CONFLICT-FREE COLORING

by

Panagiotis Cheilaris

A dissertation submitted to the Graduate Faculty in Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy, The City University of New York

2009

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirements for the degree of Doctor of Philosophy.

Stathis Zachos

Date — Chair of Examining Committee

Ted Brown

Date — Executive Officer

Amotz Bar-Noy

János Pach

Boris Aronov

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

ABSTRACT

CONFLICT-FREE COLORING

BY

PANAGIOTIS CHEILARIS

Advisers: Amotz Bar-Noy, János Pach, Stathis Zachos

Graph and hypergraph colorings constitute an important subject in combinatorics and algorithm theory. In this work, we study conflict-free coloring for hypergraphs. Conflict-free coloring is one possible generalization of traditional graph coloring. Conflict-free coloring hypergraphs induced by geometric shapes, like intervals on the line, or disks on the plane, has applications in frequency assignment in cellular networks. Colors model frequencies and since the frequency spectrum is limited and expensive, the goal of an algorithm is to minimize the number of assigned frequencies, that is, reuse frequencies as much as possible.

We concentrate on an online variation of the problem, especially in the case where the hypergraph is induced by intervals. For deterministic algorithms, we introduce a hierarchy of models ranging from static to online and we compute lower and upper bounds on the numbers of colors used.

In the randomized oblivious adversary model, we introduce a framework for conflict-free coloring a specific class of hypergraphs with a logarithmic

number of colors. This specific class includes many hypergraphs arising in geometry and gives online randomized algorithm that use fewer colors and fewer random bits than other algorithms in the literature. Based on the same framework, we initiate the study of online deterministic algorithms that recolor few points.

For the problem of conflict-free coloring points with respect to a given set of intervals, we describe an efficient algorithm that computes a coloring with at most twice the number of colors of an optimal coloring. We also show that there is a family of inputs that force our algorithm to use two times the number of colors of an optimal solution.

Then, we study conflict-free coloring problems in graphs. We compare conflict-free coloring with respect to paths of graphs to a closely related problem, called vertex ranking, or ordered coloring. For conflict-free coloring with respect to neighborhoods of vertices of graphs, we prove that number of colors in the order of the square root of the number of vertices is sufficient and sometimes necessary.

Finally, we initiate the study of Ramsey-type problems for conflict-free colorings and compute a van der Waerden-like number.

Acknowledgments

I would like to thank my supervisor Stathis Zachos. His dedication to teaching and disseminating his knowledge is exemplary. He taught me, inspired me, discussed problems with me, and spent a lot of time reading, and commenting on my manuscripts. I would also like to thank Amotz Bar-Noy for originally introducing me to the conflict-free coloring problem for intervals. Working with him has been a beneficial experience for me and he has been very generous to me. The directions he has given me for research and our collaboration have shaped the contents of this thesis. I would like to thank János Pach for spending time with me discussing several geometric and graph problems. His seminars and classes were very useful. Amotz, János, and Stathis have supported me financially, and also in other ways, during my studies. I would like to especially thank Shakhar Smorodinsky, the expert on conflict-free coloring, who has honored me with his collaboration, and has hosted me on several occasions. I would like to thank Boris Aronov for serving as the external member in my committee and for diligently reading and commenting on drafts of this work. I would like to thank Alex Delis, Yi Feng, Andreas Holmsen, Matya Katz, Mihalis Lampis, Nikos Leonardos, Nisan Lev-Tov, Ou Liu, Gila Morgenstern, Svetlana Olonetsky, David Peleg, Radoš Radoičić, Ernst Specker, Géza Tóth, for useful discussions and collaboration. I would like to thank Ted Brown, Joe Driscoll, and Lina Garcia, for their support and help related to matters of the Computer Science Program.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, we introduce the object of study of this work. Then, we review the literature. Finally, we describe in short the new results of this work.

1.1 Graphs, hypergraphs, and colorings

We define the main objects of this work, which are graphs and hypergraphs. Especially for graph theory, we follow the terminology of Bollobás [1998], Diestel [2005].

Definition 1.1. A *graph* G is an ordered pair (V, E), consisting of the *vertex set* (or ground set) V and the *edge set* E, where E is a set of two-element subsets of V.

More precisely, this type of graph is called *undirected* and *simple*, to

differentiate it from graphs where edges have a direction, or edges may have multiplicities. In this work, we are going to be concerned only with undirected simple graphs, that we just call graphs. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G)$. We usually denote the cardinality of the aforementioned sets with $n = |V(G)|$ and $m = |E(G)|$, respectively.

A *hypergraph* is a generalization of a graph in which the edge set, now called *hyperedge set*, consists of hyperedges, where each hyperedge is a non-empty subset of the underlying vertex set. The formal definition follows.

Definition 1.2. A hypergraph H is an ordered pair (V, E), consisting of the vertex set V and the hyperedge set E, where E is a set of subsets of V.

If all hyperedges in a hypergraph have the same cardinality k, then we say the hypergraph is *k-uniform*. Therefore, a graph is a 2-uniform hypergraph. The vertex set of a hypergraph H is denoted by $V(H)$ and its hyperedge set by $E(H)$. A hypergraph is also called a *set system* or a *family of sets*.

In this work, we are going to consider only graphs and hypergraphs for which the underlying vertex set is finite.

Vertex colorings are assignments of values to vertices of a graph or hypergraph. We call these values *colors*. Usually, we want them to satisfy some property.

Definition 1.3. A (traditional) *vertex coloring* of a graph $G = (V, E)$ is a function $C\colon V \to \mathbb{N}^+$ such that adjacent vertices are colored with different

colors. Formally,

$$(\forall \{v, v'\} \in E)(C(v) \neq C(v')).$$

The minimum k for which there is a (traditional) vertex coloring of G with k colors is called the *chromatic number* of G, denoted by $\chi(G)$.

Vertex coloring in hypergraphs can be defined in many ways, so that restricting the definition to simple graphs coincides with traditional graph coloring. At one extreme, it is only required that the vertices of each hyperedge are not all colored with the same color (except for singleton hyperedges).

Definition 1.4. A *non-monochromatic vertex coloring* of hypergraph $H = (V, E)$ is a function $C\colon V \to \mathbb{N}^+$ such that

$$(\forall e \in E)(\exists v \in e)(\exists v' \in e)(|e| = 1 \vee C(v) \neq C(v')).$$

The minimum k for which there is a non-monochromatic vertex coloring of H with k colors is called the *chromatic number* of H, usually denoted by $\chi(H)$.

At the other extreme, we can require that the vertices of each hyperedge are all colored with different colors.

Definition 1.5. A *colorful* or *rainbow* vertex coloring of hypergraph $H = (V, E)$ is a function $C\colon V \to \mathbb{N}^+$ such that

$$(\forall e \in E)(\forall v \in e)(\forall v' \in e)(v \neq v' \to C(v) \neq C(v')).$$

Between these two extremes, there is another possible generalization: A vertex coloring C of hypergraph H is called *conflict-free* if the vertices of each hyperedge are colored in such a way that there is a vertex whose color is unique in the hyperedge.

Definition 1.6. A *conflict-free vertex coloring* of hypergraph $H = (V, E)$ is a function $C\colon V \to \mathbb{N}^+$ such that

$$(\forall e \in E)(\exists v \in e)(\forall v' \in e)(v' \neq v \to C(v') \neq C(v)).$$

If an edge has a uniquely colored vertex, we say that this edge satisfies the *conflict-free property*.

The minimum k for which there is a conflict-free vertex coloring of H with k colors is called the *conflict-free chromatic number* of H, denoted by $\chi_{\text{cf}}(H)$.

Remark 1.7. It is not difficult to see that all three aforementioned colorings coincide with traditional graph coloring for 2-regular hypergraphs.

Conflict-free colorings of hypergraphs are the main subject of this work.

1.2 Motivation and related work

1.2.1 Conflict-free coloring

Conflict-free coloring models frequency assignment for cellular networks. A cellular network consists of two kinds of nodes: *base stations* and *mobile*

agents. Base stations have fixed positions and provide the backbone of the network; they are modeled by vertices in V. Mobile agents are the clients of the network and they are served by base stations. This is done as follows: Every base station has a fixed frequency; this is modeled by the coloring C, i.e., colors represent frequencies. If an agent's device needs to establish a link with a base station it has to tune itself to this base station's frequency. Since agents are mobile, they can be in the range of many different base stations. The range of communication of every agent is modeled by a hyperedge $e \in E$, which is the set of base stations that can communicate with the agent. To avoid interference, the system must assign frequencies to base stations in the following way: For any range, there must be a base station in the range with a frequency that is not reused by some other base station in the range. This is modeled by the conflict-free property.

One can solve the problem by assigning n different frequencies to the n base stations. However, using many frequencies is expensive, and therefore, a scheme that reuses frequencies, where possible, is preferable. For more details on frequency assignment problems, in general, see Aardal et al. [2001].

The aforementioned problem is formalized in the following. The set of points and the family of shapes given in the following definition can lie in any space, but usually we consider the plane.

Definition 1.8. Given a set P of n points and a family $\mathcal{F}$ of geometric shapes, we say that the hypergraph $H = (P, E)$ is *induced* by P and $\mathcal{F}$ if:

A subset $S \subseteq P$ is a hyperedge in E if and only if $S = P \cap Q$ for some $Q \in \mathcal{F}$.

The hypergraphs induced as above are called *geometric* hypergraphs. Possible families of shapes are

- all closed disks (in the plane),
- all unit disks (disks of unit radius),
- all halfplanes, etc.

When we conflict-free color the hypergraph induced by P and $\mathcal{F}$, we say that we conflict-free color P *with respect to* family $\mathcal{F}$. Therefore, in the aforementioned examples, we say that we conflict free color a set of points with respect to

- all closed disks (in the plane), or
- all unit disks (disks of unit radius), or
- all halfplanes, etc.

Remark 1.9. Although family $\mathcal{F}$ can be infinite, since set P is finite, the hyperedge set is finite and its cardinality is bounded by $2^{|P|}$. The motivation for considering a family of shapes such as all disks comes from the moving agents. Usually, a range of an agent can be modeled by a disk in the plane, and the agent could be anywhere in the plane. Different agents could have ranges of the same shape, but differ in communication radii (taking into

account different communication equipment), and therefore all disks in the plane can be viewed as possible ranges. If, however, all agents have the same communication equipment with the same communication radius, then all disks with some fixed radius could be considered as possible ranges (for example, unit disks after a proper scaling).

A conflict-free coloring of five points in the plane with respect to disks is shown in figure 1.1. It uses three colors and it is not difficult to see that it is optimal.

