Online conflict-free colorings for hypergraphs

Amotz Bar-Noy* Panagiotis Cheilaris† Svetalana Olonetsky ‡
Shakhar Smorodinsky§

Abstract

(i) We provide a framework for online conflict-free coloring (CF-coloring) any hypergraph. We use this framework to obtain an efficient randomized online algorithm for CF-coloring any k-degenerate hypergraph. Our algorithm uses $O(k \log n)$ colors with high probability and this bound is asymptotically optimal for any constant k. Moreover, our algorithm uses $O(k \log k \log n)$ random bits with high probability.

(ii) We show a deterministic online CF-coloring algorithm for points on the line with respect to intervals that uses $\Theta(\log n)$ colors by allowing at most one recoloring operation at any given insertion step. We also show a deterministic online CF-coloring algorithm for points in plane with respect to halfplanes that uses $\Theta(\log n)$ colors and recolors $O(n)$ points during the run.

As a corollary of (i), we obtain asymptotically optimal randomized algorithms for online CF-coloring some hypergraphs that arise in geometry and model an important version of the frequency assignment task for cellular networks. For example, hypergraphs that are induced by n points on a line with respect to intervals and also for n points in the plane with respect to unit discs. In both cases the number of colors that are used is $O(\log n)$ with high probability and this bound is asymptotically tight. Our algorithm uses exponentially fewer random bits compared to previous results for these special cases ($O(\log n)$ bits instead of $\Theta(n \log n)$ bits). This is the first general efficient online CF-coloring for arbitrary k-degenerate hypergraphs. Our framework when applied to k-degenerate graphs (also called k-inductive graphs) implies a randomized online algorithm for vertex coloring of such graphs with $O(k \log n)$ colors with high probability, a bound that was obtained deterministically by the first fit coloring algorithm.

*Computer and Information Science Department, Brooklyn College, 2900 Bedford Avenue Brooklyn, NY, 11210. E-mail: *amotz@sci.brooklyn.cuny.edu*.

†The Graduate Center, The City University of New York, 365 Fifth Avenue, New York, NY, 10016. E-mail: *philaris@sci.brooklyn.cuny.edu*.

‡Tel-Aviv University, E-mail: *olonetsk@post.tau.ac.il*.

§Courant Institute for Mathematical Sciences, New York University, 251 Mercer St., New York, NY, 10012. E-mail: *shakhar@cims.nyu.edu*. Work on this paper was supported by the NSF Mathematical Sciences Postdoctoral Fellowship award 0402492.

1 Introduction

A *hypergraph* is a pair $(V, \mathcal{E})$, where V is a finite set and $\mathcal{E} \subset 2^V$. The set V is called the *ground set* or the *vertex set* and the elements of $\mathcal{E}$ are called *hyperedges*. A *proper* k-coloring of a hypergraph $H = (V, \mathcal{E})$, for some positive integer k, is a function $\chi : V \to \{1, 2, \ldots, k\}$ such that no $S \in \mathcal{E}$ with $|S| \geq 2$ is monochromatic. Let $\chi(H)$ denote the minimum integer k for which H has a k-coloring. $\chi(H)$ is called the *chromatic number* of H. A *conflict-free* coloring (CF-coloring) of H is a coloring of V with the further restriction that for any hyperedge $S \in \mathcal{E}$ there exists a vertex $v \in S$ with a unique color (i.e., no other vertex of S has the same color as v). Both proper coloring and CF-coloring of hypergraphs are generalizations of vertex coloring of graphs (the definition coincides when the underlying hypergraph is a simple graph). Therefore the computational complexity of such colorings is at least as hard as for simple graphs.

The study of conflict-free colorings was originated in the work of Even et al. [5] and Smorodinsky [13] who were motivated by the problem of frequency assignment in cellular networks. Specifically, cellular networks are heterogeneous networks with two different types of nodes: *base stations* (that act as servers) and *clients*. Base stations are interconnected by an external fixed backbone network whereas clients are connected only to base stations. Connections between clients and base stations are implemented by radio links. Fixed frequencies are assigned to base stations to enable links to clients. Clients continuously scan frequencies in search of a base station with good reception. The fundamental problem of frequency assignment in such cellular networks is to assign frequencies to base stations so that every client, located within the receiving range of at least one station, can be served by some base station, in the sense that the client is located within the range of the station and no other station within its reception range has the same frequency (such a station would be in "conflict" with the given station due to mutual interference). The goal is to minimize the number of assigned frequencies ("colors") since the frequency spectrum is limited and costly.

Suppose we are given a set of n base stations, also referred to as *antennas*. Assume, for simplicity, that the area covered by a single antenna is given as a disk in the plane. Namely, the location of each antenna and its radius of transmission is fixed and is given (the transmission radii of the antennas are not necessarily equal). Even et al. [5] showed that one can find an assignment of frequencies to the antennas with a total of at most $O(\log n)$ frequencies such that each antenna is assigned one of the frequencies and the resulting assignment is free of conflicts, in the preceding sense. Furthermore, it was shown that this bound is worst-case optimal. Let $\mathcal{R}$ be a set of regions in the plane. For a point $p \in \cup_{r \in \mathcal{R}} r$, let $r(p) = \{r \in \mathcal{R} \mid p \in r\}$. Let $H(\mathcal{R})$ denote the hypergraph $(\mathcal{R}, \{r(p) \mid p \in \cup_{r \in \mathcal{R}}\})$. We say that $H(\mathcal{R})$ is the hypergraph *induced* by $\mathcal{R}$. Thus, Even et al. [5] showed that any hypergraph induced by a family $\mathcal{R}$ of n discs in the plane admits a CF-coloring with only $O(\log n)$ colors and that this bound is tight in the worst case. Furthermore, such a coloring can be found in deterministic polynomial time[1]. The results of [5] were further extended in [8] by combining more involved probabilistic and geometric ideas. The main result of [8] is a general randomized algorithm which CF-colors any set of n "simple" regions (not necessarily convex) whose union has "low" complexity, using a "small" number of colors.

