Algorithmica (2021) 83:337–360
https://doi.org/10.1007/s00453-020-00759-7

Tight Bounds for Online Coloring of Basic Graph Classes

Susanne Albers[1] · **Sebastian Schraink**[1]

Received: 8 January 2020 / Accepted: 7 August 2020 / Published online: 25 August 2020

Abstract
We resolve a number of long-standing open problems in online graph coloring. More specifically, we develop tight lower bounds on the performance of online algorithms for fundamental graph classes. An important contribution is that our bounds also hold for randomized online algorithms, for which hardly any results were known. Technically, we construct lower bounds for chordal graphs. The constructions then allow us to derive results on the performance of randomized online algorithms for the following further graph classes: trees, planar, bipartite, inductive, bounded-tree-width and disk graphs. It shows that the best competitive ratio of both deterministic and randomized online algorithms is $\Theta(\log n)$, where n is the number of vertices of a graph. Furthermore, we prove that this guarantee cannot be improved if an online algorithm has a lookahead of size $O(n/\log n)$ or access to a reordering buffer of size $n^{1-\epsilon}$, for any $0 < \epsilon \leq 1$. A consequence of our results is that, for all of the above mentioned graph classes except bipartite graphs, the natural *First Fit* coloring algorithm achieves an optimal performance, up to constant factors, among deterministic and randomized online algorithms.

Keywords Graph coloring · Online algorithms · Lower bounds · Randomization

1 Introduction

Online graph coloring is a classical problem in graph theory and online computation. It has applications in job scheduling, dynamic storage allocation and resource management in wireless networks [24, 28, 29]. A problem instance is defined by

A preliminary version of the paper has appeared in the 25th Annual European Symposium on Algorithms (ESA'17). Work supported by the European Research Council, Grant Agreement No. 691672, project APEG.

✉ Susanne Albers
albers@in.tum.de

Sebastian Schraink
schraink@in.tum.de

[1] Department of Computer Science, Technical University of Munich, Garching, Germany

an undirected graph $G = (V, E)$, consisting of a vertex set V and an edge set E. Let $|V| = n$. The vertices arrive one by one in a sequence $\sigma = v_1, \ldots, v_n$ that may be determined by an adversary. Whenever a new vertex v_t arrives, $1 \leq t \leq n$, its edges to previous vertices v_s with $s < t$ are revealed. An online algorithm $\mathcal{A}$ has to immediately assign a feasible color to v_t, i.e. a color that is different from those assigned to the neighbors of v_t presented so far. The goal is to minimize the total number of colors used.

For a graph G, let $\mathcal{A}(G)$ be the number of colors used by $\mathcal{A}$. Let $\chi(G)$ be the chromatic number of G, which is the minimum number of colors needed to color G offline. An online algorithm $\mathcal{A}$ is *c-competitive* if there exists a constant b such that $\mathcal{A}(G) \leq c \cdot \chi(G) + b$ holds for every graph G [30]. The additive constant b must be independent of the input graph G. If $\mathcal{A}$ is a randomized algorithm, then $E[\mathcal{A}(G)]$ is the expected number of colors used by $\mathcal{A}$. The algorithm is c-competitive against oblivious adversaries if there exists a constant b such that $E[\mathcal{A}(G)] \leq c \cdot \chi(G) + b$ holds for every G [6]. Again, b must be independent of G. An oblivious adversary, when determining σ, does not know the outcome of the random choices made by $\mathcal{A}$. We always evaluate randomized online algorithms against this type of adversary. When considering specific graph classes, for a deterministic or randomized algorithm, the competitive factor of c must hold for every graph from the given class.

The framework defined above is the standard online one. It is also interesting to explore settings where an algorithm is given more power. An online algorithm $\mathcal{A}$ has *lookahead* l if, upon the arrival of vertex v_t, the algorithm also sees the next l vertices $v_{t+1}, \ldots, v_{t+l}$ along with their adjacencies to vertices in $\{v_1, \ldots, v_{t+l}\}$. Alternatively, algorithm $\mathcal{A}$ might have a *buffer of size* b in which vertices can be stored temporarily. In any time step, if the buffer is not full, the newly arriving vertex is placed in the buffer. Then $\mathcal{A}$ may color vertices residing in the buffer and remove them from this storage. On the other hand, if the buffer is full, $\mathcal{A}$ must color either the incoming vertex or at least one from the buffer. In the latter case the new vertex is admitted to the buffer. Hence the requirement is that at the end of step t the algorithm must have colored at least $t - b$ vertices. A buffer is more powerful than lookahead because it allows the algorithm to partially reorder the input sequence and delay coloring decisions. The value of a buffer has recently been explored for a variety of online problems, see e.g. [1, 2, 16] and references therein.

Previous Work: For general graphs, the competitive ratios are high compared to the trivial upper bound of n. Lovasz, Saks and Trotter [27] developed a deterministic online algorithm that achieves a competitive factor of $O(n/\log^* n)$. Vishwanathan [31] devised a randomized algorithm that attains a competitiveness of $O(n/\sqrt{\log n})$. This bound was improved to $O(n/\log n)$ by Halldorsson [21]. Halldorsson and Szegedy [22] proved that the competitive ratio of any deterministic online algorithm is $\Omega(n/\log^2 n)$. This lower bound also holds for randomized algorithms. Moreover, it holds if a randomized algorithm has a lookahead or a buffer of size $O(\log^2 n)$ [22].

There has also been considerable research interest in online coloring for various graph classes. An early and celebrated result proved by Bean [5] in 1976 is that, for trees, every deterministic online algorithm can be forced to use $\Omega(\log n)$ colors. The *First Fit* algorithm colors every tree with $O(\log n)$ colors [20]. The natural strategy *First Fit* assigns the lowest-numbered feasible color to each

incoming vertex. Since trees have a chromatic number of 2, the best competitive ratio achievable by deterministic online algorithms is $\Theta(\log n)$. For bipartite graphs, there also exists a deterministic online algorithm that uses $O(\log n)$ colors [27]. This implies that the best competitiveness of deterministic strategies for is again $\Theta(\log n)$ because trees are bipartite. However, *First Fit* performs poorly here, as there are bipartite graphs for which it requires $\Omega(n)$ colors. Kierstead and Trotter [25] proved that, for interval graphs, the best competitive ratio of deterministic online algorithms is equal to 3.

A paper directly related to our work is by Irani [23]. She examined d-inductive graphs, also referred to as d-degenerate graphs. They are defined as the graphs which admit a numbering of the vertices such that each vertex is adjacent to at most d higher-numbered vertices. Every planar graph is 5-inductive and every chordal graph G is $(\chi(G)-1)$-inductive. Moreover, every tree is 1-inductive. Irani [23] proved that *First Fit* colors every d-inductive graph with $O(d \cdot \log n)$ colors. Furthermore, for every deterministic online algorithm $\mathcal{A}$, there exist graphs such that $\mathcal{A}$ uses $\Omega(d \cdot \log n)$ colors [23]. Since d-inductive graphs have a chromatic number of at most $d+1$, the best competitive ratio achieved by deterministic online algorithms is $\Omega(\log n)$. For planar graphs a tight bound of $\Theta(\log n)$ holds because trees are planar. However, it was an open problem if a tight competitiveness of $\Theta(\log n)$ holds for general chordal graphs. In fact, Irani [23] raised the question if, for every deterministic online algorithm $\mathcal{A}$ and every d, there exists a chordal graph with chromatic number d such that $\mathcal{A}$ uses $\Omega(d \cdot \log n)$ colors. Finally, for d-inductive graphs, Irani [23] analyzed deterministic online algorithms with lookahead l and showed that the best competitiveness is $\Theta(\min\{\log n, n/l\})$. A lower bound of $\Omega(\log \log n)$ on the competitive ratio of randomized online algorithms for d-inductive graphs was given by Leonardi and Vitaletti [26].

We address two further graph classes. Downey and McCartin [15] studied online coloring of bounded treewidth graphs. For an introduction to treewidth see [9]. For any graph of treewidth d, *First Fit* uses $O(d \cdot \log n)$ colors. This is a consequence of Irani's work [23] because a graph of treewidth d is d-inductive [15, 23]. Downey and McCartin [15] showed that, on graphs of treewidth d, *First Fit* can be forced to use $\Omega(\frac{d}{\log(d+1)} \log n)$ colors. Last but not least, a disk graph is the intersection graph of a set of disks in the Euclidean plane. Each vertex represents a disk; two vertices are adjacent if the two corresponding disks intersect. Online coloring of disk graphs has received quite some attention because it models frequency assignment problems in wireless communication networks, see [18] for a survey. The best competitiveness achieved by a deterministic online algorithm is $\Theta(\min\{\log n, \log \rho\})$, where ρ is the ratio of the largest to smallest disk radius [14, 17]. The result relies on the common assumption that an online algorithm does not use the disk representation, when making coloring decisions [14, 17, 18]. It has been repeatedly raised as an open problem if the bound of $\Theta(\min\{\log n, \log \rho\})$ can be improved using randomization [14, 17, 18].

Recent work on online graph coloring has studied scenarios where an online algorithm can query oracle information about future input [8, 11, 13] or an

adversary chooses an input uniformly at random from a set of inputs known to the algorithm [12]. Batch coloring has been considered [10]. Moreover, online coloring of hypergraphs has been explored [3, 4].

Our Contribution: In this paper we settle the performance of online coloring algorithms for fundamental and widely studied graph classes. More precisely, we prove lower bounds on the performance of online algorithms. These bounds match the best upper bounds known in the literature. An important contribution is that our bounds also hold for randomized online algorithms, for which very few results were known.

First, in Sects. 2 and 3 we investigate chordal graphs. They have been studied extensively, cf. textbook [32]. We remind the reader that a graph is chordal if every induced cycle with four or more vertices has a chord. For a chordal graph G, the chromatic number $\chi(G)$ is equal to the largest clique size $\omega(G)$. Interval graphs are a subfamily of chordal graphs. Chordal graphs in turn are perfect graphs [7], for which the offline coloring, maximum clique and independent set problems can be solved in polynomial time.

