. 237–240. IEEE Computer Society (1999). <https://doi.org/10.1109/CCC.1999.766282>
40. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity. J. Comput. Syst. Sci. **63**(4), 512–530 (2001)
41. Jansen, B.M., Kratsch, S.: Data reduction for graph coloring problems. Inf. Comput. **231**, 70–88 (2013)
42. Jansen, B.M.P., Bodlaender, H.L.: Vertex cover kernelization revisited - upper and lower bounds for a refined parameter. Theory Comput. Syst. **53**(2), 263–299 (2013). <https://doi.org/10.1007/s00224-012-9393-4>
43. Jansen, B.M.P., Pilipczuk, M.: Approximation and kernelization for chordal vertex deletion. SIAM J. Discrete Math. **32**(3), 2258–2301 (2018). <https://doi.org/10.1137/17M112035X>
44. Kaplan, H., Shamir, R., Tarjan, R.E.: Tractability of parameterized completion problems on chordal and interval graphs: Minimum fill-in and physical mapping. In: Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS), pp. 780–791. IEEE (1994)
45. Kaplan, H., Shamir, R., Tarjan, R.E.: Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs. SIAM J. Comput. **28**, 1906–1922 (1999)
46. Komusiewicz, C., Uhlmann, J.: Cluster editing with locally bounded modifications. Discrete Appl. Math. **160**(15), 2259–2270 (2012). <https://doi.org/10.1016/j.dam.2012.05.019>
47. Lekkerkerker, C.G., Boland, J.C.: Representation of a finite graph by a set of intervals on the real line. Fund. Math. **51**, 45–64 (1962/1963). <https://doi.org/10.4064/fm-51-1-45-64>
48. Liedloff, M., Montealegre, P., Todinca, I.: Beyond classes of graphs with “few” minimal separators: FPT results through potential maximal cliques. Algorithmica **81**(3), 986–1005 (2019). <https://doi.org/10.1007/s00453-018-0453-2>
49. Marx, D.: Parameterized coloring problems on chordal graphs. Theor. Comput. Sci. **351**(3), 407–424 (2006)
50. Marx, D.: Chordal deletion is fixed-parameter tractable. Algorithmica **57**(4), 747–768 (2010)
51. Matula, D.W., Beck, L.L.: Smallest-last ordering and clustering and graph coloring algorithms. J. ACM **30**(3), 417–427 (1983). <https://doi.org/10.1145/2402.322385>
52. Natanzon, A., Shamir, R., Sharan, R.: A polynomial approximation algorithm for the minimum fill-in problem. In: Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC), pp. 41–47. ACM (1998). <https://doi.org/10.1145/276698.276710>
53. Natanzon, A., Shamir, R., Sharan, R.: Complexity classification of some edge modification problems. Discrete Appl. Math. **113**(1), 109–128 (2001). [https://doi.org/10.1016/S0166-218X\(00\)00391-7](https://doi.org/10.1016/S0166-218X(00)00391-7)
54. Nešetřil, J., de Mendez, P.O.: Sparsity - Graphs, Structures, and Algorithms, *Algorithms and combinatorics*, vol. 28. Springer (2012). <https://doi.org/10.1007/978-3-642-27875-4>
55. Parter, S.: The use of linear graphs in Gauss elimination. SIAM Rev. **3**, 119–130 (1961)
56. Pilipczuk, M.: Problems parameterized by treewidth tractable in single exponential time: a logical approach. CoRR **abs/1104.3057** (2011). arXiv:1104.3057
57. Rao, S., Richa, A.W.: New approximation techniques for some linear ordering problems. SIAM J. Comput. **34**(2), 388–404 (2004). <https://doi.org/10.1137/S0097539702413197>

58. Rose, D.J.: A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In: Read, R.C. (ed.) Graph Theory and Computing, pp. 183–217. Academic Press, New York (1972)
59. Rose, D.J., Tarjan, R.E., Lueker, G.S.: Algorithmic aspects of vertex elimination on graphs. SIAM J. Comput. **5**, 266–283 (1976)
60. Takenaga, Y., Higashide, K.: Vertex coloring of comparability $+ke$ and- $-ke$ graphs. In: International Workshop on Graph-Theoretic Concepts in Computer Science (WG), pp. 102–112. Springer (2006)
61. Tarjan, R.E., 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), 566–579 (1984). <https://doi.org/10.1137/0213035>
62. Vandenberge, L., Andersen, M.S.: Chordal graphs and semidefinite optimization. Foundations and Trends® in Optimization **1**(4), 241–433 (2015)
63. Villanger, Y., Heggernes, P., Paul, C., Telle, J.A.: Interval completion is fixed parameter tractable. SIAM J. Comput. **38**(5), 2007–2020 (2009)
64. Yannakakis, M.: Computing the minimum fill-in is NP-complete. SIAM J. Alg. Disc. Meth. **2**, 77–79 (1981)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.