Inductive $k$-independent graphs and $c$-colorable subgraphs in scheduling: a review

Matthias Bentert · René van Bevern · Rolf Niedermeier

Published online: 18 December 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract
Inductive $k$-independent graphs generalize chordal graphs and have recently been advocated in the context of interference-avoiding wireless communication scheduling. The NP-hard problem of finding maximum-weight induced $c$-colorable subgraphs, which is a generalization of finding maximum independent sets, naturally occurs when selecting $c$ sets of pairwise non-conflicting jobs (modeled as graph vertices). We investigate the parameterized complexity of this problem on inductive $k$-independent graphs. We show that the MAXIMUM INDEPENDENT SET problem is W[1]-hard even on 2-simplicial 3-minoes—a subclass of inductive 2-independent graphs. In contrast, we prove that the more general MAX-WEIGHT $c$-COLORABLE SUBGRAPH problem is fixed-parameter tractable on edge-wise unions of cluster and chordal graphs, which are 2-simplicial. In both cases, the parameter is the solution size. Aside from this, we survey other graph classes between inductive 1-independent and inductive 2-independent graphs with applications in scheduling.

Keywords Independent set · Job interval selection · Interval graphs · Chordal graphs · Inductive $k$-independent graphs · NP-hard problems · Parameterized complexity

1 Introduction
Finding sets of “independent” (that is, pairwise non-conflicting) jobs is of central importance in many scheduling scenarios. In particular, the NP-hard MAXIMUM

INDEPENDENT SET problem and its generalization of finding maximum-weight induced $c$-colorable subgraphs, in both inductive $k$-independent graphs, has recently been identified as key tools in the development of polynomial-time approximation algorithms in interference-avoiding wireless communication scheduling (Ásgeirsson et al. 2017; Halldórsson 2016; Halldórsson and Tonoyan 2015).

We conduct a deeper study of the computational complexity of these problems on subclasses of inductive $k$-independent graphs. We also exhibit a rich fine structure of known graph classes that have applications in scheduling and are inductive 2-independent.

A graph is inductive $k$-independent if its vertices can be ordered from left to right so that the “right-hand” neighborhood of each vertex contains no independent vertex set of size greater than $k$. The problem we study on these graphs is MAXIMUM (WEIGHT) $c$-COLORABLE SUBGRAPH$^1$—find a maximum (weight) vertex subset that induces a subgraph allowing for coloring the vertices in $c$ colors so that no adjacent vertices have the same color. Clearly, for $c = 1$,

---

1 Ásgeirsson et al. (2017) interested in maximum-weight unions of $c$ independent sets. In the graph theory literature, the problem is known as MAX-WEIGHT $c$-COLORABLE SUBGRAPH; we prefer to stick to the established graph theory notion.
we obtain the classical MAXIMUM (WEIGHT) INDEPENDENT SET problem.

Halldórsson and Tonoyan (2015) found that, in interference-avoiding wireless communication scheduling, one typically faces these problems in inductive \( k \)-independent graphs with \( k \leq 12 \). Unfortunately, we will see that already unweighted MAXIMUM INDEPENDENT SET is hard for subclasses of inductive 2-independent graphs. Notably, however, classic scheduling problems have been studied for a number of subclasses of inductive 2-independent graphs, including interval graphs (Kolen et al. 2007), strip graphs (Halldórsson and Karlsson 2006), their superclass 2-track interval graphs (Höhn et al. 2011; van Bevern et al. 2015), and claw-free graphs (Gaur and Krishnamurti 2003; Köse et al. 2017; Köse and Médard 2017).

1.1 Our contributions and organization of this work

Our main contributions are as follows. We refer to Sect. 2 for definitions of graph and complexity classes. In Sect. 3, we explore the fine structure of graph classes below inductive 3-independent graphs, discuss relations to known graph classes (often already appearing in scheduling applications), and also survey the recognition complexities of the respective graph classes.

In Sect. 4, we show that already (unweighted) MAXIMUM INDEPENDENT SET parameterized by the solution size (number of vertices) is \( W[1] \)-hard on 2-simplicial 3-minoes, a proper subclass of inductive 2-independent graphs. Thus, there is no reason to hope for exact fixed-parameter algorithms to find even small independent sets in these graphs.

In contrast, in Sect. 5 we show that MAX-WEIGHT \( c \)-COLORABLE SUBGRAPH parameterized by the solution size is fixed-parameter tractable on a class that lies properly between inductive 1-independent and inductive 2-independent graphs and generalizes the class of strip graphs, which Halldórsson and Karlsson (2006) used to model the JOB INTERVAL SELECTION problem.

Finally, in Sect. 6, we briefly survey complexity results on MAX-WEIGHT \( c \)-COLORABLE SUBGRAPH in the other graph classes discussed in Sect. 3.

We refer to Fig. 1 for an overview of results on the parameterized complexity of MAXIMUM INDEPENDENT SET and MAX-WEIGHT \( c \)-COLORABLE SUBGRAPH on graph classes mainly below inductive 3-independent graphs.

1.2 Related work

It is well known that, on general graphs, already (unweighted) MAXIMUM INDEPENDENT SET is a notoriously hard problem from the viewpoint of polynomial-time approximability as well as from the viewpoint of parameterized complexity. Hence, it is natural to study MAXIMUM INDEPENDENT SET and related problems on special graph classes.

Our results improve on or complement the following known results on MAXIMUM INDEPENDENT SET and MAX-WEIGHT \( c \)-COLORABLE SUBGRAPH on inductive \( k \)-independent graph classes.

- **MAXIMUM INDEPENDENT SET** parameterized by solution size is fixed-parameter tractable on strip graphs (van Bevern et al. 2015); strip graphs are equivalent to the class of cluster-\( \bullet \)-\( \triangle \)-\( \triangleleft \) interval graphs (see Fig. 1), a proper subclass of inductive 2-independent graphs.

- **MAX-WEIGHT INDEPENDENT SET** parameterized by solution size is \( W[1] \)-hard already on unit 2-track interval graphs (Jiang 2010); unit 2-track interval graphs form a subclass of inductive 3-independent graphs (Ye and Borodin 2012).

- **(Unweighted)** MAXIMUM \( c \)-COLORABLE SUBGRAPH parameterized by solution size is fixed-parameter tractable on inductive 1-independent (that is, chordal) graphs (Misra et al. 2013).

2 Preliminaries

This section introduces basic notation and concepts that we will use throughout this work.

2.1 Fixed-parameter algorithms

The essential idea behind fixed-parameter algorithms is to accept exponential running times, which are seemingly inevitable in solving NP-hard problems, but to restrict them to one aspect of the problem, the parameter (Cygan et al. 2015; Downey and Fellows 2013; Flum and Grohe 2006; Niedermeier 2006).

Thus, formally, an instance of a parameterized problem \( \Pi \subseteq \Sigma^* \times \mathbb{N} \) is a pair \((x, k)\) consisting of the input \( x \) and the parameter \( k \). A parameterized problem \( \Pi \) is fixed-parameter tractable (FPT) with respect to a parameter \( k \) if there is an algorithm solving any instance of \( \Pi \) with size \( n \) in \( f(k) \cdot \text{poly}(n) \) time for some computable function \( f \). Such an algorithm is called a fixed-parameter algorithm. It is potentially efficient for small values of \( k \), in contrast to an algorithm that is merely running in polynomial time for each fixed \( k \) (thus allowing the degree of the polynomial to depend on \( k \)). FPT is the complexity class of fixed-parameter tractable parameterized problems.

2.2 Parameterized intractability

To show that a problem is presumably not fixed-parameter tractable, there is a parameterized analog of NP-hardness the-
Fig. 1 For definitions of graph classes, see Sect. 2. Our new results are in bold. White—MAX-WEIGHT c-COLORABLE SUBGRAPH is solvable in polynomial time. Light Gray—MAXIMUM c-COLORABLE SUBGRAPH is polynomial-time solvable, MAXIMUM c-COLORABLE SUBGRAPH is NP-hard, and MAX-WEIGHT c-COLORABLE SUBGRAPH parameterized by solution size is fixed-parameter tractable. Gray—MAXIMUM INDEPENDENT SET is NP-hard. MAX-WEIGHT c-COLORABLE SUBGRAPH parameterized by solution size is fixed-parameter tractable. Dark Gray—MAXIMUM INDEPENDENT SET is NP-hard and W[1]-hard parameterized by the solution size for some F if F does not contain induced subgraphs isomorphic to F. A K_{n,m} is a complete bipartite graph with n vertices on the one side and m vertices on the other.

A graph is c-colorable if one can assign each vertex one of c colors so that the endpoints of each edge have distinct colors.

We study the parameterized complexity of the following problem.

Problem 2.1 (MAX-WEIGHT c-COLORABLE SUBGRAPH)

Input: A graph G = (V, E) with vertex weights \( w: V \rightarrow \mathbb{N} \).

Task: Find a set S \( \subseteq V \) of maximum weight w(S) such that G[S] is c-colorable.

For c = 1, we obtain the classic MAX-WEIGHT INDEPENDENT SET problem.

Problem 2.2 (MAX-WEIGHT INDEPENDENT SET)

Input: A graph G = (V, E) with vertex weights \( w: V \rightarrow \mathbb{N} \).

Task: Find a set S \( \subseteq V \) of pairwise nonadjacent vertices and maximum weight w(S).

If all vertices have weight one, then we call the problems MAXIMUM c-COLORABLE SUBGRAPH and MAXIMUM INDEPENDENT SET, respectively.

2.4 Graph classes

The motivation, applications, and recognition complexity of the following graph classes are discussed in detail in Sect. 3.
We first list a number of classic graph classes and then introduce the central graph class of this work, inductive \( k \)-independent graphs (Definition 2.3).

We use \( A \cap B \) to denote the class of graphs that belong to both classes \( A \) and \( B \).

A graph is **Hamiltonian** if it contains a cycle using each vertex of the graph exactly once.

A graph is **cubic** if each vertex has exactly three neighbors.

A graph is **chordal** if it does not contain cycles of length at least four as induced subgraphs.

A graph is an **interval graph** if its vertices can be represented as intervals of the real line such that two vertices are adjacent if and only if their corresponding intervals intersect. If the vertices can be represented by intervals of equal length, then the graph is a **unit interval graph**.

**Definition 2.2** (Inductive \( k \)-independent) A graph \( G \) is inductive \( k \)-independent if there is a \( k \)-independence ordering \( v_1, v_2, \ldots, v_n \) of its vertices such that all independent sets of \( G[N[v_i] \cap \{v_i, v_{i+1}, \ldots, v_n\}] \) have size at most \( k \) for each \( i \in \{1, 2, \ldots, n\} \).

Ye and Borodin (2012) surveyed inductive \( k \)-independent graphs.

When dropping the order constraint from the definition of inductive \( k \)-independent graphs, that is, if one requires the neighborhood of each vertex to contain independent sets of size at most \( k \), then one obtains the \( K_{1,k+1} \)-free graphs—a proper subclass.

**Definition 2.4** \( (K_{1,k} \text{-free graphs}) \) A graph is \( K_{1,k} \)-free if it does not contain a \( K_{1,k} \) (a tree with one internal node and \( k \) leaves) as an induced subgraph.