In Sect. 2 we examine deterministic online coloring algorithms. We prove that, for every deterministic algorithm $\mathcal{A}$ and every integer $d \geq 2$, there exists a family of chordal graphs G with $\chi(G) = d$ such that $\mathcal{A}$ uses $\Omega(d \cdot \log n)$ colors. This resolves the open problem raised by Irani [23]. In Sect. 3 we extend this result to randomized online algorithms. The statement is identical to the one for deterministic algorithms, except that a randomized online algorithm uses an expected number of $\Omega(d \cdot \log n)$ colors. Although the result for randomized algorithms is more general, we give proofs for both deterministic and randomized policies. Our lower bound construction for deterministic algorithms exhibits an adversarial strategy for generating worst-case graphs. Given this strategy, we show how to define a probability distribution on graphs so that Yao's principle [33] can be applied. *First Fit* colors every chordal graph G with $\chi(G) = d$ using $O(d \cdot \log n)$ colors [23]. Hence, the optimal competitiveness of deterministic and randomized online algorithms is $\Theta(\log n)$.

In Sect. 4 we derive lower bounds for further graph classes, focusing on randomized online algorithms. For $d = 2$, our lower bound construction for chordal graphs generates trees. It follows that, for any randomized online algorithm $\mathcal{A}$, there exists a family of trees such that $\mathcal{A}$ needs an expected number of $\Omega(\log n)$ colors. This complements the fundamental and early result by Bean [5] for deterministic algorithms. To the best of our knowledge, no lower bound on the performance of randomized online coloring algorithms for trees was previously known. Recall that trees have a chromatic number of 2. Vishwanathan [31] gave a lower bound of $\Omega(\log n)$ on the expected number of colors used by randomized online algorithms for graphs of chromatic number 2, i.e. bipartite graphs. However, the graphs in his construction have cycles. Thus, Vishwanathan's lower bound does not apply to trees. Obviously, trees are planar and bipartite. Hence, our result for trees directly implies that every randomized online algorithm can be forced to use $\Omega(\log n)$ colors in expectation for graphs of these two classes. The lower bounds are tight because known deterministic online algorithms color trees, planar and bipartite graphs with $O(\log n)$ colors [20, 23, 27].

Section 4 also addresses inductive and bounded-treewidth graphs. Since every chordal graph G is $(\chi(G)-1)$-inductive and has treewidth $\chi(G)-1$ [9], we derive the following results. For every randomized online algorithm $\mathcal{A}$ and every $d \geq 1$, there exists a family of d-inductive graphs such that $\mathcal{A}$ uses $\Omega(d \cdot \log n)$ colors. The same statement holds for graphs of treewidth d. We further show that the statement also holds for strongly chordal graphs with chromatic number d. A chordal graph is strongly chordal if every cycle of even length consisting of at least six vertices has an odd chord, i.e. an edge connecting two vertices that have an odd distance from each other in the cycle [19]. *First Fit* colors any d-inductive graph and any graph of treewidth d using $O(d \cdot \log n)$ colors. We conclude that, for all the graph classes considered so far, $\Theta(\log n)$ is the best competitiveness of deterministic and randomized online algorithms. Finally, in Sect. 4 we study disk graphs. We prove that, for $d = 2$, every graph of the probability distribution defined in Sect. 3 translates to a disk graph. We then show that, for every randomized online algorithm $\mathcal{A}$, there exists a family of disk graphs forcing $\mathcal{A}$ to use an expected number of $\Omega(\min\{\log n, \log \rho\})$ colors, where ρ is again the ratio of the largest to smallest disk radius. Similar to earlier work we assume that $\mathcal{A}$ does not use the disk representation when making coloring decisions. Hence randomization does not improve the asymptotic performance of online coloring algorithms for disk graphs, cf. [14, 17, 18].

In Sect. 5 we explore the settings where an online algorithm has lookahead or is equipped with a reordering buffer. We show that a lookahead of size $O(n/\log n)$ does not improve the asymptotic performance of randomized online algorithms. We prove the result for chordal graphs and then derive analogous results for all the other graph classes. Irani [23] gave a similar result for deterministic algorithms, considering inductive graphs. As a final result of this paper we demonstrate that a reordering buffer of size $n^{1-\epsilon}$, for any $0 < \epsilon \leq 1$, does not yield an improvement in the asymptotic performance guarantees of deterministic online algorithms. Again, we develop the result for chordal graphs and derive corollaries for the other graph classes.

We finally remark that the additive constant b in the definition of the competitive ratio of an algorithm does not affect the lower bounds on the asymptotic competitiveness of algorithms we establish in this paper. The lower bounds we prove on the (expected) number of colors used by online algorithms depend on n, which can be arbitrarily large.

Our Proof Technique: We devise a technique for proving lower bounds that is relatively simple; we view this as a strength of our results. The main idea is to recursively construct trees of cliques, which in turn form forests. In a recursive step the construction combines forests by adding or not adding a new clique in a specific way. Our construction resembles the one by Bean [5] but differs in an important aspect that allows us to obtain lower bounds for randomized algorithms. The construction by Bean builds a tree T_k, $k \in \mathbb{N}$, by joining trees T_j, for $j < k$, so that any deterministic online algorithm must use a k-th new color for *some* vertex of T_k. This vertex then becomes the root of T_k. An oblivious adversary, playing against a randomized online algorithm, cannot identify with sufficiently high probability such vertices exhibiting a new color. Instead, our construction maintains the invariant that the root vertices of each forest use a large number of colors, given any deterministic

online algorithm. For randomized algorithms, a corresponding invariant holds with probability of at least 1/2.

Convention: Unless otherwise stated, logarithms are base 2.

2 Deterministic Online Algorithms for Chordal Graphs

We establish a lower bound on the performance of any deterministic online coloring algorithm.

Theorem 1 *Let $d \in \mathbb{N}$ with $d \geq 2$ be arbitrary. For every deterministic online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 2d^2$, there exists an n-vertex chordal graph G with chromatic number $\chi(G) = d$ such that $\mathcal{A}$ uses $\Omega(d \cdot \log n)$ colors to color G.*

The proof of Theorem 1 relies on Lemma 1, which we prove first.

Lemma 1 *Let $d \in \mathbb{N}$ with $d \geq 2$ be arbitrary. For every deterministic online algorithm $\mathcal{A}$ and every $k \in \mathbb{N}$, there exists a chordal graph G_k having chromatic number $\chi(G_k) = d$ and consisting of $n_k \leq d2^k$ vertices such that $\mathcal{A}$ is forced to use at least $c_k \geq (d-1)k/4$ colors to color G_k.*

Proof We describe how an adversary constructs a chordal graph G_k, $k \in \mathbb{N}$. Such a graph is built up recursively and consists of graphs G_j, where $j < k$. We assume that d is even. The construction of G_k can be adapted easily if d is odd; details will be given later. On a high level G_k is a forest, i.e. a collection of disjoint trees, each having a distinguished root node. In every tree T of G_k, each tree node represents a clique of size $d/2$ in G_k. If two tree nodes u_T and v_T are connected by a tree edge in T, then any two vertices $u \in u_T$ and $v \in v_T$ are connected by an edge in G_k. Hence u_T and v_T form a clique of size d in G_k. Since G_k is a forest, it consists of several connected components. One can add a final vertex and edges in order to connect the various trees; details will be given at the end of the proof.

We proceed with the concrete construction of G_k, for increasing values of $k \in \mathbb{N}$. As mentioned above, each tree T of G_k has a distinguished root node consisting of $d/2$ vertices in G_k. Let $r(T)$ be the set of these $d/2$ vertices. Moreover, let $r(G_k)$ be the union of these sets $r(T)$, taken over all T of G_k. We refer to the elements of $r(G_k)$ as the *root vertices of* G_k. They are important because the online algorithm $\mathcal{A}$ will be forced to use a large number of colors for $r(G_k)$. For any subset V' of the vertices of G_k, let $\mathcal{C}_{\mathcal{A}}(V')$ be the set of colors used by $\mathcal{A}$ to color V'.

The strategy of the adversary to generate a graph G_k is adaptive, i.e. the exact structure of the graph depends on the coloring decisions of $\mathcal{A}$. Nevertheless, during the bottom-up construction of G_k, for increasing $k \in \mathbb{N}$, the following invariants will be maintained.

1. Algorithm $\mathcal{A}$ uses at least $\frac{d}{4} \cdot k$ colors for the root vertices of G_k, i.e. $\left|\mathcal{C}_{\mathcal{A}}\big(r(G_k)\big)\right| \geq \frac{d}{4} \cdot k$.

2. G_k is a union of connected components, each of which can be represented by a tree T. Each tree node is a clique of size $d/2$. Every tree T has a distinguished root node containing a set $r(T)$ of $d/2$ root vertices in G_k.
3. G_k is chordal.
4. The maximum clique size is $\omega(G_k) = d$.
5. The number of vertices satisfies $n_k \leq \frac{d}{2} \cdot (2^{k+1} - 1)$.

Invariants (3) and (4) together imply that $\chi(G_k) = \omega(G_k) = d$ holds. Again, for a chordal graph, the chromatic number is equal to the size of the largest clique. In invariant (1) and the following technical exposition integer values are compared to expressions of the form $\frac{d}{4} \cdot k$, which might not be integer. We remark that the statements, comparisons and calculations hold without considering the rounded expressions.

Construction of the Base Graph G_1: G_1 is a clique of size d. The adversary may present the corresponding vertices in an arbitrary order. The set of root vertices $r(G_1)$ is an arbitrary subset R of size $d/2$ of the vertices of G_1. The remaining $d/2$ vertices form a second tree node. The resulting tree T is depicted in Fig. 1. We can easily verify properties (1–5).

1. Since $R = r(G_1)$ is a clique of size $d/2$, $\mathcal{A}$ uses $d/2$ colors for it, i.e. $\left|\mathcal{C}_{\mathcal{A}}(r(G_1))\right| \geq \frac{d}{4}$.
2. G_1 consists of one connected component which represents a tree, as described above and shown in Fig. 1.
3. G_1 is a clique and thus chordal.
4. The maximum clique size $\omega(G_1)$ is exactly d.
5. $n_1 = d \leq \frac{3}{2} \cdot d = \frac{d}{2} \cdot (2^{1+1} - 1)$.

