

# Online Conflict-free Colorings for Hypergraphs<sup>\*</sup>

Amotz Bar-Noy<sup>1</sup>, Panagiotis Cheilaris<sup>1,2,3,\*\*</sup>, Svetlana Olonetsky<sup>4</sup>, and  
Shakhar Smorodinsky<sup>5,\*\*\*</sup>

<sup>1</sup> Brooklyn College

<sup>2</sup> The Graduate Center

City University of New York

<sup>3</sup> School of ECE

National Technical University of Athens

<sup>4</sup> Tel-Aviv University

<sup>5</sup> Courant Institute

New York University

**Abstract.** We provide a framework for online conflict-free coloring (CF-coloring) of 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. As a corollary, we obtain asymptotically optimal randomized algorithms for online CF-coloring some hypergraphs that arise in geometry. Our algorithm uses exponentially fewer random bits compared to previous results. We introduce deterministic online CF-coloring algorithms for points on the line with respect to intervals and for points on the plane with respect to halfplanes (or unit discs) that use  $\Theta(\log n)$  colors and recolor  $O(n)$  points in total.

## 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  $\chi : V \rightarrow \{1, 2, \dots, k\}$  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

---

<sup>\*</sup> The first two authors are partially supported by the CUNY Collaborative Incentive Research Grants Program Round 11 (2004–2006).

<sup>\*\*</sup> Supported by the European Social Fund (75%) and National Resources (25%) under the program EPEAEK II, ‘Heraclitus’.

<sup>\*\*\*</sup> Work on this paper was supported by the NSF Mathematical Sciences Postdoctoral Fellowship award 0402492.

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. [6] and Smorodinsky [12] 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.

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  $(\mathcal{R}, \{r(p) \mid p \in \cup_{r \in \mathcal{R}} r\})$ . We say that  $H(\mathcal{R})$  is the hypergraph *induced* by  $\mathcal{R}$ . Even et al. [6] 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. 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., [1, 5–8, 10–13]). To capture a dynamic scenario where antennas can be added to the network, Chen et al. [7] 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, [7] 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 investigate 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 adver-

sary. 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  $V_t = \{v_1, \dots, v_t\}$  (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 [7] 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 [7], 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. Chen, 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 (randomized) bound  $\Theta(\log n)$  just for these two special cases was obtained independently by Chen [4]. 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 [4, 7, 10], when used for the special case of hypergraphs that arise in geometry, is that it uses exponentially fewer random bits. The algorithms of [4, 7, 10] use  $\Theta(n \log n)$  random bits, whereas our algorithm uses  $O(\log n)$  random bits.

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 [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 Chen et al. [7] 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 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  $\mathcal{E}' = \{e \cap V' \mid e \in \mathcal{E} \text{ and } e \cap V' \neq \emptyset\}$ .  $H(V')$  is called the induced hypergraph on  $V'$ .

**Definition 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 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 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  $\pi$  of  $V'$  such that  $V' = \{v_1, \dots, v_i\}$  (where  $i = |V'|$ ) let  $C_\pi(V') = \sum_{j=1}^i d(v_j)$ , where  $d(v_j) = |\{l < j \mid (v_j, v_l) \in G(H(\{v_1, \dots, v_j\}))\}|$ , that is,  $d(v_j)$  is the number of neighbors of  $v_j$  in the Delaunay graph of the hypergraph induced by  $\{v_1, \dots, v_j\}$ . 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 not difficult 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  $H = (V, \mathcal{E})$  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  $H(V_t)$   $t = 1, \dots, n$ , where  $V_t \subset V$  is the set of vertices revealed by time  $t$ .

For a fixed positive integer  $h$ , let  $A = \{a_1, \dots, a_h\}$  be a set of  $h$  auxiliary colors (not to be confused with the set of ‘real’ colors used for the CF-coloring:  $\{1, 2, \dots\}$ ). Let  $f : \mathbb{N} \rightarrow A$  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  $a_1, \dots, a_h$  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 halts with probability 1 and moreover it halts after a ‘‘small’’ number of entries with high probability. The above framework produces a valid CF-coloring in case it halts (proof omitted):

**Lemma 1.** *If the above framework halts for any vertex  $v_t$  then it produces a valid online CF-coloring of  $H$ .*

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: (a) If the input hypergraph is induced by points on a line with respect to intervals then the algorithm derived from the framework becomes identical to the Unique Maximum Greedy algorithm described and analyzed in [7]. (b) 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.

