Empirical Evaluation of Approximation Algorithms for Generalized Graph Coloring and Uniform Quasi-Wideness

Wojciech Nadara*   Marcin Pilipczuk†   Roman Rabinovich‡   Felix Reidl§   Sebastian Siebertz¶

Abstract

The notions of bounded expansion and nowhere denseness not only offer robust and general definitions of uniform sparseness of graphs, they also describe the tractability boundary for several important algorithmic questions. In this paper we study two structural properties of these graph classes that are of particular importance in this context, namely the property of having bounded generalized coloring numbers and the property of being uniformly quasi-wide. We provide experimental evaluations of several algorithms that approximate these parameters on real-world graphs. On the theoretical side, we provide a new algorithm for uniform quasi-wideness with polynomial size guarantees in graph classes of bounded expansion and show a lower bound indicating that the guarantees of this algorithm are close to optimal in graph classes with fixed excluded minor.

1 Introduction

The exploitation of structural properties found in sparse graphs has a long and fruitful history in the design of efficient algorithms. Besides the long list of results on planar graphs and graphs of bounded degree (which are too numerous to fairly represent here), the celebrated structure theory of graphs with excluded minors, developed by Robertson and Seymour [60] falls into this category. It not only had an immense influence on the design of efficient algorithms (see e.g. [19, 20]) it further introduced the now widely used notion of treewidth (see e.g. [9]) and gave rise to the field of parameterized complexity: “In the beginning, all we did was graph minors” (M. Fellows, pers. comm.). As such, the impact of the theory of sparse graphs on algorithmic research cannot be overstated.

Many of the algorithmic results concerning classes excluding a minor or a topological minor are in some way based on topological arguments, depending on the structure theorems (e.g. decompositions) for the class under consideration. A complete paradigm shift was initiated by Nešetřil and Ossona de Mendez with their foundational work and introduction of the notions of bounded expansion [45, 46, 47] and nowhere denseness [49]. These graph classes extend and properly contain all the aforementioned sparse classes and many arguments based on topology can be replaced by more general, and surprisingly often much simpler, methods...
arguments based on density. We refer to the textbook [50] for extensive background on the theory of sparse graph classes.

The rich structural theory for bounded expansion and nowhere dense graph classes has been successfully applied to design efficient algorithms for hard computational problems on specific sparse classes of graphs, see e.g. [6, 17, 22, 23, 24, 25, 26, 30, 32, 66]. On the other hand, several results indicate that nowhere dense graph classes form a natural limit for algorithmic methods based on sparseness arguments, see e.g. [22, 24].

One core strength of the bounded expansion/nowhere dense framework is that there exists a multitude of equivalent definitions that provide complementing perspectives. Here, we study two structural properties of these classes that are of particular importance in the algorithmic context, namely the property of having bounded generalized coloring numbers and the property of being uniformly quasi-wide. The generalized coloring numbers intuitively measure reachability properties in a linear vertex ordering of a given graph. Such an ordering yields a very weak and local form of a graph decomposition which can be exploited combinatorially [25, 57] and algorithmically [6, 22, 23, 32]. Uniform quasi-wideness was originally introduced in finite model theory [16], and soon found combinatorial and algorithmic applications on nowhere dense classes [17, 25, 30, 37, 48, 55, 63].

Even though the above results render many problems tractable in theory, many of the known algorithms have worst-case running times that involve huge constant factors and combinatorial explosions with respect to the discussed parameters. The central question of our work here is to investigate how the generalized coloring numbers and uniform quasi-wideness behave on real-world graphs, an endeavor which so far has only been conducted for a single notion of bounded expansion and on a smaller scale [21]. Controllable numbers would be a prerequisite for practical implementations of these algorithms based on such structural approaches. We provide an experimental evaluation of several algorithms that approximate these parameters on real world graphs.

On the theoretical side, we provide a new algorithm for uniform quasi-wideness with polynomial size guarantees in graph classes of bounded expansion and show a lower bound indicating that the guarantees of this algorithm are close to optimal in graph classes with fixed excluded minor.

Organization. We give background on the theory of bounded expansion and nowhere dense graphs in Section 2. In Section 3 and Section 4 we describe our approaches to compute the weak coloring numbers and uniform quasi-wideness. Our experimental setup is described in Section 5 and our results are presented in Section 6 and Section 7.

2 Preliminaries

Graphs. All graphs in this paper are finite, undirected and simple, that is, they do not have loops or multiple edges between the same pair of vertices. For a graph $G$, we denote by $V(G)$ the vertex set of $G$ and by $E(G)$ its edge set. The distance between a vertex $v$ and a vertex $w$ is the length (that is, the number of edges) of a shortest path between $v$ and $w$. For a vertex $v$ of $G$, we write $N^G(v)$ for the set of all neighbors of $v$, $N^G(v) = \{ u \in V(G) \mid \{u,v\} \in E(G) \}$, and for $r \in \mathbb{N}$ we denote by $N^G_r[v]$ the closed $r$-neighborhood of $v$, that is, the set of vertices of $G$ at distance at most $r$ from $v$. Note that we always have $v \in N^G_r[v]$. When no confusion can arise regarding the graph $G$ we are considering, we usually omit the superscript $G$. The radius of a connected graph $G$ is the minimum integer $r$ such that there exists $v \in V(G)$ with the property that all vertices of $G$ have distance at most $r$ to $v$. A set $A$ is $r$-independent if all distinct vertices of $A$ have distance greater than $r$.

Bounded expansion and nowhere denseness. A minor model of a graph $H$ in a graph $G$ is a family $(I_u)_{u \in V(H)}$ of pairwise vertex-disjoint connected subgraphs of $G$, called branch sets, such that whenever $uv$ is an edge in $H$, there are $u' \in V(I_u)$ and $v' \in V(I_v)$ for which $u'v'$ is an edge in $G$. The graph $H$ is a depth-$r$ minor of $G$, denoted $H \preceq_r G$, if there is a minor model $(I_u)_{u \in V(H)}$ of $H$ in $G$ such that each $I_u$ has radius at most $r$. A class $\mathcal{C}$ of graphs is nowhere dense if there is a function $t: \mathbb{N} \to \mathbb{N}$ such that for all $r \in \mathbb{N}$ it holds that $K_{t(r)} \not\preceq_r G$ for all $G \in \mathcal{C}$, where $K_{t(r)}$ denotes the clique on $t(r)$ vertices. The class $\mathcal{C}$ has bounded
expansion if there is a function \( d : \mathbb{N} \to \mathbb{N} \) such that for all \( r \in \mathbb{N} \) and all \( H \preceq_r G \) with \( G \in \mathcal{C} \), the edge density of \( H \), i.e., \( |E(H)|/|V(H)| \), is bounded by \( d(r) \). Note that every class of bounded expansion is nowhere dense. The converse is not necessarily true in general [50].

3 The weak coloring numbers

The coloring number \( \text{col}(G) \) of a graph \( G \) is the minimum integer \( k \) such that there is a linear order \( L \) of the vertices of \( G \) for which each vertex \( v \) has back-degree at most \( k - 1 \), i.e., at most \( k - 1 \) neighbors \( u \) with \( u <_L v \). It is well-known that for any graph \( G \), the chromatic number \( \chi(G) \) satisfies \( \chi(G) \leq \text{col}(G) \), which possibly explains the name “coloring number”.

We study a generalization of the coloring number that was introduced by Kierstead and Yang [33] in the context of coloring games and marking games on graphs. The weak coloring numbers \( \text{wcol}_r \) are a series of numbers, parameterized by a positive integer \( r \), which denotes the radius of the considered ordering.

The invariants \( \text{wcol}_r \) are defined in a way similar to the definition of the coloring number. Let \( \Pi(G) \) be the set of all linear orders of the vertices of the graph \( G \), and let \( L \in \Pi(G) \). Let \( u, v \in V(G) \). For a positive integer \( r \), we say that \( u \) is weakly \( r \)-reachable from \( v \) with respect to \( L \), if there exists a path \( P \) of length \( \ell \), \( 0 \leq \ell \leq r \), between \( u \) and \( v \) such that \( u \) is minimum among the vertices of \( P \) (with respect to \( L \)). Let \( \text{WReach}_r[G, L, v] \) be the set of vertices that are weakly \( r \)-reachable from \( v \) with respect to \( L \). Note that \( v \in \text{WReach}_r[G, L, v] \).

Consequently, we also consider an algorithm for computing treedepth in our empirical evaluation.

A related notion to weak coloring numbers are strong coloring numbers, which were also introduced in [33]. Let \( L \in \Pi(G) \), let \( r \) be a positive integer and let \( v \in V(G) \). We say that a vertex \( u \) is strongly \( r \)-reachable from \( v \) if there is a path \( P \) of length \( \ell \), \( 0 \leq \ell \leq r \) such that \( u = v \) or \( u \) is the only vertex of \( P \) smaller than \( v \) (with respect to \( L \)). Let \( \text{SReach}_r[G, L, v] \) be the set of vertices that are strongly \( r \)-reachable from \( v \) with respect to \( L \). Again, \( v \in \text{SReach}_r[G, L, v] \). The strong \( r \)-coloring number \( \text{col}_r(G) \) is defined as \( \text{col}_r(G) := \min_{L \in \Pi(G)} \max_{v \in V(G)} |\text{SReach}_r[G, L, v]| \).