In addition to the practical motivation, this new coloring model has drawn much attention of researchers through its own theoretical interest and such colorings have been the focus of several recent papers (see, e.g., [4, 5, 6, 8, 10, 12, 13, 14])

[1] In [5] it is shown that finding the minimum number of colors needed to CF-color a given collection of discs is NP-hard even when all discs are congruent, and an $O(\log n)$ approximation-ratio algorithm is provided.

To capture a dynamic scenario where antennas can be added to the network, Fiat et al. [6] initiated the study of online CF-coloring of hypergraphs. They considered a very simple hypergraph H which has its vertex set represented as a set P of n points on the line and its hyperedge set consists of all intersections of the points with some interval. The set $P \subset \mathbb{R}$ is revealed by an adversary online: Initially, P is empty, and the adversary inserts points into P, one point at a time. Let $P(t)$ denote the set P after the t-th point has been inserted. Each time a point is inserted, the algorithm needs to assign a color $c(p)$ to it, which is a positive integer. Once the color has been assigned to p, it cannot be changed in the future. The coloring should remain conflict-free at all times. That is, for any interval I that contains points of $P(t)$, there is a color that appears exactly once in I. Among other results, [6] provided a randomized algorithm for online CF-coloring n points on the line with $O(\log n \log\log n)$ colors with high probability.[2] They also provided a deterministic algorithm for online CF-coloring n points on the line with $\Theta(\log^2 n)$ colors in the worst case.

An online CF-coloring framework: In this paper, we study the most general form of online CF-coloring applied to arbitrary hypergraphs. Suppose the vertices of an underlying hypergraph $H = (V, \mathcal{E})$ are given online by an adversary. Namely, at every given time step t, a new vertex $v_t \in V$ is given and the algorithm must assign v_t a color such that the coloring is a valid conflict-free coloring of the hypergraph that is induced by the vertices $V_t = \{v_1, \ldots, v_t\}$ (see the exact definition in section 2). Once v_t is assigned a color, that color cannot be changed in the future. The goal is to find an algorithm that minimizes the maximum total number of colors used (where the maximum is taken over all permutations of the set V).

We present a general framework for online CF-coloring any hypergraph. Interestingly, this framework is a generalization of some known coloring algorithms. For example the Unique-Max Algorithm of [6] can be described as a special case of our framework. Also, when the underlying hypergraph is a simple graph then the First-Fit online algorithm is another special case of our framework.

Based on this framework, we introduce a randomized algorithm and show that the maximum number of colors used is a function of the 'degeneracy' of the hypergraph. We define the notion of a k-degenerate hypergraph as a generalization of the same notion for simple graphs. Specifically we show that if the hypergraph is k-degenerate, then our algorithm uses $O(k \log n)$ colors with high probability. This is asymptotically tight for any constant k.

As demonstrated in [6], the problem of online CF-coloring the very special hypergraph induced by points on the real line with respect to intervals is highly non-trivial. The best randomized online CF-coloring algorithm of [6] uses $O(\log n \log\log n)$ colors. Kaplan and Sharir [10] studied the special hypergraph induced by points in the plane with respect to halfplanes and unit discs and obtained a randomized online CF-coloring with $O(\log^3 n)$ colors with high probability. Recently, the bound $\Theta(\log n)$ just for these two special cases was obtained independently by Chen [3]. Our algorithm is more general and uses only $\Theta(\log n)$ colors; an interesting evidence to our algorithm being fundamentally different from the ones in [3, 6, 10], when used for the special case of hypergraphs that arise in geometry, is that it uses exponentially fewer random bits. The algorithms of [3, 10] use $\Theta(n \log n)$ random coin flips and our algorithm uses $O(\log n)$ random coin flips.

Another interesting corollary of our result is that we obtain a randomized online coloring for k-inductive graphs with $O(k \log n)$ colors with high probability. This case was studied by Irani

[2] Their algorithm assumes that the adversary is oblivious in the sense that it does not have access to the random coin flips of the algorithm.

[9] who showed that a greedy First-Fit algorithm achieves the same bound deterministically.

Deterministic online CF-coloring with recoloring: We initiate the study of online CF-coloring where at each step, in addition to the assignment of a color to the newly inserted point, we allow some recoloring of other points. The bi-criteria goal is to minimize the total number of recolorings done by the algorithm and the total number of colors used by the algorithm. We introduce an online algorithm for CF-coloring points on the line with respect to intervals, where we recolor at most one already assigned point at each step. Our algorithm uses $\Theta(\log n)$ colors. This is in contrast with the $O(\log^2 n)$ colors used by the best known deterministic algorithm by Fiat et al. [6] that does not recolor points. We also show online algorithm for CF-coloring points on the plane with respect to halfplanes that uses $\Theta(\log n)$ colors and the total number of recolorings is $O(n)$. For this problem no deterministic algorithm was known before.

Paper organization: Section 2 defines the notion of a k-degenerate hypergraph. Section 3 presents the general framework for online CF-coloring of hypergraphs. Section 4 introduces the randomized algorithm derived from the framework. Section 5 shows deterministic online algorithm for intervals and halfplanes with recoloring. Section 6 describes the results for the hypergraphs that arise from geometry. Finally, Section 7 concludes with a discussion and some open problems.