## 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 1.

**Theorem 1.** *Let  $H = (V, \mathcal{E})$  be a  $k$ -degenerate hypergraph on  $n$  vertices. Then there exists a randomized online CF-coloring algorithm 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  $A = \{a_1, \dots, a_{2k+1}\}$ . 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  $i$ ), denoted by  $X_{\geq i}^t$ , have been actually given some auxiliary color at entry  $i$  (that is,  $|V_t^i| \geq \frac{1}{2} |X_{\geq i}^t|$ ). 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  $X_{\geq i}^{t-1}$  (since the sum of these quantities is at most  $k |X_{\geq i}^t|$  and since  $V_t^i \subset X_{\geq i}^t$ ). 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 (proof omitted).

**Lemma 2.** *Let  $H = (V, \mathcal{E})$  be a  $k$ -degenerate hypergraph and let  $V_t^j$  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  $G(H(V_t^j))$  then this color together with the colors assigned to the vertices  $V_{t-1}^j$  is also a proper coloring of the hypergraph  $H(V_t^j)$ .*

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

**Lemma 3.** *Let  $H = (V, \mathcal{E})$  be a hypergraph and let  $\chi$  be a coloring produced by the above algorithm on an online input  $V = \{v_t\}$  for  $t = 1, \dots, n$ . 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  $\mathbf{E}[X] = \mathbf{E}[\mathbf{E}[X | Y]]$ . Thus

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

It is easily seen that  $\mathbf{E}[X_i | X_{\geq i}] \geq \frac{1}{2} \frac{X_{\geq i}}{2k+1}$  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

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

by linearity of expectation and by the induction hypotheses.  $\square$

**Lemma 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,

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

and we also have the following concentration result:

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

both by Markov's inequality and lemma 3.  $\square$

*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 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 insertion of a new point. The algorithm is based on the framework developed in Section 3 where we use 3 auxiliary colors  $\{a, b, c\}$  and  $f$  is the constant function  $f(l) = a \forall l$ . We refer to

points colored with  $b$  or  $c$  as  $d$ -points. In order to have only logarithmic number of entries, we slightly modify the framework (using a recoloring procedure) such that the number of points colored with  $a$  in each entry of the table is at least one third of the total points that reach that entry. To achieve this goal, our algorithm maintains the following invariant in every level: There are at most two  $d$ -points between every pair of points colored with  $a$  (i.e., between every pair that are consecutively colored  $a$  among the  $a$ -points). Therefore, at least a third of the points at each entry get color  $a$ , and two thirds are deferred for coloring in a higher entry. The total number of colors is 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:

- If  $p$  is not adjacent to a point colored with an auxiliary color  $a$  then  $p$  is assigned auxiliary color  $a$  and gets its final color in that entry.
- We color point  $p$  with  $b$  or  $c$  greedily as long as it does not break the invariant that between any two consecutive  $a$ 's we have at most two  $d$ -points.
- 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 assign to  $p$  the auxiliary color of  $q$  (thus it is a  $d$ -point) in the current entry and in all higher entries for which  $q$  got an auxiliary color and assign to it the real color of  $q$ , and we recolor  $q$  with the auxiliary color  $a$  (and delete the corresponding appearance of it in all higher entries of the table), 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  $\ell$  be the entry that caused recoloring, all entries before it remain the same, the change in the entry  $\ell$  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  $\ell$ .

It can be easily checked that this algorithm produces a valid CF-coloring. Also, it can be proved that the number of recolorings is at most  $n - (\lceil \lg n \rceil + 1)$ , and this is tight.