Another proper subclass of inductive \( k \)-independent graphs is **k-simplicial graphs** (Ye and Borodin 2012), also studied by e.g. Jamison and Mulder (2000), Kammer et al. (2010), and Halldörsson and Tonoyan (2015).

**Definition 2.5** \( (k \text{-simplicial}) \) A graph is \( k \)-simplicial if there is an ordering of its vertices \( v_1, v_2, \ldots, v_n \) such that \( G[N[v_i] \cap \{v_i, v_{i+1}, \ldots, v_n\}] \) can be partitioned into at most \( k \) cliques for each \( i \in \{1, 2, \ldots, n\} \).

When, again, dropping the order constraint, one gets \( k \)-**minoes** as introduced by Metelsky and Tyshkevich (2003), which are a proper subclass of \( K_{1,k+1} \)-free graphs:

**Definition 2.6** \( (k \text{-mino}) \) A graph is called a \( k \)-mino if each vertex is contained in at most \( k \) maximal cliques.

In several scheduling works (see Sect. 3), one encounters variants of MAX-WEIGHT INDEPENDENT SET in graphs of the following form:

\( A \gg B \) or an alike notation has been used before. Rather, previous work has been coming up with ad hoc names for the classes \( A \gg B \) for various \( A \) and \( B \), often referring to one and the same class by several names.

---

### 3 Graph classes related to scheduling

Before delving deeper into problem-specific considerations, in this section we collect a number of mostly simple properties of and relations between graph classes in the range from cluster graphs (a special case of interval graphs) on the “bottom end” and inductive 2-independent graphs on the “top end.” In this way, we lay the foundations for subsequent studies mainly related to independent set problems (with their connections to scheduling). More specifically, we exhibit the inclusion relationships between various graph classes, discuss the time complexities of their recognition problems (that is, given a graph, how hard is it to decide whether it belongs to the particular graph class), and indicate whether the graph class has already been investigated in the context of scheduling. This section might also be of independent interest to researchers outside of scheduling contexts.

First, we will discuss some kind of backbone structure of graph classes (Sect. 3.1), and then, we will enrich the picture by further natural graph classes (Sect. 3.2).

#### 3.1 Backbone graph classes

To start with, consider Fig. 2a, which depicts the fundamental backbone structure of our graph classes. Basically, it shows the relations between various levels of inductive \( k \)-independent and \( K_{1,k} \)-free graphs, enriched by interval graphs. Notably, here and also in Fig. 2b, all shown containments are proper. Next, we individually discuss the most relevant graph classes.

#### 3.1.1 Interval graphs

Interval graphs certainly are among the most fundamental graph classes studied in the context of scheduling. Among other things, they are used to model interval scheduling problems, where not only the lengths but also the start times of jobs are given and the task is to minimize the number of required machines or to select most profitable set of jobs (Kolen et al. 2007). Other uses include the problem of maximizing the weighted number of just-in-time jobs (exactly meeting their deadlines) on parallel identical machines (Sung and Vlach 2005), mutual exclusion scheduling (Baker and Coffman 1996; Gardi 2009), or selecting the optimal start times of jobs from finite sets (van Bevern et al. 2015).

Interval graphs can be recognized in linear time (Booth and Luiker 1976), contain cluster graphs, and are special cases of chordal...
Fig. 2 Graph classes discussed in our work. The gray graph classes have been studied in the scheduling literature before. Rectangular boxes show graph classes recognizable in polynomial time, and boxes with rounded corners show graph classes whose recognition is NP-hard or W[1]-hard. An edge from a lower graph class to a higher graph class means that the lower one is properly contained in the upper one.

3.1.2 Cluster graphs

These are disjoint unions of cliques (often occurring in the context of graph-based data clustering (Bansal et al. 2004; Shamir et al. 2004) and are special cases of unit interval graphs. They are exactly the $K_{1,2}$-free graphs (where $K_{1,2}$ is a path on three vertices). It is easy to see that they can be recognized in polynomial time. In scheduling, cluster graphs naturally occur as conflict graphs when, for example, modeling several variants of one job as multiple vertices and only one of them is allowed to be in a solution (for example, a job might be available at various starting times or with various processing times, which influences its profit or weight) (Halldórsson and Karlsson 2006; van Bevern et al. 2015).

3.1.3 Chordal graphs

Chordal graphs are equivalent to inductive 1-independent graphs (Aksoy et al. 2002; Blair and Peyton 1993; Ye and Borodin 2012). They are a well-known superclass of interval graphs (Dirac 1961). They can be recognized in linear time (Rose et al. 1976) and are applied in modeling throughput-maximization scheduling problems in wireless networks (Birand et al. 2012). Moreover, the problem of scheduling jobs with unit execution times and precedence constraints on parallel identical machines is polynomial-time solvable if the incomparability graph of the partial order determining the precedence constraints is chordal (Papadimitriou and Yannakakis 1979).

3.1.4 Claw-free graphs

Claw-free graphs, that is, $K_{1,3}$-free graphs, trivially contain cluster graphs and are incomparable with both interval and chordal graphs (a $K_{1,3}$ is an interval graph and a chordal graph, yet an induced 4-cycle is $K_{1,3}$-free but not chordal and thus not an interval graph). It is easy to see that they can be recognized in polynomial time. Claw-free graphs appear in the context of wireless scheduling (Köse et al. 2017; Köse and Méard 2017) as well as in classic scheduling scenarios (Gaur and Krishnamurti 2003).

3.1.5 Inductive $k$-independent graphs for $k \geq 2$

These graphs recently gained increased interest in the context of wireless scheduling (Ásgeirsson et al. 2017; Halldórsson...
and Tonoyan 2015) and clearly contain chordal graphs and claw-free graphs (both by definition).

There is a straightforward polynomial-time recognition algorithm; however, the degree of the polynomial depends on $k$, and thus, the algorithm is impractical already for small $k$. On the one hand, this is unfortunate in view of the fact that algorithms for inductive $k$-independent graphs, for example the approximation algorithm for MAX-WEIGHT $c$-COLORABLE SUBGRAPH suggested by Ye and Borodin (2012), require the $k$-independence ordering to be known. On the other hand, like in the wireless scheduling applications of Ásgeirsson et al. (2017) and Halldórsson and Tonoyan (2015), a sufficiently good $k$-independence ordering is given directly by the application data and does not have to be computed.

As observed by Ye and Borodin (2012), having the degree in the polynomial of the running time of the recognition algorithm for inductive $k$-independent graphs depending on $k$ is unavoidable unless FPT = W[1]. Since Ye and Borodin (2012) omitted the formal proof, for the sake of completeness we provide it here.

Proposition 3.1 Deciding whether a graph is inductive $k$-independent is coW[1]-hard with respect to $k$.

Proof We reduce any instance $(G = (V, E), k + 1)$ of MAXIMUM INDEPENDENT SET to an instance $G' = (V', E')$ such that $G$ contains an independent set of size $k + 1$ if and only if $G'$ is not inductive $k$-independent. Since MAXIMUM INDEPENDENT SET is W[1]-hard parameterized by $k$, this shows that recognizing non-inductive $k$-independent graphs is W[1]-hard. Thus, recognizing inductive $k$-independent graphs is coW[1]-hard.

The reduction works as follows. Let $V = \{v_1, v_2, \ldots, v_n\}$. Then, $V' := V \cup \{u_1, u_2, \ldots, u_{k+1}\}$ for $k + 1$ new vertices $u_1, u_2, \ldots, u_{k+1}$ and $E' := E \cup \{(u_i, v_j) \mid 1 \leq i \leq k + 1, 1 \leq j \leq n\}$. This completes our reduction.

Now, assume that a maximum independent set in $G$ has size at most $k$. We show that $G'$ is inductive $k$-independent. Put each of the newly introduced vertices $u_i$ for $i \in \{1, 2, \ldots, k + 1\}$ first within the $k$-independence ordering and then all vertices $v \in V$ in an arbitrary order. Since $N_G(u_i) = V$, the neighborhood of any vertex (ignoring vertices that come before it in the $k$-independence ordering) induces a subgraph of $G$. Thus, it trivially holds that this neighborhood only contains independent sets of size at most $k$.

Now, assume that $G$ contains an independent set of size at least $k + 1$. Then, $G'$ is not inductive $k$-independent: No vertex $u_i$ for $i \in \{1, 2, \ldots, k + 1\}$ can be the first in a $k$-independence ordering since $N(u_i) = V$. Moreover, no vertex $v_j$ for $j \in \{1, 2, \ldots, n\}$ can be the first in an $k$-independence ordering since $N(v_j) \supseteq \{u_1, u_2, \ldots, u_{k+1}\}$, which are pairwise nonadjacent. □

3.1.6 $K_{1,k}$-free graphs for $k \geq 4$

These are obvious superclasses of claw-free graphs, and by definition, each $K_{1,k}$-free graph is inductive $(k - 1)$-independent. Again, there is a straightforward polynomial-time recognition algorithm; however, the degree of the polynomial depends on $k$, and thus, the algorithm is impractical already for small $k$. A folklore reduction shows that this is unavoidable unless FPT = W[1].

Proposition 3.2 Recognizing $K_{1,k}$-free graphs is coW[1]-hard with respect to $k$.

Proof We will give a parameterized reduction from MAXIMUM INDEPENDENT SET, which is W[1]-hard parameterized by solution size $k$. Let $(G = (V, E), k)$ be an instance of MAXIMUM INDEPENDENT SET. We construct a graph $G'$ that is not $K_{1,k}$-free if and only if $G$ contains an independent set of size $k$.

Let $u$ be a vertex not in $V$ and $G' = (V \cup \{u\}, E \cup \{(u, v) \mid v \in V\})$. If $G$ contains an independent set $I$ of size at least $k$, then $G'[I \cup \{u\}]$ is a $K_{1,k}$. Thus, $G'$ is not $K_{1,k}$-free. Now, assume that $G$ does not contain an independent set of size at least $k$. Then, $G'$ does not contain such an independent set either since $u$ is adjacent to all other vertices of $G'$. Thus, no vertex in $G'$ can have $k$ pairwise nonadjacent neighbors in $G'$, and thus, $G'$ is $K_{1,k}$-free. □

3.2 Further graph classes

Now, we present a number of graph classes that enrich our backbone structure; some of the new graph classes have prominently appeared in the context of scheduling. Adding these further graph classes to Fig. 2a leads to the “richer” Fig. 2b. We remark that even though we could not spot literature references for some of the subsequent graph classes in the context of scheduling, we advocate their relevance in the scheduling context because they naturally generalize or specialize established “scheduling graph classes”; thus, considering these classes may lead to strengthenings of some known results.

3.2.1 Strip graphs

These are equivalent to the class of cluster $\succ$ interval graphs and thus form an obvious superclass of interval graphs. Their recognition is NP-hard; they find applications in classic scheduling (Halldórsson and Karlsson 2006), in particular in modeling the JOB INTERVAL SELECTION problem introduced by Nakajima and Hakimi (1982).

3.2.2 Cluster $\sqsupset$ chordal graphs

These graphs form an obvious superclass of strip graphs and of chordal graphs. We will see that their recognition is NP-
hard as well (under Turing reductions; see Proposition 3.6). We are not aware of a direct scheduling application.