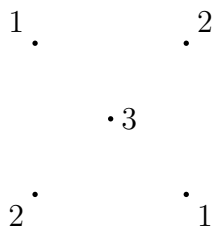

Figure 1.1: A conflict-free coloring of five points with respect to disks in the plane

The study of conflict-free colorings originated in the work of Even et al. [2003] and Smorodinsky [2003]. In addition to the practical motivation described above, 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., Even et al. [2003], Smorodinsky [2003], Pach and Tóth [2003], Har-Peled and Smorodinsky [2005], Fiat et al. [2005], Elbassioni and Mustafa [2006], Chen et al. [2006], Alon and Smorodinsky [2006], Bar-Noy et al. [2006, 2008], Chen et al. [2007], Cheilaris [2006], Cheilaris et al. [2006], Bar-Noy et al. [2007a], Ajwani et al. [2007],

Bar-Noy et al. [2007b,c,d], Smorodinsky [2007], Katz et al. [2007], Keszegh [2007], Chen et al. [2008], Chan et al. [2008]).

Even et al. [2003] also defined a dual conflict-free coloring problem, which can also model frequency assignment in cellular networks. They used a hypergraph defined as follows.

Definition 1.10. Given is a set $\mathcal{R}$ of n geometric shapes, called a range space. For every point p in the union of shapes in $\mathcal{R}$, define $R(p) \subseteq \mathcal{R}$ to be the set of shapes that contain p, i.e., $R(p) = \{r \in \mathcal{R} \mid p \in r\}$. We say that the hypergraph $H = (\mathcal{R}, E)$ is *induced* by range space $\mathcal{R}$ if:

> A subset $S \subseteq \mathcal{R}$ is a hyperedge in E if and only if there exists a point $p \in \bigcup_{r \in \mathcal{R}} r$ such that $R(p) = S$.

When we conflict-free color the hypergraph induced by $\mathcal{R}$, we say that we conflict-free color $\mathcal{R}$. Typical, examples of $\mathcal{R}$ are sets of disks, sets of unit disks, sets of axis-parallel rectangles, etc. In figure 1.2, we show a conflict-free coloring of six disks in the plane with three colors (which is optimal).

The shapes in $\mathcal{R}$ model the range of n base stations, also referred to as antennas. A moving agent that is at point p is under the influence of antennas in $R(p)$. To avoid interference, there must be an antenna in $R(p)$ assigned a unique frequency among the antennas in $R(p)$. If we model frequencies with colors, this is achieved with a conflict-free coloring of the hypergraph induced by $\mathcal{R}$. 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

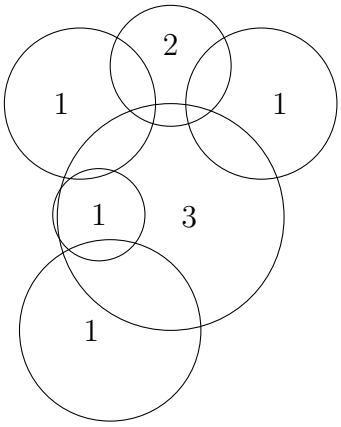

Figure 1.2: A conflict-free coloring of six disks in the plane

and its radius of transmission is fixed and is given (the transmission radii of the antennas are not necessarily equal). Even et al. [2003] 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. It was also shown that this bound is worst-case optimal. Thus, Even et al. [2003] showed that any hypergraph induced by a family $\mathcal{R}$ of n disks in the plane admits a conflict-free 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. Even et al. [2003] showed that finding the minimum number of colors needed to conflict-free color a given collection of disks is NP-hard even when all disks are congruent, and provided a $O(\log n)$ approximation-ratio algorithm. The results of Even et al. [2003] were further extended in Har-Peled and Smorodinsky [2005] by com-

bining more involved probabilistic and geometric ideas. The main result of Har-Peled and Smorodinsky [2005] is a general randomized algorithm which conflict-free colors any set of n 'simple' regions (not necessarily convex) whose union has 'low' complexity, using a 'small' number of colors.

To capture a dynamic scenario where antennas can be added to the network, Fiat et al. [2005] initiated the study of online conflict-free coloring of hypergraphs. The subject of online algorithms is important in computer science and very rich. An online computation models a situation in which an algorithm has to process an input piece-by-piece, making irrevocable decisions, without having the whole input available from the start. Usually, the performance of an online algorithm is measured against the performance of the best offline algorithm, namely an algorithm that knows the whole input from the start. In an anthropomorphic point of view, given an online algorithm, an adversary tries to chose the input in such a way that the performance of the online algorithm is as far as possible from the performance of the best offline algorithm. For more details, see the standard textbook on online algorithms by Borodin and El-Yaniv [1998].

Fiat et al. [2005] considered a very simple hypergraph H with a set P of n points on the line as its vertex set and the set of all intersections of the points with some interval as its hyperedge set. 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. The coloring must 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, Fiat et al. [2005] provided a randomized algorithm for online conflict-free coloring n points on the line with $O(\log n \log\log n)$ colors, with high probability. 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. The notion of an oblivious adversary was introduced by Raghavan and Snir [1989]. Fiat et al. [2005] also provided a deterministic algorithm for online conflict-free coloring n points on the line with $\Theta(\log^2 n)$ colors in the worst case. This is the best known deterministic algorithm in terms of the number of colors used.

Chen [2006] and Chen et al. [2007] provided a randomized algorithm in the oblivious adversary model that uses at most $O(\log n)$ colors, with high probability. The algorithms in the aforementioned papers use number of random bits linear in n.

For conflict-free coloring of n points with respect to (closed) disks, Pach and Tóth [2003] proved a lower bound of $\Omega(\log n)$ colors. They also generalized the result to homothetic copies of any convex body (i.e., scaled and translated, but not rotated copies of a fixed convex body). Har-Peled and Smorodinsky [2005] gave a conflict-free coloring of n points with respect to axis-parallel rectangles that uses $O(\sqrt{n})$ colors. Pach and Tóth [2003] improved this to $O(\sqrt{n \log\log n / \log n})$. Elbassioni and Mustafa [2006] con-

sidered an interesting variation of the problem of conflict-free coloring points with respect to axis-parallel rectangles: They proved that given any set of n points in the plane, one can add $O(n^{1-\varepsilon})$ new points, so that all points can be conflict-free colored with $\tilde{O}(n^{3(1+\varepsilon)/8})$ colors; in the $\tilde{O}$ notation polylogarithmic factors are ignored. Based on Elbassioni and Mustafa [2006], Ajwani et al. [2007] proved that any set of n points can be conflict-free colored in expected polynomial time, with respect to axis-parallel rectangles, using $\tilde{O}(n^{0.382+\varepsilon})$ colors, for any arbitrarily small $\varepsilon > 0$. Although there are only polylogarithmic lower bounds for conflict-free coloring with respect to axis-parallel rectangles, the work of Chen et al. [2008] indicates that the real answer is closer to the upper bound of Ajwani et al. [2007].

Alon and Smorodinsky [2006] considered coloring a collection of n disks in which each disk intersects at most k others so that for each point p in the union of all disks there is at least one disk in the collection containing p whose color differs from that of all other members of the collection that contain p (this is the dual problem of coloring points with respect to ranges). They managed to use just $O(\log^3 k)$ colors; their proof relies on the probabilistic method [Alon and Spencer, 2000], and especially the Lovász local lemma [Erdős and Lovász, 1975].

Smorodinsky [2006] and Smorodinsky [2007] studied non-monochromatic-edge coloring of hypergraphs that are induced by regions bounded by simple Jordan curves. He applied the aforementioned results to conflict-free coloring of regions with near linear union complexity (using a polylogarithmic number

of colors), and axis-parallel rectangles (using $O(\log^2 n)$ colors).

Katz et al. [2007] studied the problem of conflict-free coloring points on the line with respect to a given set of intervals. They claim a polynomial time algorithm that computes a solution that uses at most four times the number of colors of an optimal solution, i.e., it is a 4-approximation algorithm. For more on approximation algorithms, see Vazirani [2001].

1.2.2 Unique maximum property

We define a stronger property that implies the conflict-free property.

Definition 1.11. A *unique maximum* vertex coloring of hypergraph $H = (V, E)$ is a function $C\colon V \to \mathbb{N}^+$ such that

$$(\forall e \in E)(\exists v \in e)(\forall v' \in e)(v' \neq v \to C(v') < C(v)).$$

Namely, a unique maximum coloring is a conflict-free coloring with the additional property that in every hyperedge one of the uniquely occurring colors in the vertex is also the maximum one. The minimum k for which there is a unique maximum vertex coloring of H with k colors is called the *unique maximum chromatic number* of H and is denoted by $\chi_{\mathrm{um}}(H)$.

In the literature, most conflict-free coloring algorithms are in fact unique maximum coloring algorithms. For example, the main conflict-free coloring algorithm used by Even et al. [2003] is a unique maximum algorithm. In general, it seems that arguing about unique maximum colorings is easier

than arguing about conflict-free colorings, maybe because of their additional structure.

1.2.3 Path-related vertex colorings of graphs

Ordered colorings

So far, the hypergraphs we considered were induced by geometric shapes. Now, we consider a hypergraph induced by (simple) paths in a graph. First, we define the notion of a (simple) path.

Definition 1.12. A path in a graph is a sequence v_0, e_1, v_1, $\ldots$, e_ℓ, v_ℓ, where for every i, with $1 \leq i \leq \ell$, $e_i = \{v_{i-1}, v_i\}$. We say that the aforementioned path has length ℓ.

In most cases, we omit the edges from the sequence, and we denote the aforementioned path just with v_0, v_1, $\ldots$, v_ℓ.

Definition 1.13. Given a graph $G = (V, E)$, we say that the hypergraph $H = (V, E')$ is *induced* by the paths of graph G if:

> A subset $P \subseteq V$ is a hyperedge in E' if and only if P is the set of vertices in some path of G.

Now, we define a special vertex coloring of G.

Definition 1.14. An *ordered coloring* of G is a unique maximum coloring of the hypergraph induced by paths of G.

We can also give an equivalent, direct definition.

Definition 1.15. An *ordered coloring* of G is a function $C\colon V(G) \to \mathbb{N}^+$ such that in every path of G the maximum color occurring in the path occurs uniquely in the path. The minimum k such that a graph G has an ordered coloring with k colors is called the ordered chromatic number of G and is denoted by $\chi_{\mathrm{o}}(G)$.

Ordered coloring has been studied by Katchalski et al. [1995]. Some authors, like [Iyer et al., 1988] call it alternatively *vertex ranking*. Vertex ranking has many applications. One application is in the filed of VLSI design [Leiserson, 1980, Sen et al., 1992]. Another application is in the field of parallel Cholesky factorization of matrices [Duff, 1986, Liu, 1986, George et al., 1987, Liu, 1989]. In general graphs, finding the exact ordered chromatic number of a graph is NP-complete [Llewellyn et al., 1989, 1993]. Approximability results are given by Bodlaender et al. [1995]. The vertex ranking problem is also interesting for the Operations Research community, because it also has applications in planning efficient assembly of products in manufacturing systems [Iyer et al., 1988, Schäffer, 1989]. In general, it seems the vertex ranking problem can model situations where interrelated tasks have to be accomplished fast in parallel, with some constraints (assembly from parts, parallel query optimization in databases, etc.).