Construction of the Graph G_k, $k > 1$: Assume that the adversary can generate graphs G_j, for any $j < k$, satisfying invariants (1–5). The construction of G_k proceeds as follows. First the adversary recursively generates two independent graphs of type G_{k-1}, i.e. it twice executes the strategy for generating a graph G_{k-1}. Let G^l_{k-1} and G^r_{k-1} be these two graphs. They are created one after the other. We remark that after $\mathcal{A}$ has colored G^l_{k-1} and G^r_{k-1}, the two graphs and respective colorings need not be identical because $\mathcal{A}$'s coloring decision in one graph can affect its decisions in the other one.

Fig. 1 The tree T representing G_1

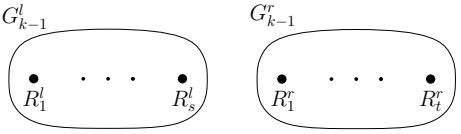

Fig. 2 The general structure of G^l_{k-1} and G^r_{k-1} restricted to the root vertices

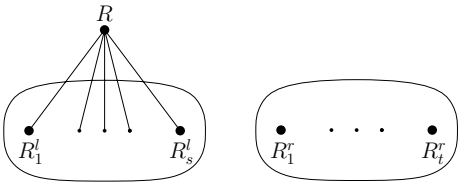

Fig. 3 The graph G_k with the new addition of R

In the following we focus on the root vertices of G^l_{k-1} and G^r_{k-1}. In particular, we consider the colors used by $\mathcal{A}$. Invariant (1) implies that $\left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))\right| \geq \frac{d}{4}(k-1)$ and $\left|\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))\right| \geq \frac{d}{4}(k-1)$. We distinguish between two cases depending on the total number of colors used, i.e. the cardinality of $\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}) \cup r(G^r_{k-1}))$. To this end we introduce some notation. Assume that G^l_{k-1} consists of s connected components, which we number in an arbitrary way. Each component/tree T^l_i has a distinguished root containing a set $r(T^l_i)$ of $d/2$ root vertices. We abbreviate $R^l_i = r(T^l_i)$, $1 \leq i \leq s$. Similarly, assume that G^r_{k-1} consists of t connected components. Let $r(T^r_j)$ be the set of root vertices in the component T^r_j and $R^r_j = r(T^r_j)$, $1 \leq j \leq t$. We have $r(G^l_{k-1}) = \bigcup_{i=1}^{s} R^l_i$ and $r(G^r_{k-1}) = \bigcup_{j=1}^{t} R^r_j$. Figure 2 shows the general structure of G^l_{k-1} and G^r_{k-1} by focusing on the roots. The left-hand side of the figure depicts G^l_{k-1} as a union of connected components rooted at $R^l_1, \ldots, R^l_s$, respectively. The right-hand side shows G^r_{k-1} as a collection of components rooted at $R^r_1, \ldots, R^r_t$.

Case 1: Assume that $\left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| \geq \frac{d}{4} \cdot k$. In this case the adversary defines G_k as the union of G^l_{k-1} and G^r_{k-1}. No further vertices or edges are added. It is easy to verify the five invariants because G^l_{k-1} and G^r_{k-1} satisfy them by inductive assumption.

1. The condition of Case 1 ensures $\left|\mathcal{C}_{\mathcal{A}}(r(G_k))\right| = \left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| \geq \frac{d}{4} \cdot k$.
2. The invariant is satisfied since G_k is the union of G^l_k and G^r_k.
3. G_k is chordal because G^l_k and G^r_k are, and no further vertices or edges have been added.
4. It holds that $\omega(G_k) = d$, as $\omega(G^l_{k-1}) = \omega(G^r_{k-1}) = d$.
5. Let n^l_{k-1} and n^r_{k-1} be the number of vertices in G^l_{k-1} and G^r_{k-1}, respectively. We have $n_k = n^l_{k-1} + n^r_{k-1} \leq 2 \cdot (\frac{d}{2} \cdot (2^k - 1)) = \frac{d}{2} \cdot (2^{k+1} - 2) \leq \frac{d}{2} \cdot (2^{k+1} - 1)$. The first inequality follows because (5) holds for n^l_{k-1} and n^r_{k-1}.

Case 2: Next assume that $\left|\mathcal{C}_A(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| < \frac{d}{4} \cdot k$. In this case the adversary adds a set R of $d/2$ vertices that form a clique. Moreover, for every vertex of R there is an edge to every vertex in R^l_i, for $i = 1, \ldots, s$. In other words, every vertex of R has edges to all root vertices of $r(G^l_{k-1})$. The vertices of R together with their adjacent edges may be presented by the adversary in an arbitrary order. The resulting structure is depicted in Fig. 3. Set R and the connected components of G^l_{k-1} rooted at $R^l_1, \ldots, R^l_s$ form a single component rooted at R. There is a tree edge between R and every R^l_i, $1 \leq i \leq s$. The newly created component forms a tree rooted at R because the components of G^l_{k-1} represent trees rooted at $R^l_1, \ldots, R^l_s$. Graph G_k is the union of the new component and the components of G^r_{k-1}. The set of root vertices of G_k consists of R and the root vertices of G^r_{k-1}. Formally, $r(G_k) = R \cup R^r_1 \cup \ldots, \cup R^r_t$. It remains to verify the five invariants.

1. We analyze the number of colors that $\mathcal{A}$ uses for the root vertices in G_k. In a first step, among the colors $\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1})) \cup \mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ for the roots of G^l_{k-1} and G^r_{k-1}, we upper bound the number q of colors occurring in $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ only. By assumption $\left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1})) \cup \mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))\right| = \left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| < \frac{d}{4} \cdot k$. It holds that $\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1})) \geq \frac{d}{4}(k-1)$. We obtain $q = \left|\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1})) \setminus \mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))\right| = \left|\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1})) \cup \mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))\right| - \left|\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))\right| < \frac{d}{4}$. Next consider the vertices in R. We upper bound the number of colors from $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ that $\mathcal{A}$ can use for R. Observe that $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ is the disjoint union of $\mathcal{C}_{\mathcal{A}}(r(G^l_{k-1})) \cap \mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ and $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1})) \setminus \mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))$. Every vertex of R is adjacent to every vertex in $r(G^l_{k-1})$. Hence, $\mathcal{A}$ cannot apply a color occurring in $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1})) \cap \mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))$ to a vertex in R. Only a color of $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1})) \setminus \mathcal{C}_{\mathcal{A}}(r(G^l_{k-1}))$ is feasible, and the latter set has cardinality $q < d/4$. Since R is a clique of size $d/2$ algorithm $\mathcal{A}$ must use at least $d/2 - q > d/4$ colors not contained in $\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))$ to color the vertices of$R$. As $r(G_k) = R \cup r(G^r_{k-1})$, we conclude $\left|\mathcal{C}_{\mathcal{A}}(r(G_k))\right| = \left|\mathcal{C}_{\mathcal{A}}(R \cup r(G^r_{k-1}))\right| = \left|\mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))\right| + \left|\mathcal{C}_{\mathcal{A}}(R) \setminus \mathcal{C}_{\mathcal{A}}(r(G^r_{k-1}))\right| \geq \frac{d}{4}(k-1) + \frac{d}{4} = \frac{d}{4}k$.
2. By construction G_k is a collection of connected components, forming trees rooted at R and $R^r_1, \ldots, R^r_t$, respectively.
3. In G_k consider a simple cycle C with at least four vertices. If C does not contain a vertex of R, then C only contains vertices of G^l_{k-1} or vertices of G^r_{k-1}. By inductive assumption, both G^l_{k-1} and G^r_{k-1} are chordal and hence C has a chord. Therefore, assume that C contains at least one vertex of R. If three or more vertices of C are in R, then there is a chord because R is a clique. If exactly two vertices of C are in R and their clique edge in R is not an edge of C, then again C has a chord.

Hence we may focus on the cases that (a) only one vertex of C is in R or (b) exactly two vertices of C are in R and their clique edge in R belongs to C. In both cases C can visit only one connected component of G^l_{k-1} because the cycle is simple. Suppose that it visits the one rooted at R^l_i.

Cycle C must contain at least two vertices of R^l_i, again because C is simple. In case (a) consider the two vertices of R^l_i visited right before/after the unique vertex

of R. The clique edge of R_i^l linking these two vertices cannot belong to C because the cycle has length at least four. Thus C has a chord. In case (b) consider any two vertices of R_i^l contained in C. Each of them has edges to both vertices of C contained in R. Hence C has a chord.

4. Set R and each R_i^l, $1 \le i \le s$, form a clique of size d. The vertices of R are not connected to any vertices outside $R_i^l, 1 \le i \le s$. Hence no other cliques are formed by the addition of R. Since $\omega(G_{k-1}^l) = \omega(G_{k-1}^r) = d$ it follows that $\omega(G_k) = d$.
5. Again, let n_{k-1}^l and n_{k-1}^r be the number of vertices in G_{k-1}^l and G_{k-1}^l. We have $n_k = n_{k-1}^l + n_{k-1}^r + \frac{d}{2} \le 2 \cdot (\frac{d}{2} \cdot (2^k - 1)) + \frac{d}{2} = \frac{d}{2} \cdot (2^{k+1} - 2) + \frac{d}{2} = \frac{d}{2} \cdot (2^{k+1} - 1)$.

The construction and analysis of G_k is complete.

Graph G_k consists of several connected components if $k > 1$. The adversary can create a connected graph by adding a final vertex v_f that has an edge to exactly one root vertex in each of the components. The resulting graph remains chordal because there is no simple cycle containing v_f. By the addition of v_f the maximum clique size does not change. Including v_f the total number of vertices is upper bounded by $\frac{d}{2}(2^{k+1} - 1) + 1 \le d2^k$ because $d \ge 2$. The lemma follows from invariants (1) and (3–5) because $\chi(G_k) = \omega(G_k) = d$.

