Online Multi-Coloring with Advice*

Marie G. Christ, Lene M. Favrholdt, and Kim S. Larsen

University of Southern Denmark, Odense, Denmark
{christm,lenem,kslarsen}@imada.sdu.dk

Abstract. We consider the problem of online graph multi-coloring with advice. Multi-coloring is often used to model frequency allocation in cellular networks. We give several nearly tight upper and lower bounds for the most standard topologies of cellular networks, paths and hexagonal graphs. For the path, negative results trivially carry over to bipartite graphs, and our positive results are also valid for bipartite graphs. The advice given represents information that is likely to be available, studying for instance the data from earlier similar periods of time.

1 Introduction

We consider the problem of graph *multi-coloring*, where each node may receive multiple requests. Whenever a node is requested, a color must be assigned to the node, and this color must be different from any color previously assigned to that node or to any of its neighbors. The goal is to use as few colors as possible. In the online version, the requests arrive one by one, and each request must be colored without any information about possible future requests. The underlying graph is known to the online algorithm in advance.

The problem is motivated by *frequency allocation* in cellular networks. These networks are formed by a number of base transceiver stations, each of which covers what is referred to as a cell. Due to possible interference, neighboring cells cannot use the same frequencies. In this paper, we use classic terminology and refer to these cells as nodes in a graph where nodes are connected by an edge if they correspond to neighboring cells in the network. Frequencies can then be modeled as colors. Multiple requests for frequencies can occur in one cell and overall bandwidth is a critical resource.

Two basic models dominate in the discussion of cellular networks, the highway and the city model. The former is modeled by linear cellular networks, corresponding to paths, and the latter by hexagonal graphs. We consider the problem of multi-coloring such graphs.

If A is a multi-coloring algorithm, we let $\mathrm{A}(I)$ denote the number of colors used by A on the input sequence I. When I is clear from the context, we simply write A instead of $\mathrm{A}(I)$. The quality of an online algorithm is often given in terms of the competitive ratio [37, 28]. An online multi-coloring algorithm is

* Supported in part by the Danish Council for Independent Research and the Villum Foundation.

c-competitive if there exists a constant α such that for all input sequences I, $\mathrm{A}(I) \leq c\,\textsc{Opt}(I) + \alpha$. The (asymptotic) *competitive ratio* of A is the infimum over all such c. Results that can be established using $\alpha = 0$ are referred to as *strict* (or absolute). Often, it is a little unclear when one refers to an *optimal* online algorithm, whether this means that the solution produced is as good as the one produced offline or that no better online algorithm can exist. For that reason, we may use the term *strictly* 1*-competitive* to emphasize that an algorithm is as good as an optimal offline algorithm, and *optimal* to mean that no better online algorithm exists under the given conditions. Throughout, we let n denote the number of requests in a given input sequence.

For practical applications, the assumption that absolutely nothing is known about the future is often unrealistic. A way of relaxing this very strict and somewhat unnatural assumption is to give the algorithm some *advice*. A recent trend in the analysis of online algorithms has been to consider advice, formalized under the notion of *advice complexity*, starting in [20].

This realization that input is not arbitrary (uniformly random, for instance) is not new, and work focused on locality of reference in input data has tried to capture this. Early work includes access graph results, starting in [8], and with references to additional related work in [10], but also more distributional models, such as [1], have been developed. In [12] an entirely different concept of accommodating sequences was introduced and further developed in [13, 9]. The idea is that for many problems requiring resources, there is a close connection between the resources available and the resources required for an optimal offline algorithm, as when capacity of transportation systems are matched with expected demand. This leans itself closely up against many of the results that we report here, where the advice needed to do better is often some information regarding the resources required by an optimal offline algorithm.

Thus, the results in this paper could have practical applications. The results establish which type of information is useful, how algorithms should be designed to exploit this information, and what the limits are for what can be obtained.

Returning to the *advice complexity modeling*, some problems need very little advice. On the other hand, complete information about the input or the desired output is a trivial upper bound on the amount of advice needed to be optimal. The first approach to formalizing the concept of advice measured the number of bits per request [20]. This model is well suited for some problems where information is tightly coupled with requests and the number of bits needed per request is constant. However, for most problems, we prefer the model where we simply measure the total advice needed throughout the execution of the algorithm. As also discussed in [5, 25], this model avoids some modeling issues present in the "per request" modeling, and at the same time makes it possible to derive sublinear advice requirements. Thus, we use the advice model from [25], where the online algorithm has access to an infinite advice tape, written by an offline oracle with infinite computation power. In other words, the online algorithm can ask for the answer to any question and read the answer from the tape. Competitiveness is defined and measured as usual, and the advice

complexity is simply the number of bits read from the tape, i.e., the maximum index of the bits read from the advice tape.