The reason is that treewidth of \( G \) can be characterized by the minimal width of an elimination ordering of \( G \) defined exactly as \( \text{col}_\infty(G) \).

Clearly, for all \( r \in \mathbb{N} \), \( \text{col}_r(G) \leq \text{wcol}_r(G) \) (and thus \( \text{tw}(G) \leq \text{td}(G) \)). Moreover, for all \( r \) we have \( \text{wcol}_r(G) \leq (\text{col}_r(G))^{r'} \) [33]. It follows that for every graph \( G \) there is some (possibly large) integer \( r \) such that \( \text{wcol}_{r-1}(G) \leq \text{tw}(G) \leq \text{wcol}_{r}(G) \). This gives a hope that an elimination ordering computed for treewidth gives a good upper bound for \( \text{wcol}_r(G) \) where \( r' \leq r - 1 \). We will evaluate orders produced by an algorithm for treewidth approximations, but interpreted as an order for weak coloring numbers.

Concrete bounds for the weak coloring numbers on restricted graph classes are given in [29, 36, 45, 56, 65, 70]. Our approximation algorithms are based on the approaches described in [45, 56, 65], which we describe in more detail in the following subsections.
3.1 Distance-constrained Transitive Fraternal Augmentations

Given a graph $G$ and a linear order $L$ of its vertices, observe that we have the following properties:

1. Let $u, v, w \in V(G)$ be such that $v \in \text{WReach}_i(G, L, u)$ and $w \in \text{WReach}_j(G, L, u)$ for some numbers $i, j$ with $i + j \leq r$. Then either $v \in \text{WReach}_{i+j}(G, L, w)$ or $w \in \text{WReach}_{i+j}(G, L, v)$.

2. Let $u, v, w \in V(G)$ be such that $u \in \text{WReach}_i(G, L, v)$ and $v \in \text{WReach}_j(G, L, w)$ for some numbers $i, j$ with $i + j \leq r$. Then $u \in \text{WReach}_{i+j}(G, L, w)$.

We can approximate the weak coloring numbers by orienting the input graph $G$ which was introduced in [56] as a more practical variant of transitive-fraternal augmentations. Introducing an arc with the aim of satisfying property 1 above is called a fracternal augmentation, while introducing an arc with the aim of satisfying property 2 is called a transitive augmentation. These operations were studied first in [46]. We are going to work with an optimized version, called distance-constrained transitive-fraternal augmentations, short dtf-augmentations, which was introduced in [56] as a more practical variant of transitive-fraternal augmentations.

Let $G$ be an undirected graph and let $\vec{G}_1$ be any orientation of $G$. Then a dtf-augmentation of $G$ is a sequence $\vec{G}_1 \subseteq \vec{G}_2 \subseteq \ldots$ of directed graphs which satisfy the following two constraints:

1. Let $u, v, w \in V(G)$ be such that $w \in E(\vec{G}_i)$ and $uw \in E(\vec{G}_j)$ are arcs of $\vec{G}_i$ and $\vec{G}_j$, respectively. Then it follows that either $vw \in \vec{G}_{i+j}$ or $wv \in \vec{G}_{i+j}$.

2. Let $u, v, w \in V(G)$ be such that $uv \in E(\vec{G}_i)$ and $uw \in E(\vec{G}_j)$ are arcs of $\vec{G}_i$ and $\vec{G}_j$, respectively. Then it follows that $wu \in \vec{G}_{i+j}$.

Just as above, arcs added because of the first item are called fraternal and arcs added because of the second item are called transitive. To simplify notation we associate a weight function $\omega_i: V(\vec{G}_i)^2 \to \{1, \ldots, \} \cup \{\infty\}$ with the $i$-th dtf-augmentation $\vec{G}_i$ where $w_1(uv) = 1$ if $uv \in E(\vec{G}_1)$ and $w_1(uv) = \infty$ if $uv \notin E(\vec{G}_1)$ and

$$
\omega_i(uv) = \begin{cases} 
\min\{\omega_{i-1}(uv), i\} & \text{if } uv \in \vec{G}_i \\
\infty & \text{else}
\end{cases}
$$

In other words: if the arc $uv$ is present in $\vec{G}_i$ but not in $\vec{G}_{i-1}$, then we have $\omega_{i-1}(uv) = i$ and $\omega_{i-1}(uv) = \infty$.

It can be shown that the arcs of weight $d$ appear exactly in augmentation $\vec{G}_d$. These augmentations behave similarly to graph powers in the following sense: consider two vertices $u, v$ that are at distance $d$ in $G$. Then in every augmentation $\vec{G}_r$ for $r \geq d$ we either find the arc $uv \in \vec{G}_d$ with $\omega_d(uv) = d$, or the the arc $vu \in \vec{G}_d$ with $\omega_d(vu) = d$, or we find a common out-neighbor $w$ of $u$ and $v$ in $\vec{G}_d$ such that $\omega_r(uw) + \omega_r(vw) = d$.

Importantly, graph classes of bounded expansion admit dtf-augmentations in which the maximum out-degree $\Delta^+(\vec{G}_r)$ is bounded by a function of depth $r$ and the graph class in question [56] (we remark that commonly in the literature one orients the graphs $\vec{G}_i$ to minimize in-degrees instead of out-degrees, however, for consistency with the weak coloring numbers we orient so that an arc $uv \in E(\vec{G}_i)$ corresponds to $u \in \text{WReach}_i(G, L, v)$). The algorithm to compute such augmentations closely follows the original algorithm for tf-augmentations (described in [46, 50]): first, the orientation $\vec{G}_1$ is chosen to be the acyclic ordering derived from the degeneracy ordering of $G$; this orientation minimizes $\Delta^+(\vec{G}_1)$. Second, we can orient the fraternal arcs added in step $r$ by first collecting all potential fraternal edges in an auxiliary graph $\vec{G}_f^t$ and then again compute an acyclic orientation $\vec{G}_f^r$ which minimizes the out-degree. We then insert the arcs into $\vec{G}_r$ according to their orientation in $\vec{G}_f^r$.

If instead of computing fraternal edges at step $r$ by searching for fraternal configurations in all pairs $\vec{G}_i, \vec{G}_j$ with $i + j = r$, it suffices to consider the pair $\vec{G}_{r-1}, \vec{G}_1$. The same optimization does not hold for transitive arcs, however.
The precise connection between dtf-augmentations and wcol-orderings is presented in the following lemma.

**Lemma 3.1** ([6, 30]). Let $\tilde{G}_r$ be the $r$-th dtf-augmentation of a graph $G$ and let $G_r$ be the underlying undirected graph. Let $L$ be an ordering of $V(G)$ such that every vertex has at most $c$ smaller neighbours with respect to $L$. Then $\text{WReach}_r(G, L, v) \leq (\Delta^+(\tilde{G}_r) + 1)c + 1$ for all $v \in G$.

Therefore we can obtain a wcol-ordering from the $r$th dtf-augmentation $\tilde{G}_r$ by simply computing a degeneracy ordering of $G_r$.

### 3.2 Flat decompositions

The following approach for approximating the weak coloring numbers was introduced in [65] and provably yields good results on graphs that exclude a fixed minor.

A *decomposition* of a graph $G$ is a sequence $\mathcal{H} = (H_1, \ldots, H_\ell)$ of non-empty subgraphs of $G$ such that the vertex sets $V(H_1), \ldots, V(H_\ell)$ partition $V(G)$. The decomposition $\mathcal{H}$ is *connected* if each $H_i$ is connected.

A decomposition of a graph $G$ induces a partial order on $V(G)$ by defining $u < v$ if $u \in V(H_i)$ and $v \in V(H_j)$ for $i < j$. A decomposition yields a good order for the weak coloring numbers if we can

1. guarantee that the $r$-neighborhood of each $v \in V(H_j)$ has a small intersection with $\bigcup_{i<j} V(H_i)$ (then, in particular, $\text{WReach}_r(G, L, v) \cap \bigcup_{i<j} V(H_i)$ is small), and

2. ensure that we can order the vertices inside each $H_i$ so that we have good weak reachability properties.

We call such a decomposition *flat*. The following procedure was proposed in [65] to compute a decomposition of a graph $G$. If $G$ excludes the complete graph $K_t$ as a minor, the resulting decomposition is flat. For a decomposition $(H_1, \ldots, H_\ell)$ of a graph $G$ and $1 \leq i \leq \ell$, we denote by $G[H_{\geq i}]$ the subgraph of $G$ induced by $\bigcup_{i \leq j \leq \ell} V(H_j)$.