We will now show that cluster $\bowtie$ chordal graphs form a proper subclass of 2-simplicial graphs and then prove the hardness of recognizing cluster $\bowtie\bowtie$ chordal graphs.

**Proposition 3.3** Cluster $\bowtie$ chordal graphs are a proper subclass of 2-simplicial graphs.

**Proof** Consider a cluster $\bowtie$ chordal graph $G$. We show that any of its induced subgraphs $G'$ contains a vertex whose neighborhood can be covered by two cliques. Repeatedly deleting such vertices from $G$ yields an ordering as required by Definition 2.5.

Note that $G'$, like its supergraph $G$, is also an edge-wise union of a chordal graph $G_1$ and a cluster graph $G_2$ on the same vertex set. Since $G_1$ is chordal, it contains a vertex $v$ whose neighborhood is a clique (Blair and Peyton 1993). Its neighborhood in $G_2$ is also a clique. Thus, the neighborhood of $v$ in $G'$ can be covered by two cliques. This concludes the proof that cluster $\bowtie$ chordal graphs form a subclass of 2-simplicial graphs.

Next we show that this inclusion is strict. To this end, we show that $K_{2,4}$ is 2-simplicial but not a cluster $\bowtie$ chordal graph. A $K_{2,4}$ is 2-simplicial since every induced subgraph contains a vertex whose neighborhood can be covered using two cliques. It is, however, not cluster $\bowtie$ chordal: If $K_{2,4}$ was the edge-wise union of a chordal graph $G_1$ and a cluster graph $G_2$, then $G_2$ would contain at most two edges of $K_{2,4}$. Independently of the choice of these two edges, the remaining edges contain an induced $C_4$ and thus cannot be part of $G_1$.

Next, we show that recognizing cluster $\bowtie$ chordal graphs is NP-hard. We will basically use the same reduction from the NP-hard HAMILTONIAN CYCLE problem on triangle-free cubic graphs that Halldórsson and Karlsson (2006) used to show that recognizing strip graphs is NP-hard. Yet we need to adapt their proof, particularly concerning the Hamiltonicity of triangle-free cubic graphs (Lemma 3.5). For the sake of self-containedness, we provide all details, including also parts that coincide with Halldórsson and Karlsson (2006).

**Problem 3.4** (HAMILTONIAN CYCLE)

**Input:** A graph $G = (V, E)$

**Question:** Is there a cycle in $G$ that passes through every vertex exactly once?

We start with an auxiliary lemma.

**Lemma 3.5** A triangle-free cubic graph $G = (V, E)$ is Hamiltonian if and only if, for each vertex $v$, there is an edge $e$ incident to $v$ such that $G' = (V, E \setminus \{e\})$ is a cluster $\bowtie\bowtie$ chordal graph.

**Proof** “⇒” Assume that $G$ contains a Hamiltonian cycle $C = (c_1, c_2, \ldots, c_n)$. Then, $\{c_i, c_{i+1}\} \in E$ for $i \in \{1, 2, \ldots, n-1\}$ and $\{c_n, c_1\} \in E$. Without loss of generality, let $v = c_n$. We will prove that $E \setminus \{c_n, c_1\}$ can be partitioned into two sets $E_1$ and $E_2$ such that $(V, E_1)$ is a cluster graph and $(V, E_2)$ is chordal. First, we define $E_2 = \{(c_i, c_{i+1}) \mid 1 \leq i \leq n - 1\}$. The graph $(V, E_2)$ is a path and, hence, chordal. Now, observe that each vertex has degree exactly one in $G_1$. Thus, $G_1$ is a perfect matching and, thus, a cluster graph.

“⇐” Assume that, for an arbitrary but fixed vertex $v$, there is an edge $\{v, w\}$ such that $G' = (V, E \setminus \{\{v, w\}\})$ is a cluster $\bowtie\bowtie$ chordal graph. We will show that $G$ contains a Hamiltonian cycle. Fix two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \cup E_2 = E$, where $G_1$ is a cluster graph and $G_2$ is a chordal graph. Since $G$ is triangle-free, it holds that $G_1$ and $G_2$ are triangle-free, too. Every vertex has degree at most one in $G_1$ since, otherwise, there would be a connected component of size at least three in $G_1$, which is a contradiction to the fact that $G_1$ is a triangle-free cluster graph. Since $G$ is a cubic graph, it holds that each vertex except for $v$ and $w$ has degree at least two in $G_2$. Since $G_2$ is chordal, it has an 1-independence ordering, that is, for each vertex $u$, the set of succeeding neighbors of $u$ induce a clique. Thus, each vertex has at most one succeeding neighbor; otherwise, $G_2$ would contain a triangle. Since there is an ordering of the vertices of $G_2$ such that each vertex has at most one succeeding neighbor and only $v$ and $w$ can have degree one in $G_2$, we get that $G_2$ is a path through all vertices with $v$ and $w$ as endpoints. Adding $\{v, w\}$ to this path gives a Hamiltonian cycle for $G$.

Using Lemma 3.5, we can now easily show the following.

**Proposition 3.6** Recognizing cluster $\bowtie\bowtie$ chordal graphs is NP-hard.

**Proof** Assume that cluster $\bowtie\bowtie$ chordal graphs are recognizable in polynomial time. Assuming this, we derive a polynomial-time algorithm for the NP-hard problem of checking whether a triangle-free cubic graph $G$ is Hamiltonian (West and Shmyos 1984), which implies P = NP.

Let $v$ be any vertex of $G$. Since $G$ is cubic, there are three edges incident to $v$ in $G$. If $G$ is Hamiltonian, then, by Lemma 3.5, deleting one out of the three edges will turn $G$ into a cluster $\bowtie\bowtie$ chordal graph. Our algorithm simply tries out all three possibilities and, for each of them, checks whether we get a cluster $\bowtie\bowtie$ chordal graph in polynomial time.

### 3.2.3 2-simplicial graphs

These graphs form a superclass of cluster $\bowtie\bowtie$ chordal graphs (Proposition 3.3) and, by definition, are inductive 2-independent.
They can be easily recognized in polynomial time by the following algorithm: As long as possible, find and delete a vertex whose neighborhood can be covered by two cliques, which is equivalent to checking whether the complement of the neighborhood is 2-colorable. The input graph is 2-simplicial if and only if this process terminates with the empty graph as result. We are not aware of a direct scheduling application.

3.2.4 3-Minoes

In 3-minoes, every vertex is contained in at most three maximal cliques. It is straightforward to see that they form a subclass of $K_{1,4}$-free graphs; they can be recognized in polynomial time (Metelsky and Tyshkevich 2003). We are not aware of a direct scheduling application.

3.2.5 Interval $\Rightarrow$ interval graphs

This graph class is also known as 2-track interval graphs and as 2-union graphs. They trivially contain interval graphs. The corresponding recognition problem is NP-hard (Gyárfás and West 1995). They have strong scheduling applications in industrial steel manufacturing (Höhn et al. 2011).

3.2.6 Unit interval $\Rightarrow$ unit interval graphs

These are special cases of 2-track interval graphs and are also called 2-track unit interval graphs. They are contained in the class of inductive 3-independent graphs (Ye and Borodin 2012) and in the class of $K_{1,5}$-free graphs because unit interval graphs are $K_{1,3}$-free (Wegner 1961). Their recognition problem is NP-hard (Jiang 2013). We are not aware of a direct scheduling application.

4 W[1]-hardness on 2-simplicial 3-minoes

MAXIMUM INDEPENDENT SET parameterized by the solution size $\ell$ is W[1]-hard on inductive 3-independent graphs; Jiang (2010) showed this on unit interval $\Rightarrow$ unit interval graphs, which are a subclass of inductive 3-independent graphs (Ye and Borodin 2012).

In contrast, the more general MAXIMUM $c$-COLORABLE SUBGRAPH problem parameterized by the solution size $\ell$ is fixed-parameter tractable on inductive 1-independent (that is, chordal) graphs (Misra et al. 2013). In Sect. 5, we will generalize this tractability result to cluster $\Rightarrow$ chordal graphs, a non-chordal proper subclass of inductive 2-independent graphs (cf. Fig. 2b).

Strengthening the result by Jiang (2010), we show that MAX-WEIGHT $c$-COLORABLE SUBGRAPH in inductive 2-independent graphs is W[1]-hard. More precisely, we show that already MAXIMUM INDEPENDENT SET parameterized by the solution size $\ell$ is W[1]-hard even on 2-simplicial 3-minoes (see Definitions 2.5 and 2.6). Since these are $K_{1,4}$-free (see Fig. 1), our result also contrasts the fact that MAX-WEIGHT INDEPENDENT SET is solvable in $O(n^3)$ time on $K_{1,3}$-free graphs (Faenza et al. 2011).

Theorem 4.1 MAXIMUM INDEPENDENT SET parameterized by solution size $\ell$ is W[1]-hard on 2-simplicial 3-minoes.

It may be tempting to prove Theorem 4.1 using a parameterized reduction from MAXIMUM INDEPENDENT SET on graphs of small degree; however, such efforts must be in vein since MAXIMUM INDEPENDENT SET parameterized by solution size is fixed-parameter tractable on graphs even with logarithmic degree via a simple search tree algorithm. To prove Theorem 4.1, we use a parameterized reduction from the MULTICOLORED CLIQUE problem, which is W[1]-hard with respect to $k$ (Fellows et al. 2009).

Problem 4.2 (MULTICOLORED CLIQUE)

Input: A graph $G$ whose vertex set is partitioned into $k$ independent sets $V_1 \cup V_2 \cup \cdots \cup V_k$.

Task: Does $G$ contain a complete subgraph of order $k$?

We will also refer to the sets $V_i$ for $i \in \{1, 2, \ldots, k\}$ as color classes. We now describe our parameterized reduction from MULTICOLORED CLIQUE to MAXIMUM INDEPENDENT SET, which is illustrated in Fig. 3. Subsequently, we prove its correctness and that it creates 2-simplicial 3-minoes.

Construction 4.3 Given a MULTICOLORED CLIQUE instance $G = (V_1 \cup V_2 \cup \cdots \cup V_k, E)$, we create an instance $G'$ of MAXIMUM INDEPENDENT SET that has a solution of size $\ell := k^2 + \binom{k}{2}$ if and only if $G$ has a clique of size $k$.

For each color class $V_i = \{v_1^{(i)}, v_2^{(i)}, \ldots, v_{n_i}^{(i)}\}$ of $G$, graph $G'$ contains a vertex selection gadget $U_i$, which consists of $k$ cliques $U_i \rightarrow j$, where $j \in \{1, 2, \ldots, k\}$. For each $j \in \{1, 2, \ldots, k\}$, clique $U_i \rightarrow j$ will be used to connect gadget $U_i$ to gadget $U_j$ and consists of newly introduced vertices $\{u^{(1)}_{i \rightarrow j}, u^{(2)}_{i \rightarrow j}, \ldots, u^{(n_i)}_{i \rightarrow j}\}$, each vertex $u^{(p)}_{i \rightarrow j}$ with $p \in \{2, 3, \ldots, n_i\}$ of which is adjacent to each vertex $u^{(q)}_{i \rightarrow j}$ with $q \in \{1, 2, \ldots, p-1\}$ and $l = (j \mod k) + 1$.