As the advice tape is infinite, we need to specify how many bits of advice the algorithm should read and if this knowledge is not implicitly available, it has to be given explicitly in the advice string. For instance, if we want OPT as advice (the number of colors an optimal offline algorithm uses on a given sequence, for instance), then we cannot merely read $\lceil \log(\text{OPT}+1) \rceil$ (all logs in this paper are base 2) bits, since this would require knowing something about the value of OPT. One can use a *self-delimiting encoding* as introduced in [22]. We use the variant from [11], defined as follows: The value of a non-negative integer X is encoded by a bit sequence, partitioned into three consecutive parts. The last part is X written in binary. The middle part gives the number of bits in the last part, written in binary. The first part gives the number of bits in the middle part, written in unary and terminated with a zero. These three parts require $\lceil \log(\lceil \log(X+1) \rceil + 1) \rceil + 1$, $\lceil \log(\lceil \log(X+1) \rceil + 1) \rceil$, and $\lceil \log(X+1) \rceil$ bits, respectively, adding a lower-order term to the number of bits of information required by an algorithm. We define $enc(x)$ to be the minimum number of bits necessary to encode a number x, and note that the encoding above is a (good) upper bound on $enc(x)$.

We now discuss *previous work* on multi-coloring and then state our results. When working with online algorithms, decisions are generally irrevocable, i.e., once a color is assigned to a node, this decision is final. However, in some applications, local changes of colors may be allowed (reassignment of frequencies). This is called *recoloring*. An algorithm is *d-recoloring* if, in the process of treating a request, it may recolor up to a distance d away from the node of the request.

For multi-coloring a path, the algorithm 4-BUCKET is $\frac{4}{3}$-competitive [19], and this is optimal [15]. Even with 0-recoloring allowed (that is, colors at the requested node may be changed), 4-BUCKET is optimal [16]. Furthermore, if 1-recoloring is allowed, the algorithm GREEDYOPT is strictly 1-competitive [16].

For multi-coloring bipartite graphs, the optimal asymptotic competitive ratio lies between $\frac{10}{7} \approx 1.428$ and $\frac{18-\sqrt{5}}{11} \approx 1.433$ [18].

In [14], it was shown that, for hexagonal graphs, no online algorithm can be better than $\frac{3}{2}$-competitive or have a better strict competitive ratio than 2. They also gave an algorithm, HYBRID, with an asymptotic competitive ratio of approximately 1.9 on hexagonal graphs. On k-colorable graphs, it is strictly $\frac{k+1}{2}$-competitive, and hence, it has an optimal strict competitive ratio on hexagonal graphs. Recoloring was studied in [27]: No d-recoloring algorithm for hexagonal graphs has an asymptotic competitive ratio better than $1+\frac{1}{4(d+1)}$. For $d=0$, the lower bound was improved to $\frac{9}{7}$. In [38], a $\frac{4}{3}$-competitive 2-recoloring algorithm is given. The best known 1-recoloring algorithm for hexagonal graphs is $\frac{33}{24}$-competitive [39]. For the offline problem of multi-coloring hexagonal graphs, no polynomial time algorithm can obtain an absolute approximation ratio better than $\frac{4}{3}$ [32, 34, 35], unless P = NP.

Table 1. Overview of our results. Recall that n denotes the number of requests in the input sequence. We mark the ratios that are strict by "s" and the ones that are asymptotic by "a". Note that a strict lower bound can be larger than an asymptotic upper bound. For each bound, we indicate the number of the theorem proving the result. For readability, many of the bounds stated are weaker than those proven in the paper. Moreover, the upper bounds for the path hold for any bipartite graph. The result of Theorem 3 in the third row of the table is valid only for *neighborhood-based* algorithms, as defined just before Theorem 3 in Section 2.

	Ratio	Lower bound	Type	Result	Upper bound	Type	Result
Path	1	$\log n - 2$	s	Thm 1	$\log n + O(\log\log n)$	s	Thm 4
	$1 + \frac{1}{2^b}$	$b - 2$	a	Thm 2	$b + 1 + O(\log\log n)$	s	Thm 5
	$< \frac{4}{3}$	$\omega(1)$	a	Thm 3			
Hex.	1				$(n+1)\lceil\log n\rceil$	s	Thm 8
	$< \frac{5}{4}$	$\Omega(n)$	a	Thm 7			
	$\frac{4}{3}$				$n + 2\lvert V\rvert$	a	Thm 10
	$\frac{3}{2}$	$\lfloor\frac{n-1}{3}\rfloor$	s	Thm 6	$\log n + O(\log\log n)$	a	Thm 9

Many other problems have been considered in the advice model, see e.g., [2, 4–7, 20, 21, 23, 29, 30]; also variants of graph coloring different from ours [3, 24, 31, 36].