Without loss of generality we may assume that $G$ is connected. We iteratively construct a connected decomposition $H_1, \ldots, H_\ell$ of $G$. To start, we choose an arbitrary vertex $v \in V(G)$ and let $H_1$ be the connected subgraph $G[v]$. Now assume that for some $q$, $1 \leq q \leq \ell - 1$, the sequence $H_1, \ldots, H_q$ has already been constructed. Fix some component $C$ of $G[H_{\geq (q+1)}]$ and denote by $Q_1, \ldots, Q_s \in \{H_1, \ldots, H_q\}$ the subgraphs that have a connection to $C$. Using that $K_t$ is excluded as a minor, one may argue that $s \leq t - 2$. Because $G$ is connected, we have $s \geq 1$. Let $v$ be a vertex of $C$ and let $T$ be a breadth-first search tree in $G[C]$ with root $v$. We choose $H_{q+1}$ to be a minimal connected subgraph of $T$ that contains $v$ and that contains for each $i$, $1 \leq i \leq s$, at least one neighbor of $Q_i$. As shown in [65], if $K_t \not\subseteq G$, then the above procedure produces a linear order $L$ that certifies that $\text{wcol}_r(G) \in O(r^{t-1})$.

Observe that this procedure leaves some freedom on how to pick the vertex $v$ of $C$ from which we start the breadth-first search and in which order to insert the vertices of $H_i$. We evaluate several options. For the choice of the root vertex, the following choices seem reasonable.

1. Choose a vertex that is maximizing the number of neighbors in some $Q_i$, to possibly obtain a set $H_{q+1}$ that is smaller than when we choose a vertex far from all $Q_i$.

2. Choose a vertex that has maximum degree in $C$, high degree vertices should be low in the order.

3. Choose a vertex that has maximum degree in $C$, but only among those that are adjacent to some $Q_i$.

For the order of the vertices of $H_i$, we check the following options.

1. The breadth-first search and the depth-first search order from the root.

2. Sorted by degrees, nonincreasingly.

3. Each of the above, but reversed.
3.3 Treedepth heuristic

Since the ‘limit’ of weak-coloring numbers is exactly the treedepth of a graph, i.e. $wcol_\infty(G) = td(G)$, we consider simply computing a treedepth decomposition and using an ordering derived from the decomposition. Our algorithm of choice, developed by Sanchez [62] and implemented by Oelschlägel [54], recursively extracts separators from the graph. To minimize the search space, only close separators are considered, that is, separators $S$ that lie in the closed neighborhood of some vertex. Furthermore, the algorithm makes use of the following proposition.

**Proposition 3.2 ([8]).** If $S \subseteq G$ is a minimal separator of a graph $G$ and $x \in S$, then for each connected component $C$ of $G - (S \cup N(x))$ the set $N(C)$ is a minimal separator of $G$.

Let $N_S(G)$ be the set of minimal separators that can be constructed from a minimal separator $S$ by applying the above proposition, where $S$ is an arbitrary minimal close separator. The algorithm then finds the separator $S_0 \in N_S(G)$ which minimizes the size of the largest connected component in $G - S_0$ (the implementation supports other heuristics, but this heuristic turned out to have an acceptable running time for the large instances).

3.4 Treewidth heuristic

A well-known approach to compute a treewidth decomposition of a graph is to find a linear order of the vertices, an elimination order, of possibly small maximum back-degree. From such an order it is easy to construct a tree decomposition of width equal to the back-degree (see, e.g. [10]). Let $L \in \Pi(G)$ and let $v \in V(G)$. The back-degree of $v$ is defined as

$$bd(v, G) := |S\text{Reach}_\infty[G, L, v]|.$$

There is a number of heuristics to produce good elimination orders. We chose one that is simple, fast and that gives rather good results for treewidth: the so-called minimum-degree heuristic [10].

The minimum-degree algorithm orders the vertices of the graphs starting from the biggest vertex which is one with minimum degree. Assume that we already ordered vertices with indices greater than $i$, we put on position $i$ a vertex with the least back-degree.

3.5 Sorting by degrees and other heuristics

Apart from algorithms with theoretical guarantees we also compared several naive heuristics.

- For $r = 1$ an optimal order is a degeneracy order, which can be easily computed. We can check if this order produces reasonable results for higher values of $r$ as well.

- Intuitively, it makes sense to sort vertices by descending degree (ties are broken arbitrarily) because from vertices of high degree more vertices can be reached in one step. This intuition is further supported by one popular network model, the Chung–Lu random graphs which sample graphs with a fixed degree distribution and successfully replicate several statistics exhibited by real-world networks [13, 14]. In this model, vertices are assigned weights (corresponding to their expected degree) and edges are sampled independently but biased according to the endpoints weights. Under this model, vertices of the same degree are exchangeable and the one ordering we can choose to minimize the number of $r$-reachable vertices is simply the descending degree ordering.

- A simple idea of generalizing the above heuristics to bigger values of $r$ is to apply them to the $r$th power $G^r$ of $G$, i.e. $G^r$ is defined as the graph with $V(G^r) = V(G)$ and $uv \in E(G^r) \iff \text{dist}_G(u, v) \leq r$.

- As a baseline we also included random ordering of vertices.
3.6 Local search

In addition to all these approaches we can try to improve their results by local search, a technique where we make small changes to a candidate solution. We applied the following local changes and tested whether they caused improvements to the current order $L$.

- Take any vertex $v$ that has biggest $\text{WReach}_r[G, L, v]$ and swap it with a random vertex that is smaller with respect to $L$.
- Take any vertex $v$ that has biggest $\text{WReach}_r[G, L, v]$ and swap it with its direct predecessor $u$ in $L$.

Both heuristics try to place a vertex with many weakly reachable vertices to the left of them and thus to make them non-weakly reachable. The advantage of the second rule is that the only possible changes are that $\text{WReach}_r[G, L, v]$ loses $u$ (if $u$ was there) and that $\text{WReach}_r[G, L, u]$ may obtain $v$. So $\text{WReach}_r[G, L, v]$ is trivial to recompute and the only computationally heavy update is for the new $\text{WReach}_r[G, L, u]$. For the first rule, recomputing $\text{WReach}$ sets is more expensive. However, the disadvantage of the second rule is that it does not lead to further improvements quickly, hence applications of only the first rule give better results than applications of the second rule only. In our implementation we did a few optimizations in order to improve the results of second rule, but we refrain from describing them in detail. Final algorithm conducting local search firstly performs round of applications of first rule and when they no longer improve results it performs round of applications of second rule. Such combination turned out to be empirically most effective.

4 Uniform quasi wideness

Intuitively, a class of graphs is wide if for every graph $G$ from the class, every radius $r \in \mathbb{N}$ and every large subset $A \subseteq V(G)$ of vertices one can find a large subset $B \subseteq A$ of vertices which are pairwise at distance greater than $r$ (recall that such a subset is called $r$-independent). The notion of uniform quasi-wideness allows to additionally delete a small number of vertices to make $B$ $r$-independent. The following definition formalises the meaning of “large” and “small”.

**Definition 4.1.** A class $C$ of graphs is uniformly quasi-wide if for every $m \in \mathbb{N}$ and every $r \in \mathbb{N}$ there exist numbers $N(m, r)$ and $s(r)$ such that the following holds.

Let $G \in C$ and let $A \subseteq V(G)$ with $|A| \geq N(m, r)$. Then there exists a set $S \subseteq V(G)$ with $|S| \leq s(r)$ and a set $B \subseteq A \setminus S$ of size at least $m$ such that for all distinct $u, v \in B$ we have $\text{dist}_{G-S}(u, v) > r$.

Uniform quasi-wideness was introduced by Dawar in [16] and it was proved by Nešetřil and Ossona de Mendez in [48] that uniform quasi-wideness is equivalent to nowhere denseness. Very recently, it was shown that the function $N$ in the above definition can be chosen to be polynomial in $m$ [37, 55]. A single exponential dependency was earlier established for classes of bounded expansion [36]. We are going to evaluate the algorithms derived from the proofs in [36, 55], as well as a new algorithm that is streamlined for bounded expansion classes and also achieves polynomial bounds in $m$. We discuss these algorithms in more detail next. We will prove in Section 8 that the bounds of our new algorithm are close to optimal.

4.1 Distance trees

We first describe the algorithm that was introduced in [55]. For simplicity, we focus on the case $r = 2$. First, observe that every graph from a nowhere dense class contains large independent sets. By definition of a nowhere dense class, some complete graph $K_t$ is excluded as a depth-0 minor, that is, simply as a subgraph. Hence, Ramsey’s Theorem immediately implies that if we consider any set $A \subseteq V(G)$ of size at least $\binom{t+m-2}{m-1}$, then there exists a set $B \subseteq A$ of size $m$ which is independent (without deleting any elements). Furthermore, the proof of Ramsey’s Theorem yielding this bound is constructive and can easily be implemented. The
difficult part is now to find in a large independent set a large 2-independent set, possibly after deleting a few elements (consider a star to see that deletion may be necessary).

Assume now that \( A \) is a large independent set. The idea is to arrange the elements of \( A \) in a binary tree \( T \), which we call a distance tree, and prove that this tree contains a long path. From this path the set \( B \) is extracted.

We identify the nodes of \( T \) with words over the alphabet \( \{0,1\} \), where \( \epsilon \) corresponds to the root, and where for a word \( w \) the word \( w0 \) is its left and the word \( w1 \) is its right successor. Fix some enumeration of the set \( A \). We define \( T \) by processing the elements of \( A \) sequentially according to the enumeration. We start with the tree that has its root labeled with the first element of \( A \). For each remaining element \( a \in A \) we execute the following procedure which results in adding a node with label \( a \) to \( T \).