Ordered colorings with few colors have been found in several families of graphs. For tree graphs, the notion of a (1/2)-separator [Jordan, 1869, Lewis II et al., 1965, Erlebach et al., 2003] is useful. A (1/2)-separator is a vertex

which, when removed, leaves connected components whose size is bounded by $n/2$. It can be proven that $\chi_{\mathrm{o}}(T) \leq 1 + \lfloor \log_2 n \rfloor$ for a tree with n vertices and this is tight for some families, like paths. See, for example, Katchalski et al. [1995]. Moreover, one can find efficiently optimal ordered colorings of trees [Iyer et al., 1988], even in linear time [Schäffer, 1989]. For a planar graph G, using separator theorems [Lipton and Tarjan, 1979, Djidjev, 1982], it can be proven that $\chi_{\mathrm{o}}(G) \leq 3(\sqrt{6}+2)\sqrt{n} \approx 13.3485\sqrt{n}$ [Katchalski et al., 1995].

Square-free colorings

We obtain another related problem by looking at colorings of paths as strings. We impose the following restriction: Every coloring of a path, when viewed as a string, shall not contain a repetition. Formally, a string $w \in (\mathbb{N}^+)^*$ is called *square-free* if there is no substring of w of the form $x^2 = xx$, where x is a nonempty string. Given a coloring C of the vertices of a graph, for every path $p = v_1 \dots v_\ell$, we define the color string of p to be $C(v_1) \dots C(v_\ell)$.

Definition 1.16. A *square-free coloring* of G is a function $C\colon V(G) \to \mathbb{N}^+$ such that for every path in the graph its color string is square-free.

The minimum k for which a graph G has a square-free coloring with k colors is called the *square-free chromatic number* of graph G and is denoted by $\chi_{\mathrm{sf}}(G)$.

Every ordered coloring is square-free, because a path whose color string is a square implies a repetition of the maximum color in the path. Every

square-free coloring is a traditional coloring, because traditional colorings can be seen as colorings with no square color strings of length two. Therefore, we have the hierarchical inclusion relation between colorings of graphs, shown in figure 1.3.

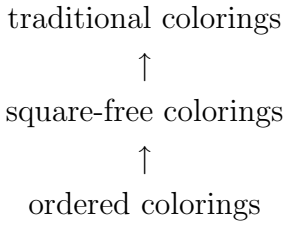

Figure 1.3: An inclusion hierarchy of graph colorings

Restating the aforementioned inclusions in terms of chromatic numbers, we have the following.

Proposition 1.17. $\chi(G) \leq \chi_{\mathrm{sf}}(G) \leq \chi_{\mathrm{o}}(G)$.

Square-free coloring a graph might require a lot fewer colors than ordered coloring. For example, a seminal result by Thue [1906] shows that three colors suffice to square-free color any path P_n. In contrast, ordered colorings of P_n require $\Omega(\log n)$ colors. Another name for square-free colorings (of graphs) is *non-repetitive* colorings. Non-repetitive colorings of graphs have been recently studied by Alon et al. [2002], Brešar et al. [2007], Barát and Wood [2005]. Another related class of colorings consists of *cube-free* colorings, where color strings of paths can not contain a x^3 substring, for x non-empty. It is known from Thue [1906], and also implicitly from Prouhet [1851], that

two colors suffice to cube-free color any path P_n. Square-free, cube-free, and related colorings have been studied extensively for strings (i.e., for the path graph in our setting). For more details, see the book by Allouche and Shallit [2003].

1.3 Contributions

In chapter 2, we investigate deterministic algorithms for online conflict-free coloring points with respect to intervals. We introduce a hierarchy of four models for the above problem: (i) static, (ii) dynamic offline, (iii) dynamic online with absolute positions, (iv) dynamic online with relative positions. We show that the hierarchy is strict. In the dynamic offline model, we give a deterministic algorithm that uses at most $\log_{3/2} n + 1 \approx 1.71 \log_2 n$ colors and exhibit inputs that need at least $3 \log_5 n + 1 \approx 1.29 \log_2 n$ colors. For the online absolute positions model, we give a deterministic algorithm that uses at most $3\lceil \log_3 n \rceil \approx 1.89 \log_2 n$ colors. This is the best known deterministic online algorithm using $O(\log n)$ colors, in a non-trivial deterministic online model.

We also consider conflict-free coloring with respect to intervals that contain at least one of the two extreme points, i.e., conflict-free coloring with respect to rays (or halflines). This is the one-dimensional version of coloring points in the plane with respect to halfplanes.

In chapter 3, we provide a framework for online conflict-free coloring any

hypergraph. We define the notion of a k-degenerate hypergraph. We use this framework to obtain an efficient randomized online algorithm for conflict-free coloring any k-degenerate hypergraph. Our algorithm uses $O(k \log n)$ colors with high probability and this bound is asymptotically optimal, because there are families of k-degenerate hypergraphs that need that many colors. 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 conflict-free coloring some hypergraphs that arise in geometry. Our algorithm uses exponentially fewer random bits compared to previous algorithms (expectedly logarithmic in n compared to linear in n).

We introduce deterministic online conflict-free coloring algorithms for points on the line with respect to intervals and for points in the plane with respect to halfplanes (or unit disks) that use $O(\log n)$ colors and performs only $O(n)$ recolorings.

Chapter 4 contains several other results of ours on conflict-free coloring.

First, as a variation of conflict-free coloring with respect to (all) intervals, we study the problem of conflict-free coloring collinear points with respect to a given set of intervals. The problem was studied by Katz et al. [2007] who claim a polynomial time 4-approximation algorithm. We propose a polynomial time 2-approximation algorithm. We also prove that the analysis of our 2-approximation algorithm is tight, by showing a family of instances which our algorithm colors with twice as many colors as the optimal solution.

We initiate the study of conflict-free coloring vertices of graphs with

respect to paths of graphs. The minimum number of colors necessary to conflict-free color a graph G as above is the conflict-free chromatic number of G (with respect to paths), denoted by $\chi_{\text{cf}}(G)$. We relate this new notion to the already studied notion of ordered colorings, which are essentially unique maximum conflict-free colorings. Conflict-free colorings of graphs are a weakening of ordered colorings. Moreover, a conflict-free coloring is always a square-free coloring, because a path whose color string is a square implies that all colors repeat in the path, i.e., the conflict-free property is not true. Therefore we have the updated inclusion hierarchy of colorings, shown in figure 1.4 (compare with figure 1.3).

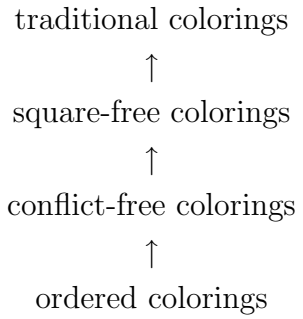

Figure 1.4: An updated inclusion hierarchy of graph colorings

As a result, in terms of chromatic numbers, we have the following.

Proposition 1.18. $\chi(G) \leq \chi_{\text{sf}}(G) \leq \chi_{\text{cf}}(G) \leq \chi_{\text{o}}(G)$.

It seems the conflict-free chromatic number behaves more like the ordered chromatic number. For many graphs, the two numbers are the same. How-

ever, we provide a family of graphs where the two chromatic numbers are different. It is interesting that in all graphs we have found with $\chi_{\mathrm{cf}}(G) < \chi_{\mathrm{o}}(G)$, the two chromatic numbers differ just by one ($\chi_{\mathrm{cf}}(G) = \chi_{\mathrm{o}}(G) - 1$).

We also initiate the study of conflict-free coloring vertices of graphs with respect to neighborhoods of graphs. The open neighborhood of a vertex v contains all the other vertices that share an edge with v. The closed neighborhood of v also contains the vertex v. We define appropriate chromatic numbers and provide upper bounds based on the dominating set number of the graph and the maximum degree of the graph. By combining these bounds, we prove that every graph with n vertices can be conflict-free colored with respect to neighborhoods with $2\sqrt{n}$ colors, and this is order tight in some cases (namely there are graphs that need $\Omega(\sqrt{n})$ colors).

Finally, we initiate the study of Ramsey-type problems for conflict-free coloring. Ramsey theory is a rich subject in combinatorics and was essentially initiated by Ramsey [1930], Erdős and Szekeres [1935], although Ramsey-type results predate the aforementioned papers [Hilbert, 1892, Schur, 1916, van der Waerden, 1927]. See the book by Graham et al. [1990] for more details. One could describe Ramsey-type results as follows. For some family of hypergraphs, given a finite number of colors, there is a large enough hypergraph, such that no matter how one colors its vertices, some monochromatic edge emerges (i.e., with all vertices with the same color). Since the emergence of a monochromatic edge implies already the emergence of a non-conflict-free edge (or more simply, a conflict edge), all monochromatic Ramsey-type

results imply conflict Ramsey-type results. A question that is also important is how big the hypergraph has to become before the emergence of a monochromatic or conflict edge. There is a monochromatic Ramsey-type theorem when the hyperedge set consists of all arithmetic progressions of length k over $\{1, \ldots, n\}$ by van der Waerden [1927]. We immediately have a conflict-free Ramsey-type theorem and we compute exactly how big the hypergraph has to be in order to have a conflict edge, in the case of two colors, for arbitrary k. In other words, we compute exactly conflict-free van der Waerden numbers for two colors. This exactness is in sharp contrast with the monochromatic van der Waerden numbers, of which very few are known exactly, even for two colors. The six non-trivial monochromatic van der Waerden numbers known exactly are shown in table 4.1 on page 129. As expected, conflict-free van der Waerden numbers are significantly smaller than monochromatic van der Waerden numbers.

Chapter 2

Deterministic conflict-free coloring for intervals

We investigate deterministic algorithms for a frequency assignment problem in cellular networks. The problem can be modeled as a special vertex coloring problem for hypergraphs: In every hyperedge there must exist a vertex with a color that occurs exactly once in the hyperedge (the conflict-free property). We concentrate on a special case of the problem, called conflict-free coloring for intervals. We introduce a hierarchy of four models for the above problem: (i) static, (ii) dynamic offline, (iii) dynamic online with absolute positions, (iv) dynamic online with relative positions. We show that the hierarchy is strict. In the dynamic offline model, we give a deterministic algorithm that uses at most $\log_{3/2} n + 1 \approx 1.71 \log_2 n$ colors and exhibit inputs that force any algorithm to use at least $3 \log_5 n + 1 \approx 1.29 \log_2 n$ colors. For the online

absolute positions model, we give a deterministic algorithm that uses at most $3\lceil \log_3 n \rceil \approx 1.89 \log_2 n$ colors. To the best of our knowledge, this is the first deterministic online algorithm using $O(\log n)$ colors, in a non-trivial online model. We also consider conflict-free coloring only with respect to intervals that contain at least one of the two extreme points.

