Tight Bounds for Online Coloring of Basic Graph Classes*[†]

Susanne Albers[1] and Sebastian Schraink[2]

1   **Technical University of Munich, Garching, Germany**
    `albers@in.tum.de`
2   **Technical University of Munich, Garching, Germany**
    `schraink@in.tum.de`

───── **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-treewidth 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.

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 [19, 23, 24]. A problem instance is defined by 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 $\mathcal{A}(G) \leq c \cdot \chi(G)$ holds for every graph G [25]. 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 $E[\mathcal{A}(G)] \leq c \cdot \chi(G)$ holds for every G [5]. 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, an algorithm might have a *buffer of size b* in which vertices can be stored temporarily. 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, 11] 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 [22] developed a deterministic online algorithm that achieves a competitive factor of $O(n/\log^* n)$. Vishwanathan [26] 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 [16]. Halldorsson and Szegedy [17] 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 $O(\log^2 n)$ [17].

There has also been considerable research interest in online coloring for various graph classes. An early and celebrated result proved by Bean [4] 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 [15]. 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 [22], implying that the best competitiveness of deterministic strategies is again $\Theta(\log n)$. However, *First Fit* performs poorly, as there are bipartite graphs for which it requires $\Omega(n)$ colors. Kierstead and Trotter [20] 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 [18]. 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. Irani [18] 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 [18]. 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 [18] 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 [18] 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 [21].

We address two further graph classes. Downey and McCartin [10] studied online coloring of bounded treewidth graphs. For an introduction to treewidth see [7]. For any graph of treewidth d, *First Fit* uses $O(d \cdot \log n)$ colors. This is a consequence of Irani's work [18] because a graph of treewidth d is d-inductive [10, 18]. Downey and McCartin [10] 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 [13] 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 [9, 12]. The result relies on the common assumption that an online algorithm does not use the disk representation, when making coloring decisions [9, 12, 13]. It has been repeatedly raised as an open problem if the bound of $\Theta(\min\{\log n, \log \rho\})$ can be improved using randomization [9, 12, 13].

Recent work on online graph coloring has studied scenarios where an online algorithm can query oracle information about future input [8, 6]. Moreover, online coloring of hypergraphs has been explored [2, 3].

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 Sections 2 and 3 we investigate chordal graphs. They have been studied extensively, cf. textbook [27]. 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, for which the offline coloring, maximum clique and independent set problems can be solved in polynomial time.

In Section 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 [18]. In Section 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 [28] can be applied. *First Fit* colors every chordal graph G with $\chi(G) = d$ using $O(d \cdot \log n)$ colors. Hence, the optimal competitiveness of deterministic and randomized online algorithms is $\Theta(\log n)$.

In Section 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 [4] 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 [26] 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 [15, 18, 22].

Section 4 also addresses inductive and bounded-treewidth graphs. Since every chordal graph G is $(\chi(G) - 1)$-inductive and has treewidth $\chi(G) - 1$, 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 [14]. *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 Section 4 we study disk graphs. We prove that, for $d = 2$, every graph of the probability distribution defined in Section 3 translates to a disk graph. We then show that, for every randomized online algorithm $\mathcal{A}$ that does not use the disk representation, 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. Hence randomization does not improve the asymptotic performance of online coloring algorithms for disk graphs, cf. [9, 12, 13].

In Section 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 [18] 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.

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 [4] 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 a 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 2, which we prove first.

▶ **Lemma 2.** *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. $|\mathcal{C}_{\mathcal{A}}(r(G_k))| \geq \frac{d}{4} \cdot k$.

(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. 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.

Figure 1 The tree T representing G_1.
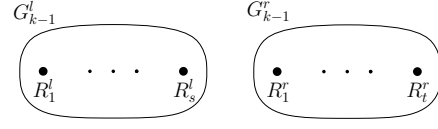
Figure 2 The general structure of G_{k-1}^l and G_{k-1}^r restricted to the root vertices.