When processing the vertex \( a \), do the following. Start with \( w \) being the empty word. While \( w \) is a node of \( T \), repeat the following step: if the distance from \( a \) to the vertex \( b \) which is at the position corresponding to \( w \) in \( T \) is at most 2, replace \( w \) by \( w0 \), otherwise, replace \( w \) by \( w1 \). Once \( w \) does not correspond to a node of \( T \), extend \( T \) by adding the node corresponding to \( w \) and label it with \( a \). In this way, we have processed the element \( a \), and now proceed to the next element of \( A \) until all elements are processed. This completes the construction of \( T \). Thus, \( T \) is a tree labeled with vertices of \( A \), and every vertex of \( A \) appears exactly once in \( T \).

Now, based on the fact that some complete graph \( K_i \) is excluded as a depth-2 minor of \( G \), it is shown that \( T \) contains a long path. This path either has many left branches or many right branches. Take a subpath that has only left branches or only right branches. Such a path corresponds to a set \( X \) such that all elements have pairwise distance 2, or all elements have pairwise distance greater than 2, that is, to a 2-independent set. In the second case, we have found the set \( B \) that we are looking for. In the other case, we proceed to show that there must exist an element \( w \in V(G) \) that is adjacent to many elements of \( X \), i.e., \( N(w) \cap X \) is large. We add the vertex \( w \) to the set \( S \) of elements to delete and repeat the above tree-classification procedure with the set \( A' = N(w) \cap X \). It is shown that this process must stop after at most \( t \) steps and yields a set \( B \) which is 2-independent in \( G - S \).

The general case reduces to the case \( r = 1 \) or \( r = 2 \) if instead of starting with an independent set \( A \) we start with an \( i \)-independent set \( A_i \) and contract the disjoint \( i/2 \) or \((i + 1)/2\)-neighborhoods of the elements of \( A_i \), respectively, to single vertices. Then one iteratively finds \( i \)-independent sets \( A_1, A_2, \ldots, A_r \) for larger and larger radii.

### 4.1.1 Implementation details

We have implemented three variants of the above method, which we denote \( \text{tree1}, \text{tree2} \) and \( \text{ld_it} \). In all variants, we get a graph \( G \), a vertex subset \( A \subseteq V(G) \) and \( r \in \mathbb{N} \) as input. We do not have the number \( m \) as input but we aim to find an \( r \)-independent subset \( B \subseteq A \) which is as large as possible while deleting as few elements as possible.

For the odd cases (which reduce to \( r = 1 \) in the description above), in each variant we use a simple heuristic for finding independent sets described in Section 4.4.

For more interesting even cases (which reduce to \( r = 2 \) in the description above), \( \text{tree2} \) computes a set of candidate solutions \((C, S)\). Here, \( C \) is a set which corresponds to a long path in the distance tree and \( S \) is the set of vertices removed so far (for this set \( C \)). At every step we compute one candidate solution \((C, S)\), remove a vertex \( w \), i.e. move it to \( S \), which has largest intersection \(|N(w) \cap A|\) and continue the process with \( N(w) \cap A \) until \( A \) becomes too small. In the end, we output the best solution from the pool of collected solutions.

In the version denoted by \( \text{tree1} \), we modify \( \text{tree2} \) as follows. We let \( C \) be a candidate for a large 2-independent set, which however, we do not choose as a subset of the currently handled set \( A \), but of the original input set \( A \). That is, we re-classify all distances of elements of the initial set \( A \) in a distance tree with vertices \( S \) that were deleted in later steps, to draw the candidate 2-independent set from a larger pool of vertices.

Finally, in the \( \text{ld_it} \) version (least degree iterated) we do not find 2-independent sets based on the distance tree, but rather in a simple greedy manner as an independent set in the graph \((G - S)^2[A]\).
4.2 Weak coloring numbers and uniform quasi-wideness

We now describe the approach of [36] which is designed for classes of bounded expansion and combines the weak coloring numbers with uniform quasi-wideness.

Let $G$ be a graph, $A \subseteq V(G)$ and $m, r \in \mathbb{N}$ be given. First, fix some order $L \in \Pi(G)$ such that \(\text{WReach}_r[G, L, v] \leq c\) for every $v \in V(G)$ (for some constant $c$). Let $H$ be the graph with vertex set $V(G)$, where we put an edge $uv \in E(H)$ if and only if $u \in \text{WReach}_r[G, L, v]$ or $v \in \text{WReach}_r[G, L, u]$. Then $L$ certifies that $H$ is $c$-degenerate, and hence, assuming that $|A| \geq (e+1) \cdot 2^m$, we can greedily find an independent set $I \subseteq A$ of size $2^m$ in $H$. By the definition of the graph $H$, we have that $\text{WReach}_r[G, L, v] \cap I = \{v\}$ for each $v \in I$. Now observe that for $v \in I$, deleting $\text{WReach}_r[G, L, v] \setminus \{v\}$ from $G$ leaves $v$ at a distance greater than $r$ (in $G - (\text{WReach}_r[G, L, v] \setminus \{v\})$) from all the other vertices of $I$.

Based on this observation, one follows the simple approach also used to prove Ramsey’s Theorem with exponential bounds. For each vertex $v$ of $I$ (in decreasing order, starting with the largest vertex with respect to $L$), we test whether $v$ is connected by a path of length at most $r$ to more than half of the remaining vertices of $I$. If this is the case, we delete the set $\text{WReach}_r[G, L, v]$ from $G$ (i.e. add it to $S$) and add the vertex $v$ to the set $B$. We continue with the subset of $I$ that had such a connection to $v$ (which is now separated by the deletion of $S$ though). Otherwise, $v$ is not connected to more than half of the remaining vertices of $I$, in which case we simply add $v$ to $B$ and do not delete anything. In this case, we continue the construction with those vertices of $I$ that are not connected to $v$. It is proved that the first case can happen at most $\text{wcol}_r(G) \leq c$ many times, hence, in total we delete at most $c^2$ vertices and arrive at a set $B$ with $m$ vertices that are pairwise at distance greater than $r$ in $G - S$.

We have implemented exactly the algorithm outlined above. We denote it by \texttt{mfcS}.

4.3 A new algorithm

Motivated by the rather conservative character of the algorithm of [36] described above, we propose here a new algorithm (albeit inspired by [36]). Furthermore, in Section 8 we show an almost tight lower bound for the guarantees of this algorithm in graphs excluding a fixed minor.

More formally, we show the following theorem.

**Theorem 4.2.** Assume we are given a graph $G$, a set $A \subseteq V(G)$, integers $r \geq 1$ and $m \geq 2$, and an ordering $L$ of $V(G)$ with $c = \max_{v \in V(G)} |\text{WReach}_r[G, L, v]|$. Furthermore, assume that $|A| \geq 4 \cdot (2cm)^r$. Then in polynomial time, one can compute sets $S \subseteq V(G)$ and $B \subseteq A \setminus S$ such that $|S| \leq c$, $|B| \geq m$, and $B$ is $r$-independent in $G - S$.

**Proof.** The algorithm iteratively constructs sets $A = A_0 \supseteq A_1 \supseteq A_2 \supseteq \ldots$, $\emptyset = S_0 \subseteq S_1 \subseteq S_2 \subseteq \ldots$, and $\emptyset = B_0 \subseteq B_1 \subseteq B_2 \subseteq \ldots$, maintaining the following invariants in every step $i$: $B_i \subseteq A \setminus S_i$, the set $B_i$ is an $r$-independent set in $G - S_i$, and every vertex of $A_i$ is within distance greater than $r$ from every vertex in $B_i$ in the graph $G - S_i$.

At step $i$, given $A_i$, $S_i$, and $B_i$, the algorithm proceeds as follows.

**(growth step)** If $|A_i| \leq 2cm$ or there exists $v \in A_i$ such that at most $|A_i|/m$ vertices of $A_i$ are within distance at most $r$ from $v$ in $G - S_i$ (i.e., $|N^G_{r-S_i}[v] \cap A_i| \leq |A_i|/m$), then move $v$ to $B_{i+1}$ and delete the conflicting vertices from $A_i$, that is set

$$A_{i+1} = A_i \setminus N^G_{r-S_i}[v]$$

$$S_{i+1} = S_i$$

$$B_{i+1} = B_i \cup \{v\}.$$ 

**(deletion step)** Otherwise, pick a vertex $z \in V(G) \setminus S_i$ that appears in a maximum number of weakly reachable sets of vertices of $A_i$. That is, pick $z \in V(G) \setminus S_i$ maximizing the quantity

$$|\{v \in A_i \mid z \in \text{WReach}_r[G, L, v]\}|.$$
Insert $z$ into $S_{i+1}$ and restrict $A_i$ to vertices containing $z$ in their weak reachable sets. More formally,

$$A_{i+1} = \{v \in A_i \mid z \in \text{WReach}_{r}(G, L, v) \setminus \{z\} \}$$

$$S_{i+1} = S_i \cup \{z\}$$

$$B_{i+1} = B_i.$$

The algorithm stops when $A_i$ becomes empty, and then returns $S = S_i$ and $B = B_i$.