2.1 Introduction

Fiat et al. [2005] considered the special case of conflict-free coloring where the hypergraph is defined as follows: Vertices are identified with points on a line and E consists of all subsets of V defined by intervals intersecting at least one vertex. A line with n points has $n(n+1)/2$ such subsets (for every $i \in \{1, \ldots, n\}$, there are $n - i + 1$ different subsets containing i points). For $n = 5$, these subsets are shown in figure 2.1. We call these subsets intervals since for our purpose, two intervals are equivalent if they contain the same vertices. We represent colorings by listing the colors of points from left to right in a *string*. For example, for the points in figure 2.1 ($n = 5$), 12312 is a conflict-free coloring, whereas 12123 is not (because the interval containing the four leftmost points does not contain a point of unique color).

Conflict-free coloring for intervals is important because it can model assignment of frequencies in networks where the agents' movement is approximately unidimensional, e.g., the cellular network that covers a single long road and has to serve agents that move along this road. In some kinds of

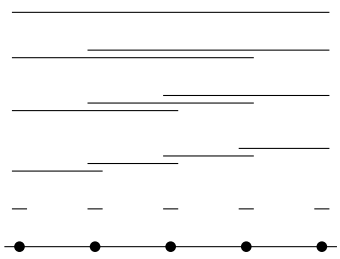

Figure 2.1: Points on a line and intervals

networks, like wireless sensor networks, the density of base stations can be very high and non-uniform. Moreover, the ranges of the agents can also be non-uniform. To ensure the absence of conflicts we require the conflict-free property for all possible intervals. Also, conflict-free coloring for intervals plays a role in the study of conflict-free coloring for more complicated range spaces and it is a special case of conflict free coloring points on the plane with respect to disks, when all points to be colored are collinear (see Even et al. [2003]).

The *static* version of the problem, where the n points are to be colored simultaneously, is solved in Even et al. [2003]. For $n = 2^k - 1$, the coloring C^k is defined recursively as follows: $C^1 = 1$ and $C^{k+1} = C^k \circ (k+1) \circ C^k$ (where $\circ$ is the concatenation operator for strings). The coloring C^k is conflict-free and uses k colors for $2^k - 1$ points. For n with $n < 2^k - 1$, the prefix (in fact, any substring) of length n of C^k is conflict-free. Even et al. [2003] also show that this coloring with $1 + \lfloor \log_2 n \rfloor$ colors is the best possible. Observe that C^k has the property that the maximum color in each interval is always unique. Coloring intervals with a unique maximum is called *vertex ranking*

of paths, or *ordered coloring* of paths, and in that context similar results were obtained in Iyer et al. [1988], Katchalski et al. [1995]. Since colorings with asymptotically $\log_2 n$ colors are important, from now on, we sometimes abbreviate the binary logarithm of n by $\lg n$.

The problem becomes more interesting when the vertices are given online by an adversary. Namely, at every given time step $t \in \{1, \ldots, n\}$, a new vertex $v_t \in V$ is given and the algorithm must assign v_t a color such that the coloring is a conflict-free coloring of the hypergraph that is induced by the vertices $V_t = \{v_1, \ldots, v_t\}$. Once v_t is assigned a color, that color cannot be changed in the future. It is desirable to avoid recoloring for the following technical reason: If a base station changes color, there might be disruption of service for all agents connected to it (a model in which a small number of recolorings is allowed is presented in section 3.5 and also in Bar-Noy et al. [2007c]). We are interested in an online setting, in which the algorithm has no knowledge of how vertices will be requested in the future. For this version of the problem, in the case of intervals, Fiat et al. [2005] provide several algorithms. Their deterministic algorithm uses $\Theta(\log^2 n)$ colors in the worst case. Their randomized algorithm uses $O(\log n \log\log n)$ colors with high probability. Randomized algorithms that use $O(\log n)$ colors with high probability have been obtained and are presented in chapter 3 (see also [Chen, 2006, Chen et al., 2007, Bar-Noy et al., 2007c]). All of the randomized algorithms assume the slightly weaker *oblivious* adversary model, in which the adversary has to commit to a specific input sequence before revealing the

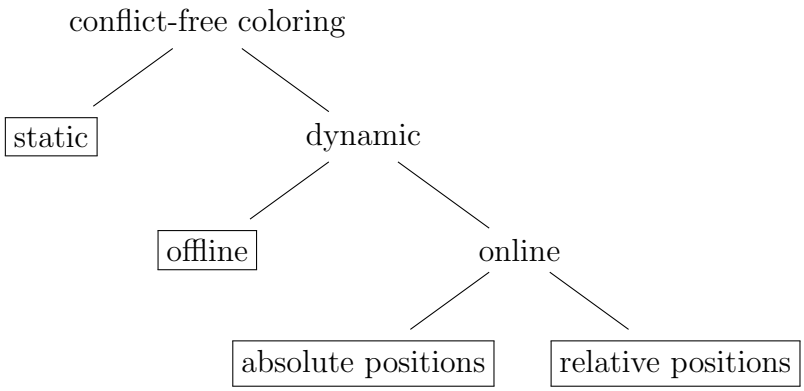

Figure 2.2: Models for conflict-free coloring for intervals

first vertex to the algorithm (see Borodin and El-Yaniv [1998]).

Our contribution

We introduce a *hierarchy* of four models for the above conflict-free coloring problem for hypergraphs: (i) static, (ii) dynamic offline, (iii) dynamic online with absolute positions, and (iv) dynamic online with relative positions. Before defining the models, the following definition is necessary.

Definition 2.1. A hypergraph $H' = (V', E')$ is an *induced subhypergraph* of $H = (V, E)$ if $V' \subseteq V$ and $E' = \{e \cap V' \mid e \in E\}$ We say that H' is *induced* by V'.

Below we define the four models. The relationship among them is shown in figure 2.2.

- In the *static model*, the complete hypergraph H is given, and a conflict-free coloring for H must be found by the algorithm.

In dynamic models, a *sequence* $\{H_t\}_{t=1}^n$ of hypergraphs (with $H = H_n$) is given where H_t has t vertices and, for $t > 1$, H_{t-1} is an induced subhypergraph of H_t; for every t a conflict-free coloring for H_t must be found that, for $t > 1$, extends the coloring of H_{t-1} (i.e., the algorithm can not change colors of vertices). Alternatively, the input is a permutation of the vertices of the final hypergraph, and H_t, the hypergraph to be colored at every time step, is the subhypergraph of H induced by the first t vertices in the permutation.

- In the *dynamic offline model*, the complete sequence $\{H_t\}_{t=1}^n$ is given.

In dynamic online models, the sequence $\{H_t\}_{t=1}^n$ is revealed incrementally, at discrete time steps $t = 1, \ldots, n$, i.e., at time t, H_t is given, and a color for the new vertex v_t must be found without knowledge of future $H_{t'}$, where $t' > t$.

- In the *dynamic online with absolute positions model*, in addition to $\{H_t\}_{t=1}^n$ being revealed incrementally, the final $H_n = H$ is given from the start as a *vertex-labeled* hypergraph and for each time t, H_t is also given as a vertex-labeled hypergraph, with the induced subhypergraph isomorphism between H_t and H_n *preserving* the labels. This means that the algorithm knows for every new vertex v_t where it is going to lie (i.e., its 'absolute' position) in the final hypergraph H.

- In the *dynamic online with relative positions model*, no information about the final hypergraph H is given (not even its size n). The only

information might be the structure of the final hypergraph (for example, when we color points on a line with respect to intervals). Thus, for every new vertex v_t, we only know its 'relative' position with respect to already inserted points, by means of the information in H_t.

From the application point of view, all models are interesting.

- In the static model, all base stations are activated at the same time.
- The dynamic offline model captures a scenario where the order of activation of base stations is known from the start. Additionally, comparison of dynamic online algorithms against algorithms in the dynamic offline model is more fair than against algorithms in the static model. Chan et al. [2008] studied the problem of coloring points on the plane with respect to unit disks in the dynamic offline model we introduced.
- The dynamic online with absolute positions model is also motivated, because revealing absolute positions is not unnatural in many scenarios: One can think of all base stations being at fixed positions, which are known to the algorithm in advance. This means that the algorithm is aware of the final hypergraph that models the situation where all base stations are activated. At the start, no base station is activated. Base stations are constructed or activated one by one, in some order, in response to increasing network traffic, and thus the order of activation is not known from the start. Every new station has to be given a color by the algorithm, such that the conflict-free property is maintained.

model	lower bound	upper bound
dynamic online, relative positions	$1 + 3\log_5 n$	$O(\log^2 n)$
dynamic online, absolute positions	$1 + 3\log_5 n$	$3\lceil \log_3 n \rceil$
dynamic offline	$1 + 3\log_5 n$	$1 + \log_{3/2} n$
static	$1 + \lfloor \log_2 n \rfloor$	$1 + \lfloor \log_2 n \rfloor$

Table 2.1: Number of colors used in deterministic algorithms for intervals

model	lower bound	upper bound
dynamic online, relative positions	$1.29\log_2 n$	$O(\log^2 n)$
dynamic online, absolute positions	$1.29\log_2 n$	$1.89\log_2 n$
dynamic offline	$1.29\log_2 n$	$1.71\log_2 n$
static	$1.00\log_2 n$	$1.00\log_2 n$

Table 2.2: Asymptotic number of colors used in deterministic algorithms for intervals scaled to logarithm with base 2

- Finally, the dynamic online with relative positions model is the one with the fewest restrictions on the adversary and can capture a situation where the exact positions of new base stations can not be planned in advance.

The four models produce a *hierarchy* of models, in the sense that an adversary in a higher model has more power and an algorithm for a higher model works also for a lower model. A summary of results for deterministic algorithms for conflict-free coloring with respect to intervals is given in tables 2.1 and 2.2. All the online algorithms considered so far in the literature work in the *relative positions* model. Our main technical result concerns a deterministic algorithm that uses $3\lceil \log_3 n \rceil \approx 1.89 \lg n$ colors, in the slightly changed online model of *absolute positions*. Our deterministic

algorithms work against an unrestricted adversary, i.e., they do not assume the (weaker) oblivious adversary. We also exhibit sequences of length $n = 3^k$ that need at least $1 + 2\log_3 n \approx 1.26 \lg n$ colors in any dynamic model, and sequences of length $n = 5^k$ that need at least $1 + 3\log_5 n \approx 1.29 \lg n$ colors in any dynamic model. The above lower bound instances are constructed with the dynamic offline model in mind, but because of the hierarchy, they apply also to the two dynamic online models, as shown in table 2.1. It is not difficult to also prove gaps for the lower bounds between the dynamic offline and the absolute positions models and between the absolute positions and the relative positions models, but they are constant additive, not constant multiplicative (like the gap in the lower bound between the static and the dynamic offline models). We defer the aforementioned discussion until we introduce ways to describe the two different types of dynamic inputs (absolute and relative) in section 2.2. For the offline model, we describe an algorithm that uses at most $1 + \log_{3/2} n \approx 1.71 \lg n$ colors in the dynamic offline model.