Furthermore, for each two color classes $V_i = \{v_1^{(1)}, v_2^{(1)}, \ldots, v_{n_i}^{(1)}\}$ and $V_j = \{v_1^{(2)}, v_2^{(2)}, \ldots, v_{n_j}^{(2)}\}$ of $G$ such that $1 \leq i < j \leq k$, graph $G'$ contains a verification gadget $E_{i \rightarrow j}$, which is a clique on the vertices $e^{i-j}_{p-q}$, newly introduced for each edge $(v_i^{(p)}, v_j^{(q)}) \in E$ of $G$.

We connect the gadgets so that choosing a vertex $e^{i-j}_{p-q}$ of a verification gadget $E_{i \rightarrow j}$ into an independent set enforces that $v_i^{(p)}$ of $V_i$ and $v_j^{(q)}$ of $V_j$ are also in the independent set, where $i$ is the smaller color index (which is reflected in the naming convention). To this end, for each edge $(v_i^{(p)}, v_j^{(q)}) \in E$ with $1 \leq i < j \leq k$ of $G$, vertex $u^{(p)}_{i \rightarrow j}$ is adjacent to all vertices $e^{i-j}_{p-q}$ such that $p \neq p'$ and $u^{(q)}_{j \rightarrow i}$ is adjacent to all vertices $e^{i-j}_{p-q}$ such that $q \neq q'$. □
We now prove the correctness of Construction 4.3. Thereafter, it remains to show that it creates 2-simplicial 3-minoes.

Lemma 4.4  A MULTICOLORED CLIQUE instance $G = (V_1 \uplus V_2 \uplus \cdots \uplus V_k, E)$ has a clique of order $k$ if and only if the graph $G'$ created by Construction 4.3 has an independent set of size $\ell := k^2 + \binom{k}{2}$.

Proof (⇒) Let $S$ be a clique of order $k$ in $G$. It contains exactly one vertex of each color class. We describe an independent set $S'$ of size $\ell$ for $G'$. To this end, denote each color class $V_i$ for $i \in \{1, 2, \ldots, k\}$ of $G$ as $V_i = \{v_i^{(1)}, v_i^{(2)}, \ldots, v_i^{(n_i)}\}$.

For each vertex $v_i^{(p)}$ in $S$, set $S'$ contains the $k$ vertices $u_i^{(p, j)}$ with $j \in \{1, 2, \ldots, k\}$ of the vertex selection gadget $U_i$. Thus, $S'$ contains exactly one vertex of each clique $U_i \rightarrow j$ and these vertices are pairwise nonadjacent: There are neither edges between vertices $u_i^{(p, j)}$ and $u_i^{(p, l)}$ for any $j \neq l$ in the set $\{j, l\} \subseteq \{1, 2, \ldots, k\}$ nor are there edges between distinct vertex selection gadgets $U_i$ and $U_j$ in $G'$.

For each edge $\{v_i^{(p)}, v_j^{(q)}\}$ with $1 \leq i < j \leq k$ of $S$, set $S'$ contains the vertex $e_{i<j}^{(p,q)}$ of the verification gadget $E_{i<j}$. Since there are no edges between verification gadgets, these vertices are pairwise nonadjacent. Moreover, note that $e_{i<j}^{(p,q)}$ has neither edges to $u_i^{(p, j)}$ nor to $u_j^{(q, i)}$, which are the only ver-
ties of $U_{j \rightarrow j}$ and $U_{j \rightarrow i}$ in $S'$, nor has it edges to any vertex in $U_{i \rightarrow j}$ for $\{i', j'\} \neq \{i, j\}$. Thus, $S'$ is an independent set.

Finally, $S'$ has size $\ell = k^2 + (\ell - 1)$. It contains $k$ vertices for each of $k$ color classes of $G$ and one vertex for each edge of the clique $S$.

(\Leftrightarrow) Let $S'$ be an independent set of size $\ell$ for $G'$. We describe a clique of order $k$ in $G$. Since each vertex of $G'$ is in one of the $\ell$ pairwise vertex-disjoint cliques $U_{i \rightarrow j}$ with $\{i, j\} \subseteq \{1, 2, \ldots, k\}$ and $E_{i \rightarrow j}$ with $1 \leq i < j \leq k$, $S'$ contains exactly one vertex of each of them. Let $S$ be the set of vertices $v^q_i$ of $G$ for $i \in \{1, 2, \ldots, k\}$ such that $u^{(i)}_{i,j}$ of $U_{i \rightarrow j}$ is in $S'$. We will prove that the vertices in $S$ are pairwise adjacent in $G$.

To this end, we first prove that $p = q$ for any two vertices $u^{(i)}_{i,j}$ and $u^{(j)}_{j,l}$ in $S'$, where $\{i, j, l\} \subseteq \{1, 2, \ldots, k\}$. To this end, note that $G'$ contains edges from vertex $u^{(i)}_{i,j}$ in $S'$ to each vertex $u^{(j)}_{j,l}$ with $q' \in \{1, 2, \ldots, p - 1\}$ and $l' = (j \mod k) + 1$. Thus, for the vertex $u^{(i)}_{i,j}$ in $S'$, we have $q' > p$. Iterating the argument, we get $q \geq q' \geq p$, and, iterating further, $p \geq q \geq q' \geq p$, and thus $p = q$.

We now prove that two arbitrary vertices $v^p_i$ and $v^q_j$ with $1 \leq i < j \leq k$ are adjacent in $G$. By choice of $S$, we have that $u^{(i)}_{i,j}$ and $u^{(j)}_{j,l}$ are in $S'$. As shown in the previous paragraph, we thus also have that $u^{(p)}_{i,j}$ and $u^{(q)}_{j,l}$ are in $S'$. Since $u^{(p)}_{i,j}$ is adjacent to all vertices $e^{p,q}_{i,j}$ of $E_{i \rightarrow j}$ such that $p = p'$ and $u^{(q)}_{j,l}$ is adjacent to all vertices $u^{(q)}_{j,l}$ such that $q = q'$, we get that $S'$ contains vertex $e^{p,q}_{i,j}$ of $E_{i \rightarrow j}$. The existence of this vertex in $E_{i \rightarrow j}$ shows that there is an edge $\{v^p_i, v^q_j\} \in E$ in $G$. \hfill \Box

We will use the following lemma to show that Construction 4.3 generates 2-inductive 3-minoes by showing that, in any induced subgraph of the graphs generated by Construction 4.3, we can find a vertex for which at least one of the three sets in the following lemma are empty or contain only one vertex:

**Lemma 4.5** Let $u^{(i)}_{i,j}$ with $1 \leq i \leq j \leq k$ be a vertex in the graph created by Construction 4.3 from an instance $G = (V_1 \uplus V_2 \uplus \cdots \uplus V_k, E)$ of MULTICOLORED CLIQUE.

Then, the neighborhood of $u^{(i)}_{i,j}$ can be covered by at most three cliques, consisting of

\begin{itemize}
  \item[(i)] $\{u^{(q)}_{i,j} \mid q \geq p\} \cup \{u^{(q)}_{j,l} \mid q < p, l = (j \mod k) + 1\}$,
  \item[(ii)] $\{u^{(q)}_{i,j} \mid q \leq p\} \cup \{u^{(q)}_{i,l} \mid q > p, j = (l \mod k) + 1\}$, and
  \item[(iii)] $u^{(p)}_{i,j}$ and its neighbors in either $E_{i < j}$ or $E_{j < i}$, if $i \neq j$.
\end{itemize}

**Proof** We first show that each of the sets (i)–(ii) is a clique. This easily follows for (iii) since $E_{i < j}$ for $1 \leq i < j \leq k$ is a clique by Construction 4.3.

We now prove that (i) and (ii) are cliques. By Construction 4.3, each vertex $u^{(p)}_{i,j}$ is adjacent to each vertex $u^{(q')}_{j,l}$ with $q' \in \{1, 2, \ldots, p - 1\}$ and $l' = (j' \mod k) + 1$. Herein, intuitively, we can think of $l'$ as the “successor” to $j'$ in the cycle $(1, 2, \ldots, k, 1)$. By setting $p' := p$ and $j' := j$, we immediately get that (i) is a clique. By choosing $p' := p + 1$ and $l' := j$, we get that (ii) is a clique, since in (ii) the left set of the union now plays the role of the “successor” of the right set.

It remains to prove that the sets (i)–(iii) cover the whole neighborhood of $u^{(p)}_{i,j}$. Vertex $u^{(p)}_{i,j}$ has neighbors in all sets $U_{i \rightarrow j}$ such that $l = j$, $l = (j \mod k) + 1$, or $j = (l \mod k) + 1$. If $i \neq j$, then it might also have neighbors in either $E_{i < j}$ or $E_{j < i}$. Thus, the union of the sets in (i)–(iii) covers all neighbors in $U_{i \rightarrow j}$, $E_{i < j}$, and $U_{j \rightarrow i}$. Moreover, set (i) covers all neighbors in $U_{i \rightarrow j}$ for $l = (j \mod k) + 1$ by Construction 4.3. Finally, if set (ii) did not cover all neighbors in $U_{i \rightarrow j}$ with $j = (l \mod k) + 1$, then this would mean that $u^{(p)}_{i,j}$ had a neighbor $u^{(q)}_{j,l}$ with $q \leq p$. This is impossible since, by Construction 4.3, $u^{(p)}_{i,j}$ is adjacent to vertices $u^{(p)}_{i,j}$ only for $p < q$. \hfill \Box

Using Lemma 4.5, the following lemma is easy to prove and finishes the proof of Theorem 4.1.

**Lemma 4.6** Construction 4.3 creates 2-simplicial 3-minoes.

**Proof** We first show that Construction 4.3 creates 3-minoes. By Lemma 4.5, the neighborhood of each vertex in vertex selection gadgets can be covered by three cliques. Also a vertex of a verification gadget $E_{i < j}$ has neighbors only in the three sets $E_{i < j}, U_{i \rightarrow j}$, and $U_{j \rightarrow i}$, each of which is a clique.

We now show that the graphs $G$ created by Construction 4.3 are 2-simplicial. To this end, it is enough to show that each induced subgraph $G'$ of $G$ contains a vertex whose neighborhood can be covered by two cliques. Then, subsequently deleting a vertex whose neighborhood can be covered by two cliques gives an ordering as required by Definition 2.5.

If $G'$ contains only vertices of verification gadgets, then the neighborhood of each vertex in $G'$ can be covered using one clique. If $G'$ contains a vertex $u^{(p)}_{i,j}$ of $G$ for any $i \in \{1, 2, \ldots, k\}$, then, by Lemma 4.5, its neighborhood can be covered by two cliques. Otherwise, there are $j \in \{1, 2, \ldots, k\}$ and $l = (j \mod k) + 1$ such that $G'$ contains vertices of $U_{i \rightarrow j}$ but no vertices of $U_{i \rightarrow j}$. Let $u^{(p)}_{i,j}$ be the vertex of $U_{i \rightarrow j}$ with maximum $p$ in $G'$. For this vertex, the set (i) in Lemma 4.5 only contains $u^{(p)}_{i,j}$ and is therefore contained in the other two sets. Thus, its neighborhood can be covered by two cliques—sets (ii) and (iii). \hfill \Box

Theorem 4.1 now follows from Lemmas 4.4 and 4.6, the fact that Construction 4.3 runs in polynomial time and that MULTICOLORED CLIQUE is W[1]-hard with respect to $k$ (Fellows et al. 2009).
5 Fixed-parameter tractability on cluster $\ll$ chordal graphs

In this section, we prove that MAX-WEIGHT $c$-COLORABLE SUBGRAPH parameterized by the solution size $\ell$ is fixed-parameter tractable on cluster $\ll$ chordal graphs. This complements the negative result of Sect. 4.