An overview of *our results* is given in Table 1. For the path, these results are nearly tight, even with upper bounds that also apply to bipartite graphs. For hexagonal graphs, note that with a linear number of advice bits, it is possible to be $\frac{4}{3}$-competitive, and the lower bound for being better than $\frac{5}{4}$-competitive is close to linear. The advice given to the algorithms is essentially (an approximation of) Opt or the maximum number of requests given to any clique in the graph. For the underlying problem of frequency allocation, guessing these values based on previous data may not be unrealistic.

Due to space restrictions, some proofs have been removed or shortened. These can be found in the full version [17].

2 The Path

As explained earlier, we establish all lower bounds for paths, and since a path is bipartite, all these negative results carry over to bipartite graphs. Similarly, all our (constructive) upper bounds are given for bipartite graphs and therefore also apply to paths. We start with three lower bound results.

Theorem 1. *Any strictly* 1*-competitive online algorithm for multi-coloring paths of at least 10 nodes has advice complexity at least* $\lceil\log(\lfloor\frac{n}{4}\rfloor + 1)\rceil$.

Proof. We let $m = \lfloor\frac{n}{4}\rfloor$ and define a set S of $m + 1$ sequences, all having the same prefix of length $2m$. The set S will have the following property: for no two

sequences in S can their prefixes be colored in the same way while ending up using the optimal number of colors on the complete sequence. Starting from one end of the path, we denote the nodes $v_1, v_2, \ldots$.

We define the set S to consist of the sequences $I_0, I_1, \ldots, I_m$, where I_i is defined in the following way. First m requests are given to each of the nodes v_1 and v_4. Then i requests to each of v_2 and v_3. To give all sequences the same length, the sequence ends with $\lceil n - 2m - 2i \rceil$ requests distributed as evenly as possible among v_6, v_8, and v_{10}. Since $\lceil \lceil n - 2m - 2i \rceil / 3 \rceil \leq m$, the optimal number of colors will not be influenced by this part of the sequence.

Note that $\textsc{Opt}(I_i) = m + i$. In order not to use more than $\textsc{Opt}(I_i)$ colors for I_i, exactly i of the colors assigned to v_4 have to be different from the colors assigned to v_1. The prefixes of length $2m$ in S are identical, so all information to distinguish between the different sequences must be given as advice. The cardinality of S is $m+1$. To specify one out of $m+1$ possible actions, $\lceil \log(m+1) \rceil$ bits are necessary. □

For algorithms that are $\frac{9}{8}$-competitive or better, we give the following lower bound.

Theorem 2. *Consider multi-coloring paths of at least 10 nodes. For any $b \geq 3$ and any $(1 + \frac{1}{2^b})$-competitive algorithm, A, there exists an $N \in \mathbb{N}$ such that A has advice complexity at least $b - 2$ on sequences of length at least N.*

Proof. For any $(1+\frac{1}{2^b})$-competitive algorithm, A, there exists an $\alpha \geq 1$ such that $\mathrm{A}(I) \leq (1 + \frac{1}{2^b})\,\textsc{Opt}(I) + \alpha$, for any input sequence I. We consider sequences of length $n \geq 2^{2b+2}\alpha + 3$.

Let $m = \lfloor \frac{n}{4} \rfloor$ and consider the same set of sequences as in the proof of Theorem 1. Recall that $\textsc{Opt}(I_i) = m + i$. For the sequence I_i, let x_i denote the number of colors that A uses on v_4 but not on v_1. Then, A uses $m + x_i$ colors in total for v_1 and v_4. On v_3, it can use at most x_i of the colors used at v_1, so the total number of colors used at v_1, v_2, and v_3 is at least $m + 2i - x_i$. Thus, $\mathrm{A}(I_i) \geq \max\{m + x_i, m + 2i - x_i\}$.

We will prove that there are $p \geq 2^{b-2}$ sequences $I_{i_1}, I_{i_2}, \ldots, I_{i_p}$ such that, for any pair $i_j \neq i_k$, we have $x_{i_j} \neq x_{j_k}$, or otherwise A would not be $(1 + \frac{1}{2^b})$-competitive. This will immediately imply that A must use at least $b - 2$ advice bits.

Let $\varepsilon = \frac{1}{2^b} + \frac{1}{2^{2b}}$. From $\mathrm{A}(I_i) \leq (1+\frac{1}{2^b})\,\textsc{Opt}(I_i)+\alpha$ and $m \geq 2^{2b}\alpha$, we obtain the inequalities

$$m + x_i \leq (1 + \varepsilon)(m + i) \qquad \text{and} \qquad m + 2i - x_i \leq (1 + \varepsilon)(m + i)$$

which reduce to

$$x_i \leq \varepsilon m + (1 + \varepsilon)i \quad (1) \qquad \text{and} \qquad i \leq \frac{x_i + \varepsilon m}{1 - \varepsilon} \quad (2)$$