2 Preliminaries

We start with some basic definitions:

Definition 2.1. Let $H = (V, \mathcal{E})$ be a hypergraph. For a subset $V' \subset V$ let $H(V')$ be the hypergraph $(V', \mathcal{E}')$ where $\mathcal{E}' = \{e \cap V' | e \in \mathcal{E}\}$. $H(V')$ is called the *induced hypergraph* on V'.

Definition 2.2. For a hypergraph $H = (V, \mathcal{E})$, the *Delaunay graph* $G(H)$ is the simple graph $G = (V, E)$ where the edge set E is defined as $E = \{(x, y) \mid \{x, y\} \in \mathcal{E}\}$ (i.e., G is the graph on the vertex set V whose edges consist of all hyperedges in H of cardinality two).

Definition 2.3. A simple graph $G = (V, E)$ is called k-degenerate (or k-inductive) for some positive integer k, if every (vertex-induced) subgraph of G has a vertex of degree at most k.

Definition 2.4. Let $k > 0$ be a fixed integer and let $H = (V, \mathcal{E})$ be a hypergraph on n vertices. Fix a subset $V' \subset V$. For a permutation π of V' such that $V' = \{v_1, ..., v_i\}$ (where $i = |V'|$) let $C_\pi(V') = \sum_{j=1}^{i} d(v_j)$, where $d(v_j) = |\{l < j | (v_j, v_l) \in G(H(\{v_1, ..., v_j\}))\}|$, that is, $d(v_j)$ is the number of neighbors of v_j in the Delaunay graph of the hypergraph induced by $\{v_1, ..., v_j\}$. Assume that $\forall V' \subset V$ and for all permutations $\pi \in S_{|V'|}$ we have $C_\pi(V') \leq k\,|V'|$. Then we say that H is *k-degenerate*.

It is easy to see that our definition of a k-degenerate hypergraph is a generalization of that of a k-degenerate graph.

3 An online CF-coloring framework

Let $H = (V, E)$ be any hypergraph. Our goal is to define a framework that colors the vertices V in an online fashion. That is, the vertices of V are revealed by an adversary one at a time.

At each time step t, the algorithm must assign a color to the newly revealed vertex v_t. This color cannot be changed in the future. The coloring has to be conflict-free for all the induced hypergraphs $H(V_t)$ $t = 1, \ldots, n$, where $V_t \subset V$ is the set of vertices revealed by time t.

For a fixed positive integer h, let $A = \{a_1, \ldots, a_h\}$ be a set of h auxiliary colors (not to be confused with the set of 'real' colors used for the CF-coloring: $\{1, 2, \ldots\}$). Let $f : \mathbb{N} \to A$ be some fixed function. We now define the framework that depends on the choice of the function f and the parameter h.

A table (to be updated online) is maintained where each entry i at time t is associated with a subset $V_t^i \subset V_t$ in addition to an auxiliary proper coloring of $H(V_t^i)$ with at most h colors. We say that $f(i)$ is the color that represents entry i in the table. At the beginning all entries of the table are empty. Suppose all entries of the table are updated until time $t-1$ and let v_t be the vertex revealed by the adversary at time t. The framework first checks if an auxiliary color can be assigned to v_t such that the auxiliary coloring of V_{t-1}^1 together with the color of v_t is a proper coloring of $H(V_{t-1}^1 \cup \{v_t\})$. Any (proper) coloring procedure can be used by the framework. For example a first-fit greedy in which all colors in the order a_1, …, a_h are checked until one is found. If such a color cannot be found for v_t, then entry 1 is left with no changes and the process continues to the next entry. If however, such a color can be assigned, then v_t is added to the set V_{t-1}^1. Let c denote such an auxiliary color assigned to v_t. If this color is the same as $f(1)$ (the auxiliary color that is associated with entry 1), then the final color in the online CF-coloring of v_t is 1 and the updating process for the t-th vertex stops. Otherwise, if an auxiliary color cannot be found or if the assigned auxiliary color is not the same as the color associated with this entry, the updating process continues to the next entry. The updating process stops at the first entry i for which v_t is both added to V_t^i and the auxiliary color assigned to v_t is the same as $f(i)$. The color of v_t in the final conflict-free coloring is then set to i.

It is possible that v_t never gets a final color. In this case we say that the framework does not halt. However, termination can be guaranteed by imposing some restrictions on the auxiliary coloring method and the choice of the function f. For example, if first-fit is used for the auxiliary colorings at any entry and if f is the constant function $f(i) = a_1$, for all i, then the framework is guaranteed to halt for any time t. In section 4 we derive a randomized online algorithm based on this framework. This algorithm always halts[3] and moreover it halts after a "small" number of entries with high probability (w.h.p.).

