Online Graph Coloring Against a Randomized Adversary

Elisabet Burjons* and Juraj Hromkovič†

Department of Computer Science, ETH Zurich, Switzerland
**elisabet.burjons@inf.ethz.ch*
†juraj.hromkovic@inf.ethz.ch

Rastislav Královič

Department of Computer Science, Comenius University, Bratislava
kralovic@dcs.fmph.uniba.sk

Richard Královič

Google Inc., Zurich, Switzerland
richard.kralovic@dcs.fmph.uniba.sk

Xavier Muñoz

Mathematics Department, UPC, Barcelona, Spain
xml@ma4.upc.edu

Walter Unger

Department of Computer Science, RWTH Aachen University, Germany
walter.unger@cs.rwth-aachen.de

We consider an online model where an adversary constructs a set of 2^s instances S instead of one single instance. The algorithm knows S and the adversary will choose one instance from S at random to present to the algorithm. We further focus on adversaries that construct sets of k-chromatic instances. In this setting, we provide upper and lower bounds on the competitive ratio for the online graph coloring problem as a function of the parameters in this model. Both bounds are linear in s and matching upper and lower bound are given for a specific set of algorithms that we call "minimalistic online algorithms".

Keywords: Online computation; information; randomization; graph coloring.

1. Introduction

For an online problem, the input is revealed gradually in consecutive time steps and an irrevocable part of the output has to be produced at each time step. In this model

there are two players, the algorithm that processes the instances piecewise and the adversary which presents the instance. A detailed introduction and an overview of online problems and algorithms can be found in [7, 21].

One of the most studied online scenarios is the problem of coloring a graph online. In the classical online model, the vertices of the graph are revealed one after the other, together with the edges connecting them to the already presented vertices. In this case, an instance I consists of a graph G and the order of presentation of the vertices. Observe that two different instances might construct the same graph, only changing the order in which the vertices are presented. The goal of the algorithm is to assign the minimum number of colors to these vertices in such a way that no two adjacent vertices get the same color. The algorithm does not have any information beforehand about the graph or the next vertices that are going to be presented. As usual in an online setting, each vertex has to be colored immediately after its appearance. The quality of an online algorithm for this problem is usually measured by the so-called competitive ratio, i.e., the ratio between the number of colors used by this algorithm (on an instance I) and an optimal coloring for the resulting graph (of this instance, $G(I)$) as it could be computed by an offline algorithm with unlimited computing power, knowing the whole graph in advance. The competitive ratio of an algorithm ALG on an instance I is then

$$\mathrm{comp}_{\mathrm{ALG}}(I) = \frac{\text{colors used by ALG in } I}{\text{chromatic number of } G(I)}.$$

The competitive ratio of ALG for a problem $\mathcal{P}$ is then computed in a worst-case manner as $\mathrm{comp}_{\mathrm{ALG}}(\mathcal{P}) = \sup_I\{\mathrm{comp}_{\mathrm{ALG}}(I)\}$.

For this problem, the competitive ratio is unbounded. This holds even for trees, for which the lower bound is in $\Omega(\log n)$ [18].

Restricting this model to specific graph classes yields improved results where the competitive ratio is bounded by some constant or even reaches optimality [2, 8, 15, 16, 18, 25, 28]. An overview of results on the online graph coloring problem can be found in [19, 20].

One can argue that adversaries as described above are unrealistic, and that in most online situations we do have some knowledge on the input instance. In the classical model, the adversary is allowed to construct one of the hardest problem instances for a given online algorithm. In the model we are studying, the power of the adversary is reduced by requesting the adversary, for a given positive integer s, to generate a set of 2^s problem instances S and to make them public. So, instead of restricting the instance to a class of graphs, as mentioned above, the adversary is restricted to a set S of graphs that the algorithm knows in advance. This same model has been applied to the paging problem [9], and the string guessing problem [27], which was originally published at the same time as this paper. A similar model is presented in [24], where the adversary is constrained within a specific instance

distribution, and in [26], where the adversarial choice for the instance must satisfy specific statistical properties.

A formal definition to measure the performance of algorithms in the model we propose can be given as follows. Let, for an $s \in \mathbb{N}$, S be a set of 2^s instances of a considered online problem $\mathcal{P}$. Let ALG be an online algorithm solving $\mathcal{P}$. The competitive ratio of ALG on S, $\mathrm{comp}_{\mathrm{ALG}}(S)$, is the expected competitive ratio of ALG on all instances in S with respect to the uniform probability distribution. An alternative definition would be the ratio between the expected cost of ALG and the expected optimal cost. We observe, however, that in the case we will look at, where the instances all have the same chromatic number, i.e., the same optimal cost, both definitions coincide.

For any S of 2^s instances, we define

$$\mathrm{comp}(S) = \min\{\mathrm{comp}_{\mathrm{ALG}'}(S) \mid \mathrm{ALG}' \text{ solves the instances in } S\}$$

as the competitive ratio of the "best" online algorithm for S. Finally,

$$\mathrm{comp}_s(\mathcal{P}) = \sup\{\mathrm{comp}(S) \mid S \text{ consists of } 2^s \text{ instances}\}$$

is the *competitive ratio for the problem $\mathcal{P}$ for an adversary with s random bits*. For an online algorithm ALG solving $\mathcal{P}$ and any $s \in \mathbb{N}$, we define the *competitive ratio of* ALG *with respect to* s as

$$\mathrm{comp}_{\mathrm{ALG}}(s) = \sup\{\mathrm{comp}_{\mathrm{ALG}}(S) \mid S \text{ consists of } 2^s \text{ instances of } \mathcal{P}\}.$$

To sum up, the competitive ratio with respect to a set S is the minimum of the competitve ratios over all online algorithms solving S. Then, the competitive ratio of the problem with respect to s is the maximum of the competitive ratios over all sets of size 2^s. This measure for the quality of online algorithms is reasonable, because in the classical model an online algorithm is bad if there exists one hard problem instance for it. Here, investigating the achievable competitive ratio with respect to s (or the size of the set of problem instances) can provide more insight into the hardness of concrete online problems. With increasing s, the set of instances grows exponentially and the competitive ratios in this setting provide a more realistic picture than the classical worst-case model.

As already mentioned, the competitive ratio for graph coloring in the classical model is unbounded. In contrast to this, we show that, within our new model, for all k-chromatic graphs with $k \geq 2$, the competitive ratio for sets S of 2^s problem instances, all of them with chromatic number k for some fixed k, is at most $(s+2)/2$.

