



Twin-width III: Max Independent Set, Min Dominating Set, and Coloring

Édouard Bonnet   

Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Colin Geniet 



University of Warsaw, Poland

Eun Jung Kim  

Université Paris-Dauphine, PSL University, CNRS UMR7243, LAMSADE, Paris, France

Stéphan Thomassé 

Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Rémi Watrigant  

Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Abstract

We recently introduced the notion of twin-width, a novel graph invariant, and showed that first-order model checking can be solved in time $f(d, k)n$ for n -vertex graphs given with a witness that the twin-width is at most d , called d -contraction sequence or d -sequence, and formulas of size k [Bonnet et al., FOCS '20]. The inevitable price to pay for such a general result is that f is a tower of exponentials of height roughly k . In this paper, we show that algorithms based on twin-width need not be impractical. We present $2^{O(k)}n$ -time algorithms for k -INDEPENDENT SET, r -SCATTERED SET, k -CLIQUE, and k -DOMINATING SET when an $O(1)$ -sequence of the graph is given in input. We further show how to solve the weighted version of k -INDEPENDENT SET, SUBGRAPH ISOMORPHISM, and INDUCED SUBGRAPH ISOMORPHISM, in the slightly worse running time $2^{O(k \log k)}n$. Up to logarithmic factors in the exponent, all these running times are optimal, unless the Exponential Time Hypothesis fails. Like our FO model checking algorithm, these new algorithms are based on a dynamic programming scheme following the sequence of contractions forward.

We then show a second algorithmic use of the contraction sequence, by starting at its end and rewinding it. As an example of such a reverse scheme, we present a polynomial-time algorithm that properly colors the vertices of a graph with relatively few colors, thereby establishing that bounded twin-width classes are χ -bounded. This significantly extends the χ -boundedness of bounded rank-width classes, and does so with a very concise proof. It readily yields a constant approximation for MAX INDEPENDENT SET on K_t -free graphs of bounded twin-width, and a $2^{O(\text{OPT})}$ -approximation for MIN COLORING on bounded twin-width graphs. We further observe that a constant approximation for MAX INDEPENDENT SET on bounded twin-width graphs (but arbitrarily large clique number) would actually imply a PTAS.

The third algorithmic use of twin-width builds on the second one. Playing the contraction sequence backward, we show that bounded twin-width graphs can be edge-partitioned into a linear number of bicliques, such that both sides of the bicliques are on consecutive vertices, in a fixed vertex ordering. This property is trivially shared with graphs of bounded average degree. Given that biclique edge-partition, we show how to solve the unweighted SINGLE-SOURCE SHORTEST PATHS and hence ALL-PAIRS SHORTEST PATHS in time $O(n \log n)$ and time $O(n^2 \log n)$, respectively. In sharp contrast, even DIAMETER does not admit a truly subquadratic algorithm on bounded twin-width graphs, unless the Strong Exponential Time Hypothesis fails.

The fourth algorithmic use of twin-width builds on the so-called *versatile tree of contractions* [Bonnet et al., SODA '21], a branching and more robust witness of low twin-width. We present constant-approximation algorithms for MIN DOMINATING SET and related problems, on bounded twin-width graphs, by showing that the integrality gap is constant. This is done by going down the versatile tree and stopping accordingly to a problem-dependent criterion. At the reached node, a greedy approach yields the desired approximation.



© Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant;

licensed under Creative Commons License CC-BY 4.0

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).

Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 35; pp. 35:1–35:20

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Fixed parameter tractability

Keywords and phrases Twin-width, Max Independent Set, Min Dominating Set, Coloring, Parameterized Algorithms, Approximation Algorithms, Exact Algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.35

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version:* <https://arxiv.org/abs/2007.14161>

1 Introduction

As the title suggests, this is the third paper of a series [4, 3] devoted to a new graph invariant called *twin-width*. All the results presented in this paper are self-contained as the relevant background is given in Section 2. In the same section, the reader can find the definitions of *contraction sequences* and *twin-width*. For now, we are content with some intuition on these notions. This will be enough to sketch the ideas and techniques leading to our results, while sparing this introduction from too much formalism.

The twin-width of a graph is a non-negative integer measuring its distance to being a cograph. Among the several characterizations of cographs, a possible definition goes as follows. A graph is a *cograph* if one can find therein two twins,¹ identify them, and iterate this process until there is only one vertex left. This corresponds to what we define as a 0-sequence in Section 2, witnessing that cographs have twin-width 0. Conversely it is also true that graphs with twin-width 0 are cographs. We generalize this identification process by allowing a controlled error on the contracted pairs of vertices. An error graph or *red graph* keeps the faulty adjacencies appearing between a contracted pair and the vertices that are neighbor of only one vertex of the pair. A *d*-sequence is an identification or contraction sequence such that the maximum degree of the error graph never exceeds *d*. The existence of such a sequence entails that the initial graph has twin-width at most *d*.

As it turns out, many graph classes have bounded twin-width: planar graphs and more generally proper minor-closed classes, bounded rank-width or clique-width graphs, proper hereditary subclasses of permutation graphs, unit interval graphs, and some particular class of cubic expanders, to name only a few.² Considering the wide variety of these classes, it might seem that our cograph generalization has gone too far to allow for a unified algorithmic treatment of bounded twin-width graphs. The first paper of the series [4] and the current one show that this is not the case. Graphs of bounded twin-width admit algorithms whose running times are provably unattainable in general graphs. We will now detail that point.

After defining any graph parameter κ , a natural question is whether some computationally hard problems can be solved more efficiently on graphs where κ is bounded. When this turns out to be the case for several problems, it may sometimes lead to a powerful meta-theorem. A standard way of capturing a large set of problems within the same framework is through the use of logic formulas over graphs, or more generally over relational structures. In the language of parameterized algorithms, one may ask for the existence of a Fixed-Parameter Tractable (FPT) algorithm parameterized by κ and the size of the graph formula φ to be tested: More precisely, an algorithm deciding in time $f(|\varphi|, \kappa(G))n^{O(1)}$, or better

¹ i.e., two vertices with the same neighborhood beside them

² A more exhaustive list is given in Theorem 7.

$f(|\varphi|, \kappa(G))n$, whether an n -vertex graph G satisfies φ , where f is some computable function. Certainly the most famous result of that kind is the celebrated Courcelle's theorem, where the parameter κ is tree-width, and the formula φ ranges over Monadic Second Order logic (MSO_2) formulas [5]. On a slightly less general logic (namely MSO_1 , where quantification over edge sets is disallowed), the result holds for the smaller parameter clique-width [6]. It implies, for instance, that deciding whether a graph on n vertices contains a subset of k pairwise non-adjacent vertices (i.e., solving k -INDEPENDENT SET) can be done in linear time on graphs of constant clique-width, while in general graphs it cannot be solved in polynomial time unless $\text{P}=\text{NP}$, nor in time $f(k)n^{O(1)}$ unless $\text{FPT}=\text{W}[1]$. Such a result is unlikely for twin-width as k -INDEPENDENT SET remains NP-hard in planar graphs, which have constant twin-width. Nevertheless, when parameterized by the solution size k , an FPT algorithm is known in planar graphs, and more generally in any proper minor-closed graph class. Actually, on the latter class, every problem expressible by a first-order (FO) formula φ can be solved in FPT time parameterized by $|\varphi|$ [9]. In the first paper of our series [4], we extended this result and obtained the following meta-theorem for twin-width.

► **Theorem 1** ([4]). *Given an n -vertex graph G , a d -sequence of G , and a first-order formula φ , one can decide $G \models \varphi$ in time $f(|\varphi|, d)n$ for some computable function f .*

The main drawback of this kind of algorithm is the obtained running time: The function f is a tower of exponentials whose height depends on the size of the formula. This is an unavoidable price to pay to solve at once all graph problems expressible in first-order logic. Indeed, it is known that testing first-order formulas on trees requires a running time whose dependence in the size of the formula is a non-elementary function, unless $\text{P} = \text{NP}$ [10]. Furthermore the running time of our FO model checking algorithm does not get better on “seemingly simpler” formulas, such as for instance, with few quantifier alternations.

Our results

We show that twin-width and its associated contraction sequence can also give rise to practical algorithms for some individual classic graph problems. In particular, we consider the following NP-complete problems, given a graph G and an integer k , decide if:

- k -INDEPENDENT SET: there are k pairwise non-adjacent vertices.
- k -CLIQUE: there are k pairwise adjacent vertices.
- (k, r) -SCATTERED SET: there are k vertices pairwise at distance at least r .
- k -DOMINATING SET: there is a set S of k vertices such that for every vertex v of G , either $v \in S$ or v has a neighbor in S .
- (k, r) -DOMINATING SET: there is a set S of k vertices such that every vertex of G is at distance at most r of some vertex in S .

These problems, parameterized by k , are $\text{W}[1]$ -hard (the last two are even $\text{W}[2]$ -complete), thus unlikely to admit an FPT algorithm, i.e., one with running time $f(k)n^{O(1)}$, on general graphs. We obtain single-exponential parameterized algorithms for all these problems when a contraction sequence witnessing “twin-width at most d ” is given. When considering the unparameterized optimization variant, we denote these five problems by MAX INDEPENDENT SET (and MIS for short), MAX CLIQUE, DISTANCE- $(r - 1)$ MIS, MIN DOMINATING SET, and MIN r -DOMINATING SET, respectively.

► **Theorem 2.** *Given an n -vertex graph G and a d -sequence $G = G_n, \dots, G_1 = K_1$, the above-mentioned five problems can be solved in time $2^{O_d(k)}n$.*

We then consider some W[1]-complete generalizations of k -INDEPENDENT SET or of k -CLIQUE. Namely:

- WEIGHTED MAX INDEPENDENT SET: given a graph G with a weight function on vertices $w : V(G) \rightarrow \mathbb{R}$ and an integer k , decide whether there exists a set S of size exactly k of pairwise non-adjacent vertices such that $\sum_{v \in S} w(v)$ is maximum.
- INDUCED SUBGRAPH ISOMORPHISM: given a graph H on k vertices and a graph G , decide whether there exists a set $S \subseteq V(G)$ such that $G[S]$, the subgraph of G induced by S , is isomorphic to H .
- SUBGRAPH ISOMORPHISM: given a graph H on k vertices and a graph G , decide whether there exists a set $S \subseteq V(G)$ such that H is isomorphic to a subgraph of $G[S]$.