Example 3.1. Consider the case where the hypergraph is induced by points with respect to intervals. Namely $V = (1, \ldots, n)$ and $\mathcal{E}$ consists of all possible discrete intervals of V (i.e., subsets of consecutive integers). Vertices appear one by one and at each time t we must have an online conflict-free coloring with respect to the discrete interval subsets of the t points revealed by time t. It is easy to see that the hypergraphs $H(V_t^i)$ can always be 3-colored (say with auxiliary colors a, b, c): Each newly inserted point has at most two immediate neighbors and thus even a first-fit coloring suffices. In Figure 1, we exhibit a run of the algorithm for the permutation $\pi = 253164$, seen as a mapping from time $t \in \{1, \ldots, 6\}$ to the corresponding vertex at position $\pi_{(t)}$. In the end the vertices look like $\pi^{-1} = v_4 v_1 v_3 v_6 v_2 v_5$, where v_t is the vertex appearing at time t. The choices are $f(1) = b$, $f(2) = a$, $f(3) = c$, $f(4) = a$, $f(5) = b$, $f(6) = a$. The six tables correspond to $t = 1, \ldots, 6$ and at the top of each table the online conflict-free coloring, so far, is shown. Entries correspond to rows in the tables, where for each entry i the following data is given: the representing color $f(i)$ and the proper auxiliary coloring

[3]It is easy to actually prove that in the randomized version the algorithm will halt with probability 1 but we omit discussing this issue here

of the vertices in the hypergraph V_t^i with three colors a, b or c.

i	$f(i)$	· **2** · · · ·
1	**b**	a
2	**a**	**a**
3		
4		
5		
6		

i	$f(i)$	· 2 · · **1** ·
1	b	a **b**
2	a	a
3		
4		
5		
6		

i	$f(i)$	· 2 **4** · 1 ·
1	b	a c b
2	a	a b
3	**c**	a
4	**a**	**a**
5		
6		

i	$f(i)$	**1** 2 4 · 1 ·
1	b	**b** a c b
2	a	a b
3	c	a
4	a	a
5		
6		

i	$f(i)$	1 2 4 · 1 **2**
1	b	b a c b a
2	a	a b **a**
3	c	a
4	a	a
5		
6		

i	$f(i)$	1 2 4 **6** 1 2
1	b	b a c a b a
2	a	a b c a
3	c	a b
4	a	a b
5	**b**	a
6	**a**	**a**

Figure 1: A run example of the framework for hypergraphs induced by points with respect to intervals. In this case the hypergraph admits a proper online 3-coloring in any entry of the table. The auxiliary colors are denoted by a, b, c.

Observe that entries 3 and 5, respectively, do not have a vertex colored with $f(3)$ and $f(5)$, respectively. As a consequence colors 3, 5 do not appear in the CF-coloring although colors 1, 2, 4, 6 do. If it is important to use consecutive colors, namely k different colors implies they are $\{1, \ldots, k\}$, the above problem can be fixed by assigning the next unused CF-color to an entry i only as soon as a vertex in entry i is colored with auxiliary color $f(i)$. The above remedy works in our general framework, not only in the specific case of this example.

We now turn to prove that the above framework produces a valid CF-coloring in case it halts.

Lemma 3.2. *If the above framework halts for any vertex v_t then it produces a valid online CF-coloring of H.*

Proof. Let $H(V_t)$ be the hypergraph induced by the vertices revealed at time t. Let S be a hyperedge in this hypergraph and let j be the maximum integer for which there is a vertex v of S colored with j. We claim that exactly one such vertex in S exists. Assume to the contrary that there is another vertex v' in S colored with j. This means that at time t both vertices v and v' were present at entry j of the table (i.e., $v, v' \in V_t^j$) and that they both got an auxiliary color (in the auxiliary coloring of the set V_t^j) which equals $f(j)$. However, since the auxiliary coloring is a proper coloring of the induced hypergraph at entry j, $S \cap V_t^j$ is not monochromatic so there must exists a third vertex $v'' \in S \cap V_t^j$ that was present at entry j and was assigned an auxiliary color different from $f(j)$. Thus v'' got its final color in an entry greater than j, a contradiction to the maximality of j in the hyperedge S. This completes the proof of the lemma. □

The above algorithmic framework can also describe some well-known deterministic algorithms. For example, if first-fit is used for auxiliary colorings and f is the constant function, $f(i) = a_1$, for all i, then:

- If the input hypergraph is induced by points on a line with respect to intervals as in example 1 then the algorithm derived from the framework becomes identical to the Unique Maximum Greedy algorithm described and analyzed in [6].
- If the input is a k-degenerate graph (also called k-inductive graph), the derived algorithm is identical to the First-Fit greedy algorithm for coloring graphs online. The performance of First-Fit for restricted classes of graphs has been analyzed in several papers [7, 9, 11]. Especially for k-inductive graphs, the First-Fit algorithm is analyzed in [9], where it is proved that it uses $O(k \log n)$ colors.

4 An online randomized CF-coloring algorithm

There is a randomized online CF-coloring in the oblivious adversary model that always produces a valid coloring and the number of colors used is related to the degeneracy of the underlying hypergraph in a manner described in theorem 4.1.

Theorem 4.1. *Let $H = (V, \mathcal{E})$ be a k-degenerate hypergraph on n vertices. Then there exists a randomized online CF-coloring for H which uses at most $O(\log_{1+\frac{1}{4k+1}} n) = O(k \log n)$ colors with high probability.*

The algorithm is based on the framework of section 3. In order to define the algorithm, we need to state what is the function f, the set of auxiliary colors of each entry and the algorithm we use for the auxiliary coloring at each entry. We use the set $A = \{a_1, \ldots, a_{2k+1}\}$. For each entry i, the representing color $f(i)$ is chosen uniformly at random from A. We use a first-fit algorithm for the auxiliary coloring.