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 Figure 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. $|\mathcal{C}_{\mathcal{A}}(r(G_1))| \geq \frac{d}{4}$.

(2) G_1 consists of one connected component which represents a tree, as described above and shown in Figure 1.

(3) G_1 is a clique and thus chordal.

(4) The maximum clique size $\omega(G_1)$ is exactly d.

(5) There holds $n_1 = d \leq \frac{3}{2} \cdot d = \frac{d}{2} \cdot (2^{1+1} - 1)$.

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_{k-1}^l and G_{k-1}^r be these two graphs. They are created one after the other. We remark that G_{k-1}^l and G_{k-1}^r need not be identical because $\mathcal{A}$'s coloring decision in one graph can affect its decisions in the other one.

In the following we focus on the root vertices of G_{k-1}^l and G_{k-1}^r. In particular, we consider the colors used by $\mathcal{A}$. Invariant (1) implies that $\left|\mathcal{C}_{\mathcal{A}}(r(G_{k-1}^l))\right| \geq \frac{d}{4}(k-1)$ and $\left|\mathcal{C}_{\mathcal{A}}(r(G_{k-1}^r))\right| \geq \frac{d}{4}(k-1)$. We distinguish two cases depending on the total number of colors used, i.e. the cardinality of $\mathcal{C}_{\mathcal{A}}(r(G_{k-1}^l) \cup r(G_{k-1}^r))$. To this end we introduce some notation. Assume that G_{k-1}^l consists of s connected components, which we number in an arbitrary way. Each component/tree T_i^l has a distinguished root containing a set $r(T_i^l)$ of $d/2$ root vertices. We abbreviate $R_i^l = r(T_i^l)$, $1 \leq i \leq s$. Similarly, assume that G_{k-1}^r consists of t connected components. Set $r(T_j^r)$ is the set of root vertices in the component T_j^r. Let $R_j^l = r(T_j^r)$, $1 \leq j \leq t$. There holds $r(G_{k-1}^l) = \bigcup_{i=1}^{s} R_i^l$ and $r(G_{k-1}^r) = \bigcup_{j=1}^{t} R_j^r$. Figure 2 shows the general structure of G_{k-1}^l and G_{k-1}^r by focusing on the roots. The left-hand side of the figure depicts G_{k-1}^l as a union of connected components rooted at $R_1^l, \ldots, R_s^l$, respectively. The right-hand side shows G_{k-1}^r as a collection of components rooted at $R_1^r, \ldots, R_s^r$.

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

(1) The condition of Case 1 ensures $|\mathcal{C}_{\mathcal{A}}(r(G_k))| = \left|\mathcal{C}_{\mathcal{A}}(r(G_{k-1}^l) \cup r(G_{k-1}^r))\right| \geq \frac{d}{4} \cdot k$.

(2) The invariant is satisfied since G_k is the union of G_k^l and G_k^r.

(3) G_k is chordal because G_k^l and G_k^r are, and no further vertices or edges have been added.

(4) There holds $\omega(G_k) = d$, as $\omega(G_{k-1}^l) = \omega(G_{k-1}^r) = d$.

(5) Let n_{k-1}^l and n_{k-1}^r be the number of vertices in G_{k-1}^l and G_{k-1}^r, respectively. There holds $n_k = n_{k-1}^l + n_{k-1}^r \leq 2 \cdot (\frac{d}{2} \cdot (2^k - 1)) = \frac{d}{2} \cdot (2^{k+1} - 2) \leq \frac{d}{2} \cdot (2^{k+1} - 1)$. The first inequality follows because (5) holds for n_{k-1}^l and n_{k-1}^r.
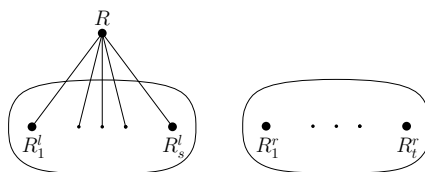
■ **Figure 3** The graph G_k with the new addition of R.