Unlike the other two problems, SUBGRAPH ISOMORPHISM is *not* a generalization of k -INDEPENDENT SET. Though it does generalize k -CLIQUE. Once the formal definition of a contraction sequence is given, it will be clear that a d -sequence for G readily yields a d -sequence for its complement, \overline{G} . Thus in the context of bounded twin-width graphs, an algorithm solving SUBGRAPH ISOMORPHISM can be used to solve k -INDEPENDENT SET. For these three problems, we now get slightly superexponential parameterized algorithms.

► **Theorem 3.** *Given an n -vertex graph G and a d -sequence $G = G_n, \dots, G_1 = K_1$, the above-mentioned three problems can be solved in time $2^{O_a(k \log k)} n$.*

The algorithms behind Theorems 2 and 3 follow the same general plan. Let us consider the n successive red graphs R_n, \dots, R_1 (error graphs) obtained after each vertex contraction.³ R_n is the edgeless n -vertex graph (since there are initially no errors) and R_1 is the 1-vertex graph. We maintain optimum partial solutions populating connected subgraphs of bounded size in each R_i . Initially in R_n , the connected subgraphs are only made of single vertices (there are no edges). So the optimum partial solutions are trivial to compute. The partial solutions for R_i are built from the partial solutions of R_{i+1} in the following way. Every partial solution *not* involving the newly contracted vertex is simply kept. Every partial solution involving the newly contracted vertex is computed by merging a bounded number of previous partial solutions on pairwise disconnected sets. The key is that, by design, there is no error between the latter partial solutions. Thus the presence or absence of edges can be decided regardless of the forgotten choices of precise vertices within the solution. Eventually a (partial) solution is computed in R_1 , which constitutes an actual solution in the entire initial graph G . In a nutshell, the algorithms may be summarized as dynamic programming over connected sets of the red graphs.

For k -INDEPENDENT SET there is not much more to it than the previous sketch. For (INDUCED) SUBGRAPH ISOMORPHISM the algorithms become more technical. Also conceptually, partial solutions are no longer necessarily feasible. For k -DOMINATING SET some new challenges appear. The partial solutions and their actual specification are not straightforward to define, as it is for k -INDEPENDENT SET.

One may wonder if subexponential parameterized algorithms are possible for any of the eight problems considered so far. We will observe that even k -INDEPENDENT SET cannot be solved in time $2^{o(k/\log k)} n^{O(1)}$ on graphs given with an $O(1)$ -sequence, unless the Exponential Time Hypothesis fails. With a similar argument, the same lower bound applies to k -DOMINATING SET. Thus, up to logarithmic factors in the exponent, the running times of Theorems 2 and 3 are optimal. Actually we will see that even algorithms running in time $2^{o(n/\log n)}$ are unlikely.

³ A reader who would want precise definitions at this point is welcome to read first the couple of paragraphs of Section 2.1.

All the previous algorithms exploit the contraction sequence forward. They follow the identification process from the initial graph G to the 1-vertex graph. What if we would start at the end, and maintain solutions as the vertices are iteratively split until the initial graph G is formed? We exemplify the idea of using the contraction sequence backward with an essentially greedy coloring procedure that is not optimal but still uses relatively few colors.

Let us be more specific. A proper k -coloring of a graph G is a mapping $c : V(G) \rightarrow \{1, \dots, k\}$ such that $c(u) \neq c(v)$ whenever $uv \in E(G)$. The chromatic number, denoted by $\chi(G)$, is the smallest integer k such that G admits a proper k -coloring. It can be seen that $\chi(G) \geq \omega(G)$, where $\omega(G)$ denotes the size of a largest clique in G , whereas many constructions of triangle-free (that is, with $\omega(G) \leq 2$) graphs G with arbitrarily large $\chi(G)$ are known. A class of graphs \mathcal{C} is χ -bounded if there is a function f such that for any graph $G \in \mathcal{C}$, we have $\chi(G) \leq f(\omega(G))$. Our coloring algorithm $(d+2)$ -colors any triangle-free graph of twin-width at most d , and more generally $(d+2)^{\omega(G)-1}$ -colors any graph G given with a d -sequence. In particular, it shows the following.

► **Theorem 4.** *Every graph class with bounded twin-width is χ -bounded.*

Algorithmically this has some direct consequences for approximating the chromatic number, as well as, in the subcase of K_t -free graphs, the independence number.

The same idea of considering the contraction sequence backward is then used to show that every graph given with an $O(1)$ -sequence admits an edge partition into $O(n)$ bicliques, each side of which is on consecutive vertices, for a fixed vertex ordering. We use this edge partition to tackle the unweighted version of some classic polynomial-time solvable problems:

- SINGLE-SOURCE SHORTEST PATHS: given a graph G and a source s , find a shortest-path tree rooted at s , spanning the connected component of s .
- ALL-PAIRS SHORTEST PATHS: given a graph G , find the distances in G between every pair of vertices.
- DIAMETER: given a graph G , report the largest distance in G between two vertices.

We show how breadth-first search (BFS) can be mimicked, when replacing “traversing an edge” by “traversing a biclique all at once”. A subtlety of the algorithm, beside the necessary data structures to get SINGLE-SOURCE SHORTEST PATHS sublinear in the total number of edges, lies in the fact that bicliques, contrary to single edges, can be traversed twice (once in both directions) before being discarded.

► **Theorem 5.** *If the input graph comes with an $O(1)$ -sequence, SINGLE-SOURCE SHORTEST PATHS can be solved in $O(n \log n)$ time, thus ALL-PAIRS SHORTEST PATHS and DIAMETER can be solved in $O(n^2 \log n)$ time. In contrast, DIAMETER cannot be solved in $O(n^{2-\varepsilon})$ for any $\varepsilon > 0$, even in that scenario, unless the Strong Exponential Time Hypothesis fails.*

Our algorithm inherently relies on unweighted edges. Nonetheless vertex-weights can be supported with the same running time.

MIN DOMINATING SET is known to be as approximable as the SET COVER problem. Thus, by classic papers by Johnson [14] and by Lovász [15], it admits a $\ln n$ -approximation and the integrality gap (i.e., the ratio between the optimum of the original problem and the optimum of the LP relaxation) of its standard LP formulation is also $\ln n$. In sharp contrast, unless $P=NP$, MIN DOMINATING SET cannot be approximated in polynomial-time within factor $(1 - o(1)) \ln n$ on n -vertex general graphs [7].

We show that, on bounded twin-width classes, the integrality gap of MIN DOMINATING SET is constant. This uses the *versatile trees of contractions* developed in the second paper of the series [3]. These are more robust witnesses of low twin-width which, instead of providing

a single contraction in a given trigraph, give linearly many disjoint ones. Placing ourselves at a right node of the versatile tree, we show that a greedy strategy in the corresponding trigraph yields a constant approximation in the original graph.

► **Theorem 6.** *If the input graph comes with an $O(1)$ -sequence, MIN DOMINATING SET, DISTANCE-2 MIS, and more generally MIN r -DOMINATING SET, DISTANCE- $2r$ MIS for every positive r , admit $O(1)$ -approximation algorithms.*

These results are particular cases of the fact that when the twin-width of a matrix A is bounded, there is a linear gap between the packing number and the minimum hitting set of the hypergraph with incidence matrix A . Bounded twin-width matrices might more generally provide linear programs with bounded duality gap. It is noteworthy that MAX INDEPENDENT SET (which corresponds to DISTANCE-1 MIS) is *not* covered by the previous theorem. We further give some evidence that MIS may have a very different approximability status than MIN DOMINATING SET on bounded twin-width graphs.

2 Preliminaries

We denote by $[i, j]$ the set of integers $\{i, i + 1, \dots, j - 1, j\}$, and by $[i]$ the set of integers $[1, i]$. If \mathcal{X} is a set of sets, we denote by $\cup \mathcal{X}$ their union. The notation $O_d(\cdot)$ gives an asymptotic behavior when d is seen as a constant. The notation $O^*(\cdot)$ suppresses polynomial factors.

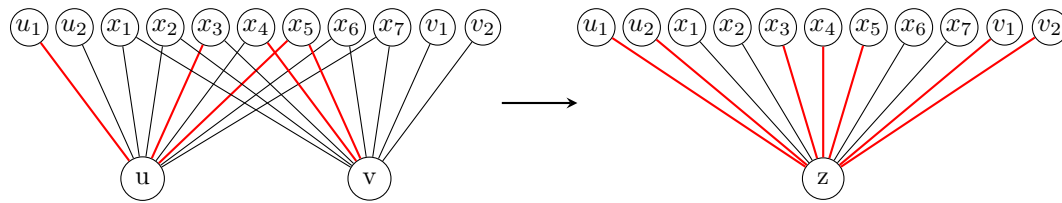
Unless stated otherwise, all graphs are assumed undirected and simple, that is, they do not have parallel edges or self-loops. We denote by $V(G)$ and $E(G)$ the set of vertices and edges respectively of a graph G . For $S \subseteq V(G)$, we denote the *open neighborhood* (or simply *neighborhood*) of S by $N_G(S)$, i.e., the set of neighbors of S deprived of S , and the *closed neighborhood* of S by $N_G[S]$, i.e., the set $N_G(S) \cup S$. We may omit the subscript if the graph is clear from the context. We denote by $G[S]$ the subgraph of G induced by S , and $G - S := G[V(G) \setminus S]$. A *connected subset* (or *connected set*) $S \subseteq V(G)$ is one such that $G[S]$ is connected. Two distinct vertices u, v such that $N(u) = N(v)$ are called *false twins*, and *true twins* if $N[u] = N[v]$. Two vertices are *twins* if they are false twins or true twins.