Our assumption on the hypergraph H (being k-degenerate) implies that at least half of the vertices up to time t that 'reached' entry i (but not necessarily added to entry i), and we denote by $X^t_{\geq i}$, have been actually given some auxiliary color at entry i (that is, $\left|V^i_t\right| \geq \frac{1}{2}\left|X^t_{\geq i}\right|$). This is easily implied by the fact that at least half of those vertices v_t have at most $2k$ neighbors in the Delaunay graph of the hypergraph induced by $X^{t-1}_{\geq i}$ (since the sum of these quantities is at most $k\left|X^t_{\geq i}\right|$ and since $V^i_t \subset X^t_{\geq i}$). Therefore since we have $2k+1$ colors available, there is always a free color to assign to such a vertex. The following lemma shows that if we use one of these 'free' colors then the updated coloring is indeed a proper coloring of the corresponding induced hypergraph as well.

Lemma 4.2. *Let $H = (V, \mathcal{E})$ be a k-degenerate hypergraph and let V^j_t be the subset of V at time t and at level j as produced by the above algorithm. Then for any j and t if v_t is assigned a color distinct from all its neighbors in the Delaunay graph $G(H(V^j_t))$ then this color together with the colors assigned to the vertices V^j_{t-1} is also a proper coloring of the hypergraph $H(V^j_t)$.*

Proof. By induction on t. The induction hypothesis is that $H(V^j_{t-1})$ is properly colored by the auxiliary coloring. Let v_t be the vertex added to the hypergraph induced by the j'th entry at time t. Any hyperedge S that contains at least two vertices of V^j_{t-1} or does not contain v_t is not monochromatic by the induction hypothesis. Thus, we are only concerned with hyperedges of cardinality two that contain v_t and exactly one vertex of V^j_{t-1}. However, we assumed that v_t got a color that is distinct from any vertex u such that $\{u, v\}$ is a hyperedge of $H(V^j_t)$ (Those are exactly the neighbors of v_t in the corresponding Delaunay graph). Thus any such hyperedge $\{u, v\}$ is also not monochromatic. This completes the inductive step and hence the proof of the lemma. □

We proceed to the analysis of the performance of the algorithm.

Lemma 4.3. *Let $H = (V, \mathcal{E})$ be a hypergraph and let χ be a coloring produced by the above algorithm on an online input $V = \{v_t\}$ for $t = 1, \ldots, n$. Let X_i (respectively $X_{\geq i}$) denote the random variable counting the number of points of V that were assigned a final color at entry i (respectively a final color at some entry $\geq i$). Let $\mathbf{E}_i = \mathbf{E}[X_i]$ and $\mathbf{E}_{\geq i} = \mathbf{E}[X_{\geq i}]$ (note that $X_{\geq i+1} = X_{\geq i} - X_i$). Then:*

$$\mathbf{E}_{\geq i} \leq \left(\frac{4k+1}{4k+2}\right)^{i-1} n$$

Proof. By induction on i. The case $i = 1$ is trivial. Assume that the statements hold for i. To complete the induction step, we need to prove that $\mathbf{E}_{\geq i+1} \leq (\frac{4k+1}{4k+2})^i n$. By the conditional expectation formula, we have for any two random variables X, Y that $\mathbf{E}[X] = \mathbf{E}[\mathbf{E}[X \mid Y]]$. Thus

$$\mathbf{E}_{\geq i+1} = \mathbf{E}[\mathbf{E}[X_{\geq i+1} \mid X_{\geq i}]] = \mathbf{E}[\mathbf{E}[X_{\geq i} - X_i \mid X_{\geq i}]] = \mathbf{E}[X_{\geq i} - \mathbf{E}[X_i \mid X_{\geq i}]].$$

It is easily seen that $\mathbf{E}[X_i \mid X_{\geq i}] \geq \frac{1}{2}\frac{X_{\geq i}}{2k+1}$ since at least half of the vertices of $X_{\geq i}$ got an auxiliary color by the above algorithm. Moreover each of those elements that got an auxiliary color had probability $\frac{1}{2k+1}$ (This is the only place where we need to assume that the adversary is oblivious and does not have access to the random bits) to get the final color i. Thus

$$\mathbf{E}[X_{\geq i} - \mathbf{E}[X_i \mid X_{\geq i}]] \leq \mathbf{E}[X_{\geq i} - \frac{1}{2(2k+1)} X_{\geq i}] = \frac{4k+1}{4k+2}\mathbf{E}[X_{\geq i}] \leq \left(\frac{4k+1}{4k+2}\right)^i n$$

by linearity of expectation and by the induction hypotheses. This completes the proof of the lemma. □

Lemma 4.4. *The expected number of colors used by the above algorithm is at most $\log_{\frac{4k+2}{4k+1}} n+1$.*

Proof. Let I_i be the indicator random variable for the following event: some points are colored with a real color in entry i. We are interested in the number of colors used, that is $Y := \sum_{i=1}^{\infty} I_i$. Let $b(k, n) = \log_{(4k+2)/(4k+1)} n$. Then,

$$\mathbf{E}[Y] = \mathbf{E}[\sum_{1 \leq i} I_i] = \mathbf{E}[\sum_{1 \leq i \leq b(k,n)} I_i] + \mathbf{E}[X_{\geq b(k,n)+1}] \leq b(k, n) + 1$$

by Markov's inequality and lemma 4.3. □

We also have the following concentration result:

$$\Pr[\text{more than } cb(k,n) \text{ colors are used}] = \Pr[X_{\geq cb(k,n)+1} \geq 1] \leq \mathbf{E}_{\geq cb(k,n)+1} \leq \frac{1}{n^{c-1}}$$

by Markov's inequality and by lemma 4.3.

This completes the performance analysis of our algorithm.