We finally address the case that d is odd. In this case the adversary executes the graph construction described above for parameter $d - 1$, which is even. In the end when G_k is generated for the desired k, the adversary adds a final vertex to each base graph G_1. This vertex has edges to every other vertex of the corresponding G_1. This increases the maximum clique size from $d - 1$ to d. The new graph remains chordal. The number of colors used by algorithm $\mathcal{A}$ is at least $\frac{d-1}{4}k$. We observe that the number of base graphs G_1 in G_k is 2^{k-1}. Hence, in the extended graph the total number of vertices is upper bounded by $\frac{d-1}{2}(2^{k+1} - 1) + 2^{k-1} \le \frac{d}{2}(2^{k+1} - 1)$. If $k > 1$, as described above, the adversary can add a final vertex to link the various components. Again the lemma follows. □

Proof of Theorem 1 Given d and n, let $k = \lfloor \log(n/d) \rfloor$. It holds that $k \in \mathbb{N}$ because $n \ge 2d^2 > 2d$. For every deterministic online algorithm, by Lemma 1, there exists a chordal graph G_k with chromatic number $\chi(G_k) = d$ such that $\mathcal{A}$ uses at least $c_k \ge (d - 1)k/4$ colors. Graph G_k has $n_k \le d2^k$ vertices. By the choice of $k = \lfloor \log(n/d) \rfloor$, we have $n_k \le n$. To G_k we add $n - n_k$ vertices, all of which have one edge to an arbitrary vertex of G_k. The resulting n-vertex graph remains chordal and $\chi(G) = d$. Since $d \ge 2$, it holds that $c_k \ge dk/8$. We have $k \ge \log n - \log d - 1$. Inequality $n \ge 2d^2$ is equivalent to $d \le \sqrt{n/2}$. Thus, $k \ge \log(n/2) - 1/2 \cdot \log(n/2) = 1/2 \cdot \log(n/2)$. As $n \ge 2d^2 \ge 4$, we have $\log(n/2) \ge 1/2 \cdot \log n$. Hence, the number of colors used by $\mathcal{A}$ is at least $c_k \ge d \log n/32$. □

In Theorem 1 the lower bound on n can be reduced from $2d^2$ to $2d^{1+\epsilon}$, for any $0 < \epsilon < 1$. If $n \ge 2d^{1+\epsilon}$, then $d \le (n/2)^{\frac{1}{1+\epsilon}}$ and in the above proof

$k \geq \log(n/2) - \frac{1}{1+\epsilon}\log(n/2) = \frac{\epsilon}{1+\epsilon}\log(n/2)$. Hence the number of colors used by $\mathcal{A}$ is $\Omega(\epsilon \cdot d \cdot \log n)$.

3 Randomized Online Algorithms for Chordal Graphs

We extend the result of Theorem 1 to randomized algorithms against oblivious adversaries.

Theorem 2 *Let $d \in \mathbb{N}$ with $d \geq 2$ be arbitrary. For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 12d^2$, there exists a n-vertex chordal graph G with chromatic number $\chi(G) = d$, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(d \cdot \log n)$.*

In order to prove Theorem 2 we resort to Yao's principle [33] and show the following Lemma 2.

Lemma 2 *Let $d \in \mathbb{N}$ with $d \geq 2$ be arbitrary. For every $k \in \mathbb{N}$, there exists a probability distribution on a set $\mathcal{G}_k$ of chordal graphs with the following properties. For every $G_k \in \mathcal{G}_k$, $\chi(G_k) = d$ and the number of vertices is at most $d \cdot 12^k$. The expected number of colors used by any deterministic online algorithm to color a graph drawn according to the distribution is at least $(d-1)k/8$.*

Proof For every $k \in \mathbb{N}$ we define a set $\mathcal{G}_k$ of chordal graphs G_k, each having a chromatic number of d. Moreover, we specify the order in which the vertices of any $G_k \in \mathcal{G}_k$ are presented to a deterministic online algorithm $\mathcal{A}$. The distribution on $\mathcal{G}_k$ is the uniform one, i.e. each $G_k \in \mathcal{G}_k$ is chosen with the same probability. We assume that d is even. The definition of $\mathcal{G}_k$ can be adapted easily if d is odd; details are given at the end of the proof.

The set $\mathcal{G}_k$ is built recursively based on $\mathcal{G}_{k-1}$. The construction of graphs $G_k \in \mathcal{G}_k$ is a generalization of the one presented in the proof of Lemma 1. A major difference is that any $G_k \in \mathcal{G}_k$ contains twelve graphs of $\mathcal{G}_{k-1}$, which are grouped into six pairs. For each pair a clique of size $d/2$ may or may not be added. As before, every $G_k \in \mathcal{G}_k$ is a union of connected components. Each such component can be represented by a tree with a distinguished root vertex. Every tree vertex is a set of $d/2$ vertices forming a clique in G_k. We reuse the notation of the proof of Lemma 1. Given $G_k \in \mathcal{G}_k$, for any component/tree T of G_k, $r(T)$ is the set of $d/2$ vertices in the root of T. Set $r(G_k)$ is the union of all $r(T)$, taken over all T of G_k. Finally $\mathcal{C}_{\mathcal{A}}(r(G_k))$ is the set of colors used by $\mathcal{A}$ for the vertices of $r(G_k)$.

During the recursive construction of $\mathcal{G}_k$, for increasing $k \in \mathbb{N}$, the following invariants are maintained. Compared to the proof of Lemma 1, (1) and (5) differ. Invariant (1) states that, for a randomly chosen G_k, every deterministic online algorithm needs, with probability greater than 1/2, at least $dk/4$ colors for the root vertices $r(G_k)$. Invariant (5) gives an adjusted bound on the size of any G_k.

1. If G_k is chosen uniformly at random from $\mathcal{G}_k$, then for any deterministic online algorithm $\mathcal{A}$, $\Pr[\left|\mathcal{C}_{\mathcal{A}}\left(r(G_k)\right)\right| \geq dk/4] > 1/2$. This holds independently of other connected components $\mathcal{A}$ might have already colored.
2. Every $G_k \in \mathcal{G}_k$ is a union of connected components, each of which can be represented by a tree T. Each tree node is a clique of size $d/2$. Every tree T has a distinguished root containing a set $r(T)$ of $d/2$ root vertices in G_k.
3. Every $G_k \in \mathcal{G}_k$ is chordal.
4. For every $G_k \in \mathcal{G}_k$, the maximum clique size is $\omega(G_k) = d$.
5. For every $G_k \in \mathcal{G}_k$, the number n_k of vertices satisfies $n_k \leq d(12^k - 1)$.

Graph Set $\mathcal{G}_1$: The set only contains G_1, the base graph used in the proof of Lemma 1, which is a clique of size d. The vertices of G_1 may be presented in any order to a deterministic online algorithm. Again, the set $r(G_1)$ of root vertices is an arbitrary subset of size $d/2$ of the vertices of G_1. The remaining $d/2$ vertices form a second tree node. Every deterministic online algorithm, with probability 1, needs $d/2$ colors for $r(G_1)$, which implies (1). Invariants (2–4) are obvious. As for (5), we have $n_1 = d \leq d(12 - 1)$.

Graph Set $\mathcal{G}_k$, $k > 1$: Assume that the set $\mathcal{G}_{k-1}$ satisfying (1–5) has been constructed. First, in order to build $\mathcal{G}_k$, all possible 12-tuples of graphs of $\mathcal{G}_{k-1}$ are formed. In assigning tuple entries, graphs of $\mathcal{G}_{k-1}$ are selected with replacement. Hence, a total of $\left|\mathcal{G}_{k-1}\right|^{12}$ tuples are built. Then, for each 12-tuple, 2^6 graphs are actually added to $\mathcal{G}_k$ in the following way. Let τ be any fixed tuple. Six graph pairs are formed. For $i = 1, \ldots, 6$, let $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$ be the graphs in tuple entries $2i - 1$ and $2i$, respectively. To the i-th pair a clique R_i of size $d/2$ may or may not be added. The possible additions, over the six pairs, can be represented by a bit vector $\mathbf{b} = (b_1, \ldots, b_6)$. Depending on the value of each bit, one of the two options of the deterministic construction is used. More specifically, given τ and any such bit vector $\mathbf{b}$, a graph G_k is constructed as follows. For $i = 1, \ldots, 6$, a subgraph G^i_k is generated. If $b_i = 0$, then G^i_k is the union of $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$. The set $r(G^i_k)$ of root vertices is the union of $r(G^{i,l}_{k-1})$ and $r(G^{i,r}_{k-1})$. If $b_i = 1$, then a clique R_i of size $d/2$ is added to $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$. Every vertex of R_i has an edge to every vertex of $r(G^{i,l}_{k-1})$. Subgraph G^i_k consists of the newly created component rooted at R_i and $r(G^{i,r}_{k-1})$, i.e. $r(G^i_k) = R_i \cup r(G^{i,r}_{k-1})$. Graph G_k is the union of the G^i_k and the set $r(G_k)$ is the union of the $r(G^i_k)$, $1 \leq i \leq 6$. When G_k is presented to $\mathcal{A}$, the subgraphs G^i_k are revealed one by one, $1 \leq i \leq 6$. For each G^i_k the graphs $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$ are presented recursively. Finally, the vertices of R_i, if they exist, are shown. It remains to verify the invariants.

1. Let G_k be a graph drawn uniformly at random from $\mathcal{G}_k$. Consider any subgraph G^i_k, $1 \leq i \leq 6$, containing $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$. By the construction of $\mathcal{G}_k$, both $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$ represent graphs drawn uniformly at random from $\mathcal{G}_{k-1}$. Let $\mathcal{A}$ be any deterministic online algorithm. Invariant (1) for $k - 1$ implies $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i,l}_{k-1}))| \geq d(k-1)/4] > 1/2 \text{and} \Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i,r}_{k-1}))| \geq d(k-1)/4] > 1/2$. Moreover, since the last two inequalities hold independently for $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$,

we have $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i,l}_{k-1}))| \geq d(k-1)/4$ and $|\mathcal{C}_{\mathcal{A}}(r(G^{i,r}_{k-1}))| \geq d(k-1)/4] > 1/4$. Let $\mathcal{E}^i$ be the latter event that $|\mathcal{C}_{\mathcal{A}}(r(G^{i,l}_{k-1}))| \geq d(k-1)/4$ and $|\mathcal{C}_{\mathcal{A}}(r(G^{i,r}_{k-1}))| \geq d(k-1)/4$ hold.

Assume that $\mathcal{E}^i$ holds. There are two cases, which correspond to those analyzed in the proof of Lemma 1. If $|\mathcal{C}_{\mathcal{A}}(r(G^{i,l}_{k-1}) \cup r(G^{i,r}_{k-1}))| \geq dk/4$, then $|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| \geq dk/4$ if R_i is not added to $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$, which happens with probability 1/2. On the other hand, if $|\mathcal{C}_{\mathcal{A}}(r(G^{i,l}_{k-1}) \cup r(G^{i,r}_{k-1}))| < dk/4$, then the addition of R_i ensures that $|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| \geq dk/4$, as argued in Case 2 in the proof of Lemma 1. Again, R_i is added with probability 1/2. In either case, given $\mathcal{E}^i$, it holds that $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| \geq dk/4] \geq 1/2$.