A graph is *H-free* if it does not contain H as an induced subgraph. However we make an exception for $H = K_{t,t}$. A $K_{t,t}$ -free graph is a graph with no biclique $K_{t,t}$ as a subgraph. A class⁴ \mathcal{C} of graphs has *property II* if every graph of \mathcal{C} has property *II*. A class is *hereditary* if it is closed under taking induced subgraphs.

2.1 Trigraphs, contraction sequences, and twin-width of a graph

A *trigraph* G has vertex set $V(G)$, (black) edge set $E(G)$, and red edge set $R(G)$ (the error edges), with $E(G)$ and $R(G)$ being disjoint. The *set of neighbors* $N_G(v)$ of a vertex v in a trigraph G consists of all the vertices adjacent to v by a black or red edge. A d -trigraph is a trigraph G such that the *red graph* $(V(G), R(G))$ has degree at most d . In that case, we also say that the trigraph has *red degree* at most d . A (vertex) *contraction* or *identification* in a trigraph G consists of merging two (non-necessarily adjacent) vertices u and v into a single vertex z , and updating the edges of G in the following way. Every vertex of the symmetric difference $N_G(u) \Delta N_G(v)$ is linked to z by a red edge. Every vertex x of the intersection $N_G(u) \cap N_G(v)$ is linked to z by a black edge if both $ux \in E(G)$ and $vx \in E(G)$, and by a red

⁴ That is, a set of graphs closed under isomorphism.



■ **Figure 1** Contraction of vertices u and v , and how the edges of the trigraph are updated.

edge otherwise. The rest of the edges (not incident to u or v) remain unchanged. We insist that the vertices u and v (together with the edges incident to these vertices) are removed from the trigraph. See Figure 1 for an illustration.

A d -sequence (or *contraction sequence*) is a sequence of d -trigraphs G_n, G_{n-1}, \dots, G_1 , where $G_n = G$, $G_1 = K_1$ is the graph on a single vertex, and G_{i-1} is obtained from G_i by performing a single contraction of two (non-necessarily adjacent) vertices. We observe that G_i has precisely i vertices, for every $i \in [n]$. The twin-width of G , denoted by $tww(G)$, is the minimum integer d such that G admits a d -sequence.

For $u \in V(G_i)$, we denote by $u(G)$ the subset of $V(G)$ that was contracted to the single vertex u in G_n, G_{n-1}, \dots, G_i . Twin-width and d -sequences can be equivalently seen as a partition refinement process on $V(G)$. We start with the finest partition $\mathcal{P}_n = \{\{v\} : v \in V(G)\}$, and end with the coarsest partition $\mathcal{P}_1 = \{V(G)\}$. There is a *partition sequence* $\mathcal{P}_n, \mathcal{P}_{n-1}, \dots, \mathcal{P}_2, \mathcal{P}_1$ mimicking the contraction sequence, where the contraction of $u, v \in V(G_i)$ corresponds to the merge of parts $u(G_i), v(G_i) \in \mathcal{P}_i$ to form the part $u(G_i) \cup v(G_i) = z(G_{i-1}) \in \mathcal{P}_{i-1}$, while all the other parts are unchanged from \mathcal{P}_i to \mathcal{P}_{i-1} . The red degree (bounded by d) of a part $P \in \mathcal{P}_i$ now corresponds to the number of other parts $P' \in \mathcal{P}_i$ which are not fully adjacent nor fully non-adjacent to P in G . We may denote by $G_{\mathcal{P}}$ the trigraph corresponding to partition \mathcal{P} over $V(G)$. Thus $G_i = G_{\mathcal{P}_i}$.

2.2 Classes with bounded twin-width and how the sequences are given

The current paper is devoted to presenting efficient algorithms when the input has bounded twin-width, and the contraction sequence is given. It is therefore important to know how realistic this scenario is. Fortunately, in the first two papers of the series [4, 3] we showed that many central sparse and dense (di)graph classes have bounded twin-width.

► **Theorem 7** ([4, 3]). *The following classes have bounded twin-width, and $O(1)$ -sequences for n -vertex members can be computed in $O(n^2)$ time.*

- Bounded clique-width/rank-width, and more generally, boolean-width graphs,
- every hereditary proper subclass of permutation graphs,
- posets of bounded antichain size (seen as digraphs),
- unit interval graphs,
- K_t -minor free graphs,
- map graphs (given with an embedding),
- subgraphs of d -dimensional grids,
- K_t -free unit d -dimensional ball graphs,
- $\Omega(\log n)$ -subdivisions of all the n -vertex graphs,
- cubic expanders defined by iterative random 2-lifts from K_4 ,
- strong products of two bounded twin-width classes one of which has also bounded degree,
- any subgraph closure of a $K_{t,t}$ -free bounded twin-width class, and
- any first-order transduction of a bounded twin-width class.

2.3 Selected results for the short version

Due to space constraints, we select a representative sample of the results announced in the introduction. This sample covers at least partially Theorems 2–4 and 6. We present the following four items on bounded twin-width classes, where the input graph comes with an $O(1)$ -sequence; each item sharply contrasting with what is possible on general graphs.

- In Section 3 we give a linear FPT algorithm for k -INDEPENDENT SET.
- In Section 4 we give a linear FPT algorithm for k -DOMINATING SET.
- In Section 5 we show a constant approximation for MIN DOMINATING SET.
- In Section 6 we show that bounded twin-width graphs are χ -bounded.

That selection presents our new conceptual ideas in their simplest form, while echoing the title of the paper. For more details on these results or for the proofs *not* covered in the short version, we refer the reader to the long version in appendix.

3 Practical algorithm for k -Independent Set

The running time analysis of the forthcoming algorithm is based on a folklore bound on the number of connected subsets of size at most k in a bounded-degree graph.

► **Lemma 8.** *The number of connected vertex sets of size at most k , intersecting a set X , in a graph of maximum degree d is at most $(d^{2k-2} + 1)|X|$. Furthermore they can be enumerated in time $O(d^{2k-2}|X|)$.*

We show how to solve k -INDEPENDENT SET by dynamic programming on the connected subsets of size at most k in the red graphs of a d -sequence given with the input graph.

► **Theorem 9.** *Given an n -vertex graph G , a positive integer k , and a d -sequence $G = G_n, \dots, G_1 = K_1$, k -INDEPENDENT SET can be solved in time $O(k^2 d^{2k} n) = 2^{O_a(k)} n$.*

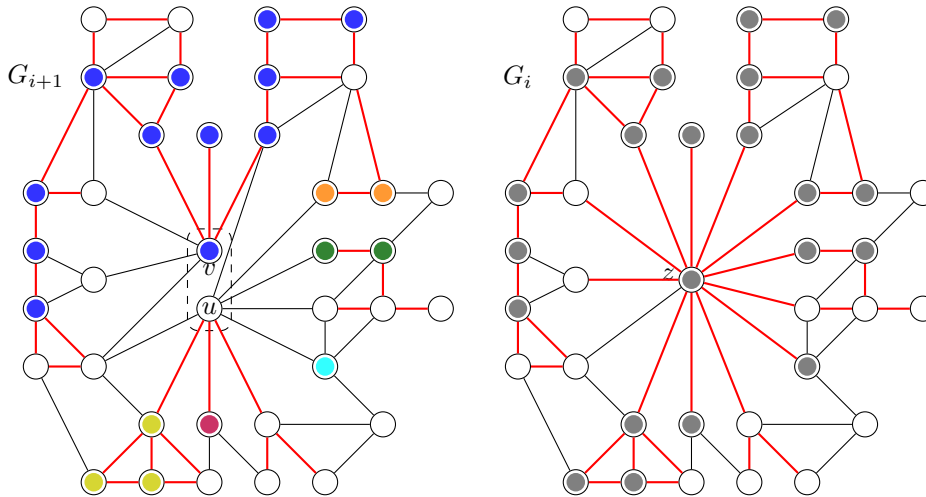
Proof. Our algorithm maintains a set of *optimum partial solutions* in the current trigraph, starting from G , and progressively going along the d -sequence. Let us start with a definition of the partial solutions and of their optimality.

A *partial solution* in the trigraph G_i is a pair (T, S) where $T \subseteq V(G_i)$ is a vertex set inducing a connected subgraph in the red graph $(V(G_i), R(G_i))$, and $S \subseteq V(G)$ is an independent set of G such that $S \subseteq \bigcup_{u \in T} u(G)$ and for every $u \in T$, $S \cap u(G) \neq \emptyset$. A partial solution (T, S) is said *optimum* if there is no partial solution (T, S') such that $|S'| < |S|$. A set $T \subseteq V(G_i)$ is said *realizable* (in G_i) if there is an $S \subseteq V(G)$ such that (T, S) is a partial solution in G_i . Notice that *not* every connected subset in the red graph is realizable. For instance, it is easy to engineer a situation where there is no independent set intersecting the three vertices of a 3-vertex red path. Initially, in G , the only connected subgraphs of the red graph are singletons (since there is no red edge). So there are exactly n (optimum) partial solutions in $G = G_n$: Each vertex v of G induces a partial solution $(\{v\}, \{v\})$. We denote by \mathcal{S}_n this set of n optimum partial solutions. It boils down to determining if there is a partial solution $(_, S)$ in G_1 (or actually in any G_i) with $|S| \geq k$. For i going from $n - 1$ down to 1, we will build a set of optimum partial solutions \mathcal{S}_i in G_i from the set \mathcal{S}_{i+1} , keeping the invariant that for every realizable set $T \subseteq V(G_i)$, there is a unique optimum partial solution (T, S) stored in \mathcal{S}_i (and no other partial solution in \mathcal{S}_i).