Remark: In the above description of the algorithm, all the random bits are chosen in advance (by deciding the values of the function f in advance). However, one can be more efficient and calculate the entry $f(i)$ only at the first time we need to update entry i, for any i. Since at each entry we need to use $O(\log k)$ random bits and we showed that the number of entries used is $O(k \log n)$ w.h.p then the total number of random bits used by our algorithm is $O(k \log k \log n)$ w.h.p.

5 Deterministic online CF-coloring algorithm with recoloring

5.1 An $O(\log n)$ algorithm with recoloring for intervals

We describe a deterministic online CF-coloring algorithm for intervals that is only allowed to recolor a single old point during each request of a new point. The algorithm uses two auxiliary colors: a and d (points colored with d correspond to ones colored with b, c above; we use only d in the presentation for simplicity). All points that get to the entry ℓ and are assigned a get real color at entry ℓ (similar to $f(\ell) = a$). First point that gets to the entry ℓ is assigned a. Points that are assigned d get their real color at a higher entry. In order to have logarithmic number of entries (and thus total colors) the size of the independent set colored with a has to be large. To achieve this goal, our algorithm maintains the following invariant in every level: For every point colored with d, there exists an adjacent point colored with a. Therefore, at least a third of the points that get to each entry get color a, and two thirds are deferred for coloring in a higher entry. Again, the colors used in total are at most $\log_{3/2} n + 1$. When a new point p arrives, it is colored according to the following algorithm, starting from entry 1:

- We color point p with a if this does not generate two consecutive a's. In this case, the point gets a real color at this entry.
- We color point p with d if it does not generate a point colored with d that will not have an adjacent point colored with a. Note that here p is deferred for real coloring at a higher entry and thus the same algorithm is recursively applied at the next entry.
- It remains to handle the case where the new point p has a point colored with a on one side and a point, say q, colored with d on the other side, such that q has no adjacent point colored with a. We color p with d in the current entry and the real color of q, and we recolor q with a, and thus we recolor q with the real color of the current entry. At this point all points have an assignment of real colors. It is not hard to check that when we recolor a point then we do not violate the invariants at any entry: Let ℓ be the entry that caused recoloring, all entries before it remain the same, the change in the entry ℓ does not break invariants, all other entries remain the same except that point p appears there instead of point q that was there before and there are no points between p and q that appear in an entry higher than ℓ.

It can be proved that the number of recolorings is at most $n - (\lfloor \lg n \rfloor + 1)$, and this is tight. An example of a run of the algorithm is shown in Figure 2.

5.2 An $O(\log n)$ recoloring algorithm for circular arcs

We define a hypergraph H closely related to the one induced by intervals: The vertex set of H is represented as a set P of n distinct points on a *circle* C and its hyperedge set consists of all intersections of the points with some *circular arc* of C. In the static case, it can be proved that n points can be optimally conflict-free colored with respect to circular arcs with $\lfloor \lg(n-1) \rfloor + 2$ colors (citation). Here, we are interested in an online setting, where the set $P \subset C$ is revealed incrementally by an adversary, and, as usual, the algorithm has to commit on a color for each point without knowing how future points will be requested. Algorithms for intervals can be used almost verbatim for circular arcs. In fact, the recoloring algorithm for intervals, given in section 5.1, can be used verbatim, if the notion of adjacency of points is adapted to the closed curve setting (for $n \geq 3$, each point has exactly 2 immediate neighboring points, whereas in

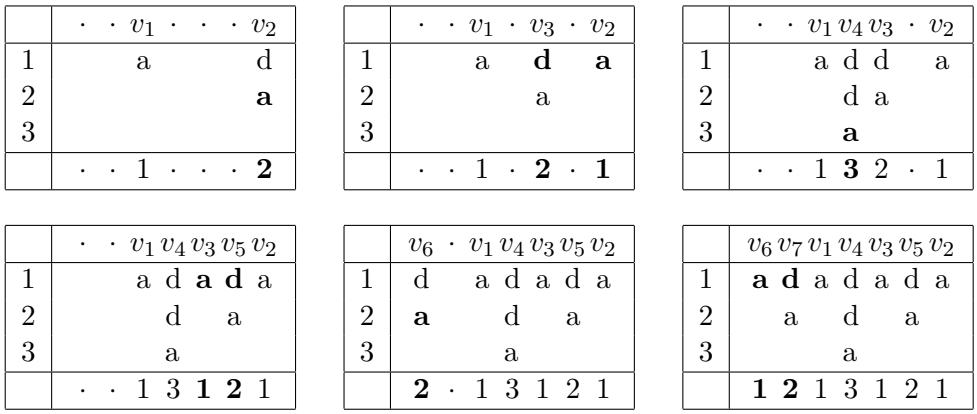

Figure 2: An example run of the recoloring algorithm. The first row of the table represents the order in which points appeared, the last row of the table shows current color allocation. The first table is at $t = 2$, after the first two points have appeared. At every time step of the run, points whose colors were changed (a new color, or a recoloring) by the last insertion are marked with bold.

the intervals case, the two extreme points have only one neighbor). Again, in each entry ℓ, at least a third of the points is assigned auxiliary color a, and thus at most $\log_{3/2} n + 1$ colors are used.

5.3 An $O(\log n)$ recoloring algorithm for circular arcs with substitution of points

We consider a variation on the problem of online conflict-free coloring with respect to circular arcs that was given in section 5.2. In this new variation, the adversary has, in addition to the 'insertion move' of a new point, a 'substitution move':

> The adversary can substitute a set Q of already requested *consecutive* points with a single new point p, and the algorithm has to color p, such that the whole set of points is conflict-free colored with respect to circular arcs (in that set, p is included, but all points in Q are removed).