We obtain $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| \geq dk/4] \geq \Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| \geq dk/4 \mid \mathcal{E}^i] \cdot \Pr[\mathcal{E}^i] \geq \frac{1}{2} \cdot \frac{1}{4} = \frac{1}{8}$. Equivalently, $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| < dk/4] \leq 7/8$. The last inequality is satisfied for each subgraph G^i_k, $1 \leq i \leq 6$, independently of other subgraphs generated so far. Again this holds true because $G^{i,l}_{k-1}$ and $G^{i,r}_{k-1}$ are graphs drawn uniformly at random from $\mathcal{G}_{k-1}$ and R_i is added or not added independently with probability 1/2.

If $|\mathcal{C}_{\mathcal{A}}(r(G_k))| < dk/4$, then $|\mathcal{C}_{\mathcal{A}}(r(G^{i}_{k}))| < dk/4$ must hold true for $i = 1, \ldots, 6$. The latter event occurs with probability at most $(7/8)^6$. We conclude $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G_k))| \geq dk/4] \geq 1 - (7/8)^6 > 1/2$. As before, this holds independently of $\mathcal{A}$'s coloring decisions made in other components.

Invariants (2–4) are immediate, based on the arguments given in the proof of Lemma 1. As for the number of vertices of any $G_k \in \mathcal{G}_k$, we observe that it is upper bounded by $12 \cdot d \cdot (12^{k-1} - 1) + 6 \cdot d/2 < d \cdot (12^k - 1)$.

If d is odd, the above construction of sets $\mathcal{G}_k$, $k \geq 1$, is performed for parameter $d - 1$. In $\mathcal{G}_1$, graph G_1 is extended by a single vertex having edges to all other vertices in G_1. Invariant (5) holds because any graph $G_k \in \mathcal{G}_k$ contains 12^{k-1} copies of G_1.

The lemma follows from (1) and (3–5). In particular, (1) implies that the expected number of colors used by any deterministic online algorithm is at least $1/2 \cdot (d-1)k/4 = (d-1)k/8$. □

Proof of Theorem 2 For the given d and n, choose $k = \lfloor \log(n/d) \rfloor$. In this proof, logarithms are base 12. We have $k \in \mathbb{N}$, because $n \geq 12d^2 > 12d$. By Lemma 2, there exists a probability distribution on a set $\mathcal{G}_k$ of chordal graphs with chromatic number d such that the expected number of colors used by every deterministic online algorithm is at least $(d-1)k/8$. The number of vertices of any graph in $\mathcal{G}_k$ is at most $d12^k$. Hence, by the choice of k, the number of vertices is upper bounded by n. For every $G_k \in \mathcal{G}_k$, we add a suitable number of vertices so that the total number of vertices is equal to n. Every new vertex has one edge to an arbitrary vertex in the original graph G_k. Hence, there exists a probability distribution on a set of n-vertex graphs with chromatic number d such that the expected number of colors used by any deterministic online algorithm is at least $(d-1)k/8$. By Yao's principle [33], for every randomized online algorithm, there exists an n-vertex chordal graph G with $\chi(G) = d$ such that the expected number of color is $c_k \geq (d-1)k/8 \geq dk/16$. We have $k \geq \log n - \log d - 1 = \log(n/12) - \log d \geq 1/2 \cdot \log(n/12)$, because $12d^2 \leq n$, and

hence $d \leq \sqrt{n/12}$. Since $d \geq 2$ and $12d^2 \leq n$, we have $\log(n/12) \geq 1/3 \cdot \log n$ and thus $c_k \in \Omega(d \cdot \log n)$. □

Again, in Theorem 2 we can reduce the lower bound on n from $12d^2$ to $12d^{1+\epsilon}$, for any $0 < \epsilon < 1$. The expected number of colors used by $\mathcal{A}$ is $\Omega(\epsilon \cdot d \cdot \log n)$.

4 Further Graph Classes

Given Theorem 2, we can derive lower bounds on the performance of randomized online coloring algorithms for other important graph classes.

4.1 Trees, Planar, Bipartite, *d*-Inductive and Bounded-Treewidth Graphs

We start with the basic class of trees.

Corollary 1 *For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 48$, there exists a n-vertex tree T, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color T is $\Omega(\log n)$.*

Proof The corollary follows from Lemma 2 and Theorem 2 because, for $d = 2$, the constructed graphs are trees. Every clique of size $d/2$ added in the construction is a singleton vertex. Indeed, every constructed graph is a forest whose components can be linked by an additional vertex. □

Since trees are planar and bipartite graphs, we obtain the following two corollaries.

Corollary 2 *For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 48$, there exists a n-vertex planar graph G, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(\log n)$.*

Corollary 3 *For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 48$, there exists a n-vertex bipartite graph G, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(\log n)$.*

Every chordal graph G is $(\chi(G) - 1)$-inductive and has treewidth $\omega(G) - 1 = \chi(G) - 1$ [9]. Hence, Theorem 2 gives the following two results.

Corollary 4 *Let $d \in \mathbb{N}$ be an arbitrary positive integer. For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 12d^2$, there exists a n-vertex d-inductive graph G, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(d \cdot \log n)$.*

Corollary 5 *Let $d \in \mathbb{N}$ be an arbitrary positive integer. For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 12d^2$, there exists a n-vertex graph G of treewidth d, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(d \cdot \log n)$.*

The following corollary gives a result for strongly chordal graphs.

Corollary 6 *Let $d \in \mathbb{N}$ be an arbitrary positive integer. For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \geq 12d^2$, there exists a n-vertex strongly chordal graph G with chromatic number $\chi(G) = d$, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(d \cdot \log n)$.*

Proof We prove that every graph $G_k \in \mathcal{G}_k$ constructed in Lemma 2 is strongly chordal. The corollary then immediately follows from Theorem 2. Let $N(v)$ denote the neighborhood of a vertex v in G_k. For $d \leq 3$, G_k can only contain cycles in the base graphs G_1 and their length is three. Hence G_k does not posses an even cycle and G_k is strongly chordal. For $d \geq 4$, consider an even cycle C of length at least six in G_k. We first argue that there must exist two non-consecutive vertices u and v in C that are part of the same tree node. If C visits only one or two tree nodes, this is obvious, given the length of C. If C visits at least three tree nodes, the desired fact follows from invariant (2) in Lemma 2, which ensures that each connected component of G_k forms a tree of tree nodes.

Hence let u and v be two non-consecutive vertices in C belonging to the same tree node w_T. As w_T is a clique, $\{u, v\}$ is an edge in G_k. Moreover $N(u)\backslash\{v\} = N(v)\backslash\{u\}$ because, again, w_T is a clique and its vertices are connected to the same vertices that are not part of w_T. Consider a neighbor s of v in C. We distinguish between two cases. First, if s is also a neighbor of u in C, then there must exist a neighbor x of u and a neighbor y of v in C because C has length at least six. Therefore, starting at x, cycle C visits vertices x, u, s, v, y in this order. We have $N(u)\backslash\{v\} = N(v)\backslash\{u\}$, which implies that $\{u, y\}$ is an edge in G_k. The distance between u and y in C is three so that $\{u, y\}$ is an odd chord of C. On the other hand, if s is not a neighbor of u in C, then $\{u, s\}$ is an edge in G_k since $N(u)\backslash\{v\} = N(v)\backslash\{u\}$. Thus $\{u, s\}$ is a chord of C. Moreover $\{u, v\}$ is a chord of C because u and v are non-consecutive in C. We conclude that either $\{u, v\}$ or $\{u, s\}$ is an odd chord of C because the distances between vertices u and v and vertices u and s in C differ by exactly one. □

4.2 Disk Graphs

A disk graph is the intersection graph of disks in the Euclidean plane. Every vertex corresponds to a disk; two vertices are connected by an edge if the respective disks intersect. The following theorem implies that it is not possible to improve on the performance of deterministic online coloring algorithms by using randomization. We use the common assumption that when an online algorithm makes coloring decisions, it does not use the disk representation [14, 17, 18].

Theorem 3 *Let $\mathcal{A}$ be an arbitrary randomized online algorithm. For every $n \in \mathbb{N}$ and $\rho \in \mathbb{R}$ with $\min\{n, \rho\} \geq 25$, there exists a n-vertex disk graph G with chromatic number $\chi(G) = 2$, presented by an oblivious adversary, in which the ratio of the largest to smallest disk radius is ρ, such that the expected number of colors used by $\mathcal{A}$ is $\Omega(\min\{\log n, \log \rho\})$.*

The proof of Theorem 3 relies on the following lemma, which we prove first.

Lemma 3 *For every $k \in \mathbb{N}$ and every $\rho \in \mathbb{R}$ with $\rho > 12^{k-1}$, there exists a probability distribution on a set $\mathcal{D}_k$ of disk graphs with the following properties. In every $D_k \in \mathcal{D}_k$, the number of vertices is at most $2 \cdot 12^k$, the ratio of the largest to smallest disk radius is ρ and $\chi(D_k) = 2$. The expected number of colors used by every deterministic online algorithm for a graph drawn according to the distribution is at least $k/8$.*

Proof We use the graph sets $\mathcal{G}_k$, constructed in Lemma 2, focusing on $d = 2$. Let $k \in \mathbb{N}$ be arbitrary. We show that if $\rho > 12^{k-1}$, every $G_k \in \mathcal{G}_k$ with n_k vertices translates to a disk graph D_k with $n_k + 1$ vertices in which the ratio of the largest to smallest disk radius is ρ.

Again $d = 2$. Assume that $\rho > 12^{k-1}$. Let $G_k \in \mathcal{G}_k$ be an arbitrary graph. The corresponding disk graph is constructed in a top-down manner. In order to slightly ease the description we also use the term *disk graph* to refer to the corresponding disk representation. In a first step we generate a graph D_k in which disks touch each other but do not intersect. At the end of the construction we slightly increase the disk radii so as to create intersections among the disks.