We shall then describe how we update the set of optimum partial solutions after a single contraction. Two partial solutions $(T, _)$ and $(T', _)$ in G_i are *disjoint* if $T \cap T' = \emptyset$, and *separate*, if they are disjoint and there is no red edge $uu' \in R(G_i)$ with $u \in T$ and

$u' \in T'$. Two separate partial solutions $(T, _)$ and $(T', _)$ are *compatible* if there is no edge $uu' \in E(G_i) \cup R(G_i)$ with $u \in T$ and $u' \in T'$. The *union* of two compatible partial solutions (T_1, S_1) and (T_2, S_2) as $(T_1, S_1) \cup (T_2, S_2) := (T_1 \cup T_2, S_1 \cup S_2)$. By definition, such a union is *not* a partial solution since T induces two connected components in its current red graph. Nevertheless we will build the new (connected) partial solutions of G_i by making unions of up to $d + 2$ pairwise compatible partial solutions in G_{i+1} . These unions will be connected in G_i , hence will correspond to partial solutions as well.

Let us be more specific. Say $u, v \in V(G_{i+1})$ are contracted into $z \in V(G_i)$ to form G_i . We say that a partial solution $(T, _)$ in G_i *intersects* a set $X \subseteq V(G_i)$ if $T \cap X \neq \emptyset$. We initialize \mathcal{S}_i with all the partial solutions of \mathcal{S}_{i+1} not intersecting $\{u, v\}$. We now add one partial solution in \mathcal{S}_i per realizable set $T \ni z$ in G_i , of size at most k . For every $T \subseteq V(G_i)$ such that $z \in T$ and T induces a connected subgraph on at most k vertices in the red graph $(V(G_i), R(G_i))$, we observe three possibilities for a potential partial solution (T, S) . Either S intersects $u(G)$ and $v(G)$, or it intersects only $u(G)$, or it intersects only $v(G)$. (It is not possible that $S \cap (u(G) \cup v(G)) = \emptyset$ since T contains z .) Therefore we take the best (meaning with the largest S , breaking ties arbitrarily) of the potential partial solutions $\bigcup \text{dec}(T \setminus \{z\} \cup \{u, v\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{u\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{v\})$, where $\text{dec}(X)$ is the set with one partial solution per connected component of X in its red graph (here $(V(G_{i+1}), R(G_{i+1}))$). See Figure 2 for an illustration of this decomposition.



■ **Figure 2** Right: In gray, a connected vertex set T in the red graph of G_i in the vicinity of the just contracted vertex $z \in T$. Left: The decomposition $\text{dec}(T \setminus \{z\} \cup \{v\})$ in the previous trigraph G_{i+1} , where each color represents a connected component. If every color class is a realizable set in G_{i+1} , then T is realizable in G_i , with (optimum) partial solution $\bigcup \text{dec}(T \setminus \{z\} \cup \{v\})$. Note that, due to black edges between u and some vertices of T , the partial solutions in $\text{dec}(T \setminus \{z\} \cup \{u, v\})$ and in $\text{dec}(T \setminus \{z\} \cup \{u\})$ cannot be pairwise compatible.

In the very possible event that at least one such connected component of X is not realizable, $\text{dec}(X) = \text{None}$. The union $\bigcup \text{dec}(X)$ of all the partial solutions of $\text{dec}(X)$ is None if $\text{dec}(X) = \text{None}$ or if there is at least one black edge between two connected components. Otherwise $\bigcup \text{dec}(X)$ is a pair (T, S) as defined in the previous paragraph, since the partial solutions of $\text{dec}(X)$ are pairwise compatible. Since T is chosen connected in $(V(G_i), R(G_i))$, (T, S) is indeed a partial solution in G_i . If $\bigcup \text{dec}(T \setminus \{z\} \cup \{u, v\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{u\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{v\})$ all three evaluate to None , then $\text{best}\{\bigcup \text{dec}(T \setminus \{z\} \cup \{u, v\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{u\}), \bigcup \text{dec}(T \setminus \{z\} \cup \{v\})\} = \text{None}$.

$\{z\} \cup \{v\}$) also returns None. This would mean that T is not realizable. If instead T is realizable, we get a partial solution (T, S) that we put in \mathcal{S}_i . If $|S| \geq k$, we already have a large enough independent set; the algorithm outputs it and terminates.

If we finally build \mathcal{S}_1 , and no independent set of size at least k was found, we output S , the unique set such that $(_, S) \in \mathcal{S}_1$. \mathcal{S}_1 is indeed a singleton since there is only one realizable set in G_1 . That finishes the description of the algorithm.

Details on the correctness and running time can be found in the long version. The correctness uses the classic inductive arguments for an algorithm based on dynamic programming. The claimed running time for **k-IndSet** essentially follows from Lemma 8.

Optimizations. We suggest some improvements or variations of **k-IndSet** to generally improve over the worst-case running time of the inner for loop. A lot of sets T will trivially be *not* realizable because they induce a black edge. When enumerating the walks starting at z of length at most $2k - 3$, one can abort every branch $zv_1 \dots v_h$ inducing at least one black edge. It can even be done in a way that the enumeration takes time $O(t)$ where t is the number of sets $T \ni z$ of size at most k , such that T is connected in the red graph, and an independent set in the black graph.

Even if a set T satisfies those properties, we have no guarantee that T is realizable. In very dense instances, it is imaginable that the realizable sets are very rare. In that case, we will lose a lot of time generating sets T to observe immediately after that there is no associated partial solution (T, S) . An alternative to **k-IndSet** is to build the new partial solutions of \mathcal{S}_i directly as unions of pairwise compatible partial solutions of \mathcal{S}_{i+1} , without anticipating the nature of the possibly realizable set $T \subseteq V(G)$.

Let us be more precise. Let R_z be the set of red neighbors of z in G_i . For every set of at most $\max(2, d + 1)$ partial solutions $(T_1, S_1), \dots, (T_h, S_h) \in \mathcal{S}_{i+1}$ intersecting R_z , at least one of which intersects $\{u, v\}$, if the partial solutions are pairwise compatible, we update the realizable set $\bigcup_{i \in [h]} T_i$ with the partial solution $\bigcup_{i \in [h]} (T_i, S_i)$ if $\bigcup_{i \in [h]} S_i$ is larger than the current best solution. Following the first improvement, we can only generate the sets that are pairwise compatible. As we know, there are at most three ways to reach a given set $T \subseteq V(G_i)$ as a union of pairwise compatible partial solutions in \mathcal{S}_{i+1} . The running time of this variation of **k-IndSet** is $O^*(\sum_{i \in [n]} |\mathcal{S}_i^{\text{new}}|)$, where $\mathcal{S}_i^{\text{new}} := \mathcal{S}_i \setminus \mathcal{S}_{i-1}$ (and $\mathcal{S}_n^{\text{new}} := \mathcal{S}_n$) represents the new partial solutions computed at step i . In practice, this can be significantly better than $O(k^2 d^{2k} n)$. Such a dynamic programming, only generating “positive” subinstances, dubbed *positive-instance driven* by Tamaki, led to a breakthrough and current state-of-the-art practical algorithm for computing optimally the treewidth of a graph [17]. ◀

Without too many changes, **k-IndSet** may support weights, that is, find an independent set of size exactly $\min(k, \alpha(G))$ with largest total weight. Instead of keeping one solution S per realizable set T , we keep up to k solutions, one per pair (T, j) with $j \in [|T|, k]$. A partial solution (T, j, S) is defined as before except S is required to have size exactly j . To compute the new partial solutions, we add a third nested for loop after line 6: We iterate over all the ways of distributing $j \leq k$ units between the red connected components induced by $T' \in \{T \setminus \{z\} \cup \{u, v\}, T \setminus \{z\} \cup \{u\}, T \setminus \{z\} \cup \{v\}\}$ so that each connected component gets a positive integer (at least equal to its size). We then add to \mathcal{S}_i one partial solution (T, j, S) (if at least one exists) maximizing the weight of S for fixed T and j . We also skip lines 8 and 9 of **k-IndSet**.

This comes with a slight increase in the running time. Namely, there is an extra $2^{O(k \log k)}$ factor accounting for the ordered partition of integer $j \leq k$ into positive integers. Thus the overall running time with weights is $2^{O(k \log k)} d^{2k} n$.

As twin-width and d -sequences are preserved when complementing the graph, we also solve k -CLIQUE in the same running time. One may wonder if the dependency in k of our $2^{O_d(k)}n$ -time algorithm can be improved. It turns out that this running time is essentially optimal. Due to the Sparsification Lemma [13] and folklore reductions, MIS restricted to subcubic n -vertex graphs cannot be solved in $2^{o(n)}$, under the Exponential Time Hypothesis⁵ (ETH) [12]. Thus, by the classic self-reduction consisting of performing an even subdivision of each edge [16], MIS cannot be solved in time $2^{o(n/\log n)}$ on $2\lceil\log n\rceil$ -subdivisions of n -vertex subcubic graphs, unless the ETH fails. In [3], we show how to find $O(1)$ -sequences in polynomial time for $2\lceil\log n\rceil$ -subdivisions of n -vertex graphs. Therefore this lower bound holds even if we are given the d -sequence. In particular, no algorithm solves k -INDEPENDENT SET in time $2^{o_d(k/\log k)}n^{O(1)}$, unless the ETH fails.

4 A practical algorithm for k -Dominating Set

We solve k -DOMINATING SET with a more involved instantiation of the scheme of the previous section. We face some new conceptual difficulties compared to the algorithm for k -INDEPENDENT SET. For one thing, the partial solutions that we maintain are not feasible solutions in the whole graph. Also we now consider balls of radius $f(d)k$ in the red graphs, and not merely of radius k . In general, the arguments are more subtle to handle partially and fully dominated vertex sets, as well as the solution trace. This entails a worse dependency in d , but the same essentially optimal $2^{O(k)}n$ when d is treated as a constant.