Our algorithm for this variation of the problem relies on the one given in section 5.2. For an insertion move of the adversary, it colors the new point like in section 5.2. For a substitution move of the adversary, it colors the new point p, with the *highest* color occurring in the points of Q. Point p also gets the entries of the unique point $q \in Q$ with the highest color. It is not difficult to see that the coloring remains conflict-free after a substitution move and a coloring of the new point p by the algorithm. We remark that a recoloring can happen only in an insertion move and that substitution moves do not make the algorithm introduce new colors. The following is true for every t:

Lemma 5.1. *After t moves, the coloring given by the algorithm uses at most $\log_{3/2} t + 1$ colors.*

5.4 An $O(\log n)$ algorithm with recoloring with respect to halfplanes

In this section we describe a deterministic algorithm for online CF-coloring points with respect to halfplanes that uses $O(\log n)$ colors and recolors $O(n)$ points. At every time instance t, the algorithm maintains the following invariant (V_t is the set of points that have appeared): All

points (strictly) inside the convex hull of V_t are colored with the same special color, say '$\star$'. The set of points on the convex hull of V_t, denoted by $\mathrm{CH}(V_t)$, are colored with another set of colors, such that every set of consecutive points on the convex hull has a point with a unique color. If one considers the points of $\mathrm{CH}(V_t)$ in the circular order they appear on the convex hull, it is enough to color them (with that circular order) conflict-free with respect to circular arcs. The number of colors used in $\mathrm{CH}(V_t)$ must be logarithmic on t. It is not difficult to see that every subset of points of V_t induced by a halfplane contains a set of consecutive points on the convex hull of V_t, and thus the whole coloring is conflict-free.

We describe how the algorithm maintains the above invariant. A new point v_{t+1} that appears at time $t+1$ is colored as follows: If it is inside the convex hull of V_t, then it gets color '$\star$'. Otherwise, the new point v_{t+1} will be on $\mathrm{CH}(V_{t+1})$, in which case we essentially use the algorithm of section 5.3 to color it. We have two case, which correspond to a substitution and an insertion move, respectively:

- It might be the case that v_{t+1} forces some points (say they comprise set Q) that where in $\mathrm{CH}(V_t)$ to get inside the convex hull of V_{t+1}, so in order to maintain the invariant, all points in Q are recolored to '$\star$', and v_{t+1} is colored with the maximum color occurring in Q (this is like a substitution move of section 5.3).

- If, on the other hand, no points of $\mathrm{CH}(V_t)$ are forced into the convex hull, then point $v_{t+1} \in \mathrm{CH}(V_{t+1})$ is colored like in an insertion move of section 5.3, with the algorithm for circular arcs. In that last case, in order to maintain logarithmic number of colors on t, one recoloring of a point in $\mathrm{CH}(V_{t+1})$ might be needed.

The total number of recolorings is guaranteed to be $O(n)$, because for every insertion, at most one recoloring happens on the new convex hull, and every point colored with '$\star$' stays with that color, because the convex hull never shrinks.

6 Application to Geometry

Our randomized algorithm has applications to conflict-free colorings of certain geometric hypergraphs studied in [3, 6, 10]. We obtain the same asymptotic result as in [3] but with better constant of proportionalities and much fewer random bits.

For example, if the hypergraph H is the same as in example 3.1 it can be proved (with an analysis similar to the one given in section 4) that for any order of insertion of n points, when the auxiliary color for each entry is chosen uniformly at random from $\{a, b, c\}$, the expected number of colors used is bounded by $\log_{\frac{3}{2}} n + 1$. It is interesting that the best known upper bound for dynamically coloring n points deterministically, when the whole insertion order is known in advance, is also $\log_{\frac{3}{2}} n + 1$ (see [2] for further details). In [3] the expected number of colors is bounded by $1 + \log_{8/7} n \approx 5.19 \log_2 n$, whereas in our algorithm it is bounded by $1 + \log_{3/2} n \approx 1.71 \log_2 n$.

When H is the hypergraph obtained by points in the plane intersected by halfplanes or unit disks, we obtain online randomized algorithms that use $O(\log n)$ colors w.h.p. We summarize it as follows:

Lemma 6.1. *Let V be a finite set of n points in the plane and let $\mathcal{E}$ be all subsets of V that can be obtained by intersecting V with a halfplane. Then the hypergraph $H = (V, \mathcal{E})$ is 3-degenerate.*

Proof. The proof uses a few geometric lemmas. Details are omitted. □

Corollary 6.2. *Let H be the hypergraph as in lemma 6.1. Then the expected number of colors used by our randomized online CF-coloring applied to H is at most $\log_{\frac{14}{13}} n + 1$. Also the actual number of colors used is $O(\log_{\frac{14}{13}} n)$ with high probability. The number of random bits is $O(\log n)$ with high probability*

Proof. The proof follows immediately from lemma 6.1, lemma 4.4 and theorem 4.1. □

Corollary 6.3. *Let V be a finite set of n points in the plane and let $\mathcal{E}$ be all subsets of V that can be obtained by intersecting V with a unit disc. Then there exists a randomized online algorithm for CF-coloring H which uses $O(\log n)$ colors and $O(\log n)$ random bits with high probability.*

Proof outline. Kaplan and Sharir [10] observed that by an appropriate partitioning of the plane one can modify any online algorithm for CF-coloring points with respect to halfplanes to a CF-coloring of points with respect to congruent discs. Using the same technique as developed in [10] and Corollary 6.2 we obtain the desired result. □