First in the construction of D_k a disk d^* of radius ρ, centered at the origin (0, 0), is placed in the Euclidean plane. Determine an $\epsilon > 0$ such that $\rho > 12^{k-1} + \epsilon(12^{k-1} - 1)12/11$. Such an ϵ exists because $\rho > 12^{k-1}$. Let S_k be the

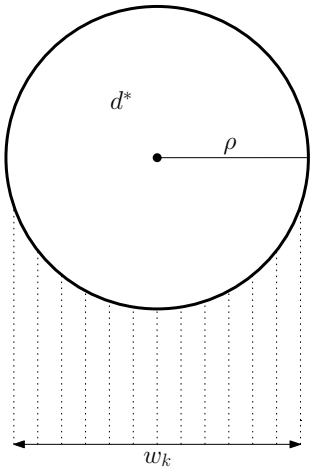

Fig. 4 The disk graph D_k

vertical strip of width $w_k = 2(12^{k-1} + \epsilon(12^{k-1} - 1)12/11)$ below d^*. The center line of S_k has x-coordinate 0. Figure 4 depicts the arrangement. Since d^* has diameter 2ρ and $2\rho > w_k$, disk d^* extends past both boundaries of S_k. In S_k a disk representation of G_k will be placed recursively below d^*. Recall that G_k consists of twelve graphs of $\mathcal{G}_{k-1}$ that form six pairs $(G^{i,l}_{k-1}, G^{i,r}_{k-1})$, $1 \le i \le 6$. To each pair a clique R_i of size $d/2$ might have been added. Since $d = 2$, this clique is a single vertex.

The disk representation of G_k in S_k is as follows. Strip S_k of width w_k is divided into twelve substrips of width $w_k/12 = 2(12^{k-2} + \epsilon(12^{k-1} - 1)/11)$ each, see again Fig. 4. In the m-th substrip a strip S^m_{k-1} of width $w_{k-1} = w_k/12 - 2\epsilon = 2(12^{k-2} + \epsilon(12^{k-2} - 1)12/11)$ is located, $1 \le m \le 12$. Strip S^m_{k-1} lies in the middle of the m-th substrip so that its left and right boundaries have a distance of ϵ from those of the substrip. The strips S^m_{k-1}, $1 \le m \le 12$, will host the representations of the twelve subgraphs of G_k.

Consider any pair $(G^{i,l}_{k-1}, G^{i,r}_{k-1})$, $1 \le i \le 6$. If R_i is added to the pair, a disk d_i of radius $r_{k-1} = w_{k-1}/2$ is placed in the odd numbered strip S^{2i-1}_{k-1} below d^*. Disk d_i is positioned so that it touches d^*. Then a representation of $G^{i,l}_{k-1}$ is placed in S^{2i-1}_{k-1} below d_i. Since $2r_k = w_k$, disk d_i fully covers S^{2i-1}_{k-1}. If R_i is not added to $(G^{i,l}_{k-1}, G^{i,r}_{k-1})$, a representation of $G^{i,l}_{k-1}$ is placed in S^{2i-1}_{k-1} below disk d^*. In any case a representation of $G^{i,r}_{k-1}$ is created in the neighboring strip S^{2i}_{k-1} below d^*. Since the left and right boundaries of S^m_{k-1} have a distance of ϵ from those of the m-th substrip containing S^m_{k-1}, disks and graph representations placed in S^m_{k-1} do not touch or overlap with disks placed in other strips S^h_{k-1}, $h \ne m$.

In general assume that a graph $G_j \in \mathcal{G}_j$, $k > j > 2$, has to be represented in a strip S_j of width $w_j = 2(12^{j-1} + \epsilon(12^{j-1} - 1)12/11)$ below a disk d. The construction proceeds in the same way as described in the last paragraph for $j = k$. Strip S_j is divided into twelve substrips, which in turn contain strips of width $w_{j-1} = 2(12^{j-2} + \epsilon(12^{j-2} - 1)12/11)$. These strips host the subgraphs of $\mathcal{G}_j$. For each pair of subgraphs for which a new vertex is added, a disk of radius $r_{j-1} = w_{j-1}/2$ is placed so that it touches disk d from below. The top-down construction ends when graphs G_1 have to be placed in a strip of width $w_1 = 2$ below a disk d. Graph G_1 is

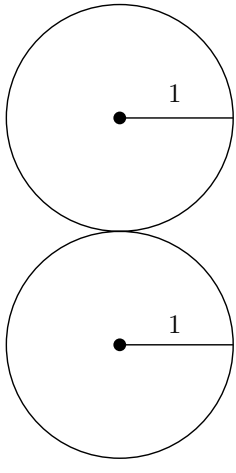

Fig. 5 The disk graph D_1

represented by a combination of two disks of radius 1, see Fig. 5. The two disks are placed on top of each other so that the upper one touches d from below.

Graph D_k is constructed in a top-down manner. However, when presented to an online coloring algorithm, the vertices are of course revealed bottom-up, with the vertices of graphs representing G_1 revealed first. Disk d^* in D_k does not correspond to a vertex in G_k. For every other vertex of G_k, exactly one disk was introduced. Hence, the analysis in the proof of Lemma 2 implies that D_k contain at most $1 + 2(12^k - 1) \leq 2 \cdot 12^k$ vertices. In D_k the contact points among disks correspond to edges in G_k. More precisely, any two disks d and d' touch each other in D_k if and only if the vertices corresponding to d and d' are connected by an edge in G_k. It is easy to modify D_k so that the contact points are replaced by real intersections among the respective disks. Let δ be the minimum distance of any disks that do not touch each other. The radius of disk d^* is increased by $\delta/2$. For every other disk the radius is increased by $\delta/(2\rho)$.

Let D_k be the set of all disk graphs generated for any $G_k \in \mathcal{G}_k$. Consider the uniform distribution on $\mathcal{D}_k$. By Lemma 2, if a graph is drawn uniformly at random from $\mathcal{D}_k$, the expected number of colors used by any deterministic online algorithm is at least $k/8$. □

Proof of Theorem 3 Let $k = \lfloor \log(\min\{n, \rho\}/2) \rfloor$. Logarithms are base 12. Since $\min\{n, \rho\} \geq 25$, there holds $k \in \mathbb{N}$. Moreover, $\rho > 12^{k-1}$. Consider the set $\mathcal{D}_k$ of disk graphs, defined in Lemma 3, each of which consists of at most $2 \cdot 12^k$ disks. Hence by the choice of k, they consist of at most n disks. To each $D_k \in \mathcal{D}_k$ with, say, n_k disks we add $n - n_k$ additional disks. Their radius may be an arbitrary value between the smallest and the largest disk radius occurring in D_k. Lemma 3, together with Yao's principle [33], implies that for every randomized online algorithm there exists an n-vertex disk graph in which the ratio of the largest to smallest disk radius is ρ such that the expected number of colors used by $\mathcal{A}$ is at least $k/8$. It holds that $k \geq \log(\min\{n, \rho\}/2) - 1 = \log(\min\{n, \rho\}) - \log(24) \geq 1/100 \cdot \log(\min\{n, \rho\})$ because $\min\{n, \rho\} \geq 25$. We conclude that $k/8 \in \Omega(\min\{\log n, \log \rho\})$. □

5 Lookahead and Buffer Reordering

We explore the settings where an online algorithm has lookahead or is equipped with a reordering buffer.

5.1 Lookahead

We first assume that a randomized online coloring algorithm $\mathcal{A}$ has lookahead l, i.e. when presented with vertex v_t the algorithm also sees $v_{t+1}, \ldots, v_{t+l}$ along with their adjacent edges in $\{v_1, \ldots, v_{t+l}\}$. Theorem 4 below shows that, for chordal graphs, a lookahead of size $O(n/\log n)$ leads to no improvement.

Theorem 4 *Let $d \in \mathbb{N}$ and $c \in \mathbb{R}$ be arbitrary numbers with $d \geq 2$ and $c \geq 1$. For every randomized online algorithm $\mathcal{A}$ with lookahead l and every $n \in \mathbb{N}$ with $n \geq \max\{12d^2, d \cdot 12^{2c}\}$ and $l \leq cn/\log(n/d)$, there exists a n-vertex chordal graph G with chromatic number $\chi(G) = d$, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(\frac{1}{c} \cdot d \cdot \log n)$.*

Proof For any $l \in \mathbb{N}$, consider the class of deterministic online algorithms with lookahead l. We refine the graph sets $\mathcal{G}_k$, defined in the proof of Lemma 2, by specifying an order in which vertices arrive and by extending the individual graphs. Let $k \in \mathbb{N}$ be arbitrary and $G_k \in \mathcal{G}_k$ be any graph. The vertices of G_k are presented to an online coloring algorithm in phases, based on the subgraphs $G_j \in \mathcal{G}_j$ with $j < k$ contained in G_k.

More precisely, at the bottom level G_k contains several instances of G_1. The vertices of all the copies of G_1 form a set P_1. They are presented first and are part of phase 1. Next G_k contains graphs $G_2 \in \mathcal{G}_2$. Let P_2 be the set of vertices in all instances of $G_2 \in \mathcal{G}_2$ that are not contained in P_1. Using the notation of the proof of Lemma 2, the vertices of P_2 belong to sets R_i that are added to graph pairs of G_1. In general, let P_j be the set of vertices in instances of $G_j \in \mathcal{G}_j$ that are not contained in $P_1 \cup \ldots \cup P_{j-1}$, $1 < j \leq k$. Graph G_k is presented to an online algorithm $\mathcal{A}$ by revealing the vertices of P_j, for increasing $j = 1, \ldots, k$. The vertex sequence of P_j forms phase j, $1 \leq j \leq k$.

In order to render $\mathcal{A}$'s lookahead useless, at the end of each phase j, exactly l new dummy vertices are presented, $1 \leq j \leq k$. These new vertices can for example be isolated vertices. Alternatively, they could be combined to form a chain of cliques having size at most d. The new vertices increase the graph size by no more than kl. When coloring the vertices of P_j, an online algorithm with lookahead l has no information about vertices of $P_{j'}$, $j' > j$. Let $\mathcal{G}_k$ be the set of all extended graphs. Lemma 2 implies that if a graph is drawn uniformly at random from $\mathcal{G}_k$, the expected number of colors used by any deterministic online algorithm with lookahead is at least $(d-1)k/8$.

We conclude that for any $k, l \in \mathbb{N}$, there exists a probability distribution on a set G_k of chordal graphs with the following properties. For every $G_k \in \mathcal{G}_k$, $\chi(G_k) = d$ and the number of vertices satisfies $n_k \leq d \cdot 12^k + kl$. The expected number of colors used by any deterministic online algorithm with lookahead l is at least $(d-1)k/8$.