► **Theorem 10.** *Given an n -vertex graph G , a positive integer k , and a d -sequence $G = G_n, \dots, G_1 = K_1$, k -DOMINATING SET can be solved in time $O(2^{2(d^2+1)(2+\log d)k}n) = 2^{O_d(k)}n$.*

Proof. As was the case with k -INDEPENDENT SET, the algorithm sequentially considers each trigraph in the d -sequence G_n, \dots, G_1 starting from G_n , and inductively updates a set of optimal partial solutions of the trigraph G_i to yield the next set for G_{i-1} . We recall that $E(G_i)$ and $R(G_i)$ respectively refer to the black and red edge set of the trigraph G_i . The ball of radius at most r in the red graph $(V(G_i), R(G_i))$ centered at a vertex $x \in V(G_i)$ is denoted as $B_i^r(x)$.

Profile of a partial solution. A *profile (of a partial solution)* of G_i is a triple (T, D, M) of vertex sets of $V(G_i)$ such that (i) T forms a connected set in the red graph $(V(G_i), R(G_i))$, (ii) $D, M \subseteq T$, and (iii) $\bigcup_{x \in D} B_i^2(x) \subseteq T$. The first entry T of a profile $P = (T, D, M)$ is called the *ground set* of P , and the size of P is defined as the size of its ground set. A profile (T, D, M) is said to be a *k -profile* if $|D| \leq k$. When the profile under consideration is clear from the context, we denote $T \setminus D$ and $T \setminus M$ by \bar{D} and \bar{M} respectively.

We say that a *profile (T, D, M) is realizable with $S \subseteq V(G)$* if the following conditions hold.

1. $S \subseteq \bigcup_{x \in T} x(G)$,
2. for every $x \in V(G_i)$, $x \in D$ if and only if $x(G) \cap S \neq \emptyset$, and
3. for every $x \in V(G_i)$, $x \in M$ if and only if $x(G)$ is (fully) dominated by S .

A profile is said to be *realizable* if there exists S with which it is realizable.

Suppose that $x, y \in V(G_{i+1})$ are contracted to yield G_i with z being the new vertex. For a vertex set $T \subseteq V(G_i)$ connected in the red graph $(V(G_i), R_i)$ and containing z , let T_1, \dots, T_ℓ be the red connected components of $T' = (T \setminus z) \cup \{x, y\}$ in G_{i+1} , i.e. the partition of T'

⁵ The assumption that there is a constant $\delta > 0$, such that 3-SAT cannot be solved in time $2^{\delta n}$.

into maximal vertex sets each of which is connected in $V(G_{i+1}, R_{i+1})$. The number of these red subgraphs does not exceed $d + 2$ because each T_i either contains x or y , or one of the newly created red neighbors of z . Notice also that ℓ can be equal to 1, which means that x and y belong to the same connected component of $(V(G_{i+1}), R(G_{i+1}))$.

For a k -profile (T, D, M) of G_i such that $z \in T$, we say that a set $\mathcal{P} = \{(T_1, D_1, M_1), \dots, (T_\ell, D_\ell, M_\ell)\}$ of k -profiles of G_{i+1} is *consistent with* (T, D, M) if the following holds. Let $T' := (T \setminus z) \cup \{x, y\}$, $D' := \bigcup_{j=1}^{\ell} D_j$ and $M' := \bigcup_{j=1}^{\ell} M_j$.

1. The ground sets of the profiles in \mathcal{P} are precisely the red components of T' in G_{i+1} .
2. $D \setminus z = D' \setminus \{x, y\}$.
3. $z \in D$ if and only if $x \in D'$ or $y \in D'$.
4. For every $u \in T \setminus z$, $u \in M$ if and only if $u \in M'$ or there exists $v \in D'$ such that uv is a black edge in G_{i+1} .
5. $z \in M$ if and only if for each $u \in \{x, y\}$, it holds that: $u \in M'$ or there exists $v \in D'$ such that uv is a black edge in G_{i+1} .

Algorithm, and how to compute τ_i from τ_{i+1} . At each iteration along the d -sequence, we maintain one mapping τ_i from k -profiles $P = (T, D, M)$ of G_i with $|T| < (d^2 + 1)k$ to a subset of $\bigcup_{t \in T} t(G)$. The assignment $\tau_i(P) = \text{nil}$ is interpreted as that P is not realizable whereas $\tau_i(P) \neq \text{nil}$ is intended to be a minimum-size vertex set of $V(G)$ realizing P . Again let G_i be obtained by contracting the vertices $x, y \in V(G_{i+1})$ and z be the new vertex. Our goal is to compute τ_i from τ_{i+1} , assuming τ_{i+1} has been computed correctly. Note that a k -profile $P = (T, D, M)$ of G_i such that $z \notin T$ is also a profile of G_i , and trivially one is realizable with S if and only if the other is realizable with S . Therefore, τ_i simply inherits the assignment of τ_{i+1} in this case as depicted in lines 6-7.

If $P = (T, D, M)$ has z in its ground set, the algorithm `k-DomSet` inspects all sets \mathcal{P} of k -profiles of G_{i+1} consistent with (T, D, M) and among the unions $\bigcup_{P \in \mathcal{P}} \tau_{i+1}(P)$ over all such \mathcal{P} , outputs the best one as $\tau_i(T, D, M)$, that is, the one of minimum cardinality is chosen. If $\bigcup_{P \in \mathcal{P}} \tau_{i+1}(P) = \text{nil}$ for each consistent \mathcal{P} , the algorithm concludes that (T, D, M) is not realizable and assigns nil . The case when \mathcal{P} contains a k -profile P with ground set of size at least $(d^2 + 1)k$, a special step is taken as τ_{i+1} is not defined on such P . In this situation, a vertex $v \in T' \setminus \bigcup_{t \in D'} B_{i+1}^2(t)$ is chosen, and the query at $(T' \setminus v, D' \setminus v, M' \setminus v)$ is made instead. Lines 15-18 handle this case. The uniqueness of k -profile in \mathcal{P} in line 16 and the existence of such v in line 17 will be discussed in the correctness proof.

Correctness. To show the correctness of Algorithm 1, it suffices to prove the following.

- (\star) For every $i \in [n]$ and every k -profile P of G_i , we have $\tau_i(P) \neq \text{nil}$ if and only if P is realizable with a set of size at most k . Furthermore, if $\tau_i(P) \neq \text{nil}$, then $\tau_i(P)$ is a set of minimum size with which P is realizable.

We prove (\star) by induction. In the base case when $i = n$, the claim trivially holds. Assume $i < n$ and let x, y be the vertices of G_{i+1} which were contracted to yield G_i , where z is the newly obtained vertex of G_i . By induction hypothesis, for any k -profile (T, D, M) of G_i with $z \notin T$ the claim holds as it is a k -profile of G_{i+1} as well.

Therefore, we assume that $z \in T$ and let $T' = (T \setminus z) \cup \{x, y\}$.

\triangleright **Claim 11.** Assume that (\star) holds for all $i' > i$ and let $P = (T, D, M)$ be a k -profile of G_i . If P is realizable with a set of size at most k , then $\tau_i(P) \neq \text{nil}$.

Proof. Suppose that $P = (T, D, M)$ is realizable with $S \subseteq V(G)$ of size at most k . Let T_1, \dots, T_ℓ be the red connected components of T' in G_i , and let $S_j = S \cap \bigcup_{t \in T_j} t(G)$ for every $j \in [\ell]$. The pairs T_j and S_j for $j = 1, \dots, \ell$ define a set of ℓ k -profiles (T_j, D_j, M_j) of G_{i+1} in a canonical way: D_j is precisely the set of vertices $t \in T_j$ such that $t(G) \cap S_j$ and M_j is the set of vertices $t \in T_j$ such that $t(G)$ is (fully) dominated by S_j . By construction, each k -profile (T_j, D_j, M_j) is realizable with S_j .

We argue that the set $\mathcal{P} = \{(T_j, D_j, M_j) : j \in [\ell]\}$ is consistent with $P = (T, D, M)$. The first and the second conditions for consistency are clearly satisfied. To verify the third condition, consider a vertex $u \in T$ distinct from z and without loss of generality we assume $u \in T_{j^*}$. If $u \in M$ and $u \notin M_{j^*}$, this means that S_{j^*} does not dominate $u(G)$ because S_{j^*} realizes $(T_{j^*}, D_{j^*}, M_{j^*})$. From $u \in M$ and the fact that S realizes (T, D, M) , we know that S dominates $u(G)$ and thus there is at least one vertex $S \setminus S_{j^*}$ which is adjacent (in G) with some vertex of $u(G)$. Consider an arbitrary vertex $v \in T$ to which some of $S \setminus S_{j^*}$ contracts to, and observe that $v \notin T_{j^*}$. This means that uv is a black edge. The converse direction of the third condition is clearly met. The fourth condition of consistency can be verified similarly as the third condition. If \mathcal{P} does not contain any k -profile whose ground set has size at least $(d^2 + 1)k$, now the claim is immediate because each (T_j, D_j, M_j) is realizable with S_j : by induction hypothesis, we have $\tau_{i+1}(T_j, D_j, M_j) \neq nil$, and thus $\tau_i(T, D, M)$ is set to $\neq nil$ at line 14.

Suppose that \mathcal{P} contains a k -profile whose ground set has size at least $(d^2 + 1)k$. One can easily see that in this case, $\ell = 1$ or equivalently T' is a red connected component in $(V(G_{i+1}), R(G_{i+1}))$ consisting of exactly $(d^2 + 1)k$ vertices. Since the union of at most k balls of radius at most 2 which is connected in $(V(G_{i+1}), R(G_{i+1}))$ have less than $(d^2 + 1)k$ vertices, there exists $v \in T' \setminus \bigcup_{t \in D'} B_{i+1}^2(t)$. Moreover, by the choice of v , $(T' \setminus v, D' \setminus v, M' \setminus v)$ is now a k -profile of G_{i+1} . To conclude that $\tau_i(T, D, M) \neq nil$, it suffices to prove that $\tau_{i+1}(T' \setminus v, D' \setminus v, M' \setminus v) \neq nil$. We do this by showing that (T, D, M) , (T', D', M') and $(T' \setminus v, D' \setminus v, M' \setminus v)$ are equivalent in regards to realizability.