7 Discussion and Open Problems

We presented a framework for online CF-coloring any hypergraph. This framework coincides with some known algorithms in the literature when restricted to the corresponding special hypergraphs. We derived a randomized online algorithm for CF-coloring any hypergraph (in the oblivious adversary model) and showed that the performance of our algorithm depends on a parameter which we refer to as the degeneracy of the hypergraph which is a generalization of the known notion of degeneracy in graphs (i.e., when the hypergraph is a simple graph then this term is the same as the classical definition of degeneracy of a graph; see Definition 2.3). Specifically, if the hypergraph is k-degenerate then our algorithm uses $O(k \log n)$ colors w.h.p, which is asymptotically optimal for any constant k, and $O(k \log k \log n)$ random bits. This is the first efficient online CF-coloring for general hypergraphs and subsumes all the previous randomized algorithmic results of [3, 6, 10]. It also substantially improves the efficiency with respect to the amount of randomness used in the special cases studied in [3, 6, 10].

Another interesting fact to note is that our algorithm when applied to k-degenerate graphs gives an online coloring of such graphs with $O(k \log n)$ colors w.h.p. Irani [9] showed that the same bound is achieved deterministically by the First-Fit algorithm.

It would be interesting to find an efficient online deterministic algorithm for conflict-free coloring k-degenerate hypergraphs. Even for the very special case of a hypergraph induced by points and intervals (as in example 3.1 where the number of neighbors of the Delaunay graph of every induced hypergraph is at most two), the best known deterministic online CF-coloring algorithm from [6] uses $\Theta(\log^2 n)$ colors. We hope that our technique together with a possible clever de-randomization technique can shed light on this problem.

As mentioned already, the framework of section 3 can describe some known algorithms such as the Unique Max Greedy of [6] for online CF-coloring points on a line with respect to intervals. No sharp asymptotic bounds are know for the performance of Unique Max Greedy. The best known upper bound is linear, whereas the best known lower bound is $\Omega(\sqrt{n})$. We believe that this new approach could help analyze the performance of Unique Max Greedy.

In Section 5 we initiate the study of online CF-coloring with recoloring: We provide a deterministic online CF-coloring for points on the real line with respect to intervals and show that our algorithm uses $\Theta(\log n)$ colors and at most one recoloring per insertion. This is in

contrast with the best known deterministic online CF-coloring for this case that uses $\Theta(\log^2 n)$ colors in the worst case and does not recolor any other point ([6]). It would be interesting to find a deterministic online CF-coloring algorithm for points in the plane with respect to halfplanes that uses $\Theta(\log n)$ colors in the worst case and recolor at most a constant number of points per insertion. We leave this as an open problem for further research.

Acknowledgements

We would like to thank Amos Fiat, Haim Kaplan, János Pach, David Peleg and Micha Sharir for helpful discussions concerning the problems studied in this paper.

References

[1] N. Alon and S. Smorodinsky. Conflict-free colorings of shallow discs. In Proc. *22nd Annual ACM Symposium on Computational Geometry* (SoCG 2006):41–43.

[2] A. Bar-Noy, P. Cheilaris, and S. Smorodinsky. Conflict-free coloring for intervals: from offline to online. In Proc. *18th ACM Symposium on Parallelism in Algorithms and Architectures* (SPAA 2006): 128–137.

[3] K. Chen. On how to play a coloring game against color-blind adversaries. In Proc. *22nd Annual ACM Symposium on Computational Geometry* (SoCG 2006): 44–51.

[4] K. Elbassioni and N. Mustafa. Conflict-Free Colorings of Rectangles Ranges. In Proc. *23rd International Symposium on Theoretical Aspects of Computer Science* (STACS 2006): 254–263.

[5] G. Even, Z. Lotker, D. Ron, and S. Smorodinsky. Conflict-free colorings of simple geometric regions with applications to frequency assignment in cellular networks. *SIAM Journal on Computing*, 33(1):94–136 (2003). Also in *Proceedings of the 43th Annual IEEE Symposium on Foundations of Computer Science* (FOCS 2002).

[6] K. Chen, A. Fiat, H. Kaplan, M. Levy, J. Matoušek, E. Mossel, J. Pach, M. Sharir, S. Smorodinsky, U. Wagner, and E. Welzl. Online conflict-free coloring for intervals. SIAM Journal on Computing, in press (2006).

[7] A. Gyarfas and J. Lehel. On-line and first fit coloring of graphs. *Journal of Graph Theory*, 12(2):217–227 (1988).

[8] S. Har-Peled and S. Smorodinsky. On conflict-free coloring of points and simple regions in the plane. *Discrete and Computational Geometry*, 34:47–70 (2005).

[9] S. Irani. Coloring inductive graphs on-line. *Algorithmica*, 11(1):53–72 (1994).

[10] H. Kaplan and M. Sharir. Online CF coloring for halfplanes, congruent disks, and axis-parallel rectangles. Manuscript, 2004.

[11] H. A. Kierstead. The linearity of first-fit coloring of interval graphs. *SIAM Journal on Discrete Mathematics*, 1(4):526–530 (1988).

[12] J. Pach and G. Tóth. Conflict free colorings. In *Discrete and Computational Geometry, The Goodman-Pollack Festschrift*, Springer Verlag, Heidelberg, 2003.

[13] S. Smorodinsky. Combinatorial Problems in Computational Geometry. Ph.D Dissertation, School of Computer Science, Tel-Aviv University, 2003.

[14] S. Smorodinsky. On the chromatic number of some geometric hypergraphs. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA 2006): 316–323.