Hence, by (1), $x_0 \leq \varepsilon m$. Thus, by (2), we can have $x_i = x_0$ only if $i \leq \frac{2\varepsilon m}{1-\varepsilon}$. Therefore, we let $i_1 = 0$ and $i_2 = \lfloor \frac{2\varepsilon m}{1-\varepsilon} + 1 \rfloor$. In general, we ensure $x_{i_j} \neq x_{i_{j+1}}$

by letting $i_{j+1} = \lfloor \frac{x_{i_j}+\varepsilon m}{1-\varepsilon} + 1 \rfloor$. Thus,

$$i_{j+1} \leq \frac{x_{i_j}+\varepsilon m}{1-\varepsilon} + 1 \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot i_j + \frac{2\varepsilon m}{1-\varepsilon} + 1,$$

where the second inequality follows from (1). Solving this recurrence relation, we get

$$i_{j+1} \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^j i_1 + \sum_{k=0}^{j-1} \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^k \left(\frac{2\varepsilon m}{1-\varepsilon} + 1\right) = \frac{\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^j - 1}{2\varepsilon} (2\varepsilon m + 1 - \varepsilon)$$

We let p equal the largest j for which $i_j \leq m$. Through arithmetic manipulations using the various bounds established above, one can show that $p \geq 2^{b-2}$. □

For the following theorem, we define the class of *neighborhood-based* algorithms: A multi-coloring algorithm, A, is called neighborhood-based, if there exists a constant d such that, when assigning a color to a request to a node v, A bases its decision only on requests to nodes a distance of at most d away from v. Note that, in particular, a neighborhood-based algorithm cannot base its decision on the current value of Opt.

Using the family of request sequences from the proofs of Theorems 1 and 2, it is fairly easy to establish a lower bound of $\omega(1)$ on the advice complexity for neighborhood-based algorithms that are better than $\frac{4}{3}$-competitive:

Theorem 3. *No neighborhood-based online algorithm for multi-coloring paths with advice complexity $O(1)$ can be better than $\frac{4}{3}$-competitive.*

We now turn to upper bounds. For multi-coloring a path, there is a strictly 1-competitive 1-recoloring algorithm, GreedyOpt [16]. GreedyOpt divides the nodes into two sets, *upper* and *lower*, such that every second node belongs to *upper* and the remaining nodes belong to *lower*. The following invariant is maintained: After each request, each node in *lower* uses consecutive colors starting with color 1 and each node in *upper* uses consecutive colors ending with a color no larger than the optimal number of colors for the request sequence seen so far.

The algorithm for paths from [16] is easily generalized to work on bipartite graphs, letting the nodes of one partition, L, belong to *lower* and the nodes of the other partition, U, belong to *upper*. Recoloring is only needed if the number of colors used by an optimal offline algorithm is not known. Hence, using *enc*(Opt) advice bits, an online algorithm can be strictly 1-competitive, even if recoloring is not allowed. For the resulting algorithm, GreedyOptAdvice, we prove the following.

Theorem 4. *Algorithm* GreedyOptAdvice *is correct, strictly* 1*-competitive, and has advice complexity enc*(Opt)*.*

We now turn to nonoptimal variants of GreedyOptAdvice using fewer than *enc*(Opt) advice bits. We show how to obtain a particular competitive ratio of $1 + \frac{1}{2^b}$, using $b + 1 + O(\log \log \text{Opt})$ bits of advice. Thus, essentially, we are approaching optimality exponentially fast in the number of bits of advice.

Theorem 5. *For any integer $b \geq 1$, there exists a strictly $(1+\frac{1}{2^{b-1}})$-competitive online algorithm for multi-coloring bipartite graphs with advice complexity $b + enc(a)$, where $a + b$ is the total number of bits in the value* Opt.

Proof. As advice, the algorithm asks for the b high order bits of the value Opt, as well as the number $a = \lceil \log(\textsc{Opt}+1) \rceil - b$ of low order bits, but not the value of these bits. The algorithm knows b and can just read the first b bits, while a needs to be encoded. Thus, $b + enc(a)$ bits are sufficient to encode the advice.

If Opt contains fewer than b bits, this is detected by a being zero. In this case, some of the b bits may be leading zeros. By Theorem 4, we can then be strictly 1-competitive. Now, assume this is not the case, and let $\textsc{Opt}_b = \lfloor \frac{\textsc{Opt}}{2^a} \rfloor$ denote the value represented by the b high order bits. The algorithm computes $m = 2^a \textsc{Opt}_b + 2^a - 1$ and runs GreedyOptAdvice with this m. Since $\textsc{Opt} \leq m \leq \textsc{Opt} + 2^a - 1$, the algorithm is correct and uses at most $\textsc{Opt} + 2^a - 1$ colors.