**Theorem 5.1** A maximum-weight $c$-colorable subgraph with at most $\ell$ vertices of a cluster $\ll$ chordal graph can be computed in $2^{\ell+c} \cdot (c+e+3)^{\ell} \cdot O(\log \ell) \cdot n^2 \cdot \log^3 n$ time if the decomposition of the input graph into a cluster and a chordal graph is given. Herein, $e$ is Euler’s number.

Note that $c \leq \ell$ holds in all nontrivial cases. Thus, Theorem 5.1 shows that MAX-WEIGHT $c$-COLORABLE SUBGRAPH parameterized by $\ell$ is fixed-parameter tractable in cluster $\ll$ chordal graphs. Moreover, note that it also shows that MAX-WEIGHT $c$-COLORABLE SUBGRAPH parameterized by the weight $W$ of the sought solution is fixed-parameter tractable, since if there is an independent set of weight $W$, then there is also one consisting of at most $W$ vertices.

On the one hand, Theorem 5.1 is a generalization of a result of Misra et al. (2013), who showed that MAXIMUM $c$-COLORABLE SUBGRAPH parameterized by the solution size $\ell$ is fixed-parameter tractable on chordal graphs. On the other hand, it generalizes a fixed-parameter tractability result on the JOB INTERVAL SELECTION problem of van Bevern et al. (2015), who showed that MAXIMUM INDEPENDENT SET parameterized by the solution size $\ell$ is fixed-parameter tractable on interval graphs. The proof of Theorem 5.1 works in three steps. First, in Sect. 5.1, we use the color coding technique due to Alon et al. (1995) to reduce MAX-WEIGHT $c$-COLORABLE SUBGRAPH to MAX-WEIGHT INDEPENDENT SET. Then, in Sect. 5.2, we again use color coding to reduce MAX-WEIGHT INDEPENDENT SET in cluster $\ll$ chordal graphs to the problem of finding a maximum-weight independent set of pairwise distinct colors in chordal graphs whose vertices are colored in $\ell$ colors. Finally, in Sect. 5.3, we use dynamic programming to find this colorful independent set.

5.1 From MAX-WEIGHT $c$-COLORABLE SUBGRAPH to MAX-WEIGHT INDEPENDENT SET

In order to prove Theorem 5.1, we show that it is enough to give a fixed-parameter algorithm for MAX-WEIGHT INDEPENDENT SET parameterized by the solution size $\ell$.

To this end, we use the color coding technique of Alon et al. (1995). This technique has been used by Misra et al. (2013) to show that MAXIMUM $c$-COLORABLE SUBGRAPH parameterized by the solution size $\ell$ is fixed-parameter tractable in chordal graphs. Their approach is neither limited to the unweighted problem nor to chordal graphs. Note, however, that the approach requires some tuning to work in the weighted setting. We describe it here in the most general form. To this end, we need the following definition.

**Definition 5.2** (Hereditary graph class) A graph class $C$ is hereditary if every induced subgraph of a graph in $C$ also belongs to $C$.

All graph classes considered in this work are hereditary.

**Lemma 5.3** Let $C$ be a hereditary graph class. Moreover, let $t: \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function, nondecreasing in both arguments, such that a maximum-weight independent set of size at most $\ell$ in $C$ can be computed in $t(\ell, n)$ time.

Then, a maximum-weight induced $c$-colorable subgraph with at most $\ell$ vertices of a graph in $C$ is computable in $2^{c+e+3^{\ell}} \cdot O(\log \ell) \cdot \log^2 n \cdot t(\ell, n)$ time.

**Proof** Our algorithm makes use of $(n, \ell, c)$-universal sets (Misra et al. 2013)—sets $F$ of functions $f: \{1, 2, \ldots, n\} \to \{1, 2, \ldots, c\}$ such that, for any set $I \subseteq \{1, 2, \ldots, n\}$ of size $|I| = \ell$ and any function $g: \{1, 2, \ldots, n\} \to \{1, 2, \ldots, c\}$, there is a function $f \in F$ with $f(i) = g(i)$ for each $i \in I$.

Our algorithm computes a maximum-weight $c$-colorable subgraph $G[S]$ with at most $\ell$ vertices of a graph $G = (\{v_1, v_2, \ldots, v_n\}, E) \in C$ as follows. First, in $c^{\ell} O(\log \ell) n \log^2 n$ time, compute an $(n, \ell, c)$-universal set $F$ of size $|F| = c^{\ell} \cdot O(\log \ell) \cdot \log^2 n$ (Misra et al. 2013). Then, iterate over each $f \in F$ and each of the $(\ell+c)^{\ell}$ possible partitions $\ell_1 + \ell_2 + \cdots + \ell_c \leq \ell$ such that $\ell_i \geq 0$ for each $i \in \{1, 2, \ldots, c\}$. In each iteration, do the following: Compute a maximum-weight independent set $S_i$ of size at most $\ell_i$ in the subgraph $G_i$ induced by the vertices $v_k$ with $f(k) = i$ for each $i \in \{1, 2, \ldots, c\}$. Return the maximum-weight set $S' = S_1 \cup S_2 \cup \cdots \cup S_c$ found in any iteration. Note that the independent set $S_i$ in each graph $G_i$ for $i \in \{1, 2, \ldots, c\}$ can be found in $t(\ell, n)$ time since $C$ is a hereditary graph class and, thus, $G_i \in C$.

It is obvious that $G[S']$ has at most $\ell$ vertices and is $c$-colorable since it is the union of $c$ independent sets. Thus, $w(S') \leq w(S)$, where $G[S]$ is a maximum-weight $c$-colorable subgraph with at most $\ell$ vertices in $G$.

It remains to show $w(S') \geq w(S)$. Let col: $V \to \mathbb{N}$ be a proper $c$-coloring of $G[S]$. Since $|S| \leq \ell$, there is a function $f \in F$ such that col($v_k$) = $f(k)$ for each $v_k \in S$. For each $i \in \{1, 2, \ldots, c\}$, let $S_i := \{v_k \in S \mid f(k) = i\}$ and $\ell_i := |S_i|$. Since $G[S]$ is a maximum-weight $c$-colorable subgraph, for each $i \in \{1, 2, \ldots, c\}$, $S_i$ is a maximum-weight independent set with at most $\ell_i$ vertices in the subgraph $G_i$ induced by the vertices $v_k$ with $f(k) = i$. Since the algorithm iterated over this $f$ and these $\ell_1 + \ell_2 + \cdots + \ell_c \leq \ell$, we get that $w(S') \geq w(S)$. 

Springer
We point out that, for an easier and faster implementation of the algorithm in the proof of Lemma 5.3, it makes sense to trade the factor \( t^{(O(\log \ell))} \log^2 n \) in the running time for a small error probability \( \varepsilon \) by simply choosing \( f(k) \in \{1, \ldots, c\} \) uniformly at random for each vertex \( v_k \) of \( G \) instead of using universal sets. With probability at least \( c^{-\ell} \), \( f \) will be a proper coloring for a maximum-weight solution and the algorithm will thus find it. Repeating the algorithm with \( O(\ell^c \ln 1/\varepsilon) \) random assignments \( f \) thus allows for finding a maximum-weight solution with error probability at most \( \varepsilon > 0 \).

5.2 From Max-Weight Independent Set to Max-Weight Colorful Independent Set

In order to show Theorem 5.1, by Lemma 5.3, it is enough to show that Max-Weight Independent Set parameterized by the solution size \( \ell \) is fixed-parameter tractable in cluster graphs. We do this using an algorithm for the following auxiliary problem.

Problem 5.4 (Max-Weight Colorful Independent Set)

Input: A graph \( G = (V, E) \) with vertex weights \( w: V \rightarrow \mathbb{N} \) and a coloring \( \text{col}: V \rightarrow \{1, 2, \ldots, c\} \) of its vertices.

Task: Find a maximum-weight independent set whose vertices have pairwise distinct colors.

Colorful independent sets of interval graphs have been studied by van Bevern et al. (2015) in order to show that Maximum Independent Set parameterized by the solution size \( \ell \) is fixed-parameter tractable on cluster graphs, thus strengthening a result on the Job Interval Selection problem of Halldórsson and Karlsson (2006).

In Sect. 5.3, we will study Max-Weight Colorful Independent Set on chordal graphs. Together with the following lemma, this will give a fixed-parameter algorithm for Max-Weight Independent Set on cluster graphs.

Lemma 5.5 Let \( C \) be a graph class and \( t: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N} \) be a function such that a maximum-weight colorable independent set in graphs in \( C \) is computable in \( t(c, n) \) time.

Then, a maximum-weight independent set of size at most \( \ell \) in cluster graphs is computable in \( e^{\ell} t^{O(\log \ell)} \log n \cdot t(\ell, n) \) time, assuming that the decomposition of the input graph into a cluster graph and a graph of \( C \) is given.

Proof Let \( G = (V, E_1 \cup E_2) \) such that \( G_1 = (V, E_1) \) is a cluster graph and \( G_2 = (V, E_2) \in C \). Let \( d \) be the number of clusters in \( G_1 \). Our algorithm uses a \((d, \ell)\)-perfect hash family \( \mathcal{F} \)—a family of functions \( f: \{1, 2, \ldots, d\} \rightarrow \{1, 2, \ldots, \ell\} \) such that, for each subset \( C \subseteq \{1, 2, \ldots, d\} \) of size \( \ell \), at least one of the functions in \( \mathcal{F} \) is a bijection between \( C \) and \( \{1, 2, \ldots, \ell\} \).

We find a maximum-weight independent set with at most \( \ell \) vertices in \( G \) as follows. First, in \( e^{\ell} t^{O(\log \ell)} d \log d \) time, construct a \((d, \ell)\)-perfect hash family \( \mathcal{F} \) with \( |\mathcal{F}| \in e^{\ell} t^{O(\log \ell)} \log d \) (Cygan et al. 2015, Definition 5.15–Theorem 5.18). Then, iterate over all functions \( f \in \mathcal{F} \). In each iteration, consider the coloring \( \text{col}_f \) that colors the vertices in each cluster \( i \in \{1, 2, \ldots, d\} \) of \( G_1 \) using color \( f(i) \in \{1, 2, \ldots, \ell\} \) and compute a maximum-weight colorable independent set in \( G_2 \) with respect to \( \text{col}_f \) in \( t(\ell, n) \) time. Return the colorful independent set \( S' \) of maximum weight found in any iteration.

We show that \( S' \) is an independent set for \( G \) with \( w(S') = w(S) \), where \( S \) is a maximum-weight independent set of size at most \( \ell \) in \( G \).