Case 2: Next assume that $\left|\mathcal{C}_\mathcal{A}(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| < \frac{d}{4} \cdot k$. 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 Figure 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, $1 \le i \le s$. 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 $\left|\mathcal{C}_\mathcal{A}(r(G^l_{k-1})) \cup \mathcal{C}_\mathcal{A}(r(G^r_{k-1}))\right| = \left|\mathcal{C}_\mathcal{A}(r(G^l_{k-1}) \cup r(G^r_{k-1}))\right| < \frac{d}{4} \cdot k$. There holds $\mathcal{C}_\mathcal{A}(r(G^l_{k-1})) \ge \frac{d}{4}(k-1)$. 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 $\left|\mathcal{C}_\mathcal{A}(r(G_k))\right| = \left|\mathcal{C}_\mathcal{A}(R \cup r(G^r_{k-1}))\right| = \left|\mathcal{C}_\mathcal{A}(r(G^r_{k-1}))\right| + \left|\mathcal{C}_\mathcal{A}(R) \setminus \mathcal{C}_\mathcal{A}(r(G^r_{k-1}))\right| \ge \frac{d}{4}(k-1) + \frac{d}{4} = \frac{d}{4}k$.

(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 and assume that at least one vertex is in R. If three or more vertices of C are in R, then there is a chord because R is a clique. If C contains one or two vertices of R, then C can visit only one connected component of G^l_{k-1}. Suppose that it visits the one rooted at R^l_i. Cycle C must contain two vertices of R^l_i. Each of these two vertices has an edge to every vertex of R in C. Hence C has a chord. Since G^l_{k-1} and G^r_{k-1}, and thus the components rooted at $R^l_1, \ldots, R^l_s$ and $R^r_1, \ldots, R^r_t$, are chordal, so is G_k.

(4) Set R and each R^l_i, $1 \le i \le s$, form a clique of size d. The vertices of R are not connected to any vertices outside R^l_i, $1 \le i \le s$. Hence no other cliques are formed by the addition of R. Since $\omega(G^l_{k-1}) = \omega(G^r_{k-1}) = d$ it follows $\omega(G_k) = d$.

(5) Again, 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}. We have $n_k = n^l_{k-1} + n^r_{k-1} + \frac{d}{2} \le 2 \cdot (\frac{d}{2} \cdot (2^k - 1)) + \frac{d}{2} = \frac{d}{2} \cdot (2^{k+1} - 2) + \frac{d}{2} = \frac{d}{2} \cdot (2^{k+1} - 1)$.

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 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 $k > 1$, 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 $k = \lfloor \log(n/d) \rfloor$. There holds $k \in \mathbb{N}$ because $n \ge 2d^2 > 2d$. For every deterministic online algorithm, by Lemma 2, 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 $d \ge 2$, there holds $c_k \ge dk/8$. 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 $n \ge 2d^2 \ge 4$, there holds $\log(n/2) \ge 1/2 \cdot \log n$. 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 $0 < \epsilon < 1$. Then 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 3.** *Let $d \in \mathbb{N}$ with $d \ge 2$ be arbitrary. For every randomized online algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$ with $n \ge 12d^2$, 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 3 we resort to Yao's principle [28] and show the following Lemma 4.

▶ **Lemma 4.** *Let $d \in \mathbb{N}$ with $d \ge 2$ 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 2. 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 2. 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 2, (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 $\mathcal{A}$, $\Pr[|\mathcal{C}_{\mathcal{A}}(r(G_k))| \geq dk/4] > 1/2$. 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 2, 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), there holds $n_1 = d \leq d(12 - 1)$.