For any number $x \geq 1$, consisting of c bits, with the most significant bit being one, $2^c \leq 2x$. Thus, $2^{b+a} \leq 2\,\textsc{Opt}$, so $2^a \leq \frac{2\,\textsc{Opt}}{2^b}$. This means that the number of colors used by GreedyOptAdvice is less than $\textsc{Opt} + \frac{2\,\textsc{Opt}}{2^b} = (1 + \frac{1}{2^{b-1}})\,\textsc{Opt}$, so the algorithm is strictly $(1 + \frac{1}{2^{b-1}})$-competitive. □

Corollary 1. *For any $\varepsilon > 0$, there exists a strictly $(1+\varepsilon)$-competitive deterministic online algorithm for multi-coloring bipartite graphs with advice complexity $O(\log \log \textsc{Opt})$.*

Proof. Except for the term b, the advice stated in Theorem 5 is $O(\log \log \textsc{Opt})$ and $\textsc{Opt} \leq n$. Thus, we just need to bound b. For a given ε, choose b large enough such that $\frac{1}{2^{b-1}} \leq \varepsilon$. Using this value for b in Theorem 5, we obtain an algorithm with a strict competitive ratio of at most $1 + \frac{1}{2^{b-1}} \leq 1 + \varepsilon$. Since, for any given ε, b is a constant, the total amount of advice is $O(\log \log \textsc{Opt})$. □

The Multi-Coloring problem is also considered in the context of request cancellations, i.e., a color already given to a node disappears again. We remark that just by allowing 0-recoloring, where requests at the node where the cancellation takes place may be recolored, we can extend the algorithm GreedyOptAdvice, using the same advice, to a strictly 1-competitive algorithm.

3 Hexagonal Graphs

A hexagonal graph is a graph that can be obtained by placing (at most) one node in each cell of a hexagonal grid (such as the one sketched in Fig. 1) and adding an edge between any pair of nodes placed in neighboring cells. Note that any hexagonal graph can be 3-colored. This is easily seen, since it is possible to use the three colors cyclically on the cells of each row of the underlying hexagonal grid, such that no two neighboring cells receive the same color.

As in the previous section, we first focus on lower bound results.

Theorem 6. *Any online algorithm for multi-coloring hexagonal graphs with a strict competitive ratio strictly smaller than $\frac{3}{2}$ has advice complexity at least $\lfloor \frac{n-1}{3} \rfloor$.*

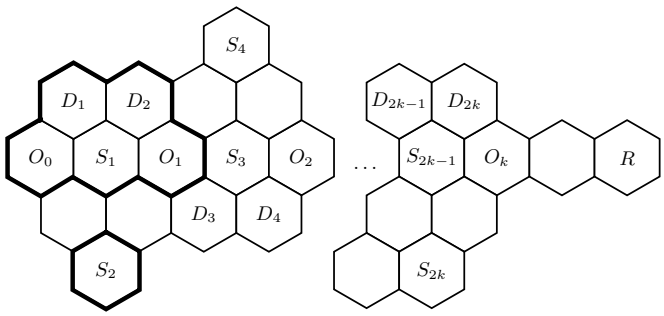

Fig. 1. Hexagonal lower bound construction.

Proof. First, we explain a small part of the construction that we will use in many copies. We consider two sequences with the same prefix of length 2. Both sequences can be colored with two colors, but this requires coloring the two prefixes of length two differently. Consider the left-most part of Fig. 1 (surrounded by thick lines) consisting of the "double" nodes D_1 and D_2, the "outer" nodes O_0 and O_1 and the "single" nodes S_1, and S_2. These nodes form the same type of configuration as the nodes D_3, D_4, O_1, O_2, S_3, and S_4.

First the nodes O_0 and O_1 get one request each. Then, either D_1 and D_2 or S_1 and S_2 receive one request each. The node S_2 is used to get up to the same sequence length in all cases. In order not to use more than two colors, the outer nodes have to use different colors if we later give requests to the two D-nodes. Similarly, the O-nodes should have the same color if we later give a request to the S-node in between them. Since the prefix of length two is $\langle O_0, O_1 \rangle$ for both sequences, all information for an algorithm to distinguish between the two sequences must be given as advice.

We can repeat this graph pattern $\lfloor \frac{n-1}{3} \rfloor$ times, as illustrated in Fig. 1 with $k = \lfloor \frac{n-1}{3} \rfloor$, giving the requests to all O-nodes first. This results in a sequence set of size $2^{\lfloor \frac{n-1}{3} \rfloor}$, implying the result. □

Theorem 7. *Any online algorithm for multi-coloring hexagonal graphs with competitive ratio strictly smaller than $\frac{5}{4}$ has advice complexity $\Omega(n)$.*

Proof. We use the basic construction from Theorem 6. Assume p requests are given to one of the components like this:

First, we give $\frac{p}{4}$ requests to each of O_0 and O_1. Let q, $0 \leq q \leq \frac{p}{4}$, denote the number of colors used at both nodes. Then following up by giving $\frac{p}{4}$ requests to each S-node results in a minimum of $\frac{3p}{4} - q$ colors used, while giving the requests to the D-nodes instead results in a minimum of $\frac{p}{2} + q$ colors.