Note that here the sets of problem instances are known to the online algorithm that can choose the appropriate working strategy with respect to the given S. If S is unknown, we have the same unbounded competitive ratio as for the classical problem since, for each online algorithm, one can take the hardest problem instance and add a few dummy vertices in order to get 2^s problem instances.

We also provide a lower bound of $(3s + 8)/4$, which is of the same order of magnitude as the upper bound for general algorithms, and a tight lower bound when restricting ourselves to online algorithms using new colors only if necessary.

In what follows, the parameter s is viewed as the number of random bits available to the adversary, because it needs s random bits to randomly pick an input from S.

Another reason to prefer viewing s as the number of random bits of the adversary instead of as a measure of the cardinality of S will be presented later in the framework of information content of online problems [17], where the game is played by three players — algorithm, adversary, and oracle.

2. Upper Bound on the Competitive Ratio of the Online Coloring Problem

In this section, we show that the competitive ratio of the problem is linear with respect to the number s of random bits available to the adversary. A practical way of computing bounds on competitive ratios in this setting is to look for the expected number of colors used by an algorithm in order to process a problem instance. Knowing the chromatic number of the problem instance and the expected number of colors used, we obtain the competitive ratio by computing their ratio.

Theorem 1. *Given an adversary that uses s random bits to generate 2^s inputs such that all of them are k-chromatic, there exists an online algorithm knowing the set of inputs that colors the input using $(s + 2)k/2$ colors in expectation.*

Proof. Given the set S of 2^s possible problem instances (defined by their request sequences of vertices), we can construct a rooted tree T_S called an *instance tree*. We call the vertices of the instance tree *nodes* in order to distinguish them from the vertices in the instances. In T_S, a node branches into two or more child nodes whenever a request in the sequence enables to distinguish between two or more groups of different problem instances. The root represents the whole set of 2^s problem instances. For example, the first vertex never gives any information about a concrete problem instance because it is the same for all 2^s instances. However, in some instance set, the second vertex might or might not be adjacent to the first vertex presented. So, looking at the first two vertices of all the request sequences, two groups of problem instances can be derived from that piece of information in this case. Note that not every single request necessarily leads to a branch in the instance tree.

Once constructed, the instance tree T_S will have at most 2^s leaves. The algorithm colors the request sequence given as follows.

First of all, it chooses one leaf ℓ in T_S according to the following criterion. For each node v in the path from the leaf ℓ to the root of the tree, the subtree T_v rooted in the node v has at least as many leaves as any of its sibling subtrees T_w. Observe that, in a balanced tree, any leaf chosen is good according to this criterion. Thus,

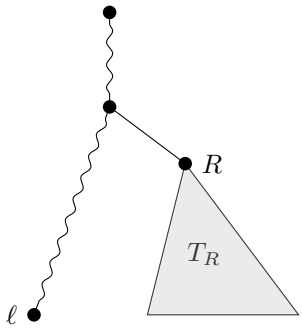


Fig. 1. Subtree T_R.

there may be more than one possible choice for the leaf. This leaf corresponds to a possible problem instance I_ℓ.

The algorithm then starts coloring the incoming vertices (requests) as if the graph given as a problem instance is the instance I_ℓ corresponding to the selected leaf ℓ. This means that the online coloring strategy used is optimal for I_ℓ.

As long as the actual problem instance I coincides with the problem instance I_ℓ, we use this strategy following the path from the root to ℓ. If a request R comes up that does not coincide with the chosen path, this means that the unknown actual problem instance I does not correspond to I_ℓ. In this case the problem instance is one that corresponds to a leaf in a subtree T_R of problem instances that coincide with I up to the request R (see Fig. 1). In this case, a leaf $\hat{\ell}$ of T_R is selected according to the same criterion as ℓ was chosen in T_S. Now, k new colors are used to color the new incoming requests according to an optimal coloring of the graph corresponding to $I_{\hat{\ell}}$. In this context, we speak about a *switch* from ℓ to $\hat{\ell}$ (or from I_ℓ to $I_{\hat{\ell}}$, respectively) during the execution of our online algorithm. The algorithm continues recursively in T_R in the same way as in T_S and makes further switches if necessary.

Coloring in this way, at most $(t+1)k$ colors are used to color the graph, where t is the number of switches until the correct leaf is reached. We have to prove that using this method no more than $s/2$ switches are performed on average. In fact, we will prove in the following lemma that altogether no more than $\frac{s2^s}{2}$ switches are required in all 2^s computations over all 2^s problem sequences considered.

Knowing that, in order to color all 2^s graphs, no more than $s2^s/2$ switches are required, we can easily conclude that on average no more than $s/2$ switches are made. Hence no more than $\frac{sk}{2} + k = \frac{(s+2)k}{2}$ colors are used in expectation. □

Lemma 2. *Let S be any set of L problem instances, and let T_S be the corresponding tree with L leaves. Let $\#(T_S)$ be the total number of switches over all L computations on the L input instances performed by the online algorithm described above. Then, $\#(T_S) \leq \frac{L \log_2 L}{2}$ for $L \geq 2$.*

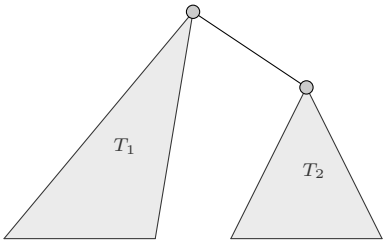


Fig. 2. Tree configuration.

Proof. We prove the lemma by induction on the number L of leaves of the instance tree. For $L = 2$, the only possible rooted tree is a root v and two leaves v_1, v_2. Either choice that is possible for ℓ is equally valid. Therefore, choosing v_1, there will be no switches for the first problem instance and one switch for the second, yielding a total number of one switch.