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 $|\mathcal{G}_{k-1}|^{12}$ tuples are built. For each tuple, 2^6 graphs are 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 $G_{k-1}^{i,l}$ and $G_{k-1}^{i,r}$ 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 $\vec{b} = (b_1, \ldots, b_6)$. More specifically, given τ and any such bit vector $\vec{b}$, a graph G_k is constructed as follows. For $i = 1, \ldots, 6$, a subgraph G_k^i is generated. If $b_i = 0$, then G_k^i is the union of $G_{k-1}^{i,l}$ and $G_{k-1}^{i,r}$. The set $r(G_k^i)$ of root vertices is the union of $r(G_k^{i,l})$ and $r(G_k^{i,r})$. If $b_i = 1$, then a clique R_i of size $d/2$ is added to $G_{k-1}^{i,l}$ and $G_{k-1}^{i,r}$. Every vertex of R_i has an edge to every vertex of $r(G_{k-1}^{i,l})$. Subgraph G_k^i consists of the newly created component rooted at R_i and $r(G_{k-1}^{i,r})$, i.e. $r(G_{k-1}^i) = R_i \cup r(G_{k-1}^{i,r})$. Graph G_k is the union of the G_k^i and the set $r(G_k)$ is the union of the $r(G_k^i)$, $1 \leq i \leq 6$. When G_k is presented to $\mathcal{A}$, the subgraphs G_k^i are revealed one by one, $1 \leq i \leq 6$. For each G_k^i the graphs $G_{k-1}^{i,l}$ and $G_{k-1}^{i,r}$

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_k^i, $1 \le i \le 6$, containing $G_k^{i,l}$ and $G_k^{i,r}$. By the construction of $\mathcal{G}_k$, both $G_k^{i,l}$ and $G_k^{i,r}$ 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 $\Pr[|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,l}))| \ge d(k-1)/4] > 1/2$ and $\Pr[|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,r}))| \ge d(k-1)/4] > 1/2$. Moreover it implies $\Pr[|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,l}))| \ge d(k-1)/4$ and $|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,r}))| \ge d(k-1)/4] > 1/4$. Let $\mathcal{E}^i$ be the latter event that $|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,l}))| \ge d(k-1)/4$ and $|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,r}))| \ge d(k-1)/4$ hold.

Assume that $\mathcal{E}^i$ holds. There are two cases, which correspond to those analyzed in the proof of Lemma 2. If $|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,l}) \cup r(G_{k-1}^{i,l}))| \ge dk/4$, then $|\mathcal{C}_\mathcal{A}(r(G_k^i))| \ge dk/4$ if R_i is not added to $G_k^{i,l}$ and $G_k^{i,r}$, which happens with probability $1/2$. On the other hand, if $|\mathcal{C}_\mathcal{A}(r(G_{k-1}^{i,l}) \cup r(G_{k-1}^{i,r}))| < dk/4$, then the addition of R_i ensures that $|\mathcal{C}_\mathcal{A}(r(G_k^i))| \ge dk/4$. Again, R_i is added with probability $1/2$. In either case, given $\mathcal{E}^i$, $\Pr[|\mathcal{C}_\mathcal{A}(r(G_k^i))| \ge dk/4] \ge 1/2$. We obtain $\Pr[|\mathcal{C}_\mathcal{A}(r(G_k^i))| \ge dk/4] \ge \Pr[|\mathcal{C}_\mathcal{A}(r(G_k^i))| \ge dk/4 \mid \mathcal{E}^i] \cdot \Pr[\mathcal{E}^i] \ge \frac{1}{2} \cdot \frac{1}{4} = \frac{1}{8}$. Equivalently, $\Pr[|\mathcal{C}_\mathcal{A}(r(G_k^i))| < dk/4] \le 7/8$. If $|\mathcal{C}_\mathcal{A}(r(G_k))| < dk/4$, then $|\mathcal{C}_\mathcal{A}(r(G_k^i))| < dk/4$ must hold true for $i = 1, \ldots, 6$. The latter event occurs with probability at most $(7/8)^6$. We conclude $\Pr[|\mathcal{C}_\mathcal{A}(r(G_k))| \ge dk/4] \ge 1 - (7/8)^6 > 1/2$. 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 2. 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 $\mathcal{G}_k$, $k \ge 1$, 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 3. For the given d and n, choose $k = \lfloor \log(n/d) \rfloor$. In this proof, logarithms are base 12. There holds $k \in \mathbb{N}$, because $n \ge 12d^2 > 12d$. By Lemma 4, 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, it 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 [28], 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 $c_k \ge (d-1)k/8 \ge dk/16$. We have $k \ge \log n - \log d - 1 = \log(n/12) - \log d \ge 1/2 \cdot \log(n/12)$, because $12d^2 \le n$, and hence $d \le \sqrt{n/12}$. Since $12d^2 \le n$, we have $\log(n/12) \ge 1/3 \cdot \log n$ and thus $c_k \in \Omega(d \cdot \log n)$. ◄