Finally, we discuss coloring with respect to a specific subset of intervals. One interesting case is coloring with respect to *rays* (or *halflines*), namely the subset of intervals that contain at least one of the two extreme points. For this case, we show a strong separation between static and dynamic offline models, in the sense that in the static model three colors suffice for any n, whereas in the dynamic offline model $\lfloor \lg n \rfloor + 1$ colors might be necessary. On the other hand, $\lfloor \lg (n-2) \rfloor + 3$ colors suffice even in the relative positions model.

In section 2.2, we introduce two ways to describe input to dynamic algorithms in the case of intervals. In section 2.3, we consider the dynamic offline model. In section 2.4, we discuss the $O(\log n)$ colors algorithm in the absolute positions model. In section 2.5, we study conflict-free coloring with respect to rays. In section 2.6, we discuss some of the results and mention open problems.

2.2 Describing dynamic model inputs

We show two ways to represent inputs for dynamic models, in the intervals case. We will be using them in subsequent sections.

In the *relative positions model*, the sequence of points inserted can be described by the position in which each new point is inserted, relative to previously inserted points. If $i-1$ points have already been inserted, the i-th point can be inserted in any of i positions described by an integer in the range $[0, i-1]$: 0 is for the new point at the start of the sequence (before any other point), and $k > 0$ is for the new point immediately after the k-th already inserted point.

An insertion sequence of length n is represented by a string of n integers, σ, where $0 \leq \sigma_{(i)} \leq i-1$. If we consider insertion sequences of the same length n ordered lexicographically, then the first and last elements in that order are: $s_n^{\text{first}} = [0, 0, 0, \ldots, 0]$, $s_n^{\text{last}} = [0, 1, 2, \ldots, n-1]$. In the relative positions online model, an insertion sequence is revealed from left to right,

one by one, to the online algorithm. There are $n!$ possible insertion sequences of length n.

In the *absolute positions model*, initially the algorithm knows the total number of points to be inserted. Then, for each new point the absolute position of that point in the final sequence is revealed to the algorithm. The absolute position can be any number in $\{1, \ldots, n\}$ which has not appeared before. Thus, the input to the algorithm is a permutation $\pi \in \mathbf{S}_n$ that is revealed one by one, from left to right. Therefore, there are again $n!$ possible insertion sequences of length n.

In the dynamic *offline* setting, the input can be given in either absolute or relative positions, because the two representations are easily convertible to each other if the *whole* sequence is known. For example, the insertion sequence $\sigma = 00121$ (relative positions) corresponds to the permutation $\pi = 51342$ (absolute positions), which means the first point inserted is at the 5^{th} absolute position (rightmost), the second point inserted is at the 1^{st} absolute position (leftmost), and so on.

Strict hierarchy of models. In the following, we show that the hierarchy of models shown in figure 2.2 is strict, in the sense that deterministic algorithms in less constrained models have strictly more power and can be made to use less colors, and therefore we prove a separation result for the models.

The dynamic version of the problem is more difficult than the static version of the problem, because there might be more constraints to be satisfied.

Even for $n = 3$, we have the static coloring 121, but the insertion sequence $\sigma = 011$ ($\pi = 132$ in absolute positions) needs three different colors to be colored dynamically. In section 2.3, we exhibit sequences that separate the two models by more than one color.

In order to separate the offline dynamic model from the online models, we need to exhibit two insertion sequences of absolute positions that have a common prefix, but the optimal offline algorithm colors essentially differently the sequences, up to that prefix. Two such sequences of absolute positions are: $\pi = 123654$ and $\pi' = 123465$, which share the common prefix 123. Two optimal colorings for the above sequences are: $C = 123121$ and $C' = 121312$, respectively, and it is not difficult to see (after some case analysis) that in an optimal coloring, for π the algorithm has to use a new color for the point appearing at time $t = 3$, whereas for π' the algorithm has to reuse a previously used color for the aforementioned point. Therefore, there is no deterministic online algorithm which computes the optimal coloring for both π and π', whereas some offline algorithm can compute the optimal coloring for both.

Similarly, in order to separate the absolute position from the relative position online model, we need to exhibit two insertion sequences of relative positions that have a common prefix, but for which the respective absolute positions inputs have different prefixes, so knowledge of absolute positions might allow us to color the sequences better. Two possible sequences of relative positions are: $\sigma = 0122456$ and $\sigma' = 0123456$ with common prefix

012. The corresponding absolute positions sequences are: $\pi = 1243567$ and $\pi' = 1234567$, i.e., they differ in the third position (or, alternatively, the point appearing at time $t = 3$). In both cases the optimal coloring is of the form 1213121 and it is computed by an algorithm only if for σ the algorithm uses a new color for the point appearing at time $t = 3$, whereas for σ' the algorithm reuses a previously used color for the aforementioned point. Therefore, there is no deterministic online algorithm on relative positions that can color both input sequences optimally, because no relative positions algorithm can differentiate between the two sequences early enough, in contrast with a deterministic online algorithm on absolute positions, which can compute the optimal coloring for both inputs.

2.3 Dynamic offline model

Lower bound We exhibit insertion sequences that need asymptotically $c \lg n$ colors, where $c > 1$. This is in contrast with the static model in which asymptotically $\lg n$ colors are enough in all cases. First, some definitions are needed.

Definition 2.2. Given a string π of numbers and $x \in \mathbb{N}$, the string $(\pi + x)$ is defined by adding x to each element of π, i.e., $(\pi + x)_{(i)} = \pi_{(i)} + x$, for $i \in \{1, \ldots, |\pi|\}$, where $|\pi|$ is the length of π.

We also define a sum-like operator for concatenation of strings:

$$\bigcirc_{i=1}^{p} s_i := s_1 \circ \cdots \circ s_p \,.$$

Permutations can be viewed as strings of numbers. Here we first explain and then formally define a sequence of larger and larger permutations: π, π^2, π^3, ... In general, for $k > 1$, π^k consists of n π^{k-1}-like components that are inserted in the π order. We alert the reader that π^k should not be confused with concatenation of k copies of the string π: $\bigcirc_{i=1}^{k} \pi$. For example, if $\pi = 132$, then $n = |\pi| = 3$ and $\pi^2 = 132798465$, i.e., $|\pi^2| = 3^2 = 9$. Notice that π^2 consists of three π-like components (132, 798, 465), that are inserted in the π order.

Definition 2.3. Given a permutation π, with $|\pi| = n$, and $k \in \mathbb{N}$, the permutation π^k is defined recursively:

$$\begin{aligned} \pi^0 &= 1 \ , \\ \pi^{k+1} &= \bigcirc_{i=1}^{n} (\pi^k + (\pi_{(i)} - 1) \cdot n^k) \ . \end{aligned}$$

It is not difficult to prove that if π is a permutation, then indeed π^k is also a permutation. The proof is by induction on k. We have that $\pi_0 = 1$ is trivially a permutation. For the inductive step, assuming that π^k is a permutation, in π^{k+1} the n different concatenation components range in $\{1, \ldots, n^k\}$, $\{n^k+1, \ldots, 2n^k\}$, $\ldots$, $\{(n-1)n^k+1, \ldots, n^k\}$, and therefore π^{k+1}

is also a permutation.

The idea behind our lower bound proofs is to find a permutation π such that for every $k > 0$, π^k needs y more colors than π^{k-1}, where y is fixed. In the following, we prove two lower bounds. The first is simpler, it is based on $\pi = 132$ (with $y = 2$), and gives a $2\log_3 n + 1 \approx 1.26 \lg n$ lower bound. The second is more elaborate, it is based on $\pi = 15432$ (with $y = 3$), its proof relies on the previous lower bound, and it gives a lower bound of $3\log_5 n + 1 \approx 1.29 \lg n$.

For every k, we exhibit an insertion sequence π^k of absolute positions that has length $n = 3^k$ and needs $2k + 1 = 2\log_3 n + 1$ colors to be conflict-free colored.

Proposition 2.4. *For $\pi = 132$, input π^k needs at least $2k + 1$ colors in the dynamic model.*

Proof. The proof is by induction. For the base case, $k = 0$, input $\pi^0 = 1$ needs one color.

For the inductive step, assume input π^{k-1} needs at least $2k - 1$ colors. Consider the colorings of the three π^{k-1} components of π^k: A, B, C, in increasing time order of appearance. Since they are inserted in the 132 order, the π^k coloring incrementally looks like: A, then AB, and finally ACB. By the inductive hypothesis each of A, B, C uses at least $2k - 1$ colors.

Assume, for the sake of contradiction, that ACB uses at most $2k$ colors. At least one color that appears in ACB must be unique. If the unique color

is in A, then CB uses at most $2k-1$ colors; if in B, then AC uses at most $2k-1$ colors; if in C, then AB uses at most $2k-1$ colors. However, all of CB, AC, AB appear at some point in the coloring of π^k: CB and AC appear in ACB, and AB appears just before the insertion of C. Thus, in each of CB, AC, AB, there must be a unique color. Now, take $J \in \{\text{CB}, \text{AC}, \text{AB}\}$ that is using at most $2k-1$ colors: This J has a unique color among the colors used in J, that can only appear in one of the two π^{k-1} components of J, therefore, the other π^{k-1} component of J is using at most $2k-2$ colors; a contradiction to the inductive hypothesis. □

For every k, we exhibit an insertion sequence π^k of absolute positions that has length $n = 5^k$ and needs $3k+1 = 3\log_5 n + 1$ colors to be conflict-free colored.

Proposition 2.5. *For $\pi = 15432$, input π^k needs at least $3k+1$ colors in the dynamic model.*

Proof. The proof is by induction. For the base case, $k = 0$, input $\pi^0 = 1$ needs one color.

For the inductive step, assume input π^{k-1} needs at least $3k-2$ colors. Consider the colorings of the five π^{k-1} components of π^k: A, B, C, D, E, in increasing time order of appearance. Since they are inserted in the 15432 order, the π^k coloring incrementally looks like: A, AB, ACB, ADCB, and finally AEDCB. By the inductive hypothesis each of A, B, C, D, E uses at least $3k-2$ colors.

Assume, for the sake of contradiction, that AEDCB uses at most $3k$ colors. At least one color that appears in AEDCB must be unique. If the unique color is in A, then EDCB uses at most $3k-1$ colors; if in B, then AED uses at most $3k-1$ colors; if in C, then AED uses at most $3k-1$ colors; if in D, then ACB uses at most $3k-1$ colors; if in E, then ACB uses at most $3k-1$ colors. However, all of EDCB, AED, ACB appear at some point in the coloring of π^k. For AED, ACB, since the π^{k-1} components are inserted in the 132 order, an argument along the lines of the proof of proposition 2.4 gives a π^{k-1} component using at most $3k-3$ colors; a contradiction. For EDCB, the π^{k-1} components are inserted in the 4321 order: One of them has a unique color, which leaves at least two consecutive π^{k-1} components using at most $3k-2$ colors. One of these two consecutive components has a unique color, and thus the other component uses at most $3k-3$ colors; a contradiction to the inductive hypothesis. □

It is an open problem whether other permutations can improve on the $3\log_5 n+1$ lower bound. We remark that the above lower bound applies to all three dynamic models.