Suppose the lemma is valid for any tree with less than L leaves. For a tree with L leaves, the root will have two or more child nodes. We pick the one which has the smallest subtree and name it T_2 with ℓ_2 leaves and name the rest of the tree with the original root T_1 with ℓ_1 leaves (see Fig. 2). The selected leaf will always be in T_1 (as T_2 is the smallest subtree with a child root of T_S). The total number of switches is $\#(T_S) = \#(T_1) + \ell_2 + \#(T_2)$. This corresponds to the switches inside T_1, the switches inside T_2, and the first switch in the computation for each of the ℓ_2 instances in T_2. Using the induction hypothesis, we get

$$\begin{aligned}
\#(T_S) &= \#(T_1) + \ell_2 + \#(T_2) \\
&\leq \frac{\ell_1}{2}\log_2 \ell_1 + \frac{\ell_2}{2} + \frac{\ell_2}{2}\log_2 \ell_2 + \frac{\ell_2}{2} \\
&= \frac{\ell_1}{2}\left(\log_2 \ell_1 + \frac{\ell_2}{\ell_1}\right) + \frac{\ell_2}{2}\left(\log_2 \ell_2 + 1\right) \\
&= \frac{\ell_1}{2}\log_2\left(2^{\ell_2/\ell_1}\ell_1\right) + \frac{\ell_2}{2}\log_2(2\ell_2)\,.
\end{aligned}$$

Now we use the facts that $\ell_2 \leq \ell_1$ and $2\ell_2 \leq L$, and $2^{\ell_2/\ell_1} \leq 1 + \ell_2/\ell_1$, because $\ell_2/\ell_1 \leq 1$, in the first term and conclude

$$\begin{aligned}
\#(T_S) &\leq \frac{\ell_1}{2}\log_2\left(2^{\ell_2/\ell_1}\ell_1\right) + \frac{\ell_2}{2}\log_2(2\ell_2) \\
&\leq \frac{\ell_1}{2}\log_2\left(\left(1 + \frac{\ell_2}{\ell_1}\right)\ell_1\right) + \frac{\ell_2}{2}\log_2 L \\
&= \frac{\ell_1}{2}\log_2 L + \frac{\ell_2}{2}\log_2 L \\
&= \frac{L}{2}\log_2 L.
\end{aligned}$$

Observe that, if all of the child subtrees of the root have the same number of leaves, we can still choose the selected leaf among T_1 or select a subtree T_2 where the chosen leaf is not included. □

Following Lemma 2 the expected number of switches is at most

$$\frac{\frac{L \log_2 L}{2}}{L} = \frac{\log_2 L}{2}. \tag{1}$$

For $L = 2^s$, the expected number of switches is at most $s/2$, and so the expected number of colors used over all 2^s input instances is at most $k + sk/2$. If the optimal coloring of all the instances uses k colors, then the competitive ratio is at most $1 + s/2$.

3. General Lower Bound

First, we present a lower bound for bipartite graphs, i.e., 2-colorable graphs [30]. Afterwards, we generalize the bound to k-chromatic graphs. Note that the lower bound has the same order of magnitude as the upper bound, but it is not matching. In Sec. 4, we will present a matching lower bound for a restricted class of algorithms.

An adversarial strategy constructing a set of instances for which any algorithm uses a large number of colors in expectation, is used for proving a lower bound on the competitive ratio.

3.1. *Bipartite case*

Consider an arbitrary set S of 2^s bipartite graph instances, and a deterministic algorithm ALG that knows S. An instance $I \in S$ is selected uniformly at random, and is presented online to ALG. ALG has to produce a valid coloring by assigning every incoming vertex some color.

Theorem 3. *Any algorithm* ALG *requires at least* $(3s + 8)/4$ *colors in expectation* (*taken over all graphs from* S *for the worst-case set* S).

Proof. The adversarial strategy is the following. The instances can be described by a tree-like structure, i.e., by a tree where each of its nodes defines an instance. The set S will be a subset of 2^s instances of this tree.

There is one instance of level 0, namely the complete graph with 2 vertices, K_2. There are 2^ℓ instances of level ℓ, constructed as follows: consider any instance I from the $2^{\ell-1}$ instances of level $\ell - 1$. Let $V(I) = L \cup R$ be the bipartition of the graph generated by I, $G(I)$. From I, we create two instances of level ℓ (called *straight* and *flipped*). The straight instance consists of two copies of I, called I_1 and I_2, with vertex sets $V(I_1) = L_1 \cup R_1$, $V(I_2) = L_2 \cup R_2$, and two additional vertices u and v connected by an edge, such that all vertices from $L_1 \cup L_2$ are connected to u, and all vertices from $R_1 \cup R_2$ are connected to v. The flipped instance is constructed

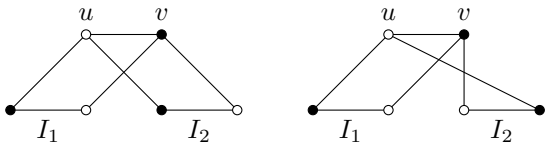


Fig. 3. Adversarial instances of level 1. The 2-coloring helps to visualize that both instances are bipartite.

similarly, except that the vertices from $L_1 \cup R_2$ are connected to u, and the vertices from $R_1 \cup L_2$ are connected to v. We say that the vertices u and v are vertices of level ℓ. The two instances of level 1 are as shown in Fig. 3.

For the set S, consider the 2^s instances of level s; in each instance, the vertices are presented in the order of increasing levels, and from left to right within a level.

The following representation of the instance set S aids in proving the lower bound. Every instance starts by presenting 2^s copies of K_2 that are the vertices of level 0. Then 2^{s-1} copies of K_2 are presented consisting of the vertices of level 1, connected to the corresponding vertices of level 0 by either a straight or flipped connection. Next, 2^{s-2} copies of K_2 that form the vertices of level 2 are presented, again connected either in a straight or flipped way. An instance can be described by an s-bit binary vector, and the whole S can be viewed as a butterfly graph of dimension s as shown in Fig. 4.

The graph has $s+1$ rows and 2^s columns: the top-most row is row s, the bottom-most row is row 0. The vertices of the butterfly graph will be called nodes. Each node of row s has a corresponding subtree that represents one instance. Each leaf corresponds to two vertices of level 0. The nodes in row 1 represent the vertices of level 1, either straight or flipped, etc. Each subtree of nodes corresponds to a bipartite graph of vertices. We call the bipartitions of the graph left and right based on the order of appearance of the first leaf (i.e., for a leaf, the two vertices appear in

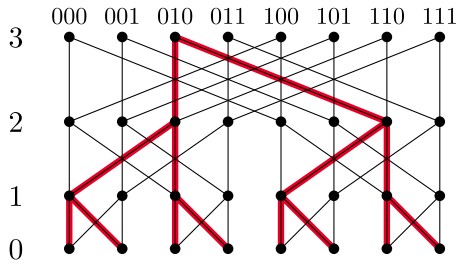


Fig. 4. Butterfly graph of dimension 3. The marked tree corresponds to the representation of an instance.

the order left-right; for an internal node, the left bipartition is the one that contains the left vertex of the first leaf).