“\( \leq \)”: Observe that \( S' \), being an independent set for \( G_2 \), contains no edges of \( G_2 \). Moreover, being colorable for some coloring \( \text{col}_f \), \( S' \) neither contains edges of \( G_1 \). The endpoints of such an edge belong to the same cluster \( i \) and thus get the same color \( \text{col}_f(i) \). Thus, \( S' \) is an independent set of size at most \( \ell \) for \( G \) and, therefore, \( w(S') \leq w(S) \).

“\( \geq \)”: Since \( S \) uses at most one vertex of each cluster of \( G_1 \), the set \( S \) of clusters that contain vertices of \( S \) has size at most \( \ell \). Thus, \( \mathcal{F} \) contains a bijection \( f: C_S \rightarrow \{1, 2, \ldots, |C_S|\} \). Note that \( S \) is a colorful independent set for \( G_2 \) with respect to \( \text{col}_f \). Thus, \( w(S') \geq w(S) \).

As in the case with Lemma 5.3, for an easier and faster implementation of the algorithm in the proof of Lemma 5.5, it makes sense to trade the factor \( t^{O(\log \ell)} \log d \) in the running time for a small error probability by randomly coloring the clusters instead of using perfect hash families: The vertices of maximum-weight solution will have pairwise distinct colors with probability at least \( \ell!/\ell^\ell \Theta(1) \). Repeating the algorithm with \( O(\ell^c \ln 1/\varepsilon) \) random colorings will thus find a maximum-weight solution with error probability at most \( \varepsilon > 0 \).

5.3 Max-Weight Colorful Independent Set in Chordal Graphs

To complete our proof of Theorem 5.1, we finally show the following proposition, which, together with Lemmas 5.3 and 5.5, yields Theorem 5.1.

Proposition 5.6 Max-Weight Colorful Independent Set in Chordal Graphs is Solvable in \( O(3^c n^2) \) time.

Proposition 5.6 generalizes a result of Halldórsson and Karlsson (2006), who showed that Max-Weight Colorful Independent Set parameterized by the number \( c \) of colors is fixed-parameter tractable in interval graphs. Whereas the algorithm of Halldórsson and Karlsson (2006) is based on dynamic programming over the interval representation of interval graphs, we will exploit dynamic programming over tree decompositions of chordal graphs.
**Definition 5.7** (Tree decomposition) A tree decomposition of a graph $G = (V, E)$ is a pair $(\mathcal{X}, T)$, where $\mathcal{X} \subseteq 2^V$ and $T$ is a tree on $\mathcal{X}$, such that

(i) $\bigcup_{X \in \mathcal{X}} X = V$, the sets $X \in \mathcal{X}$ are called bags,
(ii) for each edge $e \in E$, there is a bag $X \in \mathcal{X}$ with $e \subseteq X$,
(iii) for any three bags $X, Y, Z \in \mathcal{X}$ such that $Y$ is on the path between $X$ and $Z$ in $T$, one has $X \cap Z \subseteq Y$.

Chordal graphs have tree decompositions in which each bag induces a clique and such tree decomposition can be computed in linear time (Blair and Peyton 1993).

We will additionally assume that the tree decomposition is rooted at an arbitrary bag and that each bag $X$ with more than one child in $T$ has exactly two children $Y$ and $Z$ in $T$ such that $X = Y = Z$. This property can be established step by step as follows: if some bag $X$ has more than two children or its children are not all equal to $X$, then we can attach two copies $X_1$ and $X_2$ of $X$ to $X$ as children in $T$ and attach one original child of $X$ to $X_1$ and the other children of $X$ to $X_2$.

This procedure, in the worst case, triples the number of bags in the tree decomposition and works in linear time.

Using this linear-time computable tree decomposition and the following lemma with $\alpha = 1$ yields Proposition 5.6, which, together with Lemmas 5.3 and 5.5, concludes the proof of Theorem 5.1.

**Lemma 5.8** Given a tree decomposition, each bag of which has at most $\alpha$ pairwise nonadjacent vertices, MAX-WEIGHT COLORFUL INDEPENDENT SET can be solved in $O(3^\alpha \cdot n^{\alpha+1})$ time.

**Proof** Let $G = (V, E)$ be a graph and let $(\mathcal{X}, T)$ be a tree decomposition such that each bag of $\mathcal{X}$ contains at most $\alpha$ pairwise nonadjacent vertices. Root $T$ at an arbitrary bag $R \in \mathcal{X}$. For each bag $X \in \mathcal{X}$, let $V_X \subseteq V$ denote the set of vertices in $X$ and its descendant bags in $T$. Moreover, let $G_X := G[V_X]$ and $\operatorname{col}(S) := \{\operatorname{col}(v) \mid v \in S\}$.

For each bag $X \in \mathcal{X}$, each $S \subseteq X$, each subset $C \subseteq \{1, 2, \ldots, c\}$ of colors, let $T[X, C, S]$ denote the maximum weight of an independent set $I$ of $G_X$ with $I \cap X = S$ whose vertices have pairwise distinct colors from $C$. Then, by Definition 5.7(i), the solution to MAX-WEIGHT COLORFUL INDEPENDENT SET is \[ \max_{S \subseteq R} T[R, \{1, 2, \ldots, c\}, S]. \]

5.3.1 Infeasible solutions

Since no bag has an independent set with more than $\alpha$ vertices, for each $X \in \mathcal{X}$, each $S \subseteq X$, and each $C \subseteq \{1, 2, \ldots, c\}$, we have $T[X, C, S] = -\infty$ if $|S| > \alpha$, or if $S$ is not independent, or if the vertices of $S$ do not have pairwise distinct colors from $C$.

In the following, we will thus assume that $S \subseteq X$ is always an independent set and that its vertices have pairwise distinct colors from $C \subseteq \{1, 2, \ldots, c\}$.

5.3.2 Bags without children

For each bag $X \in \mathcal{X}$ without children,

\[ T[X, C, S] = w(S). \]

5.3.3 Bags with a single child

Consider a bag $X \in \mathcal{X}$ that has one child $Y$. We show that

\[ T[X, C, S] = \max \left\{ T[Y, C \setminus \operatorname{col}(S \setminus S'), S'] + w(S \setminus S') \right\} \]

for each independent set $S' \subseteq Y$ such that $S \cap X \cap Y = S' \cap X \cap Y$.

Herein, the condition $S \cap X \cap Y = S' \cap X \cap Y$ ensure that independent sets for $G_X$ and $G_Y$ agree on the vertices that are both in the bag $X$ and its child $Y$. Next, we show validity of (5.2).

“$\preceq$”: Consider a maximum-weight independent set $I$ for $G_X$ with $I \cap X = S$ whose vertices have pairwise distinct colors from $C$. Then, $I' := I \cap V_Y$ is an independent set for $G_Y$. For $S' := I' \cap Y$, we have that $S' \cap X \cap Y = I' \cap X \cap Y = I \cap X \cap Y = S \cap X \cap Y$. Since $I \cap V_Y \subseteq X$ and, by Definition 5.7(iii), $I' \cap X \subseteq Y$, one has $I \cap I' = (I \cap X) \setminus (I' \cap X) = (I \cap X) \setminus (I' \cap X \cap Y) = (I \cap X \cap Y) \setminus (I' \cap Y) = S \setminus S'$. Thus, the vertices of $I'$ have pairwise distinct colors from $C' := C \setminus \operatorname{col}(S \setminus S')$ and

\[ T[X, C, S] = w(I) = w(I') + w(S \setminus S') \]

\[ \leq T[Y, C \setminus \operatorname{col}(S \setminus S'), S'] + w(S \setminus S'). \]

“$\succeq$”: Consider any independent set $S' \subseteq Y$ with $S \cap X \cap Y = S' \cap X \cap Y$. Let $I'$ be a maximum-weight independent set for $G_Y$ with $I' \cap Y = S'$ whose vertices use pairwise distinct colors from $C' := C \setminus \operatorname{col}(S \setminus S')$.

Then, $I := I' \cup S = I' \cup (S \setminus S')$ uses pairwise distinct colors in $C$. Moreover, by Definition 5.7(iii), $I' \cap X \subseteq V_Y \cap X \subseteq Y$ and, therefore, $I \cap X = (I' \cup S) \cap X = (I' \cap X) \cup (S \setminus S') = (S' \cap X \cap Y) \cup S = (S' \cap X \cap Y) \cup S = (S \cap X \cap Y) \cup S = S \cap X \cap Y \cup S \subseteq S$.

We show that $I$ is independent. Toward a contradiction, assume that there is an edge $e = \{x, y\} \subseteq I \cap I' \cup S$. Since $e \not\subseteq I'$ and $e \not\subseteq S$, we have $x \in S \setminus I' \subseteq X$ and $y \in I' \setminus S \subseteq V_Y$. Since $x \not\in I' \supseteq S' \cap X \cap Y = S \cap X \cap Y$ but $x \in S = I \cap X$, we get $x \not\in Y$. Consequently, $x \not\in V_Y$ by Definition 5.7(iii) and $\{x, y\} \subseteq X$ by Definition 5.7(ii).

Thus, we showed that $\{x, y\} \subseteq X \cap I = S$, a contradiction.

\[ \square \]
Finally since \( I' \cap X \subseteq I' \cap Y = S' \), sets \( I' \) and \( S \setminus S' \) are disjoint and we get
\[
T[X, C, S] \geq w(I) = w(I') + w(S) = T[Y, C', S'] + w(S \setminus S').
\]

### 5.3.4 Bags with two children

For each bag \( X \in \mathcal{X} \) with two children \( Y = Z = X \), we show that
\[
T[X, C, S] = \max \left\{ \begin{array}{l}
T[Y, C_Y, S] + T[Z, C_Z, S] - w(S) \\
\text{such that } C_Y \cup C_Z = C \\
\text{and } C_Y \cap C_Z = \text{col}(S) \end{array} \right\}.
\]

\( \leq \): Let \( I \) be a maximum-weight independent set for \( G_X \) with \( X \cap I = S \) whose vertices use pairwise distinct colors from \( C \). Then, \( I_Y := I \cap Y \) and \( I_Z := I \cap Z \) are independent sets for \( G_Y \) and \( G_Z \), respectively. One has \( Y \cap I_Y = Y \cap I \cap V_Y = Y \cap I = X \cap I \) and \( S \) is and, likewise, \( Z \cap I_Z = S \).