## 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 shown that  $n$  points can be optimally conflict-free colored with respect to circular arcs with  $\lceil \lg(n - 1) \rceil + 2$  colors. 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 the intervals case, the two extreme points have only one neighbor). Again, in each entry  $\ell$ , 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 $O(\log n)$ algorithm for circular arcs with substitution

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 each move. 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  $t$  (proof omitted):

**Lemma 5.** *After  $t$  moves, the above coloring algorithm uses at most  $\log_{3/2} t + 1$  colors.*

### 5.4 An $O(\log n)$ algorithm with recoloring for 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. Thus it can also be modified for CF-coloring points in the plane with respect to unit discs as remarked in Section 6. 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  $\text{CH}(V_t)$ , 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  $\text{CH}(V_t)$  in the circular order they appear on the convex hull, it is enough to CF-color them (with that circular order) with respect to circular arcs. The number of colors used in  $\text{CH}(V_t)$  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  $\text{CH}(V_{t+1})$ , in which case we essentially use the algorithm of section 5.3 to color it. We have two cases, 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 were in  $\text{CH}(V_t)$  to belong to the interior of  $\text{CH}(V_{t+1})$ , 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  $\text{CH}(V_t)$  are forced into the convex hull, then point  $v_{t+1} \in \text{CH}(V_{t+1})$  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  $\text{CH}(V_{t+1})$  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 ‘ $\star$ ’ remains with that color.

## 6 Application to Geometry

Our randomized algorithm has applications to conflict-free colorings of certain geometric hypergraphs studied in [4, 7, 10]. We obtain the same asymptotic result as in [4] but with better constant of proportionalities and much fewer random bits. For example, if the hypergraph  $H$  is induced by intervals, 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 [3] for further details). In our algorithm the expected number of colors is bounded by  $1 + \log_{3/2} n \approx 1.71 \log_2 n$ , whereas in [4] by  $1 + \log_{8/7} n \approx 5.19 \log_2 n$ , three times our bound. 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.** *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 4-degenerate.*

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

**Corollary 1.** *Let  $H$  be the hypergraph as in lemma 6. Then the expected number of colors used by our randomized online CF-coloring applied to  $H$  is at most*

$\log_{\frac{18}{17}} n + 1$ . Also the actual number of colors used is  $O(\log_{\frac{18}{17}} n)$  with high probability. The number of random bits is  $O(\log n)$  with high probability

*Proof.* The proof follows immediately from lemmata 6, 4 and theorem 1.  $\square$

**Corollary 2.** *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).* Chen, 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 1 we obtain the desired result.  $\square$

## 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 some special underlying 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 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 [4, 7, 10]. It also substantially improves the efficiency with respect to the amount of randomness used in the special cases studied in [4, 7, 10].

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 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 [7] 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 [7] for online CF-coloring points on a line with respect to intervals. No sharp asymptotic bounds are known 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 ([7]). We also present deterministic online algorithms for CF-coloring points with respect to circular arcs and halfplanes (and unit discs) that use  $O(\log n)$  colors and  $O(n)$  recolorings in the worst case. It would be interesting to obtain an online CF-coloring algorithm with  $O(\log n)$  and  $O(1)$  recolorings per insertion for the case of halfplanes.

*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. D. Ajwani, K. Elbassioni, S. Govindarajan, and S. Ray. Conflict-free coloring for rectangle ranges using  $O(n^{0.382+\varepsilon})$  colors. To appear in *19th ACM Symposium on Parallelism in Algorithms and Architectures* (SPAA 2007).
2. N. Alon and S. Smorodinsky. Conflict-free colorings of shallow discs. In Proc. *22nd Annual ACM Symposium on Computational Geometry* (SoCG 2006):41–43.
3. 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.
4. 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.
5. 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.
6. 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).
7. 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*, 36: 1342–1359 (2006).
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. K. Chen, H. Kaplan and M. Sharir. Online CF coloring for halfplanes, congruent disks, and axis-parallel rectangles. Manuscript, 2005.
11. J. Pach and G. Tóth. Conflict free colorings. In *Discrete and Computational Geometry, The Goodman-Pollack Festschrift*, Springer Verlag, Heidelberg, 2003.
12. S. Smorodinsky. Combinatorial Problems in Computational Geometry. Ph.D Dissertation, School of Computer Science, Tel-Aviv University, 2003.
13. 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.