To each node v, we assign a number $c(v)$ that is the number of colors the algorithm uses in the corresponding subtree. To prove the theorem, we need to show that

$$\sum_{v \text{ is in row } s} c(v) \geq 2^s(3s+8)/4. \tag{2}$$

Instead of proving (2) directly, we assign to a node v two other numbers, $l(v)$ and $r(v)$, that represent the number of colors used in the left and the right bipartitions, respectively. Let $f(v) := l(v) + r(v)$; we prove for each row ℓ that

$$\sum_{v \text{ is in row } \ell} f(v) \geq 2^s(3\ell+4)/2 = 2^s(3\ell/2+2). \tag{3}$$

Observe that

$$l(v) + r(v) + 2 \leq 2c(v),$$

as every color from $c(v)$ can contribute to both $l(v)$ and $r(v)$ except for the two vertices of level ℓ that can contribute only to one part as they are neighbors of all the vertices in the other part. Since $f(v) + 2 \leq 2c(v)$, (3) implies (2).

We prove (3) by induction on ℓ. For nodes on row $\ell = 0$, $l(v) = r(v) = 1$, and the claim clearly holds. Consider now row $\ell + 1$. The first ℓ rows form $2^{s-\ell}$ butterfly graphs, and consecutive pairs of them are joined in row $\ell + 1$. Denote the nodes of row ℓ by $v_1, \ldots, v_{2^{s-1}}$, $w_1, \ldots, w_{2^{s-1}}$ as shown in Fig. 5.

Nodes v_i, w_i of row ℓ correspond to nodes v'_i, w'_i of row $\ell + 1$. We prove the following:

$$\forall 1 \leq i \leq 2^{s-1} : \; f(v'_i) + f(w'_i) \geq f(v_i) + f(w_i) + 3. \tag{4}$$

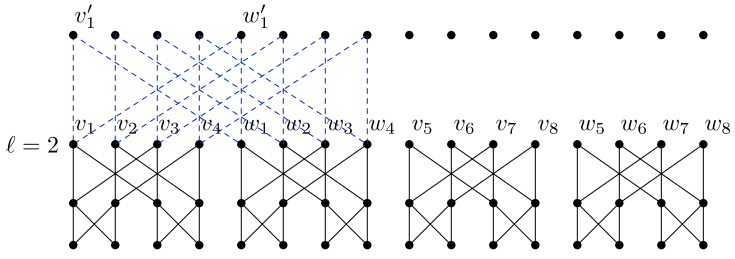


Fig. 5. construction of level $\ell = 3$ from level $\ell = 2$ in the butterfly graph.

Before proving (4), let us show how the induction step of (3) follows from it:

$$\begin{aligned}\sum_{v \text{ is in row } \ell+1} f(v) &= \sum_{i=1}^{2^{s-1}} f(v'_i) + f(w'_i) \\ &\geq \sum_{i=1}^{2^{s-1}} (f(v_i) + f(w_i) + 3) \\ &\geq 3 \cdot 2^{s-1} + \sum_{v \text{ is in row } \ell} f(v) \\ &\geq 3 \cdot 2^{s-1} + 2^s(3\ell/2 + 2) \\ &\geq 2^s(3(\ell+1)/2 + 2).\end{aligned}$$

It remains to show (4). Consider two nodes $v = v_i$ and $w = w_i$. Let L_v, R_v, L_w, R_w be the sets of colors used to color the left and right partitions of the subtrees of v and w, respectively. Then the left partition of v' contains all colors of $L_v \cup L_w$ (plus maybe one color used to color the corresponding vertex of level $\ell + 1$), and the right partition of v' contains colors $R_v \cup R_w$. Similar observations can be made for w'.

The situation looks like in Fig. 6.

Let v_L, v_R, w_L, w_R be the colors used for the nodes of level $\ell + 1$ in v' and w', respectively, and let χ_{v_L}, χ_{v_R}, χ_{w_L}, χ_{w_R} be the indicator variables for the events that v_L, v_R, w_L, w_R are not in L_v, R_v, L_w, R_w, respectively.

Observe that

$$f(v')+f(w') = |L_v \cup L_w \cup v_L| + |R_v \cup R_w \cup v_R| + |L_v \cup R_w \cup w_L| + |L_w \cup R_v \cup w_R|. \quad (5)$$

Now, for any two sets A and B, $|A \cup B| = |A| + |B \setminus A|$ and $|A \cup B| = 1/2(|A| + |B \setminus A|) + 1/2(|B| + |A \setminus B|)$. Substituting in (5), we get three different expressions for $f(v') + f(w')$:

$$\begin{aligned}f(v') + f(w') = f(v) + f(w) &+ \chi_{v_L} + \chi_{v_R} + \chi_{w_L} + \chi_{w_R} \\ &+ 1/2(|L_v \setminus L_w| + |L_w \setminus L_v| + |R_v \setminus R_w| + |R_w \setminus R_v| \\ &+ |L_v \setminus R_w| + |R_w \setminus L_v| + |L_w \setminus R_v| + |R_v \setminus L_w|), \quad (6)\end{aligned}$$

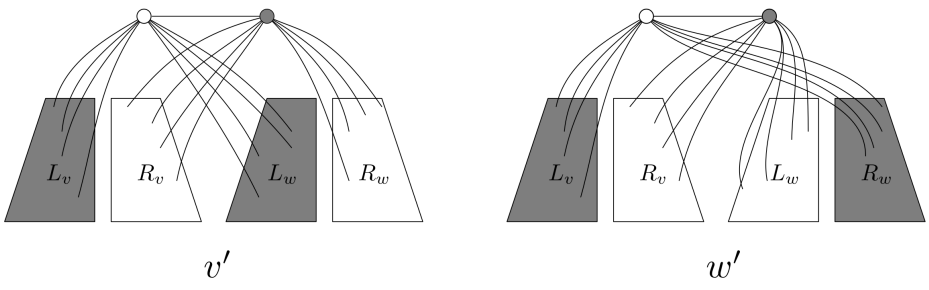


Fig. 6. Diagram that shows the left and right partitions of the graph in straight and flipped states.

$$f(v') + f(w') = f(v) + f(w) + \chi_{v_L} + \chi_{v_R} + \chi_{w_L} + \chi_{w_R} + |L_v \setminus L_w| + |R_v \setminus R_w| + |R_w \setminus L_v| + |L_w \setminus R_v|, \quad (7)$$

and

$$f(v') + f(w') = f(v) + f(w) + \chi_{v_L} + \chi_{v_R} + \chi_{w_L} + \chi_{w_R} + |L_w \setminus L_v| + |R_w \setminus R_v| + |L_v \setminus R_w| + |R_v \setminus L_w|. \quad (8)$$

The two vertices of level ℓ in v and w can contribute only to one part (either L or R) as they are neighbors of all the vertices in the other part. Therefore, $|L_v \setminus R_v|$, $|R_v \setminus L_v|$, $|L_w \setminus R_w|$, $|R_w \setminus L_w| \geq 1$.