Let us now analyze the algorithm. The fact that in the growth step we remove from $A_{i+1}$ the vertices of $A_i$ that are within distance at most $r$ from $v$ preserves the invariant that the distance between $A_i$ and $B_i$ in $G - S_i$ is greater than $r$. This invariant, in turn, proves that $B_i$ is an $r$-independent set in $G - S_i$. It remains to show the bounds on the sizes of $S$ and $B$. To this end, we show the following two claims.

**Claim 4.3.** At every step $i$, for every $z \in S_i$ and $v \in A_i$, we have that $z \in \text{WReach}_{r}(G, L, v)$.

**Proof.** The claim follows directly from the fact that in the deletion step, we restrict $A_{i+1}$ to the set of those vertices of $A_i$ that have $z$ in their weak reachability set.

**Claim 4.4.** At every step $i$, if the is no vertex $v \in A_i$ with $|N_{G^c}^{G^c-S_i}[v] \cap A_i| \leq |A_i|/m$, then there exists $z \in V(G) \setminus S_i$ with at least $|A_i|/(cm)$ vertices $v \in A_i$ satisfying $z \in \text{WReach}_{r}(G, L, v)$.

**Proof.** Let $v \in A_i$ be the least vertex of $A_i$ in the ordering $L$. Since the growth step is not applicable, we have that the set $X := N_{G^c}^{G^c-S_i}[v] \cap A_i$ is of size at least $|A_i|/m$. For every $x \in X$, fix a path $P_x$ of length at most $r$ between $x$ and $v$ in $G - S_i$, and let $z_x$ be the $L$-minimal vertex on this path. The subpath of $P_x$ from $z_x$ to $v$ shows that $z_x \in \text{WReach}_{r}(G, L, v)$ and the subpath of $P_x$ from $z_x$ to $x$ shows that $z_x \in \text{WReach}_{r}(G, L, x)$. Since $|\text{WReach}_{r}(G, L, v)| \leq c$, while $|X| > |A_i|/m$, there exists $z \in \text{WReach}_{r}(G, L, v)$ with

$$|\{v \in A_i \mid z \in \text{WReach}_{r}(G, L, v)\}| \geq \frac{|X|}{|\text{WReach}_{r}(G, L, v)|} \geq \frac{|A_i|}{cm}.$$

This finishes the proof of the claim.

Consequently, when the algorithm executes the deletion step, we have $|A_{i+1}| \geq |A_i|/(cm) - 1$ (the $-1$ comes from the case $z \in A_i$).

In particular, we have that the last step of the algorithm is the growth step: the deletion step executes only if $|A_i| > 2cm$, and then $|A_{i+1}| \geq |A_i|/(cm) - 1 > 1$. Let $v$ be the vertex added to $B_{i+1}$ in this last growth step. Then we have that $S = S_{i+1} = S_i \subseteq \text{WReach}_{r}(G, L, v)$. Consequently, the algorithm executed at most $c$ deletion steps and $|S| \leq c$.

For the bound on the size of set $B$, let $i$ be the minimum index with $|A_i| \leq 2cm$. For every $0 \leq j < i$ that executed a deletion step, we have

$$|A_{j+1}| \geq \frac{|A_j|}{cm} - 1 \geq \frac{|A_j|}{2cm}.$$

For every $0 \leq j < i$ that executed a growth step, we have

$$|A_{j+1}| \geq |A_j| - \frac{|A_j|}{m} = \left(1 - \frac{1}{m}\right)|A_j|.$$

In particular, we have $A_i \neq \emptyset$ due to $m \geq 2$. Consequently, since the algorithm executed $|S_i|$ deletion steps and $|B_i|$ growth steps, we have

$$1 \leq |A_i| \leq |A| \cdot \left(1 - \frac{1}{m}\right)^{|B_i|} \cdot \left(2cm\right)^{-|S_i|}.$$

Hence, since $(1 - 1/m)^m \geq 1/4$ for every $m \geq 2$, if $|A| \geq 4 \cdot (2cm)^c$, then we have $|B_i| \geq m$. This finishes the proof. □
The actual implementation of the above algorithm differs in a number of aspects. First, we found the threshold $|A_i|/m$ for the distinction between the growth step and the deletion step too small in practice, despite working well in the proof above. Moreover, experiments with this algorithm showed that it is unstable in the sense that small changes in this threshold can trigger big changes in produced result which are, a priori, hard to predict. Because of that our implementation has a fixed constant $k$ and executes the above algorithm with thresholds $\frac{1}{k+1}, \frac{2}{k+1}, \ldots, \frac{k}{k+1}$ and chooses the best result (we will address comparing different results later).

Second, the above algorithm can be modified so that the growth step is applied only in cases where least vertex of $A_i$ with respect to $L$ has only a small number of conflicts, in which case we use that first vertex to enlarge $B$. Note that such an algorithm also satisfies the theorem, because in the analysis of the algorithm we used only the fact that if the growth step is not applicable, then this condition is not satisfied for the first vertex of $A_i$. Such a variant is present in our implementation.

Third, in the proof above, the algorithm always applies the growth step when the size of $A_i$ drops below the threshold $2cm$. This is a minor technical detail, and can be omitted at the cost of some more hassle in the proof (in the analysis of the last steps of the algorithm) and somewhat worse bounds for $|S|$ and $|A|$. In the implementation, we do not have this threshold, but instead we roll back the unnecessary deletion steps that were performed by the algorithm near the end of the execution. It is straightforward (but a bit more tedious) to adapt the above analysis to this variant.

### 4.3.1 Implementation details

We have implemented three variants of the above described method, which we denote new1, new2 and new1d. In the outlined algorithm, when we consider a vertex $v$, we compute the set of vertices from $A$ conflicting with $v$. In new1, we consider two vertices to be conflicting if their WReach$_v$ sets intersect. In new2 and new1d, two vertices are considered to be conflicting if the distance between them in the remaining part of the graph is at most $r$. Moreover, new1d after every step tries to fill its partial solution with the heuristic described in Section 4.4 to find independent set in $(G - S)^r \cap A$, where $S$ is a set of already removed vertices.

### 4.4 Other naive approaches and heuristic optimizations

Since uniform quasi-widthness for $r = 1$ is exactly finding independent sets, it makes sense to include heuristics for finding independent sets as a baseline. Moreover the problem of finding independent sets is also used as a subroutine in the approach based on distance trees. We used the following simple greedy algorithm to find independent sets. As long as our graph is nonempty, take any vertex that has the smallest degree, add it to the independent set and remove it and its neighbors from the graph.

The following algorithm is what we came up with as a naive but reasonable heuristic for larger values of $r$. For every number $k \in \{0, 1, \ldots, K\}$ (where $K$ is some hardcoded constant) computes the biggest independent set in graph $(G - S_k)^r[A]$ using the greedy procedure described above, where $S_k$ is a set of $k$ vertices with biggest degrees. This heuristic is based on the fact that independent sets in $G^r$ correspond to $r$-independent sets in $G$. Without any other knowledge about the graph, vertices with the biggest degree seem to be the best candidates to be removed. In the end, we output the best solution obtained in this manner. In the following, we abbreviate this approach as 1d (least degree on power graph).

### 4.5 Comparing different results

Uniform quasi-widthness is a two-dimensional measure: we have to measure both the size $m$ of the $r$-independent set $B$ which we desire to find, as well as the size $s(r)$ of vertices to be deleted. In order to compare the performance of our studied methods we propose the following approach that arises from applications of uniform quasi-widthness in several algorithms [17, 22, 55, 63]. Let $G, A \subseteq V(G), r \in \mathbb{N}$ be an input to any of our algorithms (note that none of our algorithms takes the target size of the $r$-independent set as input) and let $S \subseteq V(G)$ and $B \subseteq A \setminus S$ such that $B$ is $r$-independent in $G - S$ be its output. Let us define $\pi_r[v, S]$ – the $r$-distance profile of $v$ on $S$ – as the function from $S$ to
\{0, 1, \ldots, r, \infty\}$ so that $\pi_r[v,S](a) = \text{dist}_G(v,a)$ if this distance is at most $r$, and $\pi_r[v,S](a) = \infty$ otherwise. The performance of the algorithms [17, 22, 55, 63] strongly depends on the size of the largest equivalence class on $B$ defined by $u \sim v$ if $\pi_r[u,S] = \pi_r[v,S]$ for $u, v \in B$.

We hence decided to use the size of the largest equivalence class in the above relation as the scoring function to measure the performance of our algorithms. Note that number of different $r$-distance profiles is bounded by $(r + 2)^{|S|}$, so if $r$ is fixed and $|S|$ is bounded then the number of different $r$-distance profiles is also bounded, so having a big $r$-independent set implies having a big subset of this set with equal $r$-distance profiles on $S$.

This well defined scoring function makes it possible to compare the results of the algorithms. Furthermore, in our code the implementation of the scoring function can be easily exchanged, so if different scoring functions are preferred, re-computation and re-evaluation is easily possible.