The equivalence of the first two is obvious. For the equivalence of the last two, note that if S realizes (T', D', M') , S does not intersect $v(G)$, and thus S trivially realizes $(T' \setminus v, D' \setminus v, M' \setminus v)$. Conversely, suppose that $(T' \setminus v, D' \setminus v, M' \setminus v)$ is realizable with S' . The crucial observation is that v has no red neighbor in D' since otherwise, v belongs to the union $\bigcup_{t \in D'} B_{i+1}^2(t)$, contradicting the choice of v . Therefore, we know that $v \in M'$ if and only if there exists $u \in D' \setminus v$ such that uv is a black edge. In the case when $v \in M'$, there exists a black neighbor $u \in D' \setminus v$ of v , and any S' realizing $(T' \setminus v, D' \setminus v, M' \setminus v)$ intersects $u(G)$. It follows that S' fully dominates $v(G)$ and S' realizes (T', D', M') . Else if $v \notin M'$, this means that not only the red neighbors of v are disjoint from D' but also no black neighbor of v is contained in D' . As a consequence $v(G)$ is not dominated by S' , thus S' realizes (T', D', M') . This proves the equivalence of (T', D', M') and $(T' \setminus v, D' \setminus v, M' \setminus v)$, and completes the proof of the claim. \triangleleft

To establish the other direction, suppose that $\tau_i(T, D, M) \neq nil$ and let \mathcal{P}^* be the set consistent with P such that $\tau_i(T, D, M) = \bigcup_{P \in \mathcal{P}^*} \tau_{i+1}(P)$ or $\tau_i(T, D, M) = \tau_{i+1}(T' \setminus v, D' \setminus v, M' \setminus v)$ for some v . Such \mathcal{P}^* clearly exists since otherwise only nil can be output. In the former case, it is tedious to verify that if each (T_i, D_i, M_i) of \mathcal{P}^* is realizable with S_i , then $\bigcup_{i \in [\ell]} S_i$ realizes (T, D, M) . In the latter case, we simply recall that (T, D, M) and $(T' \setminus v, D' \setminus v, M' \setminus v)$ are equivalent in regards to realizability. This completes the proof of the first statement of (\star) . The second statement immediately follows.

■ **Algorithm 1** k -DomSet.

Input : A graph G , a positive integer k , and a d -sequence $G = G_n, \dots, G_1 = K_1$.
Output: A dominating set of G of size at most k , or report *nil* (NO-instance).

- 1 **for** $v \in V(G_n)$ **do**
- 2 $\tau_n(\{v\}, \{v\}, \{v\}) = \{v\}$, $\tau_n(\{v\}, \emptyset, \emptyset) = \emptyset$, $\tau_n(P) = \text{nil}$ for all other k -profiles P
- 3 **for** $i = n - 1 \rightarrow 1$ **do**
- 4 $x, y \leftarrow$ contracted pair in $G_{i+1} \rightarrow G_i$
- 5 $z \leftarrow$ contraction of x and y in G_i
- 6 **for every** k -profile (T, D, M) of G_i of size less than $(d^2 + 1)k$ s.t. $z \notin T$ **do**
- 7 $\tau_i(T, D, M) \leftarrow \tau_{i+1}(T, D, M)$
- 8 **for every** k -profile (T, D, M) of G_i of size less than $(d^2 + 1)k$ s.t. $z \in T$ **do**
- 9 $\tau_i(T, D, M) \leftarrow \text{nil}$
- 10 $T' \leftarrow (T \setminus z) \cup \{x, y\}$
- 11 **for every set** \mathcal{P} of k -profiles of G_{i+1} consistent with (T, D, M) **do**
- 12 **if each** k -profile of \mathcal{P} has size less than $(d^2 + 1)k$ **then**
- 13 **if** $\tau_{i+1}(P) \neq \text{nil}$ for all $P \in \mathcal{P}$ **then**
- 14 $\tau_i(T, D, M) \leftarrow \text{best}\{\tau_i(T, D, M), \bigcup_{P \in \mathcal{P}} \tau_{i+1}(P)\}$
- 15 **else**
- 16 Let (T', D', M') be the unique k -profile contained in \mathcal{P} .
- 17 Choose $v \in T' \setminus \bigcup_{t \in D'} B_{i+1}^2(t)$
- 18 $\tau_i(T, D, M) \leftarrow \text{best}\{\tau_i(T, D, M), \tau_{i+1}(T' \setminus v, D' \setminus v, M' \setminus v)\}$
- 19 **if** $\tau_i(T, D, M) \neq \text{nil}$ and has size larger than k **then**
- 20 $\tau_i(T, D, M) \leftarrow \text{nil}$
- 21 **return** $\tau_1(V(G_1), V(G_1), V(G_1))$

Running time. In an actual implementation of Algorithm 1, we maintain a single mapping τ . As we proceed from G_{i+1} to G_i , we modify the domain of τ consisting of k -profiles so that new k -profiles involving z are added and after calculating the assignments for the new k -profiles, all the domains and corresponding assignments involving x or y shall be discarded. Therefore, it suffices to check the running time for updating τ , which is performed in the inner loop of lines 6-20. By Lemma 8, there are $O(d^{2(d^2+1)k-2} \cdot 2^{2(d^2+1)k})$ new profiles of G_i to compute. For each k -profile (T, D, M) with $z \in T$, the ground sets T_1, \dots, T_ℓ of a potentially consistent set \mathcal{P} is already determined. Hence, we exhaust all possibilities of appending each T_i by M_i and D_i to form a k -profile and the inner loop of 8-20 will consider at most $2^{(d^2+1)k} \cdot 2^{(d^2+1)k}$ sets \mathcal{P} . The consistency of \mathcal{P} with (T, D, M) can be routinely verified. This establishes the claimed running time. ◀

5 Approximation algorithms

5.1 Constant approximation for Min Dominating Set

In this section, we prove that MIN DOMINATING SET has bounded integrality gap in classes of bounded twin-width. A constant factor approximation algorithm readily follows. We will use the following technical lemma from the second paper of the series.

► **Theorem 12** (Section 3, Lemma 20 in [3]). *For every integer t , there are integers s and t' such that every graph G with a t -sequence admits a rooted tree \mathcal{T} with the following properties.*

- *Every node of \mathcal{T} is labeled by a t' -trigraph.*
- *The root of \mathcal{T} is labeled by G .*
- *All the leaves of \mathcal{T} are labeled by the 1-vertex graph K_1 .*
- *If a node x of \mathcal{T} is labeled by H , and a child node of x is labeled by H' , there is a t' -contraction in H that yields H' . In particular $|V(H)| = |V(H')| + 1$.*
- *Every internal node of \mathcal{T} labeled by H has at least $|V(H)|/s$ children coming from t' -contractions on pairwise disjoint pairs of vertices of H .*

Such a tree is called an *s -versatile tree of t' -contractions*. Informally Theorem 12 says that, by degrading the twin-width bound, one can move away from the “linear nature” of the contraction sequence to a profusely branching contraction witness.

Theorem 12 is effective: The s -versatile tree of t' -contractions can be computed in polynomial time, if a t -sequence for G is provided.

► **Theorem 13.** *In classes of bounded twin-width, MIN DOMINATING SET has bounded integrality gap.*

Proof. Let G be a graph of twin-width at most t . By Theorem 12, there exist t', s functions of t only such that G admits an s -versatile tree of t' -contraction. Let $w^* : V(G) \rightarrow [0, 1]$ be the weight function of a minimum fractional dominating set, with total weight γ^* . Thus w^* is an optimum solution of the linear program

$$\begin{aligned} & \text{minimize} && \sum_{x \in V(G)} w(x) \\ & \text{with } \forall x \in V(G), && \sum_{y \in N[x]} w(y) \geq 1, \text{ and } 0 \leq w(x) \leq 1, \end{aligned}$$

and $\gamma^* = \sum_{x \in V(G)} w^*(x)$. The weight function w^* is extended to subsets of vertices by sum. We assume that G has at least one vertex, so $\gamma^* \geq 1$.

We now greedily perform contractions in G following the versatile tree of contractions with a restriction: contractions involving a part of total weight at least $\frac{1}{2(t'+1)}$ are forbidden. Let us explain what this means in more detail. We start at the root, labeled G , of the versatile tree. We move to a(ny) child node along an edge corresponding to a non-forbidden t' -contraction. A *non-forbidden* contraction is one of u, v with $w^*(u(G)) < \frac{1}{2(t'+1)}$ and $w^*(v(G)) < \frac{1}{2(t'+1)}$. We iterate that until we get stuck (every child of the current node entails a forbidden contraction).

We adopt the partition viewpoint of the t' -sequence. Let \mathcal{P} be the partition of $V(G)$ obtained when this process finishes, and let $G_{\mathcal{P}}$ be the corresponding trigraph (that is, the label of the node where we stop). We observe that we cannot end at a leaf of the versatile tree. Indeed that would mean that the last contraction merged a bipartition $\{X, Y\}$ of $V(G)$ into $\{V(G)\}$. As $\gamma^* \geq 1$, this would imply that $w^*(X) \geq 1/2$ or $w^*(Y) \geq 1/2$, contradicting $\max(w^*(X), w^*(Y)) < \frac{1}{2(t'+1)}$.

▷ **Claim 14.** The partition \mathcal{P} has at most $2s(t'+1)\gamma^*$ classes.