Upper bound The dynamic offline case can be viewed as a static problem, because dynamically coloring the sequence $\{H_t\}_{t=1}^n$ is equivalent to statically coloring the hypergraph $H=(V,\bigcup_{t=1}^n E_t)$, where E_t is the hyperedge set of H_t. In Even et al. [2003], a general framework for conflict-free coloring is presented: The authors provide an algorithm (Algorithm 1 in the paper) that

colors the points in iterations. At the ℓ-th iteration, some points are colored with color ℓ and these colored points are not considered in the subsequent iterations. To simplify the presentation, we do not consider singleton hyperedges, since they are conflict-free colored for free. The notion of a Delaunay graph is useful.

Definition 2.6. The *Delaunay graph* $G(H)$ of the hypergraph $H = (V, E)$ is the graph (V, D), where the edge set $D \subseteq E$ is the subset of all hyperedges in E of cardinality two.

For example, the hypergraph

$$H = (\{1, 2, 3, 4, 5\}, \{\{1, 2\}, \{2, 3\}, \{1, 2, 3\}, \{4, 5\}\})$$

has the Delaunay graph

$$G(H) = (\{1, 2, 3, 4, 5\}, \{\{1, 2\}, \{2, 3\}, \{4, 5\}\}).$$

We set $E = \bigcup_{t=1}^{n} E_t$, and adapt the framework of Even et al. [2003] to get the algorithm in figure 2.3.

Our algorithm in figure 2.3 is slightly different from the one in Even et al. [2003]. When the algorithm in Even et al. [2003] computes $E^{\ell+1}$ from E^{ℓ}, it does not consider hyperedges for which $|e \cap V^{\ell}| = 1$, since they are for sure conflict-free colored. On the other hand, our algorithm includes those hyperedges in the subsequent iteration. We made the above modification

$\ell \leftarrow 1$; $V^1 \leftarrow V$; $E^1 \leftarrow E$
while $V^\ell \neq \emptyset$ do:
    $I^\ell \leftarrow$ an independent set of the Delaunay graph of (V^ℓ, E^ℓ)
    color every v in I^ℓ with color ℓ
    $V^{\ell+1} \leftarrow V^\ell \setminus I^\ell$
    $E^{\ell+1} \leftarrow \{e \cap V^{\ell+1} \mid e \in E^\ell\}$
    $\ell \leftarrow \ell + 1$

Figure 2.3: Algorithm for the dynamic offline model

because the hypergraph (V^ℓ, E^ℓ) arises by inserting the points in V^ℓ in their time order and considering all possible intervals at each time. With this observation, it is easier to argue about the Delaunay graph of (V^ℓ, E^ℓ), and the key to get an efficient conflict-free coloring is to find a big independent set in the Delaunay hypergraph. Also, it is easier to implement our algorithm. The correctness of our algorithm is an immediate consequence of the correctness of the algorithm in Even et al. [2003], since ours just conflict-free colors more hyperedges. In order to bound the number of colors used, the following lemma is needed.

Lemma 2.7. *In the case of intervals, the Delaunay graph of* (V^ℓ, E^ℓ) *is* 3*-colorable.*

Proof. Consider the vertices in V^ℓ ordered according to the time of insertion. Each appearing vertex is immediately adjacent to at most two other vertices, and thus it can be colored greedily by one of three given colors. □

Corollary 2.8. *In a 3-coloring of the Delaunay graph of* (V^ℓ, E^ℓ)*, the largest color class is an independent set of size at least* $\lceil |V^\ell|/3 \rceil$.

Theorem 2.9. *There is an algorithm that uses at most* $\log_{3/2} n + 1$ *colors in the dynamic offline model.*

Proof. Apply the algorithm in figure 2.3, where the independent set chosen at each iteration ℓ is the largest size color class of any 3-coloring of the Delaunay graph of (V^ℓ, E^ℓ). By corollary 2.8, at each iteration at least $\lceil |V^\ell|/3 \rceil$ are colored. Therefore, the number of iterations (and thus the number of colors) is bounded by $\log_{3/2} n + 1$. □

Remark 2.10. Recently, Chan et al. [2008] gave an algorithm that colors points on the plane with respect to unit disks in the dynamic offline model we introduced.

2.4 Absolute positions model

In this section we present an algorithm that uses $O(\log n)$ colors in the absolute positions model. Roughly speaking, a point p with a unique color in an interval acts as a *separator*: Points to the left of p and points to the right of p can be colored independently, and colors can be freely reused. Our algorithm uses only a logarithmic number of colors by choosing the correct points as separators.

In a preliminary version of this work (see Bar-Noy et al. [2006]), we presented an algorithm that chooses the separators in each level and uses recursion to independently color the left and the right side of the separators. Here, we present an algorithm that adopts the opposite approach of coloring two

thirds of the points with a greedy non-recursive scheme in each level and of coloring the separators by using the recursion. The algorithm we present here is better in terms of the number of colors it uses (asymptotically $1.89 \lg n$) in the worst case, compared to the algorithm of Bar-Noy et al. [2006] ($2 \lg n$ colors).

2.4.1 An algorithm that uses $3\lceil \log_3 n \rceil$ colors

We provide a recursive algorithm in the absolute positions model that uses $3\lceil \log_3 n \rceil \approx 1.89 \lg n$ colors to color any input of size n. Triples of points with consecutive positions play a major role in the algorithm and therefore we call it the *triples* algorithm.

To prove the above bound, it suffices to show a method of conflict-free coloring any input of size 3^k with $3k$ colors, because, in that case, if $3^{k-1} < n \leq 3^k$ then the algorithm takes the n-sized input, attaches (in any insertion order) $3^k - n$ dummy points to the right of the n points, solves the 3^k-sized instance with the method to get a conflict-free coloring with $3k$ colors, and then discards the colors of the dummy points to get a conflict-free coloring of the original n points.

If $n = 3^k$, points are colored in k levels that correspond to recursion call levels of the algorithm and each level uses three colors. At each level $\ell \in \{1, \ldots, k\}$, some of the points are colored and the rest are deferred for coloring at a higher level. More precisely at each level ℓ, with $\ell < k$, two thirds of the points are colored in that level and the rest (one third) are

		points		
level	input size	colored	deferred	colors used
1	3^k	$2 \cdot 3^{k-1}$	3^{k-1}	1, 2, 3
...	...	...	...	...
ℓ	$3^{k+1-\ell}$	$2 \cdot 3^{k-\ell}$	$3^{k-\ell}$	$3\ell-2$, $3\ell-1$, 3ℓ
...	...	...	...	...
$k-1$	9	6	3	$3k-5$, $3k-4$, $3k-3$
k	3	3	0	$3k-2$, $3k-1$, $3k$

Table 2.3: Recursion levels of the triples algorithm

deferred. Thus, for each level $\ell < k$ of the recursion, out of the $3^{k+1-\ell}$ points that reach the level, $2 \cdot 3^{k-\ell}$ are colored in that level and $3^{k-\ell}$ are deferred for coloring in a higher level. The final level k is special because all three points that reach it are colored in that level. This situation is shown in table 2.3, where, by convention, level ℓ uses colors $3\ell-2$, $3\ell-1$, and 3ℓ.

Now, we describe how the algorithm decides at each level which points to color and which to defer. At each level ℓ, with $\ell < k$, the algorithm partitions the points in triples, according to their absolute positions: The three leftmost points are in the first triple, the second three leftmost points are in the second triple, and so on, until the final triple which contains the three rightmost points. The decision at each level ℓ is made as follows.

> For every triple, the first point that is requested to be colored in the triple is deferred for coloring in a higher level, whereas the other two points are colored at level ℓ.

Also, for $\ell < k$, the input at level ℓ, which we denote by $\pi_{\langle \ell \rangle}$, induces an input

$\pi_{\langle \ell+1 \rangle}$ at level $\ell+1$ as follows: The absolute positions of triples at level ℓ give the absolute positions of points and points at level $\ell+1$ are requested in the same order as the first points of triples at level ℓ. Initially, the input at level 1, i.e., $\pi_{\langle 1 \rangle}$, is set equal to the original input π. For example, consider the input $\pi = 923745618$, revealed to the online algorithm, one by one element, from left to right. In order to better illustrate how the algorithm runs, we take the inverse permutation of π, which maps absolute positions of points to the time they are requested:

$$\pi^{-1} = \pi^{-1}_{\langle 1 \rangle} = \ 8\mathbf{2}3\ \mathbf{5}67\ 49\mathbf{1}\ .$$

This is the input for level 1 (as denoted by the subscript) and we have also highlighted the first (i.e., earliest) point requested in every triple. Input π induces the following input for level 2: $\pi^{-1}_{\langle 2 \rangle} = 231$, or $\pi_{\langle 2 \rangle} = 312$.

The triples algorithm relies on a dynamic first-fit greedy coloring scheme, which colors every new point that appears with the smallest color such that the conflict-free property is maintained in all intervals. The following lemma proves useful.

Lemma 2.11. *Any conflict-free coloring of $x < 4$ points, can be extended to a conflict-free coloring of $x+1$ points, with at most three colors, for any position of the $x+1$-th point, by using the dynamic first-fit greedy coloring scheme.*

Proof. If the coloring of x points is using fewer than three colors, then the

greedy scheme introduces at most one new color. If the coloring of x points is using three colors, then it must be the case that $x = 3$ and the coloring looks like abc. In that case, the first-fit greedy scheme can color any new point by reusing the minimum color among the colors of non-adjacent points to the new point. □

Now, we explain how the algorithm colors points at a specific level. If a new point p is requested that is decided to be colored at level ℓ (i.e., not deferred for coloring at a higher level), the algorithm finds the set of all points P already colored at level ℓ with the following property: p' is in P, if there is no point deferred for coloring at a higher level between p and p'. It can be proven that P contains at most three points. We postpone the proof of the above statement, and include it later in the proof of the correctness of the algorithm, in proposition 2.12. The algorithm chooses the color of p using a greedy coloring scheme, i.e., by choosing the minimum color possible among $3\ell-2$, $3\ell-1$, and 3ℓ, so that the interval containing $P\cup\{p\}$ remains conflict-free (we proved in lemma 2.11 that this is always possible). The coloring at each level ℓ, corresponding to input $\pi_{\langle\ell\rangle}$, is denoted by $C_{\langle\ell\rangle}$; it is a partial coloring for $\ell < k$, because only two thirds of the points are colored.

The run of the algorithm on the input example mentioned above is shown in figure 2.4. Each '$*$' denotes a point that is to be colored at a higher level and C denotes the final coloring.