5 Experimental setup

5.1 Hard- and Software

The experiments on generalized coloring numbers has been performed on an Asus K53SC laptop with Intel Core i3-2330M CPU @ 2.20GHz x 2 processor and with 7.7GiB of RAM. Weak coloring numbers of a larger number of graphs for the statistics in Section 6.4 (presented without running times) were produced on a cluster at the Logic and Semantics Research Group, Technische Universitt Berlin. The experiments on uniform quasi-wideness have been performed on a cluster of 16 computers at the Institute of Informatics, University of Warsaw. Each machine was equipped with Intel Xeon E3-1240v6 3.70GHz processor and 16 GB RAM. All machines shared the same NFS drive. Since the size of the inputs and outputs to the programs is relatively small, the network communication was negligible for tests with substantial running times. The dtf implementation has been done in Python, while all other code in C++ or C. The code is available at [43, 3].

5.2 Test data

Our dataset consists of a number of graphs from different sources.

Real-world data We collected appropriately-sized networks from several collections [1, 35, 41, 7, 61, 38]. Our selection contains classic social networks [69, 12], collaboration networks [40, 52, 51] contact networks [64, 42], communication patterns [40, 59, 34, 39, 58, 4], protein-protein interaction [11], gene expression [28], infrastructure [67], tournament data [27], and neural networks [68]. We kept the names assigned to these files by the respective source.

PACE 2016 Feedback Vertex Set The Parameterized Algorithms and Computational Experiments Challenge is an annual programming challenge started in 2016 that aims at investigate the applicability of algorithmic ideas studied and developed in the subfields of multivariate, fine-grained, parameterized, or fixed-parameter tractable algorithms (from the PACE webpage). In the first edition, one of the tracks focused on the Feedback Vertex Set problem [18], providing 230 instances from various sources and of different sizes. We have chosen a number of instances with small feedback vertex set number, guaranteeing their very strong sparsity properties (in particular, low treewidth). In our result tables, they are named fvs???, where ?? is the number in the PACE 2016 dataset.

Random planar graphs In their seminal paper, Alber, Fellows, and Niedermeier [5] initiated the very fruitful direction of developing of polynomial kernels (preprocessing routines rigorously analyzed through the framework of parameterized complexity) in sparse graph classes by providing a linear kernel for Dominating Set in planar graphs. Dominating Set soon turned out to be the pacemaker of the development of fixed-parameter and kernelization algorithms in bounded expansion and nowhere dense graph classes [6, 17, 22, 23]. In [5], an experimental evaluation is conducted on random planar graphs generated by the LEDA library [2]. We followed their setup and included a number of random planar graphs with various size and average degree. In our result tables, they are named planarN, where N stands for the number of vertices.
Random graphs with bounded expansion A number of random graph models has been shown to produce almost surely graphs of bounded expansion [21]. We include a number of graphs generated by O’Brien and Sullivan [53] using the following models: the stochastic block model (sb-? in our dataset) [31] and the Chung-Lu model with households (clh-?) and without households (cl-?) [15]. We refer to [21, 53] for more discussion on these sources.

The graphs have been partitioned into four groups, depending on their size: the small group gathers graphs up to 1000 edges, medium between 1000 and 10 000 edges, big between 10 000 and 48 000 edges, and huge above 48 000 edges. The random planar graphs in every test group have respectively 900, 3 900, 21 000, and 150 000 edges. The whole dataset is available for download at [3].

6 Weak coloring numbers: results

6.1 Fine-tuning flat decompositions

| option    | average appx. ratio | option    | average appx. ratio | option    | average appx. ratio |
|-----------|---------------------|-----------|---------------------|-----------|---------------------|
| BFS/(1)   | 1.159               | DFS/(1)   | 1.156               | SORT/(1)  | 1.072               |
| BFS/(2)   | 1.131               | DFS/(2)   | 1.117               | SORT/(2)  | **1.039**           |
| BFS/(3)   | 1.147               | DFS/(3)   | 1.135               | SORT/(3)  | 1.054               |
| BFS/(1)   | 1.363               | DFS/(1)   | 1.368               | SORT/(1)  | 1.41                |
| BFS/(2)   | 1.277               | DFS/(2)   | 1.291               | SORT/(2)  | 1.329               |
| BFS/(3)   | 1.309               | DFS/(3)   | 1.324               | SORT/(3)  | 1.36                |

Table 1: Comparison of different flat decomposition variants: sorting vertices of the new blobs $B_i$ by the BFS, DFS, by degree (nonincreasing), or these orders reversed; the second coordinate refers to the choice of the root vertex: (1) maximizing the number of neighbors already processed, (2) maximizing degree in $U$, (3) as previous, but only among neighbors of already processed vertices. The value is the average of the approximation ratios to the best generalized coloring numbers found by all versions of this algorithm.

As discussed in Section 3.2, we have experimented with a number of variants of the flat decompositions approach, with regards to the choice of the next root vertex and the internal order of the vertices of the next $B_i$. The results for the big dataset are presented in Table 1. They clearly indicate that (a) all reversed orders performed much worse, and (b) among other options, the best is to sort the vertices of a new $B_i$ nonincreasingly by degree and choose as the next root the vertex of maximum degree. In the subsequent tests, we use this best configuration for comparison with other approaches.

6.2 Comparison of all approaches

Table 2 presents the results of our experiments on all test instances and all approaches, summarized as follows:

dtf dft-augmentations with the respective radius $r$ supplied as the distance bound;

flat the best configuration of the flat decompositions approach (see previous section);

treedepth the treedepth approximation heuristic;

treewidth the treewidth heuristic;

degree sort the heuristic which sorts the vertices nonincreasing by degree.
Table 2: *Gray columns*: Comparison of the main approaches and their average approximation ratio to the best found coloring number. Some of the approaches did not finish in time on larger graphs or ran out of memory. *White columns*: Total running time of the main approaches. Note that for some approaches the ordering (and thus running time) is independent of the radius.

| tests | r  | dtf | flat | treedepth | treewidth | degree sort |
|-------|----|-----|------|-----------|-----------|-------------|
| small | 2  | 1.19| 0:04.20 | 1.2 | 1.408 | 1.12 | 0:00.16 | 0:00.09 |
|       | 3  | 1.439 | 0:05.08 | 1.239 | 1.438 | 1.124 | 0:08.97 | 0:00.34 |
|       | 4  | 1.558 | 0:05.74 | 1.288 | 1.384 | 1.135 | 0:00.16 | 0:00.09 |
|       | 5  | 1.718 | 0:06.55 | 1.353 | 1.414 | 1.167 | 0:00.16 | 0:00.09 |
| medium| 2  | 1.177 | 0:27.97 | 1.362 | 2.171 | 1.524 | 0:01.97 | 0:23.64 |
|       | 3  | 1.258 | 1:02.31 | 1.43 | 1.918 | 1.283 | 0:02.17 | 0:00.56 |
|       | 4  | 1.499 | 1:53.21 | 1.451 | 1.698 | 1.159 | 0:01.97 | 0:23.64 |
|       | 5  | 1.595 | 2:15.04 | 1.469 | 1.612 | 1.093 | 0:01.97 | 0:23.64 |
| big   | 2  | 1.107 | 0:32.82 | 1.43 | — | 2.278 | 0:19.08 | 0:03.30 |
|       | 3  | 1.243 | — | 1.419 | — | 1.895 | 0:19.08 | 0:03.30 |
|       | 4  | — | — | 1.414 | — | 1.434 | 0:19.08 | 0:03.30 |
|       | 5  | — | — | 1.415 | — | 1.189 | 0:19.08 | 0:03.30 |
| huge  | 2  | — | — | 1.727 | — | — | 0:19.08 | 0:03.30 |
|       | 3  | — | — | 2.156 | — | — | 0:19.08 | 0:03.30 |
|       | 4  | — | — | 2.13 | — | — | 0:19.08 | 0:03.30 |
|       | 5  | — | — | 2.095 | — | — | 0:19.08 | 0:03.30 |

Table 2: *Gray columns*: Comparison of the main approaches and their average approximation ratio to the best found coloring number. Some of the approaches did not finish in time on larger graphs or ran out of memory. *White columns*: Total running time of the main approaches. Note that for some approaches the ordering (and thus running time) is independent of the radius.

Out of all simple heuristics (c.f. Section 3.5) the degree sorting was supreme and we skip the results of inferior heuristics (see [43, 3] for full data). Interestingly, this heuristic also outperformed all other (much more involved) approaches on larger graphs. On small graphs, the treewidth heuristic takes the lead. An explanation why the treewidth heuristic is better on smaller graphs $G$ might be that $\mathrm{tw}(G) = \col_{\infty}(G)$ and on small graphs the difference between $\col_{\infty}(G)$ and $\col_r(G)$ for the considered $r$ is not that big. However, this does not explain why treedepth does not perform better than treewidth. (Recall that $\mathrm{td}(G) = \wcol_{\infty}(G)$.) It is worth observing that on larger graphs (the big group) the performance of the flat decomposition matches or outperforms the one of the treewidth heuristic for radii $r = 2, 3, 4$. However, the treewidth heuristic outperforms all approaches with proved guarantees for $r = 5$ on test sets up to the big group.

Table 2 gathers total running time of our programs on discussed data sets. These results clearly indicate large discrepancy between consumed resources for different approaches. Out of the approaches with provable guarantees on the output coloring number, the flat decompositions approach is clearly the most efficient.