Then, either $|L_v \setminus L_w| \geq 1$ or $L_v \subseteq L_w$ and $|R_w \setminus L_v| \geq |R_w \setminus L_w| \geq 1$, so $|L_v \setminus L_w| + |R_w \setminus L_v| \geq 1$.

Analogously, we can pair up $|L_w \setminus L_v|$ and $|R_v \setminus L_w|$ etc., and we get the following set of restrictions:

$$\begin{aligned} |L_v \setminus L_w| + |R_w \setminus L_v| &\geq 1, \\ |L_w \setminus L_v| + |R_v \setminus L_w| &\geq 1, \\ |R_v \setminus R_w| + |L_w \setminus R_v| &\geq 1, \\ |R_w \setminus R_v| + |L_v \setminus R_w| &\geq 1. \end{aligned} \quad (9)$$

Moreover, either $|L_v \setminus L_w| \geq 1$ or $L_v \subseteq L_w$ and $|L_w \setminus R_v| \geq |L_v \setminus R_v| \geq 1$, so $|L_v \setminus L_w| + |L_w \setminus R_v| \geq 1$. Using the same argument on the different sets, we obtain another set of restrictions:

$$\begin{aligned} |L_v \setminus L_w| + |L_w \setminus R_v| &\geq 1, \\ |L_w \setminus L_v| + |L_v \setminus R_w| &\geq 1, \\ |R_v \setminus R_w| + |R_w \setminus L_v| &\geq 1, \\ |R_w \setminus R_v| + |R_v \setminus L_w| &\geq 1. \end{aligned} \quad (10)$$

Looking at the vertices of level ℓ, we deduce that $\chi_{v_L} = 1$ if $(L_v \cup L_w) \setminus (R_v \cup R_w) = \emptyset$. However, $(L_v \cup L_w) \setminus (R_v \cup R_w) = ((L_v \setminus R_v) \cap (L_v \setminus R_w)) \cup ((L_w \setminus R_v) \cap (L_w \setminus R_w))$. So $\chi_{v_L} = 1$ if $|L_v \setminus R_v| + |L_w \setminus R_w| = 0$. It follows that:

$$\begin{aligned} |L_v \setminus R_w| + |L_w \setminus R_v| + \chi_{v_L} &\geq 1, \\ |R_w \setminus L_v| + |R_v \setminus L_w| + \chi_{v_R} &\geq 1, \\ |L_v \setminus L_w| + |R_w \setminus R_v| + \chi_{w_L} &\geq 1, \\ |L_w \setminus L_v| + |R_v \setminus R_w| + \chi_{w_R} &\geq 1. \end{aligned} \quad (11)$$

With all these restrictions observe the following cases:

(1) $\mathbf{L_v = L_w}$

From this fact we directly deduce that $|L_v \setminus L_w| = |L_w \setminus L_v| = 0$.

Replacing these in (9), we get $|R_w \setminus L_v| \geq 1$, $|R_v \setminus L_w| \geq 1$.

And from (10) it follows that $|L_w \setminus R_v| \geq 1$, $|L_v \setminus R_w| \geq 1$.

(a) $\mathbf{R_v = R_w}$
From this fact we directly deduce that $|R_v \setminus R_w| = |R_w \setminus R_v| = 0$.
Replacing these equalities and the previous ones in (11), we get $\chi_{w_L} = \chi_{w_R} = 1$.
Replacing all of them in (6, 7 or 8), we obtain $f(w') + f(v') \geq f(v) + f(w) + 4$.

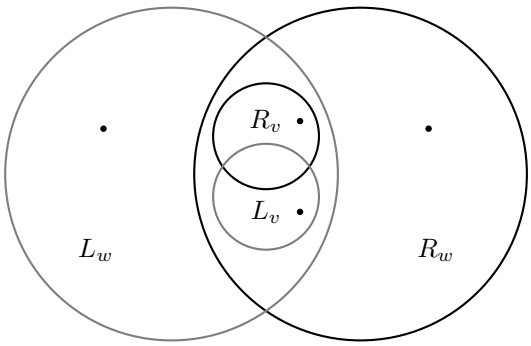
(b) $\mathbf{R_v \subsetneq R_w}$
From this fact, we directly deduce that $|R_v \setminus R_w| = 0$ and $|R_w \setminus R_v| \geq 1$.
Replacing these equalities and the previous ones in (11), we get $\chi_{w_R} = 1$.
Replacing them in (8), we obtain $f(w') + f(v') \geq f(v) + f(w) + 4$.

(c) $\mathbf{R_v \not\subseteq R_w}$ **and** $\mathbf{R_w \not\subseteq R_v}$
From this fact, we directly deduce that $|R_v \setminus R_w| \geq 1$, $|R_w \setminus R_v| \geq 1$.
Replacing them in (6, 7 or 8), we obtain $f(w') + f(v') \geq f(v) + f(w) + 3$.

(2) $\mathbf{L_v \subsetneq L_w}$
From this fact, we directly deduce that $|L_v \setminus L_w| = 0$ and $|L_w \setminus L_v| \geq 1$.
Replacing these in (9), we get $|R_w \setminus L_v| \geq 1$.
And from (10), we obtain $|L_w \setminus R_v| \geq 1$.

(a) $\mathbf{R_v \subsetneq R_w}$
From this fact, we directly deduce that $|R_v \setminus R_w| = 0$ and $|R_w \setminus R_v| \geq 1$.
We cannot infer anything about $|R_v \setminus L_w|$ and $|L_v \setminus R_w|$, but we can make the following observations:

- If $|R_v \setminus L_w| = 0$ and $|L_v \setminus R_w| = 0$, then $R_v \subset L_w$ and $L_v \subset R_w$. Recall that $R_v \setminus L_v \neq \emptyset$, $L_w \setminus R_w \neq \emptyset$, $L_v \setminus R_v \neq \emptyset$ and $R_w \setminus L_w \neq \emptyset$. Drawing a Venn diagram of the sets and placing elements in the nonempty subsets, we realize that $|R_w \setminus R_v| \geq 2$, $|L_w \setminus L_v| \geq 2$, $|R_w \setminus L_v| \geq 2$, and $|L_w \setminus R_v| \geq 2$.



  Replacing them in (6), we obtain $f(w') + f(v') \geq f(v) + f(w) + 4$.
- If either $|R_v \setminus L_w| \geq 1$ or $|L_v \setminus R_w| \geq 1$ (or both), replacing them in (8), we obtain $f(w') + f(v') \geq f(v) + f(w) + 3$.