Again, in Theorem 3 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 3, 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

▶ **Corollary 5.** *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 $\Omega(\log n)$.*

The proof is given in the full version of the paper. Since trees are planar and bipartite graphs, we obtain the following two corollaries.

▶ **Corollary 6.** *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 7.** *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 $\omega(G) - 1 = \chi(G) - 1$ [7]. Hence, Theorem 3 gives the following two results.

▶ **Corollary 8.** *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 9.** *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 graphs used in the proof of Theorem 3 are strongly chordal, which yields the following corollary. The proof can be found in the full version of the paper.

▶ **Corollary 10.** *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 $\Omega(d \cdot \log n)$.*

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 [9, 12, 13]. The proof of Theorem 11 is presented in the full version of the paper.

▶ **Theorem 11.** *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 $\Omega(\min\{\log n, \log \rho\})$.*

5   Lookahead and buffer reordering

Lookahead: We first assume that a randomized online coloring algorithm $\mathcal{A}$ has lookahead l. Theorem 12 below shows that, for chordal graphs, a lookahead of size $O(n/\log n)$ leads to no improvement. The proof is given in the full version of the paper.

▶ **Theorem 12.** *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 $\Omega(\frac{1}{c} \cdot d \cdot \log n)$.*

Based on Theorem 12 we can derive analogous results for all the other graph classes considered in Section 4. Loosely speaking, a lookahead of size $O(n/\log n)$ is of no help. The next Corollary 13 addresses trees. Exactly the same statement holds for planar and bipartite graphs, respectively. For brevity, we omit the corresponding corollaries.

▶ **Corollary 13.** *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 12 directly carries over. In fact, the result holds for all integers $d \geq 1$. For disk graphs, Theorems 11 and 12 give the following corollary.

▶ **Corollary 14.** *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 $\Omega(\frac{1}{c} \cdot \log n)$.*

Buffer reordering: Next we examine the setting in which a deterministic online coloring algorithm $\mathcal{A}$ has a reordering buffer. 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 15.** *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 $\Omega(\epsilon \cdot d \cdot \log n)$.*

The proof of Theorem 15 is presented in the full version of the paper. Given Theorem 15, we derive analogous results for the other graph classes. Corollary 16 shows a result for trees. Identical statements hold for planar and bipartite graphs. Again, for brevity, we omit the corresponding corollaries.

▶ **Corollary 16.** *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 15 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 17.** *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 $\Omega(\epsilon \cdot \min\{\log n, \log \rho\})$.*

─── **References** ───

1 N. Avigdor-Elgrabli and Y. Rabani. An optimal randomized online algorithm for reordering buffer management. In *Proc. 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 2013.

2 A. Bar-Noy, P. Cheilaris, S. Olonetsky, and S. Smorodinsky. Online conflict-free colouring for hypergraphs. *Combinatorics, Probability & Computing*, 19(4):493–516, 2010.