Note that we applied different timeout policies for generating different data. For generating time of execution and for applying local search we set timeout to be 1 minute, however for generating orders and wcol numbers we set timeout to be 5 minutes, but for the sake of completeness we sometimes allowed some programs to run longer.

In summary, on our data sets the simple heuristic is consistently the fastest and produces the best results, save for the smallest graphs on which the treewidth heuristic won. We remark here that it is simple to “fool” the degree-sorting heuristic by adding multiple pendant vertices of degree one and thus forcing it to take arbitrarily bad ordering, but such adversarial obstacles seem to be absent in real-world graphs. If one is to choose an algorithm with provable guarantees, the discussed variant of the flat decompositions approach appears to be the best choice.
Table 3: Gray columns: Comparison of average approximation ratio after local search. White columns: Relative improvement of local search for ordering output by the studied approaches.

6.3 Local search

In a second round of experiments we applied a simple local-search routine that, given an ordering output by one of the approaches, tries to improve it by moving vertices with the largest weakly reachable sets earlier in the ordering. The white columns in Table 3 show how local search improved orderings output by discussed approaches, and the gray columns show average approximation ratios of orderings improved by local search. Two remarks are in place. First, regardless of how the ordering was computed, a local search step always significantly improves the ordering (we have no good explanation on why local search is significantly less effective on the orderings output by the treewidth heuristic for bigger radii). Second, the local search step does not improve the orderings enough to change the relative order of the performance of the base approaches except for one remarkable case. On medium group the treewidth heuristic gave best results on \(r=5\), however degree sort regained the lead after application of local search due to its low performance on larger radii for treewidth heuristic. We therefore recommend the local search improvement as a relatively cheap post-processing improvement to any existing algorithm.

6.4 Correlation of weak coloring numbers with other parameters

While it is undeniable that weak coloring numbers have immense algorithmic power from a theoretical perspective, the efficient computation of such weak coloring orders is only one component to leverage them in practice: we also need these numbers to be reasonably low. So far, this had only been established on a smaller scale [21, 56] for a related measure. Here, we computed the weak coloring number for \(r \in \{1, \ldots, 5\}\) for 1675 real-world networks from various sources [38, 41, 61, 7, 1]. Figure 1 summarizes our findings: for \(r \in \{1, \ldots, 3\}\) we find a modest correlation with \(n\) and a significant correlation with \(m\). The correlation with \(n\) becomes quite pronounced for \(r=5\); the probable reason being that for all networks involved \(\log n \leq 10\). Still, even in the worst examples \(wcol_5\) is at least one order of magnitude smaller than \(n\) or \(m\). We further see a high correlation between \(wcol_1\) and the average degree \(\bar{d}\) which vanishes for larger radii. It is no big surprise that \(\bar{d}\) and the degeneracy \(wcol_1\) are highly correlated since these values are only far apart in graphs with highly inhomogeneous densities.

The low dependence on the maximum degree confirms the findings of [21]: the exact shape of the degree distribution’s tail is much more relevant than the singular value of the maximum degree. Finally, note that in our graphs the degeneracy \(wcol_1\) practically does not grow with \(n\).
7 Uniform quasi wideness: results

Table 4 gathers aggregated data from our experiments on medium dataset. (Full data can be downloaded from [43, 3].) Every tested algorithm has been run on every test with timeout 10 minutes and with radii \( r \in \{2, 3, 4, 5\} \) and with the starting set either \( A = V(G) \) or a random subset of 20\% of vertices of \( V(G) \).

Data indicate the simple heuristic, \( 1d \), as the best choice in most scenarios, as it has always best or nearly-best total score and runs relatively quickly. The third variant of the new algorithm \( \text{new}\_ld \) has comparable results, but is inefficient and does not finish within the timeout. Other variants \( \text{new1} \) and \( \text{new2} \) as well as \( \text{mfcs} \) are significantly outperformed by other approaches. Out of other approaches with provable guarantees, the variants \( \text{tree1}, \text{tree2}, \) and \( \text{ld\_it} \) provide results in most cases less than 10\% worse than the heuristic \( 1d \), with \( \text{tree2} \) being consistently worse.

To sum up, our experiments show that the simple heuristic \( 1d \) gives best results, but if one is interested in algorithm with provable guarantees, one should choose one of the variant \( \text{tree1} \) over \( \text{mfcs} \) or \( \text{new1/new2} \).

8 A lower bound to the TGV algorithm

In this section we observe that the construction of [29] shows also that the bounds of our new uniform quasi wideness algorithm of Section 4.3 are close to optimal. More precisely, we show the following corollary of the construction of [29].

**Theorem 8.1.** For every integers \( k, r \geq 1 \) and every integer \( m' > c \) where \( c = \binom{k+r}{r} \), there exists a graph \( G_{k,r,m'} \) with the following properties:

- the treewidth of \( G_{k,r,m'} \) is at most \( k \);
- \( \text{wcol}_r(G_{k,r,m'}) = c \);
- \( |V(G)| \geq (m' - 1)^c \).
Table 4: Aggregated results of uniform quasi-wideness on medium set: total size of all deleted and independent sets, total score (total size of largest equivalence classes w.r.t. deleted vertices), and total running time.

- for every pair of disjoint sets $B, Z \subseteq V(G)$ such that $B$ is $2r$-independent in $G - Z$, we have $|B| \leq |Z| \cdot m' + 1$; in particular, if $|B| \geq cm' + 1$ then $|Z| \geq c$ and if $|Z| \leq c$ then $|B| \leq cm' + 1$.

Before we proceed with the proof, let us discuss the statement and its implications. Most importantly, the example of Theorem 8.1 is weak in the sense that it treats $2r$-independent sets, as opposed to $r$-independent sets output by the algorithm of Section 4.3. However, it shows that even in bounded treewidth graph classes the dependency between the size of the input set $A$ and the size of the output independent set $B$ needs to be polynomial with degree depending on the quality of the graph class in question (here, $c = \text{wcol}_r(G_{k,r,m'})$).
Apart from this slackness, the bounds in Theorem 8.1 are very similar to the ones of Theorem 4.2: to get a independent set of size \( m := cm' + 1 \) in a graph with \( \text{wcol}_{L} = c \) one needs a vertex set of a graph of size \( (m' - 1)c \sim (m/c)c \) and deletion of \( c \) vertices.

**Proof of Theorem 8.1.** We start by recalling the construction of [29]. Fix a branching degree \( d \). For every \( k, r \geq 1 \) let \( T(k, r) \) be a rooted tree of depth \( c = \binom{k+r}{r} \) and branching degree \( d \). We define graphs \( G(k, r) \) inductively as follows.

First, we start with \( T(k, r) \) being a spanning tree of \( G(k, r) \). We will maintain the invariant that every edge of \( G(k, r) \) connects an ancestor and a descendant in \( T(k, r) \) (i.e., \( G(k, r) \) is a subgraph of ancestor-descendant closure of \( T(k, r) \)).

For \( k = 1 \), we take \( G(k, r) = T(k, r) \). For \( r = 1 \), we take \( G(k, r) \) to be the whole ancestor-descendant closure of \( T(k, r) \), that is, we add \( uv \) to \( E(G(k, r)) \) whenever \( u \) is an ancestor of \( v \) in \( T(k, r) \). For \( k, r \geq 2 \), note that one can equivalently construct \( T(k, r) \) as follows: start with \( T(k, r - 1) \) and for every leaf \( v \) of \( T(k, r - 1) \), create \( d \) copies of \( T(k - 1, r) \) and connect their roots to \( v \). To define \( G(k, r) \), we proceed as follows: we start with \( G(k, r - 1) \) and for every leaf \( v \) of the spanning tree \( T(k, r - 1) \) of \( G(k, r - 1) \), we create \( d \) copies of \( G(k - 1, r) \) and make all of them fully adjacent to \( v \).

In [29], it is shown that the treewidth of \( G(k, r) \) is \( k \), and that as long as \( d \geq c = \binom{k+r}{r} \), in every ordering of \( V(G(k, r)) \) there exists a leaf \( v \) of \( T(k, r) \) with its every ancestor belonging to \( WReach_{L}(G(k, r), L, v) \) (in particular, \( \text{wcol}_{L}(G(k, r)) \geq c \)). We take \( G(k, r, m') = G(k, r) \) for branching degree \( d = m' - 1 \); recall that \( m' > c = \binom{k+r}{r} \). The bound on the number of vertices of \( G(k, r, m') \) is straightforward. It remains to show the last property of \( G(k, r, m') \).

We start by observing the following.

**Claim 8.2.** For every \( v \in V(G(k, r)) \) and its ancestor \( u \) in \( T(k, r) \), there exists a path from \( v \) to \( u \) of length at most \( r \) that traverses only vertices on the unique path from \( v \) to \( u \) in \( T(k, r) \).

**Proof.** We proceed by induction on \( k + r \). For \( k = 1 \) or \( r = 1 \) the statement is straightforward. Assume then \( k, r \geq 2 \), and recall that \( G(k, r) \) consists of \( G(k, r - 1) \) and \( d \) copies of \( G(k - 1, r) \) attached to every leaf of \( T(k, r - 1) \).