(b) $\mathbf{R_w \subsetneq R_v}$
From this fact, we directly deduce that $|R_v \setminus R_w| \geq 1$ and $|R_w \setminus R_v| = 0$.

Replacing these in (9), we get $|L_v \setminus R_w| \geq 1$.
From (10), it follows that $|R_v \setminus L_w| \geq 1$.
And from (11), we get $\chi_{w_L} = 1$.
Replacing them in (8), we obtain $f(w') + f(v') \geq f(v) + f(w) + 4$.

(c) $\mathbf{R_v \not\subseteq R_w}$ **and** $\mathbf{R_w \not\subseteq R_v}$
From this fact, we directly deduce that $|R_v \setminus R_w| \geq 1$ and $|R_w \setminus R_v| \geq 1$.
Replacing them in (7), we obtain $f(w') + f(v') \geq f(v) + f(w) + 3$.

(3) $\mathbf{L_v \not\subseteq L_w}$ **and** $\mathbf{L_w \not\subseteq L_v}$
From this fact, we directly deduce that $|L_v \setminus L_w| \geq 1$ and $|L_w \setminus L_v| \geq 1$.

(a) $\mathbf{R_v \not\subseteq R_w}$ **and** $\mathbf{R_w \not\subseteq R_v}$
From this fact, we directly deduce that $|R_v \setminus R_w| \geq 1$ and $|R_w \setminus R_v| \geq 1$.
We know from (11) that

$$|L_v \setminus R_w| + |L_w \setminus R_v| + \chi_{v_L} \geq 1.$$
$$|R_w \setminus L_v| + |R_v \setminus L_w| + \chi_{v_R} \geq 1.$$

Replacing them in (6), we obtain $f(w') + f(v') \geq f(v) + f(w) + 3$.

As all cases have been examined, (4) follows and with this the theorem holds. □

3.2. *k-chromatic graphs*

We now extend the lower bound to k-chromatic graphs by using the same graph structure described previously in the following way.

If k is even ($k = 2m$), each of the 2^s instances consists of m identical copies of one bipartite problem instance, namely $I_1, \ldots, I_m$. Moreover, a vertex v_s^ℓ in the level ℓ of the graph I_s is adjacent to every vertex in I_t for all $t \neq s$.

The bipartite lower bound tells us that, to color each I_s, any algorithm ALG requires at least $(3s + 8)/4$ colors in expectation. Due to all the adjacencies in between the bipartite instances, no colors may be repeated in I_s, I_t for all $s \neq t$. Adding up the expected number of colors for each I_s, we get an expected lower bound of $k(3s + 8)/8$.

If k is odd ($k = 2m + 1$), each instance consists of one copy of an even instance I_{2m} and an extra vertex in each level ℓ adjacent to all the vertices of I_{2m} in the levels $\leq \ell$. This makes the instances $(2m + 1)$-chromatic and an extra color is required for this vertex. And we obtain the following lower bound.

Theorem 4. *Any algorithm* ALG *requires in expectation (taken over all k-chromatic graphs from S for the worst-case set S) at least $\lfloor k(3s + 8)/8 \rfloor$ colors.*

4. Lower Bound for Minimalistic Online Algorithms

In order to prove matching lower bounds to the upper bounds proved in Sec. 2, we restrict our attention to a natural subclass of so-called *minimalistic online algorithms* having the following property: They never use a new color if the presented vertex

can be colored with one of the colors used up to the current step. We leave the existence of a matching lower bound for unrestricted algorithms as an open problem.

4.1. *Bipartite graphs*

We start again with the bipartite case by proving the lower bound $s + 2$ on the expected number of colors if the adversary has s random bits. We use the same adversarial instances as in the previous section and we analyze the performance of minimalistic online algorithms on them.

To get some intuition about the proof strategy, consider first the instances of level 1 in Fig. 3. A minimalistic online algorithm ALG will receive first I_1 and I_2, and will use 2 colors to color them. Without loss of generality, we can assume that the left vertices in I_1 and I_2 receive one color c_1 and the right ones receive c_2. In this case, in the straight example, ALG can color u with c_2 and v with c_1, however, ALG will need two extra colors for u and v in the flipped case where both u and v have c_1 and c_2 in neighboring vertices. On average, ALG will use $3 = (1 + 2)$ colors to color a level 1 instance.

Observation 1. Let $I(w)$ be an instance of level m with $1 \leq m \leq s$ represented by the vertex w in the butterfly graph. In any coloring of $I(w)$ by a minimalistic online algorithm ALG, either

$$l(w) = r(w) \text{ and } |L_w \setminus R_w| = |R_w \setminus L_w| = 1$$

or

$$|l(w) - r(w)| = 1.$$

Proof. The claim follows from the fact that ALG will only use another color if it is not possible to use a previously used color. Without loss of generality, suppose that the left partition needs a new color. If it is not possible to reuse a color for the top-most vertex of $L(w)$, this means that all the colors are used in both $L(w)$ and $R(w)$, except maybe one that is only used in $R(w)$, i.e., $r(w) = l(w)$ or $r(w) = l(w) + 1$. This means that either they share all the colors except for the top-most layer (where each partition has one color of its own), or they share all the colors but one (in the top layer of one side). □

We call an online algorithm on the set S of instances *symmetric* if it uses the same strategies to color two different copies of the same subgraph of level $m \leq s$ with respect to the order in which the vertices are presented.

Lemma 5. *For each $s \in \mathbb{N}$, any minimalistic symmetric online algorithm* ALG *working on the set of instances S uses at least $s + 2$ colors on average.*

Proof. Let us prove this fact by induction on the number of random bits s. Let $s = 0$. Then there is only $2^s = 1$ instance, which is K_2, and any minimalistic symmetric online algorithm ALG needs $s + 2 = 2$ colors. Let us assume now that, on any instance of level $m \leq s$, ALG uses $m + 2$ colors on average.