In order to prove the theorem, for d, c and n with the stated properties, choose $k = \lfloor \frac{2}{3c} \cdot \log(\frac{n}{d}) \rfloor$. Logarithms are base 12. We have $k \in \mathbb{N}$ because $n \geq d \cdot 12^{2c}$. Consider the set $\mathcal{G}_k$ of extended graphs defined above. Each graph in $\mathcal{G}_k$ has at most $d \cdot 12^k + kl$ vertices. We argue that, for $l \leq cn/\log(n/d)$, this expression is upper bounded by n. For the chosen k, we have $d \cdot 12^k \leq d^{1/3} n^{2/3} \leq n/3$. The first inequality holds because $c \geq 1$. The second inequality is equivalent to $27 \leq n/d$, which holds for $n \geq d \cdot 12^{2c}$. By the choice of k, if $l \leq cn/\log(n/d)$, then $kl \leq 2n/3$. Hence, as desired, $d \cdot 12^k + kl \leq n$. To each graph of $\mathcal{G}_k$ we add a suitable number of vertices so that the graph size is exactly n.

Using Yao's principle we obtain that, for every randomized online algorithm $\mathcal{A}$ with lookahead l, where $l \leq cn/\log(n/d)$, there exists an n-vertex chordal graph G with $\chi(G) = d$ such that the expected number of colors used

by $\mathcal{A}$ is at least $c_k \geq (d-1)k/8$, which in turn is lower bounded by $dk/16$. We have $k \geq 2/(3c) \cdot \log(n/d) - 1 = 2/(3c) \cdot (\log(n/d) - (3c/2))$. Moreover, $\log(n/d) - (3c/2) \geq (1/4) \cdot \log(n/d)$, for $n \geq d \cdot 12^{2c}$. Also, $\log(n/d) \geq 1/2 \cdot \log n$, for $n \geq d^2$. In conclusion, $k \geq 1/(12c) \cdot \log(n)$ and therefore $c_k \in \Omega(\frac{1}{c} \cdot d \cdot \log(n))$. □

Based on Theorem 4 we can derive analogous results for all the other graph classes considered in Sect. 4. Loosely speaking, a lookahead of size $O(n/\log n)$ is of no help. The next Corollary 7 addresses trees. It directly follows from Theorem 4 by setting $d = 2$. The statement of Corollary 7 also holds in exactly the same formulation for planar and bipartite graphs, respectively. For brevity, we omit the corresponding corollaries.

Corollary 7 *Let $c \geq 1$ be an arbitrary real number. For every randomized online algorithm $\mathcal{A}$ with lookahead l and every $n \in \mathbb{N}$ with $n \geq \max\{48, 2 \cdot 12^{2c}\}$ and $l \leq cn/\log(n/2)$, there exists a n-vertex tree G, presented by an oblivious adversary, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(\frac{1}{c} \cdot \log n)$.*

For d-inductive graphs, graphs of treewidth d and strongly chordal graphs with chromatic number d, the formulation of Theorem 4 directly carries over. In fact, the result holds for all integers $d \geq 1$. For disk graphs, Theorem 4 together with the disk graph construction of Theorem 3 gives the following corollary.

Corollary 8 *Let $c \in \mathbb{R}$ with $c \geq 1$ be arbitrary. For every randomized online algorithm $\mathcal{A}$ with lookahead l, every $n \in \mathbb{N}$ and $\rho \in \mathbb{R}$ with $\min\{n, \rho\} \geq 2 \cdot 12^{2c}$ and $l \leq cn/\log(n/2)$, there exists a n-vertex disk graph G with chromatic number $\chi(G) = 2$, presented by an oblivious adversary, in which the ratio of the largest to smallest disk radius is ρ, such that the expected number of colors used by $\mathcal{A}$ to color G is $\Omega(\frac{1}{c} \cdot \log n)$.*

5.2 Buffer Reordering

Next we examine the setting in which a deterministic online coloring algorithm $\mathcal{A}$ has a reordering buffer. A newly arriving vertex may be stored in the buffer if capacity permits. In any time step algorithm $\mathcal{A}$ may color the incoming vertex of vertices from the buffer. Given a buffer of size b, the constraint is that, after time step t, $\mathcal{A}$ must have colored at least $t - b$ vertices. We prove that a buffer of size $n^{1-\epsilon}$, for any $0 < \epsilon \leq 1$, does not improve the asymptotic performance of the algorithms.

Theorem 5 *Let $d \in \mathbb{N}$ and $\epsilon \in \mathbb{R}$ be arbitrary numbers with $d \geq 2$ and $0 < \epsilon \leq 1$. For every deterministic online algorithm $\mathcal{A}$ having a buffer of size b and every $n \in \mathbb{N}$ with $b \leq n^{1-\epsilon}$ and $n \geq \max\{2d^2, 2^{7/\epsilon}\}$, there exists a n-vertex chordal graph G with chromatic number $\chi(G) = d$ such that the number of colors used by $\mathcal{A}$ is $\Omega(\epsilon \cdot d \cdot \log n)$.*

Proof We extend the adaptive graph construction presented in the proof of Lemma 1 and, as in the proof of Theorem 4, let the adversary generate a graph in a bottom-up fashion. Given d, ϵ and n with the stated properties, let $k = \lfloor \log(n/d) \rfloor$. The adversary constructs a graph $G_k \in \mathcal{G}_k$ consisting of 2^{k-j} subgraphs $G_j \in \mathcal{G}_j$, for any $1 \leq j \leq k$. In phase 1, 2^{k-1} graphs G_1 are constructed. As always each G_1 is a clique of size d in which $d/2$ arbitrary vertices form a set of distinguished root vertices. We assume that d is even and address the case that d is odd at the end of the proof. The vertices of all the copies of G_1 may be presented in an arbitrary order to the deterministic online algorithm $\mathcal{A}$. In general in phase j, $1 < j \leq k$, the adversary presents the vertices of subgraphs $G_j \in \mathcal{G}_j$ that have not been revealed in previous phases.

More specifically, let $k' = \lfloor \frac{1}{2}(k - \log(4n^{1-\epsilon}/d)) \rfloor$. We have $k' \geq 1$: The last inequality is satisfied if $\frac{1}{2}(\log(n/d) - 1 - \log(4n^{1-\epsilon}/d)) - 1 \geq 1$. This inequality in turn is equivalent to $n \geq 2^{7/\epsilon}$, which holds true by the choice of n. Consider any phase j, $1 \leq j \leq k'$. We say that algorithm $\mathcal{A}$ *has made progress on a subgraph* $G_j \in \mathcal{G}_j$ if at the end of phase j the algorithm has colored at least half of the root vertices of G_j. We prove that at the end of every phase j, $1 \leq j \leq k'$, the following invariant (1) holds. Invariants (2–5) are as in the proof of Lemma 1.

1. At the end of phase j, $\mathcal{A}$ has made progress on at least 2^{k-2j} subgraphs $G_j \in \mathcal{G}_j$. For each of these subgraphs, $\left|\mathcal{C}_A\left(r(G_j)\right)\right| \geq \frac{d}{8}j$.

For $j = 1$, the analysis is simple. Suppose that at the end of phase 1 $\mathcal{A}$ has made progress on less than 2^{k-2} subgraphs G_1. Then there exist more than $2^{k-1} - 2^{k-2} = 2^{k-2}$ graphs G_1 for which less than half of the root vertices have been colored. Thus at the end of phase 1 the buffer must contain more than $\frac{d}{4}2^{k-2}$ vertices. We observe that for any j with $1 \leq j \leq k'$, it holds that $\frac{d}{4}2^{k-2j} \geq n^{1-\epsilon}$ because the latter inequality is equivalent to $\frac{1}{2}(k - \log(4n^{1-\epsilon}/d)) \geq j$, which is satisfied by the choice of k'. Since the buffer size is at most $n^{1-\epsilon}$, $\mathcal{A}$ cannot store more than $\frac{d}{4}2^{k-2}$ vertices in the buffer at the end of phase 1. Hence $\mathcal{A}$ must have made progress on at least 2^{k-2} subgraphs G_1. For each of those subgraphs at least half of the root vertices have been colored, i.e. at least $d/4 > d/8$ colors have been used.

Assume that invariant (1) holds for phases $1, \ldots, j-1$, where $j \leq k'$. The adversary takes $2^{k-2(j-1)}$ subgraphs $G_{j-1} \in \mathcal{G}_{j-1}$ for which $\mathcal{A}$ has made progress and $\left|\mathcal{C}_A\left(r(G_{j-1})\right)\right| \geq \frac{d}{8}(j-1)$ holds. The adversary pairs them in an arbitrary way so that 2^{k-2j+1} graph pairs are formed. Consider any such pair (G^l_{j-1}, G^r_{j-1}). By inductive assumption $|\mathcal{C}_\mathcal{A}(r(G^l_{j-1}))| \geq \frac{d}{8}(j-1)$ and $|\mathcal{C}_\mathcal{A}(r(G^r_{j-1}))| \geq \frac{d}{8}(j-1)$. If $|\mathcal{C}_\mathcal{A}(r(G^l_{j-1}) \cup r(G^r_{j-1}))| \geq \frac{d}{8}j$, then the adversary creates a graph G_j that is simply the union of G^l_{j-1} and G^r_{j-1}. No further vertices are added. On the other hand, if $|\mathcal{C}_\mathcal{A}(r(G^l_{j-1}) \cup r(G^r_{j-1}))| < \frac{d}{8}j$, the adversary creates a graph G_j by adding a clique R of size $d/2$ to (G^l_{j-1}, G^r_{j-1}). Each vertex of R has an edge to every vertex of $r(G^l_{j-1})$. As in the proof of Lemma 1 we can show that if $\mathcal{A}$ has colored at least half of the vertices of R, it holds that $|\mathcal{C}_\mathcal{A}(r(G_j))| \geq \frac{d}{8}j$. Phase j consists of the arrival of the vertices of R, taken over all the 2^{k-2j+1} graph pairs for which such a clique is added.

Finally, the adversary takes the subgraphs G_{j-1} not combined so far and pairs them in an arbitrary way so as to create graphs G_j. No further vertices are added.