Note that $\textsc{Opt} = \frac{p}{2}$, independent of in which of the two ways the sequence is continued. Thus, for any $\varepsilon > 0$, any $(\frac{5}{4} - \varepsilon)$-competitive algorithm must choose q

such that, for some constant α, $\frac{3p}{4} - q \leq \left(\frac{5}{4} - \varepsilon\right)\frac{p}{2} + \alpha$ and $\frac{p}{2} + q \leq \left(\frac{5}{4} - \varepsilon\right)\frac{p}{2} + \alpha$. Adding these two inequalities, we obtain $\frac{5p}{4} \leq (\frac{5}{4} - \varepsilon)p + 2\alpha$ which is equivalent to $\varepsilon p \leq 2\alpha$. Thus, if p is non-constant, no $(\frac{5}{4} - \varepsilon)$-competitive algorithm can use the same value of q for both sequences.

Now assume for the sake of contradiction that for some advice of $g(n) \in o(n)$ bits, we can obtain a ratio of $\frac{5}{4} - \varepsilon$. Let $f(n) = \frac{1}{2}\frac{n}{g(n)}$. Since $g(n) \in o(n)$, $f(n) \in \omega(1)$. The idea is now to repeat the construction as in the proof of Theorem 6 and give $f(n)$ requests to each construction ($f(n)$ has the role of p in the above). Since a pair of neighboring constructions share $f(n)/4$ requests, this results in $\frac{n-f(n)/4}{3f(n)/4} = \frac{4n-f(n)}{3f(n)} \geq \frac{n}{f(n)}$ constructions. We assume without loss of generality that all our divisions result in integers.

In order to be $(\frac{5}{4} - \varepsilon)$-competitive, an online algorithm must, for each two neighboring O-nodes, choose between at least two different values of q. These are independent decisions, and the ratio only ends up strictly better than $\frac{5}{4}$ if the algorithm decides correctly in every subconstruction. Thus, it needs at least $\frac{n}{f(n)}$ bits of advice. However, $\frac{n}{f(n)} = \frac{n}{\frac{1}{2}\frac{n}{g(n)}} = 2g(n) > g(n)$, a contradiction. □

For upper bounds, we first have the following trivial upper bound on the advice necessary to be optimal, independent of the graph topology:

Theorem 8. *There is a strictly* 1*-competitive online multi-coloring algorithm with advice complexity* $(n+1)\lceil \log \textsc{Opt} \rceil$.

In the following, we will show how two known approximation algorithms can be converted to online algorithms with advice. In the description of the algorithms, we let the *weight* of a clique denote the total number of requests to the nodes of the clique. Note that the only maximal cliques in a hexagonal graph are isolated nodes, edges, or triangles. We let ω denote the maximum weight of any clique in the graph.[1]

A $\frac{3}{2}$-competitive algorithm called the Fixed Preference Allocation algorithm, FPA, was proposed in [26]. In [33], the strategy was simplified and it was noted that the algorithm can be converted to a 1-recoloring online algorithm. We describe the simplified offline algorithm below.

The algorithm uses three color classes, R, G, and B. The color classes represent a partitioning of the nodes in the graph so that no two neighbors are in the same partition. Each of the three color classes has its own set of $\lceil \frac{\omega}{2} \rceil$ colors, and each node in a given color class uses the colors of its color class, starting with the smallest. This set of colors is also referred to as the node's *private* colors. If more than $\lceil \frac{\omega}{2} \rceil$ requests are given to a node, then it borrows colors from the private colors of one of its neighbors, taking the highest available color. R nodes can borrow colors from G nodes, G from B, and B from R. Since $\lceil \frac{\omega}{2} \rceil \leq \lceil \frac{\textsc{Opt}}{2} \rceil$, we can give $\lceil \frac{\omega}{2} \rceil$ as advice and obtain the following:

[1] The Greek letter ω is traditionally used here, so we will also do that. Since there is no argument, this should not give rise to confusion with the $\omega(f)$, stemming from asymptotic notation.

Theorem 9. *There is a $\frac{3}{2}$-competitive online algorithm for multi-coloring hexagonal graphs with advice complexity $enc(\lceil \frac{\text{OPT}}{2} \rceil)$.*

In [32], an algorithm with an approximation ratio of $\frac{4}{3}$ was introduced. This algorithm uses color classes in the same way as FPA, except that the private color sets contain only $\lfloor \frac{\omega+1}{3} \rfloor$ colors each. In describing the algorithm, we use the following notation. For any node v, we let n_v denote the number of requests to v. Furthermore, b_v denotes the maximum number of colors that v can borrow, i.e., $b_v = \max\{0, \lfloor \frac{\omega+1}{3} \rfloor - n'_v\}$, where n'_v is the maximum number of requests to any of the neighboring nodes in the color class that v can borrow from.