In general, let us consider two instances of level $s - 1$, $I(v)$ and $I(w)$, that merge together in the straight $I(v')$ and flipped $I(w')$ instances at random as described in the butterfly graph. By induction, ALG uses $(s - 1) + 2 = s + 1$ colors on average on all of the possible 2^{s-1} instances of level $s - 1$. By symmetry, ALG has used the same strategy for $I(v)$ as for $I(w)$, so either $L(v) = L(w)$ and $R(v) = R(w)$, or $L(v) = R(w)$ and $R(v) = L(w)$. In both cases, either the straight or the flipped instance will require two extra colors while the other will require none. This means that any symmetric ALG will require on average one more color to cover the last step, which concludes the proof. □

We are not completely satisfied with forcing the online algorithms to be symmetric, because this looks like a strong restriction. We can remove this requirement by using the full power of the adversary. For a given minimalistic online algorithm ALG, the adversary reorders the sequences of vertices in all $I \in S$ in such a way that ALG behaves on every instance as if it were a symmetric algorithm. The adversary can do this because it can select a hardest set of instances S' in a worst-case manner with respect to ALG. In particular, any instance $I \in S$ can be modified to force ALG to behave symmetrically in S' by presenting the vertices in the following order.

When presenting the vertices of level 0, the adversary presents first all the vertices on the "left" that will be isolated at this point. Now, since ALG is minimalistic, it will color all the vertices with the same color. Thus, when the vertices on the "right" are presented, each connected to a vertex on the "left", all of them will receive a second color, the same for all of them. By induction, assume that level $\ell - 1$ has been presented and we have $2^{s-(\ell-1)}$ copies of the same subgraph colored with the same set of colors in the "left" and "right" partitions, $L_{\ell-1}$ and $R_{\ell-1}$, respectively (the naming is arbitrary because the straight and flipped states will alter the tags of left and right in some subgraphs). Now every vertex of level ℓ will be the neighbor of two partitions, either $L^1_{\ell-1}$ and $L^2_{\ell-1}$ or $R^1_{\ell-1}$ and $R^2_{\ell-1}$ in the straight case, or $L^1_{\ell-1}$ and $R^2_{\ell-1}$ or $R^1_{\ell-1}$ and $L^2_{\ell-1}$ in the flipped case. As all vertices of level ℓ will be straight or flipped, we can present all the vertices of the left partition first, attached to $R^1_{\ell-1}$ and $R^2_{\ell-1}$ (or $R^1_{\ell-1}$ and $L^2_{\ell-1}$ in the flipped case). If a new color is needed, it will be needed for all of the left vertices of this level and the same color will be used for all of them by minimality. If no new color is needed, only one color will be available by minimality, so again no conflict arises. Hence, when the right vertices are presented, all of them will receive the same color analogously and, since each pair of vertices of level ℓ connects two of the subgraphs of level $\ell - 1$, we will obtain $2^{s-\ell}$ copies of the same subgraph colored with the same set of colors. This finally yields the following result.

Theorem 6. *For each $s \in \mathbb{N}$, any minimalistic online algorithm working on the set of instances S described in the previous section uses at least $s+2$ colors on average.* □

4.2. *k-chromatic graphs*

We now extend the lower bound to k-chromatic graphs in a similar way as in Subsec. 3.2.

Creating more copies of an instance of level s and connecting them by adding edges from each vertex of one copy to all vertices of all other copies, one gets the same structure as in Subsec. 3.2, and using exactly the same argument as in Subsec. 3.2 and the lower bound for minimalistic online algorithms in bipartite graphs we obtain the following theorem.

Theorem 7. *For any $k, s \in \mathbb{N}$, and any minimalistic online coloring algorithm* ALG*, there exists an adversary using s random bits to construct 2^s problem instances that are k-chromatic such that the competitive ratio of* ALG *is at least $(s+2)/2$.* □

5. Conclusions and Open Problems

In this paper, we presented upper and lower bounds within the same order of magnitude for the coloring problem against a randomized adversary. An idea to continue this line of research would be to prove a matching lower bound for unrestricted algorithms.

The original point of view on the model we have worked with is given by looking into advice complexity. Advice complexity was introduced in [11] and revised in [5, 12, 17] in order to measure the information content of online problems. The question is how many bits about the future are necessary and sufficient for an online algorithm in order to be able to solve a given problem in an optimal way or to guarantee a concrete competitive ratio. Studying online problems from this point of view by getting tradeoffs between the solution quality and the size of advice provided, one obtained a new instrument for measuring the hardness of online problems. Other important conceptual contributions are the development of a powerful method for proving lower bounds on the achievable competitive ratios of randomized online algorithms and new insights on the potential power of information.

The investigations in a series of papers [1–4, 6, 10, 13, 14, 22, 23, 29] show very different behaviors of the tradeoff between the solution quality measured by the competitive ratio and the amount of information of the unknown future. Sometimes this behavior looks really surprising. The typical patterns occurring for different problems are the following ones.

(1) The achievable competitive ratio improves continuously with the number of advice bits provided. Sometimes very quickly, sometimes very slowly.

(2) The number of advice bits provided does not help at all until a special threshold value is reached. After crossing this threshold, the quality of solutions may jump to a significantly better competitive ratio.
(3) The pattern can be a mix of (1) and (2), depending on the interval of the number of advice bits offered.

In the classical advice model, there are three players, namely an online algorithm, the adversary, and the advisor (oracle). As usual in the classical model, the adversary knows everything about a given online algorithm and constructs the hardest problem instance for it with respect to the achievable competitive ratio. The advisor in the extended game is very powerful, it knows everything about the future, i. e., it knows the whole input instance that will be presented request by request. The advisor writes its advice on the oracle tape, and the number of bits read by the online algorithm from the tape is the advice complexity on this problem instance. The advice complexity of an online algorithm is in general a function of the input size defined in a worst-case manner as the maximum over all inputs of the same size.

The model we propose is now the following. The adversary constructs a set of 2^s instances S, but the algorithm does not know anything about S. The advisor provides an advice tape to the algorithm with some information about S. However, the advisor does not know the outcome of the random selection (the random bits of the adversary), so it will not be able to provide information on the real input, only on the input set.

The goal would be to obtain results on the tradeoff between the advice complexity and the competitive ratio to see whether the behavior is similar to any of those observed in the classical online model. The results of this paper are only a starting point for this research, because they deal only with the extremal situation when the full information about S is offered to the algorithm.

Acknowledgments

This work has been partially supported by the SNF grant 200021-146372 and the Spanish government under project TEC2013-47960-C4-1-P. An extended abstract of this paper was published at SOFSEM 2016. Rastislav Královič was partially supported by grant VEGA 2/0165/16.

References

[1] Kfir Barhum, Hans-Joachim Böckenhauer, Michal Forišek, Heidi Gebauer, Juraj Hromkovič, Sacha Krug, Jasmin Smula, and Björn Steffen. On the power of advice and randomization for the disjoint path allocation problem. In Viliam Geffert, Bart Preneel, Branislav Rovan, Július Štuller, and A. Min Tjoa, editors, *SOFSEM 2014: Theory and Practice of Computer Science: 40th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 26-29, 2014, Proceedings*, Springer International Publishing, Cham, 2014, pages 89–101.