It remains to verify that invariant (1) holds. Again, consider the 2^{k-2j+1} graph pairs composed of subgraphs G_{j-1} satisfying invariant (1) for phase $j-1$. The adversary has constructed 2^{k-2j+1} corresponding subgraphs G_j. We argue that at the end of phase j, $\mathcal{A}$ has made progress on at least half of them. Consider any G_j, based on graph pair (G^l_{j-1}, G^r_{j-1}). If no clique has been added, then $\mathcal{A}$ has made progress on G_j, because by inductive assumption $\mathcal{A}$ has colored at least half of the root vertices of G^l_{j-1} and G^r_{j-1}. On the other hand, if a clique R had been added and $\mathcal{A}$ has not made progress on G_j, then more than $d/4$ vertices of R must reside in the buffer at the end of phase j. Hence, if $\mathcal{A}$ had not made progress on more than half of the 2^{k-2j+1} considered subgraphs G_j, then more than $\frac{d}{4} \cdot \frac{1}{2} 2^{k-2j+1} = \frac{d}{4} 2^{k-2j}$ vertices must reside in the buffer. This is impossible because, as verified above when analyzing the case $j = 1$, $\frac{d}{4} 2^{k-2j} \geq n^{1-\epsilon}$. We obtain that $\mathcal{A}$ has made progress on at least $\frac{1}{2} 2^{k-2j+1} = 2^{k-2j}$ subgraphs G_j. For each of these subgraphs, the potential addition of a clique R ensures that $\mathcal{A}$ must use at least $\frac{d}{8} j$ colors for the root vertices.

After phase k' the formation of graphs G_j, $k' < j \leq k$ is simple. The adversary takes arbitrary pairs of graphs G_{j-1} and combines them to form graphs G_j. No further vertices are added. Finally, when the generation of a graph G_k consisting of, say, n_k vertices is complete, the adversary adds $n - n_k$ vertices to form a final graph G with n vertices. Invariant (1) for $j = k'$ ensures that $\mathcal{A}$ uses at least $\frac{d}{8} k'$ colors. We show that k' is in $\Omega(\epsilon \cdot \log n)$. It holds that $k' \geq \frac{1}{2}(\log(n/d) - 1 - \log(4n^{1-\epsilon}/d)) - 1 = \frac{1}{2}(\epsilon \log n - 5) \geq \frac{1}{8} \epsilon \log n$, for $n \geq 2^{7/\epsilon}$.

Finally, if d is odd, the above graph construction is performed for $d-1$. A single vertex is added to each subgraph G_1 to form a graph with clique size d. □

We remark that Theorem 5, as presented, does not hold for randomized algorithms. The graph construction involves an adversarial strategy, picking and combining subgraphs for which an algorithm has made progress. Choosing them at random will not give a sufficiently high success probability when going through the phases of the construction.

Given Theorem 5, we derive analogous results for the other graph classes. Corollary 9 shows a result for trees, by choosing again $d = 2$. Identical statements hold for planar and bipartite graphs. Again, for brevity, we omit the corresponding corollaries.

Corollary 9 *Let $\epsilon \in \mathbb{R}$ with $0 < \epsilon \leq 1$ be arbitrary. For every deterministic online algorithm $\mathcal{A}$ having a buffer of size b and every $n \in \mathbb{N}$ with $b \leq n^{1-\epsilon}$ and $n \geq 2^{7/\epsilon}$, there exists a n-vertex tree G such that the number of colors used by $\mathcal{A}$ is $\Omega(\epsilon \cdot \log n)$.*

For d-inductive graphs, graphs of treewidth d and strongly chordal graphs with chromatic number d, the statement of Theorem 5 directly carries over. In this case it holds for any $d \geq 1$. The corollaries are omitted here. Finally, we give a result for disk graphs.

Corollary 10 *Let $\mathcal{A}$ be an arbitrary deterministic online algorithm having a buffer of size b and let $\epsilon \in \mathbb{R}$ be an arbitrary real number with $0 < \epsilon \leq 1$. For every $n \in \mathbb{N}$ and $\rho \in \mathbb{R}$ with $b \leq \min\{n^{1-\epsilon}, \rho^{1-\epsilon}\}$ and $\min\{n, \rho\} \geq 2^{7/\epsilon}$, there exists a n-vertex disk graph G with chromatic number $\chi(G) = 2$, in which the ratio of the largest to smallest disk radius is ρ, such that the number of colors used by $\mathcal{A}$ is $\Omega(\epsilon \cdot \min\{\log n, \log \rho\})$.*

Acknowledgements We thank anonymous referees for their valuable comments.

Funding Open Access funding provided by Projekt DEAL.

References

1. Avigdor-Elgrabli, N., Rabani, Y.: An optimal randomized online algorithm for reordering buffer management. In: Proceedings of 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS'13), pp. 1–10 (2013)
2. Avigdor-Elgrabli, N., Rabani, Y.: An improved competitive algorithm for reordering buffer management. ACM Trans. Algorithms **11**(4), 35:1–35:15 (2015)
3. Bar-Noy, A., Cheilaris, P., Olonetsky, S., Smorodinsky, S.: Online conflict-free colouring for hypergraphs. Comb. Probab. Comput. **19**(4), 493–516 (2010)
4. Bar-Noy, A., Cheilaris, P., Smorodinsky, S.: Deterministic conflict-free coloring for intervals: from offline to online. ACM Trans. Algorithms **4**(4), 44:1–44:18 (2008)
5. Bean, D.: Effective coloration. J. Symb. Log. **41**(2), 469–480 (1976)
6. Ben-David, S., Borodin, A., Karp, R., Tardos, G., Wigderson, A.: On the power of randomization in on-line algorithms. Algorithmica **11**(1), 2–14 (1994)
7. Berge, C.: Färbung von Graphen, deren sämtliche bzw. deren ungerade Kreise starr sind. Wiss. Z. Martin-Luther Univ. Halle-Wittenberg Math. Natur. Reihe **10**(4), 114 (1961)
8. Bianchi, M.P., Böckenhauer, H.-J., Hromkovic, J., Keller, L.: Online coloring of bipartite graphs with and without advice. Algorithmica **70**(1), 92–111 (2014)
9. Bodlaender, H.L.: A tourist guide through treewidth. Acta Cybern. **11**(1–2), 1–21 (1993)
10. Boyar, J., Epstein, L., Favrholdt, L.M., Larsen, K.S., Levin, A.: Batch coloring of graphs. Algorithmica **80**(11), 3293–3315 (2018)
11. Boyar, J., Favrholdt, L.M., Kudahl, C., Larsen, K.S., Mikkelsen, J.W.: Online algorithms with advice: a survey. ACM Comput. Surv. **50**(2), 19:1–19:34 (2017)
12. Burjons, E., Hromkovic, J., Královic, R., Královic, R., Muñoz, X., Unger, W.: Online graph coloring against a randomized adversary. Int. J. Found. Comput. Sci. **29**(4), 551–569 (2018)
13. Burjons, E., Hromkovic, J., Muñoz, X., Unger, W.: Online graph coloring with advice and randomized adversary. In: Proceedings of the 42nd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'16), pp. 229–240. LNCS 9587, Springer (2016)
14. Caragiannis, I., Fishkin, A.V., Kaklamanis, C., Papaioannou, E.: A tight bound for online colouring of disk graphs. Theoret. Comput. Sci. **384**(2), 152–160 (2007)

15. Downey, R.G., McCartin, C.: Online promise problems with online width metrics. J. Comput. Syst. Sci. **73**(1), 57–72 (2007)
16. Englert, M., Özmen, D., Westermann, M.: The power of reordering for online minimum makespan scheduling. SIAM J. Comput. **43**(3), 1220–1237 (2014)
17. Erlebach, T., Fiala, J.: On-line coloring of geometric intersection graphs. Comput. Geom. **23**(2), 243–255 (2002)
18. Erlebach, T., Fiala, J.: Independence and coloring problems on intersection graphs of disks. In: Bampis, E., Jansen, K., Kenyon, C. (eds.) Efficient Approximation and Online Algorithms: Recent Progress on Classical Combinatorial Optimization Problems and New Applications, pp. 135–155. LNCS 3484, Springer (2006)
19. Farber, M.: Characterizations of strongly chordal graphs. Discrete Math. **43**(2–3), 173–189 (1983)
20. Gyárfás, A., Lehel, J.: On-line and first fit colorings of graphs. J. Graph Theory **12**(2), 217–227 (1988)
21. Halldórsson, M.M.: Parallel and on-line graph coloring. J. Algorithms **23**(2), 265–280 (1997)
22. Halldórsson, M.M., Szegedy, M.: Lower bounds for on-line graph coloring. Theoret. Comput. Sci. **130**(1), 163–174 (1994)
23. Irani, S.: Coloring inductive graphs on-line. Algorithmica **11**(1), 53–72 (1994)
24. Kierstead, H.A.: Coloring graphs on-line. In: Fiat, A., Woeginger, G.J. (eds.) Online Algorithms, pp. 281–305. LNCS 1442, Springer (1998)
25. Kierstead, H.A., Trotter, W.A.: An extremal problem in recursive combinatorics. Congres. Numer. **33**, 143–153 (1981)
26. Leonardi, S., Vitaletti, A.: Randomized lower bounds for online path coloring. In: Proceedings of the 2nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM'98), pp. 232–247. LNCS 1518, Springer (1998)
27. Lovász, L., Saks, M., Trotter, W.T.: An on-line graph coloring algorithm with sublinear performance ratio. Ann. Discret. Math. **43**, 319–325 (1989)
28. Marx, D.: Graph colouring problems and their applications in scheduling. Period. Polytech. Electr. Eng. **48**(1–2), 11–16 (2004)
29. Narayanan, L.: Channel assignment and graph multicoloring. In: Handbook of Wireless Networks and Mobile Computing, pp. 71–94 (2004)
30. Sleator, D.D., Tarjan, R.E.: Amortized efficiency of list update and paging rules. Commun. ACM **28**(2), 202–208 (1985)
31. Vishwanathan, S.: Randomized online graph coloring. J. Algorithms **13**(4), 657–669 (1992)
32. West, D.B.: Introduction to Graph Theory, 2nd edn. Pearson, London (2001)
33. Yao, A.C.C.: Probabilistic computations: toward a unified measure of complexity. In: Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS'77), pp. 222–227 (1977)