The algorithm can be seen as working in up to three phases: In the first phase, the algorithm colors $\min\{n_v, \lfloor \frac{\omega+1}{3} \rfloor\}$ requests to each node, v, using the node's private colors. In the second phase, each node v with more than $\lfloor \frac{\omega+1}{3} \rfloor$ requests borrows $\min\{n_v - \lfloor \frac{\omega+1}{3} \rfloor, b_v\}$ colors. Let G_2 be the graph induced by nodes that still have uncolored requests after Phase 2. In [32] it is proven that G_2 is bipartite and that any pair of neighbors in G_2 has a total of at most $\omega - 2\lfloor \frac{\omega+1}{3} \rfloor \leq \lfloor \frac{\omega+1}{3} \rfloor + 1$ uncolored requests after Phase 2. Thus, in the third phase, the remaining requests can be colored with GreedyOpt (see the path section) using $\lfloor \frac{\omega+1}{3} \rfloor + 1$ additional colors.

We now show how an online algorithm, given the right advice, can behave like the offline $\frac{4}{3}$-approximation algorithm. Note that the three phases of the offline $\frac{4}{3}$-approximation algorithm are characterized by the coloring strategy (using the node's own private colors, borrowing private colors from neighbors, or coloring a bipartite graph). However, when requests arrive online, the nodes may not go from one phase to the next simultaneously.

Theorem 10. *There is a $\frac{4}{3}$-competitive online algorithm for multi-coloring hexagonal graphs with advice complexity at most $n + 2|V|$.*

Proof. Initially, each node is in Phase 1. On a request, the algorithm reads an advice bit and if it is zero, the next color from its private colors is used. If, instead, a one is read, this is treated as a stop bit for Phase 1, and this particular node enters Phase 2.

The algorithm starts with empty private color sets, and adds one color to each set whenever necessary, i.e., whenever a Phase 1 node that has already used all its private colors receives an additional request (this includes the first request to the node). As soon as a node leaves Phase 1, the algorithm knows that this node received $\lfloor \frac{\omega+1}{3} \rfloor$ requests, which is then the final size of each private color set. Knowing the size of the private color sets, the algorithm can calculate the maximum color for the complete coloring of the graph as $m = 4\lfloor \frac{\omega+1}{3} \rfloor + 1$.

In Phase 2, every zero indicates that the algorithm should borrow a color. When another stop bit is received (which could be after no zeros at all if the borrowing phase is empty), it moves to Phase 3. In Phase 3, it reads one bit to decide which partition, upper or lower, of the bipartite graph it is in, and does not need more information after that, since it simply uses the colors $3\lfloor \frac{\omega+1}{3} \rfloor + 1, \ldots, m$, either top-down or bottom-up.

If we allow the algorithm one bit per request, it needs at most two more bits per node, since the stop bits are the only bits that do not immediately tell the algorithm which action to take. Thus, $n + 2|V|$ bits of advice suffice. □

Acknowledgments. We would like to thank anonymous referees for comments, improving the presentation of our results.

References

1. S. Albers, L.M. Favrholdt, and O. Giel. On paging with locality of reference. *J. Comput. System Sci.*, 70(2):145–175, 2005.
2. K. Barhum, H.-J. Böckenhauer, M. Forisek, H. Gebauer, J. Hromkovič, S. Krug, J. Smula, and B. Steffen. On the power of advice and randomization for the disjoint path allocation problem. In *SOFSEM*, volume 8327 of *Lecture Notes in Computer Science*, pages 89–101. Springer, 2014.
3. M. Paola Bianchi, H.-J. Böckenhauer, J. Hromkovic, and L. Keller. Online coloring of bipartite graphs with and without advice. In *COCOON*, volume 7434 of *LNCS*, pages 519–530, 2012.
4. H.-J. Böckenhauer, D. Komm, R. Královič, and R. Královič. On the advice complexity of the k-server problem. In *ICALP*, volume 6755 of *LNCS*, pages 207–218, 2011.
5. H.-J. Böckenhauer, D. Komm, R. Královič, R. Královič, and T. Mömke. On the advice complexity of online problems. In *ISAAC*, volume 5878 of *LNCS*, pages 331–340, 2009.
6. H.-J. Böckenhauer, D. Komm, R. Královič, and P. Rossmanith. On the advice complexity of the knapsack problem. In *LATIN*, pages 61–72, 2012.
7. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
8. A. Borodin, S. Irani, P. Raghavan, and B. Schieber. Competitive paging with locality of reference. *J. Comput. System Sci.*, 50(2):244–258, 1995.
9. J. Boyar, L.M. Favrholdt, K.S. Larsen, and M.N. Nielsen. Extending the Accommodating Function. *Acta Informatica*, 40(1):3–35, 2003.
10. J. Boyar, S. Gupta, and K.S. Larsen. Access graphs results for LRU versus FIFO under relative worst order analysis. In *SWAT*, volume 7357 of *LNCS*, pages 328–339. Springer, 2012.
11. J. Boyar, S. Kamali, K.S. Larsen, and A. López-Ortiz. Online bin packing with advice. In *Thirty-First International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 25 of *LIPIcs*, pages 174–186. Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, 2014.
12. J. Boyar and K.S. Larsen. The Seat Reservation Problem. *Algorithmica*, 25(4):403–417, 1999.
13. J. Boyar, K.S. Larsen, and M.N. Nielsen. The Accommodating Function: a generalization of the competitive ratio. *SIAM J. Comput.*, 31(1):233–258, 2001.
14. J.W.-T Chan, F.Y.L. Chin, D. Ye, and Y. Zhang. Absolute and asymptotic bounds for online frequency allocation in cellular networks. *Algorithmica*, 58(2):498–515, 2010.
15. J.W.-T. Chan, F.Y.L. Chin, D. Ye, Y. Zhang, and H. Zhu. Frequency allocation problems for linear cellular networks. In *ISAAC*, volume 4288 of *LNCS*, pages 61–70. Springer, 2006.