For example, the point colored at $t = 4$ gets color 2 (at level 1), because it is not the first point that appeared in its triple, and because there is a

$\pi^{-1}_{\langle 1\rangle} =$	8	2	3	5	6	7	4	9	1
$C_{\langle 1\rangle} =$	1	$*$	1	$*$	1	3	2	1	$*$
$\pi^{-1}_{\langle 2\rangle} =$		2		3					1
$C_{\langle 2\rangle} =$		5		6					4
$C =$	1	5	1	6	1	3	2	1	4

Figure 2.4: A run of the triples algorithm

point requested at $t = 3$ which is colored with color 1.

The correctness of the algorithm is immediate from the following proposition.

Proposition 2.12. *At any time $t \in \{1, \ldots, n\}$, in any interval I of points, there is a point in I colored with a unique color in I. Moreover, a uniquely colored point can always be found among the points that were colored in the deepest recursive level of points in I.*

Proof. Recall that the three leftmost points are in the first triple, the second three leftmost points are in the second triple, and so on, until the final triple which contains the three rightmost points. At some time t, we say that a triple is *empty* if no point of it has yet been requested to be colored. At any time t, take any interval I and consider the points in I that were colored at the highest level ℓ. If $\ell = k$, then these are at most three points in I and by lemma 2.11 there is a unique color in I. If $\ell < k$, then I can not span a whole non-empty triple in level ℓ, because it will include a point colored at a level higher than ℓ. Also, I can not span parts of more than two non-empty

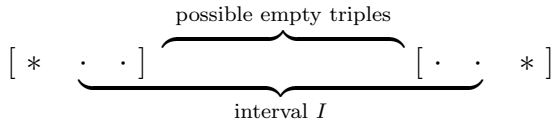

Figure 2.5: At most four points spanned at level $\ell < k$ by I

triples, because then it would span a whole triple between the two extreme triples. Thus, I spans at most two triples and at most two points in each of them (see figure 2.5). By the description of the algorithm these points are colored by a greedy scheme with colors from $3\ell - 2$, $3\ell - 1$, and 3ℓ, and thus by lemma 2.11 there is a unique color in I. □

Thus, we have established the following theorem.

Theorem 2.13. *In the dynamic online absolute positions model, there is a deterministic algorithm that uses at most* $3\lceil \log_3 n \rceil$ *colors.*

2.5 Coloring with respect to rays

We relax the conflict-free coloring problem of intervals as follows. Instead of the requirement that *all* intervals need to have a uniquely colored point, it is required that the conflict-free condition holds only for intervals in a specific *subset* of the set of all intervals. Examples are coloring with respect to all intervals of a specific length, say k, or all intervals of length up to k. Recently, the problem has been also studied by Katz et al. [2007], where approximation algorithms have been obtained; see section 4.1 for an algorithm with improved approximation ratio over the ones by Katz et al. [2007].

Another interesting case arises from the intervals that contain either of the two extreme points. Equivalently, these are intervals that are defined by halflines (infinite intervals), or *rays*. We therefore refer to the problem as conflict-free coloring with respect to *rays*. The motivation for considering this restricted subset comes from agents whose movement range is not strictly inside the line segment between the two extreme points. We also want to point out how different are the results and the gaps between models, compared to the all intervals case.

For n points there are $2n - 1$ ray defined intervals, of which n contain the leftmost point and are called *prefix* intervals and n contain the rightmost point and are called *suffix* intervals (the interval containing all points is both a prefix and a suffix interval).

In the static model, the coloring $133 \ldots 332$ (i.e., color the extreme points with unique colors and use the same color for all non-extreme points) suffices for all n and uses three colors. It is not hard to see that three colors are required for $n \geq 4$ (for $n = 3$ the coloring 121 with two colors is a conflict-free coloring).

To analyze the problem in the dynamic models, we consider first coloring with respect to *prefix intervals* only (the suffix case has the same bounds, because it is symmetric). In the static model for prefixes, the coloring $122 \ldots 22$ is a conflict-free coloring with 2 colors. Obviously, this coloring is optimal. In the dynamic models for prefixes, we will first prove a lower bound of $1 + \lfloor \lg n \rfloor$ even for the dynamic offline model and then provide an algorithm

using $2 + \lfloor \lg(n-1) \rfloor$ colors already in the relative position model.

Proposition 2.14. *In the dynamic (offline) model, input $\sigma = 0^n$ needs $1 + \lfloor \lg n \rfloor$ colors to be conflict-free colored with respect to prefixes.*

Proof. By viewing the dynamic problem as a static problem (as we did in the upper bound discussion of section 2.3), it can be proven that coloring 0^n with respect to prefixes is equivalent to coloring n points statically with respect to (all) intervals. This is because the point inserted at $t = i$ is always at the left of all previously inserted points, contributing i new intervals, and therefore, after all n points have been inserted, all $n(n+1)/2$ intervals are defined. Thus, at least $1 + \lfloor \lg n \rfloor$ colors are needed. □

We propose the following algorithm for coloring prefixes: The algorithm colors differently

(a) points that appear to the left of all previously inserted points,

(b) points that appear to the right of at least one previously inserted point.

The first group of points contains points for which $\sigma_{(i)} = 0$, and the second group points for which $\sigma_{(i)} > 0$. Therefore, it is possible to distinguish between the two groups even in the relative positions model. Points in the first group are colored according to static coloring for intervals: ...41213121. Points in the second group are all colored with the same color, which is different from colors used in the first group. For example, input $\sigma = 010120020$

is colored as $131\star2\star\star1\star$, where '$\star$' is the color used for points in the second group. It is not difficult to see that subsets of points which must have the conflict-free property never contain a '$\star$'-colored point as their leftmost point. Thus, every subset of points that must be conflict-free colored contains a point of the first group. Additionally, every such subset contains points with consecutive positions in the first group, and is therefore already conflict-free colored. If the input is 0^n, exactly $1 + \lfloor \lg n \rfloor$ colors are used. Otherwise, at least one point is '$\star$'-colored and at most $n-1$ points are in the first group, which implies that at most $2 + \lfloor \lg(n-1) \rfloor$ are used. We just proved the following.

Proposition 2.15. *For $n \geq 2$, there is an algorithm that conflict-free colors, with respect to prefixes, any insertion sequence σ (in the relative positions model) with at most $2 + \lfloor \lg(n-1) \rfloor$ colors.*

Finally, we use the upper bound for prefixes (and suffixes) to prove an upper bound for rays. We claim that for dynamically coloring with respect to rays, one more color than the prefix (or suffix) case suffices. The idea is to use a unique color for the first point p inserted, and then color independently points to the left of p and points to the right of p: color whatever is inserted to the left of p with respect to prefixes and whatever is inserted to the right of p with respect to suffixes. From the above discussion, the following result is immediate.

Proposition 2.16. *For $n \geq 3$, there is an algorithm that conflict-free colors,*

model	lower bound	upper bound
all dynamic	$1 + \lfloor \lg n \rfloor$	$3 + \lfloor \lg(n-2) \rfloor$
static	3	3

Table 2.4: Number of colors used in deterministic algorithms for rays for n at least 3

with respect to rays, any insertion sequence σ (in the relative positions model) with at most $3 + \lfloor \lg(n-2) \rfloor$ colors.

The above analysis gives a separation between static and dynamic models for coloring with respect to rays: The number of colors used is a logarithmic factor apart. All the results are shown in table 2.4. This is in contrast with the all-intervals case in which the separation result between static and dynamic offline model is weaker, just a constant factor apart, $1 + \lg n$ and $1 + \log_{3/2} n$ colors, respectively.

2.6 Discussion and open problems

We introduced a hierarchy of models for conflict-free coloring ranging from a completely static model (weakest adversary model) to a fully online model (strongest adversary model). We concentrated on deterministic conflict-free coloring with respect to intervals. For this special case, we proposed algorithms for some of the models and gave upper bounds on their worst-case performance. We also provided lower bounds on the number of colors used in some models.

There are still gaps between lower and upper bounds (see table 2.1). For example, in the dynamic offline model the lower bound is $1 + 2\log_3 n \approx 1.26 \lg n$, whereas the upper bound is $1+\log_{3/2} n \approx 1.71 \lg n$, a constant factor apart. The situation is similar in the absolute positions model where the upper bound is approximately $1.89 \lg n$. The most important open problem is narrowing the gap between lower and upper bound in the relative positions model: $\Omega(\log n)$, and $O(\log^2 n)$, respectively, which are a logarithmic factor apart.

So far in the literature, only the static and the fully online (relative positions) models had been considered. If there is a gap on the number of colors used between these two extreme models, the hierarchy can help pin-point exactly where the 'jump' occurs, and thus give a better understanding of the problem. In the case of all-intervals, static uses $O(\log n)$ and the best known online deterministic algorithm $O(\log^2 n)$ colors, but this logarithmic factor 'jump' is not a result of the online model, because it occurs just between the absolute positions model and the fully online (relative positions) model. However, in the rays case, a logarithmic factor jump occurs between the static and the dynamic offline model.

In the dynamic online absolute positions setting, the algorithm has access to the total number n of points that will be inserted, from the start. The triples algorithm exploits this information to achieve $O(\log n)$ colorings. However, in the triples algorithm, for final size of input $n = 3^k$, the adversary can request the first k points in such a way such that the algorithm uses

k different colors. An open problem in the absolute positions model is to maintain an $O(\log k)$ coloring after the first k points have been inserted for all k with $0 < k \leq n$.

Finally, the hierarchy of models is not limited to problems for points on the real line. It can be used in conflict-free coloring for hypergraphs, in general. A possible use of the hierarchy would be to better understand conflict-free coloring problems in the plane, or even in higher dimensions.

Chapter 3

Online framework for conflict-free coloring hypergraphs

We provide a framework for online conflict-free coloring any hypergraph. We use this framework to obtain an efficient randomized online algorithm for conflict-free coloring any k-degenerate hypergraph. Our algorithm uses $O(k \log n)$ colors with high probability and this bound is asymptotically optimal, because there are families of k-degenerate hypergraphs that need that many colors. 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 conflict-free coloring some hypergraphs that arise in geometry. Our algorithm uses exponentially fewer random bits than pre-

vious algorithms.

We introduce deterministic online conflict-free coloring algorithms for points on the line with respect to intervals and for points on the plane with respect to halfplanes (or unit disks) that use $\Theta(\log n)$ colors and recolor $O(n)$ points in total.

3.1 Introduction

An online conflict-free coloring framework. In this section, we investigate the most general form of online conflict-free coloring applied to arbitrary hypergraphs. Suppose the vertices of an underlying hypergraph $H = (V, E)$ 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 3.2). 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 conflict-free coloring any hypergraph. Interestingly, this framework is a generalization of some known coloring algorithms. For example the Unique-Max Algorithm of Fiat et al. [2005] can be described as a special case of our framework. Also, when the underlying hypergraph is a simple graph then the first-fit greedy online al-

gorithm 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 Fiat et al. [2005], the problem of online conflict-free coloring the very special hypergraph induced by points on the real line with respect to intervals is highly non-trivial. The best randomized online conflict-free coloring algorithm of Fiat et al. [2005] uses $O(\log n \log \log n)$ colors. Kaplan and Sharir [2004] studied the special hypergraph induced by points in the plane with respect to halfplanes and unit disks and obtained a randomized online conflict-free coloring with $O(\log^3 n)$ colors with high probability. Recently, the bound $\Theta(\log n)$ just for these two special cases was obtained by Chen [2006] (see also Chen et al. [2007]). We also obtained independently an algorithm which is more general and uses only $\Theta(\log n)$ colors; an interesting evidence to our algorithm being fundamentally different from the ones in Chen [2006], Fiat et al. [2005], Kaplan and Sharir [2004], when used for the special case of hypergraphs that arise in geometry, is that our algorithm uses exponentially fewer random bits. The algorithms of Chen [2006], Kaplan and Sharir [2004] use $\Theta(n)$ random bits and 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 [1994] who showed that a greedy first-fit algorithm achieves the same bound deterministically.