By Definition 5.7(iii), one has \( I_Y \cap I_Z \subseteq X \) and thus \( I_Y \cap I_Z = I_Y \cap I_Z \cap X = S \). Since \( I \) contains only one vertex of each color from \( C \), we also get \( \text{col}(I_Y) \cap \text{col}(I_Z) = \text{col}(S) \). Thus, the vertices of \( I_Y \) have pairwise distinct colors in \( C_Y := \text{col}(I_Y) \) and the vertices of \( I_Z \) have pairwise distinct colors in \( C_Z := (C \setminus C_Y) \cup \text{col}(S) \). One has \( C_Y \cup C_Z = C \) and \( C_Y \cap C_Z = \text{col}(S) \). Thus,
\[
T[X, C, S] = w(I) = w(I_Y) + w(I_Z) - w(I \cap I_Z) \\
= w(I_Y) + w(I_Z) - w(S) \\
\leq T[Y, C_Y, S] + T[Z, C_Z, S] - w(S).
\]

\( \geq \): Let \( C_Y \) and \( C_Z \) be color sets such that \( C_Y \cap C_Z = \text{col}(S) \) and \( C_Y \cup C_Z = C \). Let \( I_Y \) and \( I_Z \) be maximum-weight independent sets for \( G_Y \) and \( G_Z \) with \( I_Y \cap Y = I_Z \cap Z = S \) and whose vertices use pairwise distinct colors from \( C_Y \) and \( C_Z \), respectively. Then, for \( I := I_Y \cup I_Z \), one has \( I \cap X = (I_Y \cup I_Z) \cap X = (I_Y \cap X) \cup (I_Z \cap X) = S \).

We show that \( I \) is an independent set. Assume toward a contradiction that there is an edge \( e \subseteq I = I_Y \cup I_Z \). Since \( e \notin I_Y \) and \( e \notin I_Z \), we have \( y \in I_Y \) and \( z \in I_Z \). Definition 5.7(iii) gives \( V_Y \cap V_Z \subseteq X \). Thus, by Definition 5.7(ii), \( e \notin V_Z \) or \( e \notin V_Y \). By symmetry, assume that \( e \notin V_Z \). Then, \( y \in V_Y \cap V_Z \subseteq X \). Since \( y \in X \cap I_Y = X \cap I \), it follows that \( \{y, z\} \subseteq I_Z \), a contradiction to \( I_Z \) being an independent set.

We show that the colors of the vertices in \( I \) are pairwise distinct. Let there be two vertices \( \{u, v\} \subseteq I \) with \( \text{col}(u) = \text{col}(v) = c^* \). Then, \( c^* \in \text{col}(S) \) since \( C_Y \cap C_Z = \text{col}(S) \).

Thus, there is a vertex \( w \in S \subseteq I_Y \cap I_Z \) with \( \text{col}(w) = c^* \). Since \( I_Y \) and \( I_Z \) contain only one vertex of each color, it holds that \( w = u = v \). Hence \( I \) only contains one vertex with color \( c^* \). Finally, Definition 5.7(iii) yields \( I_Y \cap I_Z \subseteq X \).

Thus,
\[
T[X, C, S] \geq w(I) = w(I_Y) + w(I_Z) - w(I \cap I_Z) \\
= w(I_Y) + w(I_Z) - w(I \cap I_Z \cap X) \\
= w(I_Y) + w(I_Z) - w(S) \\
= T[Y, C_Y, S] + T[Z, C_Z, S] - w(S).
\]

### 5.3.5 The algorithm and its running time

There are \( O(n) \) bags. The algorithm first computes (5.1) for all leaf bags. Then, for each bag whose children have been processed, it computes (5.2) and (5.3).

For each leaf bag \( X \), we compute (5.1) as follows. First, for each of the at most \( n^\alpha \) subsets \( S \subseteq X \) with \( |S| \leq \alpha \), we once compute \( w(S) \) and check the independence of \( S \) in \( O(\alpha^2) \) time. We then use this information in the computation for all subsets \( C \subseteq \{1, 2, \ldots, c\} \). Thus, all leaf nodes are processed in \( O(n \cdot (2^c + \alpha^2 \cdot n^\alpha)) \) time.

For each bag \( X \) with two children, we compute (5.3) as follows. First, for each of the at most \( n^\alpha \) subsets \( S \subseteq X \) with \( |S| \leq \alpha \), we once compute \( w(S) \) and check the independence of \( S \) in \( O(\alpha^2) \) time. We then use this information in the computation for all subsets \( C \subseteq \{1, 2, \ldots, c\} \). To this end, iterate over all subsets \( C_Y \subseteq C \), whereby \( C_Z \) can be computed from \( C_Y \) in \( O(c) \) time. Note that, in total, one iterates over at most \( 3^c \) subsets \( C_Y \subseteq C \subseteq \{1, 2, \ldots, c\} \). Each color is either not in \( C \) or in \( C \) but not in \( C_Y \), or in \( C_Y \). Thus, bags with two children can be processed in total time \( O(n \cdot (3^c c + \alpha^2 \cdot n^\alpha)) \).

For each bag \( X \) with a single child \( Y \), we compute (5.2) as follows. First, for each of the at most \( n^\alpha \) subsets \( S' \subseteq Y \) with \( |S'| \leq \alpha \), we once compute the intersection \( S' \cap X \) and check the independence of \( S' \cap X \) in \( O(\alpha^2) \) time. During this computation, for each encountered intersection \( S^* \) and color set \( C \subseteq \{1, 2, \ldots, c\} \), remember the maximum value \( T_{S}[Y, C] \) taken by \( T[Y, C, S'] \) for any \( S' \subseteq Y \) with \( S^* = S' \cap X \). This works in \( O(2^c \cdot \alpha) \) time. We then iterate over all subsets \( C \subseteq \{1, 2, \ldots, c\} \), each of the encountered intersections \( S^* \subseteq X \cap Y \), and each independent set \( S \) with \( S^* \subseteq S \subseteq X \) and compute \( T[Y, C, \text{col}(S \setminus S')] \) as the maximum value \( T_{S}[C, \text{col}(S \setminus S')] \) encountered for any intersection \( S^* \) that yielded \( S \). For each \( \alpha' \leq \alpha \) and each of \( n^\alpha \) intersections \( S^* \) of size exactly \( \alpha' \), we enumerate at most \( n^{\alpha'-\alpha} \) independent sets \( S \) such that \( S^* \subseteq S \subseteq X \). Thus, bags with a single child can be processed in \( O(n \cdot (2^c + \alpha' \cdot n^\alpha)) \) total time.

Altogether, the overall algorithm runs in \( O(3^c n^{\alpha+1}) \) time.

\[ \square \]
6 Parameterized complexity on other graph classes

In Sect. 4, we have shown that MAXIMUM INDEPENDENT SET is W[1]-hard parameterized by the solution size even on 2-simplicial 3-minoes. In contrast, in Sect. 5, we have shown that MAX-WEIGHT C-COLORABLE SUBGRAPH is fixed-parameter tractable parameterized by the solution size on cluster $\gg\gg$ chordal graphs.

In this section, we survey the parameterized complexity of MAX-WEIGHT C-COLORABLE SUBGRAPH on the neighboring graph classes discussed in Sect. 3, thus completing the computational complexity picture given in Fig. 1. Most results in this section are known or easy to obtain. Yet since they are scattered throughout the literature using terminology from this section are known or easy to obtain. Yet for cluster $\approx\approx$ interval graphs, one can give a slightly better running time.

6.1 $K_{1,3}$-free graphs

Section 4 has shown that MAXIMUM INDEPENDENT SET is W[1]-hard parameterized by the solution size on 3-minoes, which are a proper subclass of $K_{1,4}$-free graphs. This is complemented as follows.

Proposition 6.1 On $K_{1,3}$-free graphs,

(i) MAXIMUM $c$-COLORABLE SUBGRAPH is NP-hard for each fixed $c \geq 3$ and

(ii) a maximum-weight $c$-colorable subgraph with at most $\ell$ vertices is computable in $2^{\ell+c} \cdot c^\ell \cdot \mathcal{O}(\log \ell) \cdot n^{3} \log^{2} n$ time.

Proof Proposition 6.1(i) follows from the fact that checking 3-colorability of line graphs is NP-complete, where line graphs are $K_{1,3}$-free.

Proposition 6.1(ii) follows directly from Lemma 5.3 using the fact that $K_{1,3}$-free graphs are a hereditary graph class and that MAX-WIGHT INDEPENDENT SET is solvable in $O(n^{3})$ time on $K_{1,3}$-free graphs (Faenza et al. 2011).

Note that $c \leq \ell$ holds in all nontrivial cases. Thus, Proposition 6.1 shows that MAX-WEIGHT $c$-COLORABLE SUBGRAPH is fixed-parameter tractable parameterized by $\ell$ on $K_{1,3}$-free graphs. The complexity for the case $c = 2$ seems to be open.

6.2 Cluster $\gg\gg$ interval graphs

These graphs form a subclass of cluster $\gg\gg$ chordal graphs and therefore Theorem 5.1 applies to them. Yet for cluster $\approx\approx$ interval graphs, one can give a slightly better running time.

Proposition 6.2 On cluster $\gg\gg$ interval graphs,

(i) MAX-WEIGHT INDEPENDENT SET is NP-hard and

(ii) a maximum-weight $c$-colorable subgraph with at most $\ell$ vertices is computable in $2^{\ell+c} \cdot (c + e + 2)^\ell \cdot \mathcal{O}(\log \ell) \cdot (n + m) \cdot \log^{3} n$ time if the decomposition of the input graph into a cluster graph and an interval graph is given.

Herein, $e$ is Euler’s number.

Proposition 6.2(i) is due to Halldórsson and Karlsson (2006). Toward proving Proposition 6.2(ii), van Bevern et al. (2015) noted that Halldórsson and Karlsson (2006) showed that a maximum-weight independent set of vertices of pairwise distinct colors in an interval graph whose vertices are colored in $c$ colors is computable in $O(2^{c} \cdot n)$ time if an interval representation with sorted intervals is given, which can be computed in $O(n + m)$ time. Since cluster $\gg\gg$ interval graphs are a hereditary graph class, Proposition 6.2(ii) follows from this result of Halldórsson and Karlsson (2006) and Lemmas 5.3 and 5.5.

6.3 Chordal graphs

Theorem 4.1 shows that MAXIMUM INDEPENDENT SET is W[1]-hard parameterized by the solution size on inductive 2-independent graphs. Chordal graphs are exactly the inductive 1-independent graphs (Blair and Peyton 1993; Ye and Borodin 2012).

Proposition 6.3 In chordal graphs, MAX-WEIGHT $c$-COLORABLE SUBGRAPH is

(i) NP-hard,

(ii) polynomial-time solvable for each fixed $c$ yet W[2]-hard parameterized by $c$, and

(iii) a maximum-weight $c$-colorable subgraph with at most $\ell$ vertices is computable in $2^{\ell+c} \cdot c^\ell \cdot \mathcal{O}(\log \ell) \cdot (n + m) \cdot \log^{3} n$ time.

Proposition 6.3(i) and (ii) were shown by Yannakakis and Gavril (1987). Proposition 6.3(iii) follows from the fact that MAX-WIGHT INDEPENDENT SET is linear-time solvable in chordal graphs (Frank 1975) and using Lemma 5.3. This was shown for the unweighted variant by Misra et al. (2013), and our Lemma 5.3 is, essentially, just a slightly adapted weighted variant of this result.

6.4 Interval graphs

Yannakakis and Gavril (1987) noted that MAX-WEIGHT $c$-COLORABLE SUBGRAPH in interval graphs is solvable in polynomial time by modeling it as a totally unimodular linear program.

Using a flow formulation, Arkin and Silverberg (1987) proved that the equivalent problem of scheduling a maximum-weight subset of intervals on $c$ parallel identical machines...
such that the intervals on each machine are pairwise non-intersecting is solvable in \(O(n^2 \log n)\) time. Since the interval representation of an interval graph is computable in the same time (Corneil et al. 2009), one gets the following result.