Proof. As we explained, we cannot end up with a partition \mathcal{P} at a leaf of the versatile tree. Thus at least $|\mathcal{P}|/s$ disjoint pairs of vertices are t' -contractions in $G_{\mathcal{P}}$. Therefore all these contractions must be forbidden by our restriction imposed on the weights. It follows that at least $|\mathcal{P}|/s$ parts of \mathcal{P} have weight at least $\frac{1}{2(t'+1)}$. Since the sum of all weights in \mathcal{P} is γ^* , it follows that $|\mathcal{P}| \leq 2s(t'+1)\gamma^*$. ◁

▷ **Claim 15.** Let $P \in \mathcal{P}$ be any part. Either $w^*(P) < \frac{1}{t'+1}$ or P is a singleton.

Proof. Let $P \in \mathcal{P}$, and assume that P is not a singleton. Then P has been obtained by contracting two parts P_1, P_2 during the contraction sequence leading to \mathcal{P} . The restriction on the contraction sequence ensures that $w^*(P_1) < \frac{1}{2(t'+1)}$ and $w^*(P_2) < \frac{1}{2(t'+1)}$. Therefore $w^*(P) = w^*(P_1) + w^*(P_2) < \frac{1}{t'+1}$. ◀

Let $D \subseteq V(G)$ be obtained by picking arbitrarily one vertex x_P in each part $P \in \mathcal{P}$. By Claim 14, $|D| \leq 2s(t'+1)\gamma^*$, which is linear in γ^* when t is fixed. Let us prove that D is a dominating set. We let $P \in \mathcal{P}$, and prove that all vertices of P are dominated by D .

Suppose first that there exists $P' \in \mathcal{P}$ such that P, P' is a black edge in $G_{\mathcal{P}}$. Then $x_{P'} \in P'$ is adjacent to all vertices of P , which are thus dominated by D .

Hence we may instead assume that P does not have any black neighbor in $G_{\mathcal{P}}$. Consider any vertex $y \in P$, and let P_1, \dots, P_k the parts of $\mathcal{P} \setminus \{P\}$ such that there exists an edge between y and some vertex of P_i . Then P_1, \dots, P_k are neighbors of P in $G_{\mathcal{P}}$, and must be red neighbors since P has no black neighbor. Since $G_{\mathcal{P}}$ is a t' -trigraph, it follows that $k \leq t'$.

We now claim that one of the parts P, P_1, \dots, P_k must be a singleton. Indeed, since w^* is a fractional dominating set, and since $P \cup \bigcup_{i=1}^k P_i$ contains y and its neighborhood, it must be that $w^*(P) + \sum_{i=1}^k w^*(P_i) \geq 1$. Because $k \leq t'$, it follows that one part among P, P_1, \dots, P_k has weight at least $\frac{1}{t'+1}$. By Claim 15, that same part P_h must be a singleton. Let z be the single vertex in P_h . Necessarily $z \in D$. If this singleton part is P , then $z = y$. Otherwise z is a neighbor of y by definition of P_1, \dots, P_k . In either case y is dominated in D by z . ◀

5.2 A constant approximation for MIS would imply a PTAS

A pessimistic stance on the result of this section is that, perhaps surprisingly, the constant approximation of MIN DOMINATING SET is unlikely to be generalizable to the closely related MIS. We indeed observe that the self-improving reduction of Feige et al. [8] preserves the twin-width. As a consequence a constant approximation for MIS would provide a polynomial-time approximation scheme (PTAS).

► **Theorem 16.** *If MAX INDEPENDENT SET on graphs of twin-width at most d has a constant-approximation algorithm, then it admits a PTAS.*

For G_1 and G_2 two non-empty graphs, and $u \in V(G_1)$, we denote by $G_1(u \leftarrow G_2)$ the substitution in G_1 of u by G_2 . That is, u is replaced by G_2 , and every vertex of $V(G_1) \setminus \{u\}$ initially adjacent to u is made adjacent to the whole $V(G_2)$.

► **Lemma 17.** $tw(G_1(u \leftarrow G_2)) = \max(tw(G_1), tw(G_2))$.

For G a graph, let G^t be the graph on the vertex set $V(G)^t$, such that for $\bar{x} = (x_1, \dots, x_t)$, $\bar{y} = (y_1, \dots, y_t)$ distinct vertices, $\bar{x}\bar{y} \in E(G^t)$ if and only if $x_i y_i \in E(G)$ where i is the smallest index such that $x_i \neq y_i$. This definition can be restated inductively: G^0 is the 1-vertex graph, and G^t is obtained from G by substituting each vertex by a copy of G^{t-1} . With the notations of the initial definition, for $x \in V(G)$, the set of vertices of G^t of the form (x, x_2, \dots, x_t) is a copy isomorphic to G^{t-1} .

The following holds as a direct consequence of Lemma 17.

► **Lemma 18.** *For any graph G and integer $t > 0$, $tw(G^t) = tw(G)$.*

We now show that the independence number of G^t is tightly related to the one of G .

► **Lemma 19.** *For any graph G , both following conditions hold.*

1. *Given any independent set of size k in G , one can compute an independent of size k^t in G^t , in time $O(k^t)$.*
2. *Given any independent set of size k' in G^t , one can compute an independent of size at least $\sqrt[t]{k'}$ in G , in time $O(k')$.*

As an immediate corollary, $\alpha(G^t) = \alpha(G)^t$ where, we recall, $\alpha(H)$ denotes the size of a maximum independent set in H .

Proof of Theorem 16. Assume there is a polynomial-time β -approximation for MIS on graphs of twin-width at most d . Let G be a graph with twin-width at most d . By Lemma 18 the algorithm can be ran on G^t to obtain an independent set of size at least $\frac{\alpha(G^t)}{\beta} = \frac{\alpha(G)^t}{\beta}$. By Lemma 19, this independent set in G^t can be turned into an independent set in G of size at least $\alpha(G)/\sqrt[t]{\beta}$. This gives a polynomial-time $\sqrt[t]{\beta}$ -approximation for arbitrary t . Thus the approximation ratio can be made arbitrarily close to 1. ◀

6 Bounded twin-width classes are χ -bounded

So far, our algorithms use the contraction sequence (or tree) “forward”. This is the original scheme of Guillemot and Marx [11], and of our model checking algorithm [4]. We now see how it can be useful to consider the contraction process “backward”. We start with the case of triangle-free graphs, which will be the base case for the proof of the χ -boundedness.

► **Theorem 20.** *Every triangle-free graph with twin-width at most d is $(d+2)$ -colorable.*

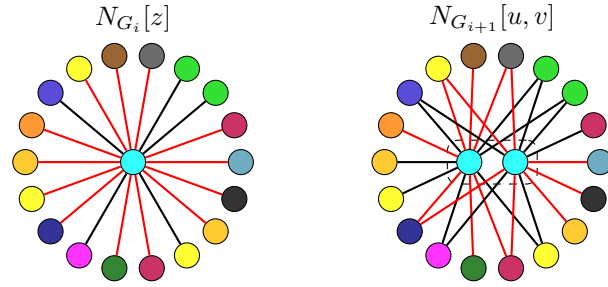
Proof. Let G be an n -vertex triangle-free graph of twin-width at most d , and let $G = G_n, \dots, G_1 = K_1$ be a d -sequence of G . We show how to color G with $d+2$ colors starting from G_1 , and iteratively coloring G_{i+1} based on the coloring of G_i . We give the unique vertex of $G_1 = K_1$ color 1. This defines coloring C_1 . For every i from 1 to $n-1$, let z be the vertex of G_i split into $u, v \in V(G_{i+1})$. In coloring C_{i+1} , every vertex of $V(G_{i+1}) \setminus \{u, v\}$ keeps the color it received by C_i . Vertex u receives color $C_i(z)$. Finally, v receives color $C_i(z)$ if uv is a non-edge in G_{i+1} , and the smallest positive integer *not* appearing in its neighborhood (black and red neighbors) in G_{i+1} , otherwise. We will now show that C_n is a proper coloring of G using at most $d+2$ distinct colors.

We show by induction on i that C_i is a proper $(d+2)$ -coloring of the graph $G'_i := (V(G_i), E(G_i) \cup R(G_i))$. Coloring C_1 is indeed proper in G'_1 and uses $1 \leq d+2$ color. We assume that C_i is a proper $(d+2)$ -coloring of G'_i , and distinguish two cases. If there is a black edge $yz \in E(G_i)$ (recall that z is the vertex split into u, v), then uv has to be a non-edge in G_{i+1} . Otherwise there is at least one edge between $u(G)$ and $v(G)$, and this edge forms a triangle with any vertex in $y(G)$. Thus in that case, $C_{i+1}(u) = C_{i+1}(v) = C_i(z)$. So the number of distinct colors given by C_{i+1} is still at most $d+2$ (see Figure 3).

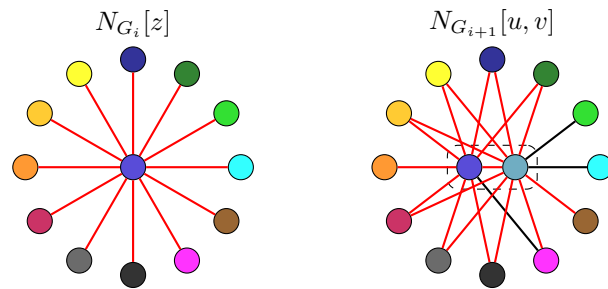
And C_{i+1} is a proper coloring of G'_{i+1} since $N_{G'_{i+1}}(\{u, v\}) = N_{G'_i}(z)$. If instead z has only red neighbors in G_i , then z has at most d neighbors in G'_i . Furthermore let us assume that $uv \in E(G'_{i+1})$, otherwise we conclude as previously. In that case, v is properly colored by C_{i+1} in G'_{i+1} by construction, and vertex u as well, since $N_{G'_{i+1}}(u) \setminus \{v\} \subseteq N_{G'_i}(z)$. Finally $C_{i+1}(v)$ is the smallest positive integer not appearing in a set of at most $d+1$ positive integers. Thus $C_{i+1}(v) \leq d+2$, and C_{i+1} is overall a proper $(d+2)$ -coloring of G'_{i+1} (see Figure 4).