If \( u \) and \( v \) both belong to \( G(k, r - 1) \) or to the same copy of \( G(k - 1, r) \), then we are done by inductive hypothesis. Otherwise, \( v \) belongs to a copy of \( G(k - 1, r) \) attached to a leaf \( w \) of \( T(k, r - 1) \), and \( u \) belongs to \( G(k, r - 1) \). By induction hypothesis, there exists a path of length at most \( r - 1 \) from \( w \) to \( u \) that uses only vertices on the path from \( w \) to \( u \) in \( T(k, r - 1) \). Together with the edge \( vw \), this path forms the desired path from \( v \) to \( u \).

Consequently, for every subtree \( T \) of \( T(k, r) \), every vertex of \( T \) is within distance at most \( r \) from the topmost vertex of \( T \) in \( G(k, r) \), and, consequently, the vertex set of \( T \) induces a graph of diameter at most \( 2r \) in \( G(k, r) \).

Consider now a pair of disjoint sets \( B, Z \subseteq V(G) \) such that \( B \) is \( 2r \)-independent in \( G - Z \). The observation from the preceeding paragraph implies that every connected component of \( T(k, r) - Z \) contains at most one vertex of \( B \). On the other hand, the maximum degree of \( T(k, r) \) is \( d + 1 = m' \). Consequently, \( |B| \leq |Z|m' + 1 \). This finishes the proof. \( \square \)

## 9 Conclusions

We have conducted a thorough empirical evaluation of algorithms for computing generalized coloring numbers and uniform quasi-wideness. In both cases, one of the simplest heuristics, without any theoretical guarantees, outperformed the rest. In particular, our new algorithm for uniform quasi-wideness, whose development was motivated by conservativeness of the previous approach of [36], performed rather poorly in the experiments. From the algorithms with provable guarantees, the experiments indicated a variant of the algorithm of [65] as the algorithm of choice for generalized coloring numbers and a variant of the algorithm of [55] as the algorithm of choice for uniform quasi-wideness.
Acknowledgments

We thank Christoph Dittmann for providing us with his code for the mfcs algorithm, which we partially used for our implementation. We thank Michał Pilipczuk for many hours of fruitful discussions.

References

[1] Gephi datasets. https://github.com/gephi/gephi/wiki/Datasets.
[2] LEDA. http://www.algorithmic-solutions.com/leda/index.htm.
[3] Recent trends in kernelization: theory and experimental evaluation — project website, 2018. http://kernelization-experiments.mimuw.edu.pl.
[4] Lada A Adamic and Natalie Glance. The political blogosphere and the 2004 US election: divided they blog. In Proceedings of the 3rd International Workshop on Link Discovery, pages 36–43. ACM, 2005.
[5] Jochen Alber, Michael R. Fellows, and Rolf Niedermeier. Polynomial-time data reduction for dominating set. Journal of the ACM, 51(3):363–384, 2004.
[6] Saeed Akhoomian Amiri, Patrice Ossona de Mendez, Roman Rabinovich, and Sebastian Siebertz. Distributed domination on graph classes of bounded expansion. CoRR, abs/1702.02848, 2017.
[7] Vladimir Batagelj and Andrej Mrvar. Pajek datasets. http://vlado.fmf.uni-lj.si/pub/networks/data/, 2006.
[8] Anne Berry, Jean-Paul Bordat, and Olivier Cogis. Generating all the minimal separators of a graph. International Journal of Foundations of Computer Science, 11(03):397–403, 2000.
[9] Hans L. Bodlaender. Treewidth: Algorithmic techniques and results. In International Symposium on Mathematical Foundations of Computer Science, pages 19–36. Springer, 1997.
[10] Hans L. Bodlaender and Arie M. C. A. Koster. Treewidth computations I. Upper bounds. Information and Computation, 208(3):259–275, 2010.
[11] Dongbo Bu, Yi Zhao, Lun Cai, Hong Xue, Xiaopeng Zhu, Hongchao Lu, Jingfen Zhang, Shiwei Sun, Lunjiang Ling, Nan Zhang, et al. Topological structure analysis of the protein–protein interaction network in budding yeast. Nucleic acids research, 31(9):2443–2450, 2003.
[12] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. pages 1082–1090.
[13] Fan Chung and Linyuan Lu. The average distances in random graphs with given expected degrees. Proceedings of the National Academy of Sciences, 99(25):15879–15882, 2002.
[14] Fan Chung and Linyuan Lu. Connected components in random graphs with given expected degree sequences. Annals of combinatorics, 6(2):125–145, 2002.
[15] Fan R. K. Chung and Linyuan Lu. The average distance in a random graph with given expected degrees. Internet Mathematics, 1(1):91–113, 2003.
[16] Anuj Dawar. Homomorphism preservation on quasi-wide classes. Journal of Computer and System Sciences, 76(5):324–332, 2010.
[17] Anuj Dawar and Stephan Kreutzer. Domination problems in nowhere-dense classes. In IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS, pages 157–168, 2009.
[18] Holger Dell, Thore Husfeldt, Bart M. P. Jansen, Petteri Kaski, Christian Komusiewicz, and Frances A. Rosamond. The First Parameterized Algorithms and Computational Experiments Challenge. In 11th International Symposium on Parameterized and Exact Computation (IPEC 2016), volume 63 of Leibniz International Proceedings in Informatics (LIPIcs), pages 30:1–30:9, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
[19] Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. The Computer Journal, 51(3):292–302, 2007.
[20] Erik D. Demaine, MohammadTaghi Hajiaghayi, and Ken-ichi Kawarabayashi. Algorithmic graph minor theory: Improved grid minor bounds and wagner’s contraction. Algorithmica, 54(2):142–180, 2009.
[21] Erik D. Demaine, Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, Somnath Sikdar, and Blair D. Sullivan. Structural sparsity of complex networks: Random graph models and linear algorithms. CoRR, abs/1406.2557, 2014.
[22] Pál Grønás Drange, Markus Sortland Dregi, Fedor V. Fomin, Stephan Kreutzer, Daniel Lokshitanov, Marcin Pilipczuk, Michal Pilipczuk, Felix Reidl, Fernando Sánchez Villaamil, Saket Saurabh, Sebastian Siebertz, and Somnath Sikdar. Kernelization and sparseness: the case of dominating set. In 33rd Symposium on Theoretical Aspects of Computer Science, STACS, pages 31:1–31:14, 2016.

[23] Zdenek Dvorak. Constant-factor approximation of the domination number in sparse graphs. European Journal of Combinatorics, 34(5):833–840, 2013.

[24] Zdenek Dvorak, Daniel Kráľ, and Robin Thomas. Testing first-order properties for subclasses of sparse graphs. Journal of the ACM, 60(5):36:1–36:24, 2013.

[25] Kord Eickmeyer, Archontia C. Giannopoulou, Stephan Kreutzer, O-joung Kwon, Michal Pilipczuk, Roman Rabinovich, and Sebastian Siebertz. Neighborhood complexity and kernelization for nowhere dense classes of graphs. In 44th International Colloquium on Automata, Languages, and Programming, ICALP, pages 63:1–63:14, 2017.

[26] Jakub Gajarský, Petr Hliněný, Jan Obdržálek, Sebastian Ordyniak, Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, and Somnath Sikdar. Kernelization using structural parameters on sparse graph classes. Journal of Computer and System Sciences, 84:219–242, 2017.

[27] Michelle Girvan and Mark E. J. Newman. Community structure in social and biological networks. Proceedings of the National Academy of Sciences, 99(12):7821–7826, 2002.

[28] Kwang-Il Goh, Michael E. Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-László Barabási. The human disease network. Proceedings of the National Academy of Sciences, 104(21):8685–8690, 2007.

[29] Martin Grohe, Stephan Kreutzer, Roman Rabinovich, Konstantinos Stavropoulos. Colouring and covering nowhere dense graphs. In 41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016.

[30] Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. Journal of the ACM, 64(3):17:1–17:32, 2017.

[31] Paul W. Holland, Kathryn B. Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. Social networks, 5(2):109–137, 1983.

[32] Wojciech Kazana and Luc Segoufin. Enumeration of first-order queries on classes of structures with bounded expansion. In 32nd Symposium on Principles of Database Systems, PODS, pages 297–308, 2013.

[33] Henry A. Kierstead and Daqing Yang. Orderings on graphs and game coloring number. Order, 20:255–264, 2003.

[34] Bryan Klimt and Yiming Yang. Introducing the Enron corpus. In CEAS, 2004.

[35] Donald Ervin Knuth. The Stanford GraphBase: a platform for combinatorial computing, volume 37. Addison-Wesley Reading, 1993.

[36] Stephan Kreutzer, Michal Pilipczuk, Roman Rabinovich, and Sebastian Siebertz. The generalised colouring numbers on classes of bounded expansion. In 41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016.

[37] Stephan Kreutzer, Roman Rabinovich, and Sebastian Siebertz. Polynomial kernels and wideness properties of nowhere dense graph classes. In Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1533–1545, 2017.

[38] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 177–187. ACM, 2005.

[39] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[40] Wojciech Nadara, Marcin Pilipczuk, Felix Reidl, Roman Rabinovich, and Sebastian Siebertz. Empirical evaluation of approximation algorithms for generalized graph coloring and uniform quasi-wideness. code repository, 2018. https://bitbucket.org/marcin.pilipczuk/wcol-uqw-experiments.