**Proposition 6.4**: MAX-WEIGHT \(c\)-COLORABLE SUBGRAPH is solvable in \(O(n^2 \log n)\) time on interval graphs.

### 6.5 Inductive \(k\)-independent graphs

Of course, our hardness result of Theorem 4.1 also holds for MAXIMUM INDEPENDENT SET in inductive \(k\)-independent graphs.

On the positive side, Ye and Borodin (2012) showed that MAX-WEIGHT INDEPENDENT SET is polynomial-time \(k\)-approximable in inductive \(k\)-independent graphs. The algorithm requires the \(k\)-independence ordering of Definition 2.3, which cannot be computed efficiently (cf. Proposition 3.1), yet sometimes is given by the application data (Ásgeirsson et al. 2017; Halldórsson and Tonoyan 2015).

### 7 Conclusion

Motivated by recent work on wireless scheduling (Ásgeirsson et al. 2017; Halldórsson and Tonoyan 2015), we performed an extensive analysis of problems related to independent sets on graphs that lie between interval graphs on the “bottom end” and inductive 2-independent graphs on the “top end”. Some of our (computational hardness) results might be discouraging given that, in real-world scheduling applications, one often has to expect inductive \(k\)-independent graphs with constant \(k > 2\). This negative impression is alleviated by the following observations.

First, there are several tractability results for classic scheduling problems that can be modeled as independent set problems at this fairly low level in the graph classes hierarchy.

Second, our parameterized complexity studies are basically focused on the parameter solution size. In the spirit of multivariate complexity analysis (Fellows et al. 2013; Komusiewicz and Niedermeier 2012; Niedermeier 2010), this encourages the studies of further and also “combined” parameterizations. Indeed, with real-world data at hand, one might first measure numerous parameter values in the instance (say characteristics such as maximum degree, treewidth, degree distribution, feedback edge and vertex numbers) that, if small, altogether might be exploited to get strong fixed-parameter tractability results.

Third, we point out that combining approximation algorithms [there are several results in the literature including Ásgeirsson et al. (2017); Halldórsson and Tonoyan (2015); and Ye and Borodin (2012)] with problem-specific parameterizations might lead to parameterized approximation algorithms with improved approximation factors and/or running times in relevant application scenarios; see van Bevern et al. (2017) for a concrete example.

We remark that our main positive result (fixed-parameter tractability of finding maximum-weight \(c\)-colorable subgraphs on cluster \(\gg\) chordal graphs, Theorem 5.1) indeed also holds for the so far seemingly neglected class of graphs with a given tree decomposition where the independent sets in each bag have size at most \(\alpha\): It can be solved in \(O(3^\alpha \cdot n^{\alpha+1})\) time (Lemma 5.8). We feel that this class of graphs might be of independent interest.

Finally, Sect. 3 with its classification of graph classes between interval graphs and inductive 3-independent graphs might prove useful even for (graph-algorithmic) studies not touching scheduling and independent set problems. On a different note, we advocate that parameterized complexity analysis may prove useful for further studies on hard scheduling problems—so far the number of studies in this direction is fairly small and leaves numerous challenges for future work (Mnich and van Bevern 2018).

**Acknowledgements** We are grateful to Andreas Krebs (Tübingen) for fruitful discussions concerning parts of this work. We thank the anonymous referees of *Journal of Scheduling* for constructive feedback.

---

**References**

Akcoglu, K., Aspnes, J., DasGupta, B., & Kao, M. Y. (2002). Opportunity-cost algorithms for combinatorial auctions. In *Applied optimization 74*: *Computational methods in decision-making, economics and finance* (pp. 455–479). Dordrecht: Kluwer. https://doi.org/10.1007/978-1-4757-3613-7_23

Alon, N., Yuster, R., & Zwick, U. (1995). Color-coding. *Journal of the ACM*, 42(4), 844–856. https://doi.org/10.1145/210332.210337.

Arkin, E. M., & Silverberg, E. B. (1987). Scheduling jobs with fixed start and end times. *Discrete Applied Mathematics*, 18(1), 1–8. https://doi.org/10.1016/0166-218X(87)90037-0.

Ásgeirsson, E. I., Halldórsson, M. M., & Tonoyan, T. (2017). Universal framework for wireless scheduling problems. In *44th international colloquium on automata, languages, and programming*, ICALP, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (pp 129:1–129:15). https://doi.org/10.4230/LIPIcs.ICALP.2017.129.

Baker, B. S., & Coffman, E. G. (1996). Mutual exclusion scheduling. *Theoretical Computer Science*, 162(2), 225–243. https://doi.org/10.1016/0304-3975(96)00031-X.

Bansal, N., Blum, A., & Chawla, S. (2004). Correlation clustering. *Machine Learning*, 56(1), 89–113. https://doi.org/10.1023/B:MACH.0000033116.37574.95.

van Bevern, R., Komusiewicz, C., & Sorge, M. (2017). A parameterized approximation algorithm for the mixed and windy capacitated arc routing problem: Theory and experiments. *Networks*, 70(3), 262–278. https://doi.org/10.1002/net.21742 (WARP 2 special issue).

van Bevern, R., Mnich, M., Niedermeier, R., & Weller, M. (2015). Interval scheduling and colorful independent sets. *Journal of Scheduling*, 18, 449–469. https://doi.org/10.1007/s10951-014-0398-5.
Birand, B., Chudnovsky, M., Ries, B., Seymour, P., Zassman, G., & Zwols, Y. (2012). Analyzing the performance of greedy maximal scheduling via local pooling and graph theory. *IEEE/ACM Transactions on Networks*, 20(1), 163–176. https://doi.org/10.1109/TNET.2011.2157831.

Blair, J. R. S., & Peyton, B. (1993). An introduction to chordal graphs and clique trees. In *Graph theory and sparse matrix computation*, *IMA volumes in mathematics and its applications* (vol. 56, pp. 1–29). Berlin: Springer. https://doi.org/10.1007/978-1-4613-4869-1_7.

Booth, K. S., & Lueker, G. S. (1976). Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3), 335–379. https://doi.org/10.1016/0022-0000(76)90045-1.

Birand, B., Chudnovsky, M., Ries, B., Seymour, P., Zassman, G., & Gyárfás, A., & West, D. (1995). Multitrack interval graphs. *Congressus Numerantium*, 100, 1–7. Available on arXiv:1712.00658.

Booth, K. S., & Lueker, G. S. (1976). Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3), 335–379. https://doi.org/10.1016/0022-0000(76)90045-1.

Downey, R. G., & Fellows, M. R. (2013). *Fundamentals of parameterized complexity*. Berlin: Springer. https://doi.org/10.1007/978-3-642-32589-7_1.

Fellows, M. R., Jansen, B. M. P., & Rosamond, F. A. (2013). Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity. *European Journal of Combinatorics*, 34(3), 541–566. https://doi.org/10.1016/j.ejc.2012.04.008.

Kammer, F., Tholey, T., & Voepel, H. (2010). Approximation algorithms for intersection graphs. In *Proceedings of the 13th international workshop on approximation algorithms for combinatorial optimization problems and 14th workshop on randomized techniques in computation: Algorithms and techniques (APPROX'10 and RANDOM'10), Lecture notes in computer science* (vol. 6302, pp. 260–273). Berlin: Springer. https://doi.org/10.1007/978-3-642-15369-3_20.

Kolm, A. W. J., Lenstra, J. K., Papadimitriou, C. H., & Spieksma, F. C. R. (2007). Interval scheduling: A survey. *Naval Research Logistics*, 54(5), 530–543. https://doi.org/10.1002/nav.20231.

Misra, C., & Niedermeier, R. (2012). New races in parameterized algorithmics. In *Proceedings of the 37th international symposium on mathematical foundations of computer science (MFCS'12), Lecture notes in computer science* (vol. 7464, pp. 19–30). Berlin: Springer. https://doi.org/10.1007/978-3-642-32589-2_2.

Köse, A., Evirgen, N., Gökcetu, H., Gökcetu, K., & Médard, M. (2017). Nearly optimal scheduling of wireless ad hoc networks in polynomial time. Available on arXiv:1712.00658.

Köse, A., & Médard, M. (2017). Scheduling wireless ad hoc networks in polynomial time using claw-free conflict graphs. In *Proceedings of the 28th IEEE annual international symposium on personal, indoor, and mobile radio communications (PIMRC'17)* (pp. 1–7). New York: IEEE. https://doi.org/10.1109/PIMRC.2017.8292404.

Misra, N., Panolan, F., Rai, A., Raman, V., & Saurabh, S. (2013). Parameterized algorithms for max colorable induced subgraph problem on perfect graphs. In *Proceedings of the 39th international workshop on graph-theoretic concepts in computer science (WG'13)* (pp. 370–381). Berlin: Springer. https://doi.org/10.1007/978-3-642-45043-3_32.

Mnich, M., & van Bevern, R. (2018). Parameterized complexity of machine scheduling: 15 open problems. *Computers and Operations Research*, 100, 254–261. https://doi.org/10.1016/j.cor.2018.07.020.

Nakajima, K., & Jakim, S. L. (1982). Complexity results for scheduling tasks with discrete starting times. *Journal of Algorithms*, 3(4), 344–361. https://doi.org/10.1016/0196-6774(82)90030-X.

Niedermeier, R. (2006). *Invitation to fixed-parameter algorithms*. Oxford: Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198566076.001.0001.

Niedermeier, R. (2010). Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th international symposium on theoretical aspects of computer science* (pp. 197–218). Berlin: Springer.
Papadimitriou, C., & Yannakakis, M. (1979). Scheduling interval-ordered tasks. *SIAM Journal on Computing*, 8(3), 405–409. https://doi.org/10.1137/0208031.

Rose, D. J., Tarjan, R. E., & Lueker, G. S. (1976). Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2), 266–283. https://doi.org/10.1137/0205021.

Shamir, R., Sharan, R., & Tsur, D. (2004). Cluster graph modification problems. *Discrete Applied Mathematics, 144*(1–2), 173–182. https://doi.org/10.1016/j.dam.2004.01.007.

Sung, S. C., & Vlach, M. (2005). Maximizing weighted number of just-in-time jobs on unrelated parallel machines. *Journal of Scheduling*, 8(5), 453–460. https://doi.org/10.1007/s10951-005-2863-7.

Wegner, G. (1961). *Eigenschaften der Nerven homologisch-einfacher Familien im R^n*. Ph.D. Thesis, Universität Göttingen.

West, D. B., & Shmoys, D. B. (1984). Recognizing graphs with fixed interval number is NP-complete. *Discrete Applied Mathematics*, 8(3), 295–305. https://doi.org/10.1016/0166-218X(84)90127-6.

Yannakakis, M., & Gavril, F. (1987). The maximum k-colorable subgraph problem for chordal graphs. *Information Processing Letters*, 24(2), 133–137. https://doi.org/10.1016/0020-0190(87)90107-4.

Ye, Y., & Borodin, A. (2012). Elimination graphs. *ACM Transactions on Algorithms*, 8(2), 14:1–14:23. https://doi.org/10.1145/2151171.2151177.

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