3 A. Bar-Noy, P. Cheilaris, and S. Smorodinsky. Deterministic conflict-free coloring for intervals: From offline to online. *ACM Trans. Algorithms*, 4(4):44:1–44:18, 2008.

4 D. Bean. Effective coloration. *J. Symbolic Logic*, 41(2):469–480, 1976.

5 S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11(1):2–14, 1994.

6 M. P. Bianchi, H.-J. Böckenhauer, J. Hromkovic, and L. Keller. Online coloring of bipartite graphs with and without advice. *Algorithmica*, 70(1):92–111, 2014.

7 H. L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11(1-2):1–21, 1993.

8 E. Burjons, J. Hromkovic, X. Muñoz, and W. Unger. Online graph coloring with advice and randomized adversary. In *Proc. 42nd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'16)*, pages 229–240. Springer LNCS 9587, 2016.

9 I. Caragiannis, A. V. Fishkin, C. Kaklamanis, and E. Papaioannou. A tight bound for online colouring of disk graphs. *Theoretical Computer Science*, 384(2):152–160, 2007.

10 R. G. Downey and C. McCartin. Online promise problems with online width metrics. *Journal of Computer and System Sciences*, 73(1):57–72, 2007.

11 M. Englert, D. Özmen, and M. Westermann. The power of reordering for online minimum makespan scheduling. *SIAM J. Comput.*, 43(3):1220–1237, 2014.

12 T. Erlebach and J. Fiala. On-line coloring of geometric intersection graphs. *Computational Geometry*, 23(2):243–255, 2002.

13 T. Erlebach and J. Fiala. Independence and coloring problems on intersection graphs of disks. In E. Bampis, K. Jansen, and C. Kenyon, editors, *Efficient Approximation and Online Algorithms: Recent Progress on Classical Combinatorial Optimization Problems and New Applications*, pages 135–155. Springer LNCS 3484, 2006.

14 Martin Farber. Characterizations of strongly chordal graphs. *Discrete Mathematics*, 43(2-3):173–189, 1983.

15 A. Gyárfás and J. Lehel. On-line and first fit colorings of graphs. *Journal of Graph Theory*, 12(2):217–227, 1988.

16 M. M. Halldórsson. Parallel and on-line graph coloring. *J. Algorithms*, 23(2):265–280, 1997.

17 M. M. Halldórsson and M. Szegedy. Lower bounds for on-line graph coloring. *Theoretical Computer Science*, 130(1):163–174, 1994.

18 S. Irani. Coloring inductive graphs on-line. *Algorithmica*, 11(1):53–72, 1994. Preliminary version in *FOCS'90*.

19 H. A. Kierstead. Coloring graphs on-line. In A. Fiat and G. J. Woeginger, editors, *Online Algorithms*, pages 281–305. Springer LNCS 1442, 1998.

20   H. A. Kierstead and W. A. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.

21   S. Leonardi and A. Vitaletti. Randomized lower bounds for online path coloring. In *Proc. 2nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM'98)*, pages 232–247. Springer LNCS 1518, 1998.

22   L. Lovász, M. Saks, and W. T. Trotter. An on-line graph coloring algorithm with sublinear performance ratio. *Annals of Discrete Mathematics*, 43:319–325, 1989.

23   D. Marx. Graph colouring problems and their applications in scheduling. *Periodica Polytechnica, Electrical Engineering*, 48(1–2):11–16, 2004.

24   L. Narayanan. Channel assignment and graph multicoloring. *Handbook of Wireless Networks and Mobile Computing*, pages 71–94, 2004.

25   D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985.

26   S. Vishwanathan. Randomized online graph coloring. *J. Algorithms*, 13(4):657–669, 1992. Preliminary version in *FOCS'90*.

27   D. B. West. *Introduction to Graph Theory, 2nd Edition*. Pearson, 2001.

28   A. C. C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proc. 18th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.