16. M.G. Christ, L.M. Favrholdt, and K.S. Larsen. Online multi-coloring on the path revisited. *Acta Informatica*, 50(5–6):343–357, 2013.
17. M.G. Christ, L.M. Favrholdt, and K.S. Larsen. Online multi-coloring with advice. arXiv:1409.1722 [cs.DS], 2014.
18. M. Chrobak, L. Jez, and J. Sgall. Better bounds for incremental frequency allocation in bipartite graphs. *Theoretical Computer Science*, 514:75–83, 2013.
19. M. Chrobak and J. Sgall. Three results on frequency assignment in linear cellular networks. *Theor. Comput. Sci.*, 411(1):131–137, 2010.
20. S. Dobrev, R. Královič, and D. Pardubská. Measuring the problem-relevant information in input. *RAIRO Theor. Inf. Appl.*, 43(3):585–613, 2009.
21. R. Dorrigiv, M. He, and N. Zeh. On the advice complexity of buffer management. In *ISAAC*, volume 7676 of *LNCS*, pages 136–145, 2012.
22. P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21(2):194–203, 1975.
23. Y. Emek, P. Fraigniaud, A. Korman, and A. Rosén. Online computation with advice. *Theoretical Computer Science*, 412(24):2642–2656, 2011.
24. M. Forisek, L. Keller, and M. Steinová. Advice complexity of online coloring for paths. In *LATA*, volume 7183 of *LNCS*, pages 228–239, 2012.
25. J. Hromkovič, R. Královič, and R. Královič. Information complexity of online problems. In *MFCS*, volume 6281 of *LNCS*, pages 24–36, 2010.
26. J. Janssen, K. Kilakos, and O. Marcotte. Fixed preference channel assignment for cellular telephone systems. *IEEE Trans. Veh. Technol.*, 48(2):533–541, 1999.
27. J. Janssen, D. Krizanc, L. Narayanan, and S.M. Shende. Distributed online frequency assignment in cellular networks. *J. Algorithms*, 36(2):119–151, 2000.
28. A.R. Karlin, M.S. Manasse, L. Rudolph, and D.D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:79–119, 1988.
29. D. Komm and R. Královič. Advice complexity and barely random algorithms. *RAIRO Theor. Inf. Appl*, 45(2):249–267, 2011.
30. D. Komm, R. Královič, and T. Mömke. On the advice complexity of the set cover problem. In *CSR*, volume 7353 of *LNCS*, pages 241–252, 2012.
31. M. P. Bianchi and H.-J. Böckenhauer and J. Hromkovič and S. Krug and B. Steffen. On the advice complexity of the online $l(2,1)$-coloring problem on paths and cycles. In *COCOON*, volume 7936 of *Lecture Notes in Computer Science*, pages 53–64. Springer, 2013.
32. C. McDiarmid and B.A. Reed. Channel assignment and weighted coloring. *Networks*, 36(2):114–117, 2000.
33. L. Narayanan. *Channel Assignment and Graph Multicoloring*, pages 71–94. John Wiley & Sons, Inc., 2002.
34. L. Narayanan and S.M. Shende. Static frequency assignment in cellular networks. *Algorithmica*, 29(3):396–409, 2001.
35. L. Narayanan and S.M. Shende. Corrigendum: Static frequency assignment in cellular networks. *Algorithmica*, 32(4):679, 2002.
36. S. Seibert, A. Sprock, and W. Unger. Advice complexity of the online coloring problem. In *CIAC*, volume 7878 of *LNCS*, pages 345–357, 2013.
37. D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.
38. P. Sparl and J. Zerovnik. 2-local 4/3-competitive algorithm for multicoloring hexagonal graphs. *Journal of Algorithms*, 55(1):29–41, 2005.
39. R. Witkowski and J. Zerovnik. 1-local 33/24-competitive algorithm for multicoloring hexagonal graphs. In *WAW*, volume 6732 of *LNCS*, pages 74–84, 2011.