[2] Maria Paola Bianchi, Hans-Joachim Böckenhauer, Juraj Hromkovič, and Lucia Keller. Online coloring of bipartite graphs with and without advice. *Algorithmica*, **70**(1):92–111, Sep 2014.

[3] Hans-Joachim Böckenhauer, Juraj Hromkovič, Dennis Komm, Sacha Krug, Jasmin Smula, and Andreas Sprock. The string guessing problem as a method to prove lower bounds on the advice complexity. *Theoretical Computer Science*, **554**(Supplement C):95–108, 2014. Computing and Combinatorics.

[4] Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královič, and Richard Královič. On the advice complexity of the k-server problem. *Journal of Computer and System Sciences*, **86**(Supplement C):159–170, 2017.

[5] Hans-Joachim Böckenhauer, Dennis Komm, Rastislav Královič, Richard Královič, and Tobias Mömke. On the advice complexity of online problems. In Yingfei Dong, Ding-Zhu Du, and Oscar Ibarra, editors, *Algorithms and Computation: 20th International Symposium, ISAAC 2009, Honolulu, Hawaii, USA, December 16-18, 2009. Proceedings*, pages 331–340, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[6] Hans-Joachim Böckenhauer, Dennis Komm, Richard Královič, and Peter Rossmanith. The online knapsack problem: Advice and randomization. *Theoretical Computer Science*, **527**(Supplement C):61–72, 2014.

[7] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis.* (Cambridge University Press, 1998).

[8] Hajo J. Broersma, Agostino Capponi, and Daniël Paulusma. A new algorithm for on-line coloring bipartite graphs. *SIAM Journal on Discrete Mathematics*, **22**(1):72–91, 2008.

[9] Stefan Dobrev, Juraj Hromkovič, Dennis Komm, Richard Královič, Rastislav Královič, and Tobias Mömke. The complexity of paging against a probabilistic adversary. In Rūsiņš Mārtiņš Freivalds, Gregor Engels, and Barbara Catania, editors, *SOFSEM 2016: Theory and Practice of Computer Science: 42nd International Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 23-28, 2016, Proceedings*, pages 265–276, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[10] Stefan Dobrev, Rastislav Královič, and Euripides Markou. Online graph exploration with advice. In Guy Even and Magnús M. Halldórsson, editors, *Structural Information and Communication Complexity: 19th International Colloquium, SIROCCO 2012, Reykjavik, Iceland, June 30-July 2, 2012, Revised Selected Papers*, pages 267–278, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[11] Stefan Dobrev, Rastislav Královič, and Dana Pardubská. Measuring the problem-relevant information in input. *RAIRO — Theoretical Informatics and Applications*, 43(3):585613, 2009.

[12] Yuval Emek, Pierre Fraigniaud, Amos Korman, and Adi Rosén. Online computation with advice. *Theoretical Computer Science*, **412**(24):2642–2656, 2011. Selected Papers from 36th International Colloquium on Automata, Languages and Programming (ICALP 2009).

[13] Michal Forišek, Lucia Keller, and Monika Steinová. Advice complexity of online coloring for paths. In Adrian-Horia Dediu and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications: 6th International Conference, LATA 2012, A Coruña, Spain, March 5-9, 2012. Proceedings*, pages 228–239, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[14] Heidi Gebauer, Dennis Komm, Rastislav Královič, Richard Královič, and Jasmin Smula. Disjoint path allocation with sublinear advice. In Dachuan Xu, Donglei Du, and Dingzhu Du, editors, *Computing and Combinatorics: 21st International Conference,*

COCOON 2015, Beijing, China, August 4-6, 2015, Proceedings, pages 417–429, Cham, 2015. Springer International Publishing.

[15] Andras Gyárfás, Zoltan Király, and Jenö Lehel. On-line competitive coloring algorithms. Technical Report TR-9703-1, 1997.

[16] Andras Gyárfás and Jenö Lehel. On-line and first fit colorings of graphs. *Journal of Graph Theory*, 12(2):217–227, 1988.

[17] Juraj Hromkovič, Rastislav Královič, and Richard Královič. Information complexity of online problems. In Petr Hliněný and Antonín Kučera, editors, *Mathematical Foundations of Computer Science 2010: 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, pages 24–36, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[18] Lucia Keller. *Complexity of Optimization Problems: Advice and Approximation*. PhD thesis, ETH Zürich, 2014.

[19] H. A. Kierstead and W. T. Trotter. On-line graph coloring. *On-line Algorithms, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 7:85–92, 1992.

[20] H.A. Kierstead. Chapter 18 recursive and on-line graph coloring. In Yu. L. Ershov, S.S. Goncharov, A. Nerode, J.B. Remmel, and V.W. Marek, editors, *Handbook of Recursive Mathematics*, volume 139 of *Studies in Logic and the Foundations of Mathematics*, pages 1233–1269. Elsevier, 1998.

[21] Dennis Komm. *An Introduction to Online Computation - Determinism, Randomization, Advice*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016.

[22] Dennis Komm and Richard Královič. Advice complexity and barely random algorithms. *RAIRO — Theor. Inf. and Applic.*, 45(2):249–267, 2011.

[23] Dennis Komm, Richard Královič, and Tobias Mömke. On the advice complexity of the set cover problem. In *CSR*, Volume 7353 of *Lecture Notes in Computer Science*, pages 241–252. Springer, 2012.

[24] Elias Koutsoupias and Christos H. Papadimitriou. Beyond competitive analysis. *SIAM J. Comput.*, 30(1):300–317, 2000.

[25] László Lovász, Michael Saks, and W.T. Trotter. An on-line graph coloring algorithm with sublinear performance ratio. *Discrete Mathematics*, 75(1):319–325, 1989.

[26] P. Raghavan. A statistical adversary for on-line algorithms. *On-line Algorithms, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 79–83, 1991.

[27] Jasmin Smula. *Information Content of Online Problems*. PhD thesis, ETH Zürich, 2015.

[28] Sundar Vishwanathan. Randomized online graph coloring. *Journal of Algorithms*, 13(4):657–669, 1992.

[29] David Wehner. Advice complexity of fine-grained job shop scheduling. In Vangelis Th. Paschos and Peter Widmayer, editors, *Algorithms and Complexity: 9th International Conference, CIAC 2015, Paris, France, May 20-22, 2015. Proceedings*, pages 416–428, Cham, 2015. Springer International Publishing.

[30] Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 2 edition, September 2001.