In particular, C_n is a proper $(d+2)$ -coloring of $G'_n = G_n = G$. ◀

As a side note, it is, to our knowledge, possible that every triangle-free K_t -minor free graph has twin-width $O(t)$. If this turns out to be true, it offers a seemingly different approach to getting improved bounds in the triangle-free case of the Hadwiger’s conjecture:



■ **Figure 3** Split, when z is incident to a black edge in G_i . As G is triangle-free, there cannot be an edge (red or black) between u and v . Thus both u and v can take the color of z , which does not appear in their neighborhood.



■ **Figure 4** Split, when z is only incident to red edges. Even if the red neighbors of z have d distinct colors, vertex v can find a color in $[d + 2]$ which avoids these d colors plus the color of z and u .

Instead of trying to color these graphs, one could try to design contraction sequences for them. We now show how to color any K_t -free graph G given with a d -sequence, with at most $(d + 2)^{t-2}$ colors.

► **Theorem 21.** *For every integer $t \geq 3$, every K_t -free graph with twin-width at most d is $(d + 2)^{t-2}$ -colorable.*

Proof. Let G_n, \dots, G_1 be a d -sequence of a K_t -free graph G with $t \geq 3$. In Theorem 20, whenever a vertex $x \in V(G_{i+1})$ was incident to a black edge for the first time (going from G_1 to G_n), the color of all the vertices in $x(G)$ was eventually set to the same value, namely $C_{i+1}(x)$. Now such a set $x(G)$ is not necessarily an independent set, but rather induces a K_{t-1} -free graph. Indeed, a K_{t-1} in $G[x(G)]$ would form a K_t in G with any vertex of $y(G)$, where $xy \in E(G_{i+1})$. By induction on t , we may color $G[x(G)]$ with tuples of at most $t - 3$ integers of $[d + 2]$, and prepends $C_{i+1}(x)$ to these tuples. The base case $t = 3$ is Theorem 20. We make the general idea a bit more precise.

For every $i \in [n]$, we define G_i^* as the *graph* obtained from G_i by blowing every vertex $x \in V(G_i)$ into $G[x(G)]$ whenever x is incident to a black edge, and then turning every red edge into a black edge. We define the successive colorings C'_1, \dots, C'_n of G_1^*, \dots, G_n^* , respectively, following the algorithm of Theorem 20. While there are no black edge in the current trigraph G_i , we set $C'_i := C_i$, where C_i is the coloring in the triangle-free case. Say, at least one black edge appears for the first time in G_{i+1} (this is well-defined since G_n has only black edges). Again we adopt the convention that $z \in V(G_i)$ was split into $u, v \in V(G_{i+1})$. Let S be the set of (at most $d + 2$) vertices with an incident black edge in G_{i+1} . (One may notice that $S \subseteq \{u, v\} \cup N_{G_i}(z)$ and $S \cap \{u, v\} \neq \emptyset$.) Every vertex $w \in V(G_{i+1}) \setminus S$ receives

color $C_{i+1}(w)$. As we observed, for every $x \in S$, $G[x(G)]$ is K_{t-1} -free. By induction there is a coloring C^x of $G[x(G)]$ with tuples of at most $t - 3$ integers from $[d + 2]$. We *permanently* color every vertex $y \in x(G)$ by $(C_{i+1}(x), C^x(y))$. This defines the coloring C'_{i+1} of G_{i+1}^* .

We continue to follow Theorem 20, with the ensuing precisions. We go through all the splits, including the ones between two permanently colored vertices, since they may make some other vertices incident to a black edge for the first time. If the split vertex $z \in V(G_j)$ is *not* such that $z(G)$ was already permanently colored, the colors of the new vertices $u, v \in V(G_{j+1})$ are chosen according to the rules of Theorem 20 where we consider the trigraphs G_j and G_{j+1} (and not the graphs G_j^* and G_{j+1}^*), and the coloring C_j of $V(G_j)$ is defined as: $C_j(y)$ is the first coordinate of $C'_j(y)$ (or $C'_j(y)$ itself if it is not a tuple) if $y \in V(G_j^*)$, and the first coordinate of the color of any vertex in $y(G)$, otherwise. (One may observe that C_j is *not* necessarily a proper coloring of $(V(G_j), E(G_j) \cup R(G_j))$, but all the conflict edges lie within a permanently colored subgraph.) Every time a vertex x becomes incident to a black edge, we permanently color $x(G)$. This defines the sequence of colorings C'_1, \dots, C'_n .

We show by induction on i that C'_i properly colors G_i^* . Coloring C'_1 is indeed a proper coloring of $G_1^* = K_1$. We assume that C'_i is a proper coloring of G_i^* , and let xy be any edge in $E(G_{i+1}^*)$. By the outermost induction on t , if xy lies within a K_{t-1} -free graph permanently colored, then $C'_{i+1}(x) \neq C'_{i+1}(y)$. If instead x and y belong to two distinct vertices of G_{i+1} , by the proof of Theorem 20 and the fact that C'_i is a proper coloring of G_i^* , the first coordinate of $C'_{i+1}(x)$ and of $C'_{i+1}(y)$ differ. In particular C'_n is a proper coloring of $G_n^* = G_n = G$. We pad every tuple $C'_n(x)$ of length $t' < t$ with $t - t'$ entries 1. From the previous proof, it can be observed that this new coloring of G is still proper, and uses at most $(d + 2)^{t-2}$ colors. ◀

Theorem 21 directly implies that, provided $O(1)$ -sequences are given, MIN COLORING can be $2^{O(\text{OPT})}$ -approximated on bounded twin-width graphs, and MAX INDEPENDENT SET can be $O(1)$ -approximated on K_t -free graphs of bounded twin-width. It would be interesting to determine if bounded twin-width classes are polynomially χ -bounded, that is, satisfies for some constant c , $\chi(G) = O(\omega(G)^c)$ for every graph G in the class. Bounded clique-width or rank-width classes were shown polynomially χ -bounded only recently [2]. We show however that bounded twin-width classes satisfy the related *strong Erdős-Hajnal property*. We recall that a class \mathcal{C} of graphs satisfies the *strong Erdős-Hajnal property* if there exists an $\varepsilon > 0$ such that every $G \in \mathcal{C}$ contains two disjoint subsets of vertices X, Y , both of size at least $\varepsilon|V(G)|$, with either all edges or no edges between X and Y . The strong Erdős-Hajnal property of a hereditary class implies the existence of a clique or a stable set of polynomial size, that is, the Erdős-Hajnal property [1].

► **Theorem 22.** *The class of graphs with twin-width at most d satisfies the strong Erdős-Hajnal property with $\varepsilon = 1/(d + 4)$.*

Proof. Let G be an n -vertex graph with twin-width at most d . Consider in a fixed d -sequence G_n, \dots, G_1 the maximum index i such that there is a vertex $z \in V(G_i)$ satisfying $|z(G)| \geq n/(d + 4)$. Since $X := z(G)$ is the union of $u(G)$ and $v(G)$ for some $u, v \in V(G_{i+1})$, its size is at most $2n/(d + 4)$. Vertex z has at most d red neighbors in G_i . These neighbors constitute a set $S \subseteq V(G)$ of at most $d \cdot n/(d + 4)$ vertices. Thus $|V(G) \setminus (z(G) \cup S)| \geq n - 2n/(d + 4) - dn/(d + 4) = 2n/(d + 4)$. By construction, every vertex in $V(G) \setminus (z(G) \cup S)$ is fully adjacent to X or fully non-adjacent to X . Let $Y \subseteq V(G) \setminus (z(G) \cup S)$ be the subset of all vertices in the majority regarding these two outcomes. Set Y has size at least $n/(d + 4)$ vertices and X, Y is therefore an appropriate pair. ◀

References

- 1 Noga Alon, János Pach, Rom Pinchasi, Radoš Radoičić, and Micha Sharir. Crossing patterns of semi-algebraic sets. *Journal of Combinatorial Theory, Series A*, 111(2):310–326, 2005. doi:10.1016/j.jcta.2004.12.008.
- 2 Marthe Bonamy and Michal Pilipczuk. Graphs of bounded cliquewidth are polynomially χ -bounded. *CoRR*, abs/1910.00697, 2019. arXiv:1910.00697.
- 3 Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width II: small classes. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1977–1996, 2021. doi:10.1137/1.9781611976465.118.
- 4 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 601–612. IEEE, 2020. doi:10.1109/FOCS46700.2020.00062.
- 5 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.
- 6 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000. doi:10.1007/s002249910009.
- 7 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633. ACM, 2014. doi:10.1145/2591796.2591884.
- 8 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Approximating Clique is almost NP-complete (preliminary version). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 2–12. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185341.
- 9 Jörg Flum and Martin Grohe. Fixed-parameter tractability, definability, and model-checking. *SIAM J. Comput.*, 31(1):113–145, 2001. doi:10.1137/S0097539799360768.
- 10 Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Log.*, 130(1-3):3–31, 2004. doi:10.1016/j.apal.2004.01.007.
- 11 Sylvain Guillemot and Dániel Marx. Finding small patterns in permutations in linear time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 82–101, 2014. doi:10.1137/1.9781611973402.7.
- 12 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.
- 13 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.
- 14 David S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.*, 9(3):256–278, 1974. doi:10.1016/S0022-0000(74)80044-9.
- 15 László Lovász. On the ratio of optimal integral and fractional covers. *Discret. Math.*, 13(4):383–390, 1975. doi:10.1016/0012-365X(75)90058-8.
- 16 Svatopluk Poljak. A note on stable sets and colorings of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15(2):307–309, 1974.
- 17 Hisao Tamaki. Positive-instance driven dynamic programming for treewidth. *J. Comb. Optim.*, 37(4):1283–1311, 2019. doi:10.1007/s10878-018-0353-z.