Deterministic online conflict-free coloring with recoloring. We initiate the study of online conflict-free 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 conflict-free 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. [2005] that does not recolor points. We also show online algorithm for conflict-free 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.

From the application point of view, there is motivation to study this recoloring model. The frequency spectrum is quite expensive, so a solution which strictly uses a logarithmic number of colors is desirable. On the other hand excessive recoloring is not desirable, because if a base station is given another color there is a disruption of service for all agents connected to it.

Organization. In section 3.2 we define the notion of a k-degenerate hypergraph. In section 3.3 we present the general framework for online conflict-free coloring of hypergraphs. In section 3.4 we introduce the randomized algorithm derived from the framework. In section 3.5 we show deterministic online algorithms for intervals and halfplanes with recoloring. In section 3.6 we describe the results for the hypergraphs that arise from geometry. Finally, in section 3.7 we conclude with a discussion and some open problems.

3.2 Preliminaries

We will use the notion of *induced hypergraph* given in definition 2.1 on page 27 and the notion of *Delaunay graph* of a hypergraph given in definition 2.6 on page 40. We give some more definitions.

Here is a graph theoretic common definition:

Definition 3.1. 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.

We sensibly extend to a similar definition for hypergraphs.

Definition 3.2. Let $k > 0$ be a fixed integer and let $H = (V, E)$ be a hypergraph on the n vertices $v_1, \ldots, v_n$. For a permutation $\pi\colon \{1, \ldots, n\} \to$

$\{1, \ldots, n\}$ define the n partial sums, indexed by $t = 1, \ldots, n$,

$$S_t^\pi = \sum_{j=1}^{t} d(v_{\pi(j)}),$$

where

$$d(v_{\pi(j)}) = \left|\{i < j \mid \{v_{\pi(i)}, v_{\pi(j)}\} \in G(H(\{v_{\pi(1)}, ..., v_{\pi(j)}\}))\}\right|,$$

that is, $d(v_{\pi(j)})$ is the number of neighbors of $v_{\pi(j)}$ in the Delaunay graph of the hypergraph induced by $\{v_{\pi(1)}, ..., v_{\pi(j)}\}$. Assume that for all permutations π and for every $t \in \{1, \ldots, n\}$ we have

$$S_t^\pi \leq kt. \tag{3.1}$$

Then, we say that H is *k-degenerate*.

3.3 A framework for online conflict-free coloring

Let $H = (V, E)$ be any hypergraph. Our goal is to define a framework that colors the vertices V in an online fashion, i.e., when 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 future times $t' > t$. The coloring has to be conflict-free for all

the induced hypergraphs $H(V_t)$ with $t = 1, \ldots, n$, where $V_t \subseteq V$ 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 *main* colors used for the conflict-free 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 with row entries indexed by the variable i with range in $\mathbb{N}^+$. Each row entry i at time t is associated with a subset $V_t^i \subseteq V_t$ in addition to an auxiliary proper non-monochromatic 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 non-monochromatic coloring of $H(V_{t-1}^1 \cup \{v_t\})$. Any (proper non-monochromatic) coloring procedure can be used by the framework. For example a first-fit greedy method in which all colors in the order $a_1, \ldots, a_h$ 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 conflict-free 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 $f(1)$, then 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 $f(i)$. Then, the color of v_t in the final conflict-free coloring is 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. An example instantiation of the framework for conflict-free coloring with respect to intervals is given in example 3.14 on page 84. In section 3.4 we derive a randomized online algorithm based on this framework. This algorithm always halts, or to be more precise halts with probability 1, and moreover it halts after a 'small' number of entries with high probability. We prove that the above framework produces a valid conflict-free coloring in case it halts.

Lemma 3.3. *If the above framework halts for any vertex v_t then it produces a valid online conflict-free coloring of H.*

Proof. Let $H(V_t)$ be the hypergraph induced by the vertices already 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 non-monochromatic coloring of the induced hypergraph at entry j, $S \cap V_t^j$ is not monochromatic so there must exist 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 3.5 then the algorithm derived from the framework becomes identical to the Unique Maximum Greedy algorithm described and analyzed in Fiat et al. [2005].

- 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 the first-fit greedy algorithm for restricted classes of graphs has been analyzed in several papers

[Gyárfás and Lehel, 1988, Kierstead, 1988, Irani, 1994]. Especially for k-inductive graphs, the first-fit greedy algorithm is analyzed by Irani [1994], who proved that it uses $O(k \log n)$ colors. Our framework can be used to give an alternative simpler proof of the aforementioned result (see Smorodinsky [2008] for details).

3.4 An online randomized conflict-free coloring algorithm

There is a randomized online conflict-free coloring algorithm 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 the following theorem.

Theorem 3.4. *Let $H = (V, E)$ be a k-degenerate hypergraph on n vertices. Then, there exists a randomized online conflict-free coloring algorithm for H which uses at most $O(\log_{1+\frac{1}{4k+1}} n) = O(k \log n)$ colors with high probability against an oblivious adversary.*

The algorithm is based on the framework of section 3.3. 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), denoted by $X^t_{\geq i}$, have been actually given some auxiliary color at entry i (that is, $|V^i_t| \geq \frac{1}{2}\left|X^t_{\geq i}\right|$). This is due to 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 $V^i_t \subseteq X^t_{\geq i}$). Therefore since we have $2k+1$ colors available, there is always an available color to assign to such a vertex. The following lemma shows that if we use one of these available colors then the updated coloring is indeed a proper non-monochromatic coloring of the corresponding induced hypergraph as well.

Lemma 3.5. *Let $H = (V, E)$ 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 non-monochromatic 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 non-monochromatically 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_{t-1}^j. However, we assumed that v_t obtained a color that is distinct from any vertex u such that $\{u, v_t\}$ is a hyperedge of $H(V_t^j)$ (Those are exactly the neighbors of v_t in the corresponding Delaunay graph). Thus, any such hyperedge $\{u, v_t\}$ is also not monochromatic. This completes the inductive step and hence the proof of the lemma. □

We also prove that for every vertex v_t, our algorithm always halts, or more precisely halts with probability 1.

Proposition 3.6. *For every vertex v_t, the algorithm halts with probability 1.*

Proof. In order for the framework to not halt for vertex v_t, it must be the case that vertex v_t reaches every entry $i \in \mathbb{N}^+$ and in every entry i the auxiliary color of v_t is different from $f(i)$. If an entry is empty at time t and v_t reaches that entry, then the probability that v_t does not get a main color in that entry is $1 - h^{-1}$, where $h = 2k + 1$ is the number of auxiliary colors. The aforementioned events are independent for empty entries. At time t, all but at most $t - 1$ entries are empty. The above discussion implies the following.

$$\Pr[\text{algorithm does not halt for } v_t] \leq$$

$$\Pr[\text{algorithm does not color } v_t \text{ in empty entries}] =$$

$$\Pr[\bigcap_{i\text{: empty entry}} (\text{algorithm does not color } v_t \text{ in entry } i)] =$$

$$\prod_{i\text{: empty entry}} \Pr[\text{algorithm does not color } v_t \text{ in entry } i] =$$

$$\prod_{i\text{: empty entry}} (1-h^{-1}) = \lim_{j\to\infty}(1-h^{-1})^j = 0$$

and therefore

$$\Pr[\text{algorithm halts for } v_t] = 1. \qquad \square$$

We proceed to the analysis of the number of colors used by the algorithm, proving theorem 3.4.

Lemma 3.7. *Let $H=(V,E)$ be a hypergraph and let C 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 statement holds 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 $\frac{1}{2k+1}$ to get the final color i. This is the only place where we need to assume that the adversary is oblivious and does not have access to the random bits. 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 3.8. *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 main 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,

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

by Markov's inequality and lemma 3.7. □

We notice that:

$$b(k,n) = \frac{\ln n}{\ln \frac{4k+2}{4k+1}} \leq (4k+2)\ln n = O(k \log n).$$

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 3.7.

This completes the performance analysis of our algorithm.

Remark 3.9. 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 $O(k \log n)$ with high probability then the total number of random bits used by our algorithm is $O(k \log k \log n)$ with high probability.

3.5 Deterministic online algorithms with recoloring

In this section, we relax the requirement that an online algorithm has to commit to the color of every point, by allowing the algorithm to recolor a 'few' of the points that have appeared in the past. Our goal is to find deterministic online algorithms that use a logarithmic number of colors and perform a total number of recolorings which is linear in n. We manage to find such algorithms with respect to intervals and halfplanes. The algorithm for halfplanes relies on an algorithm that colors points on a disk with respect to circular arcs, where the adversary can additionally ask the algorithm to substitute a set of consecutive points on the disk with a single point (we call this a substitution move). As always, the coloring must remain conflict-free at all times.

3.5.1 An $O(\log n)$ colors algorithm for intervals

We describe a deterministic online conflict-free coloring algorithm for intervals that is allowed to recolor just a single old point during each insertion of a new point. The algorithm is based on the framework developed in section 3.3 where we use 3 auxiliary colors $\{a, b, c\}$ and f is the constant function $f(l) = a$, for every l. We refer to points colored with b or c as d-points. In order to have only a logarithmic number of entries, we slightly modify the framework (using a recoloring procedure) such that the number of points col-

ored 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 obtained an auxiliary color and assign to it the main color of q, and we recolor q with the auxiliary color a (and delete the corresponding appearance of it in all higher entries of he table), and thus we recolor q with the main color of the current entry.

At this point all points have an assignment of main 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.

In the following, we give an example run of the recoloring algorithm.

Example 3.10. An example run of the recoloring algorithm is shown in figure 3.1 for input $\pi = 3754612$. Vertex v_t appears at time t, where t ranges from 1 to 7. The first row of the table represents the order in which points appeared, the last row of the table shows current color allocation. 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.

Recolorings happen at $t = 3$ for v_2, at $t = 5$ for v_3, and at $t = 7$, for v_6.

It can be easily checked that the recoloring algorithm produces a valid conflict-free coloring, because it is essentially an instance of the general framework: After every insertion (and a possible recoloring), the point of highest entry in each interval is uniquely colored.

Also, it can be proven that the number of recolorings is at most $n - (\lfloor \lg n \rfloor + 1)$, and this is tight.

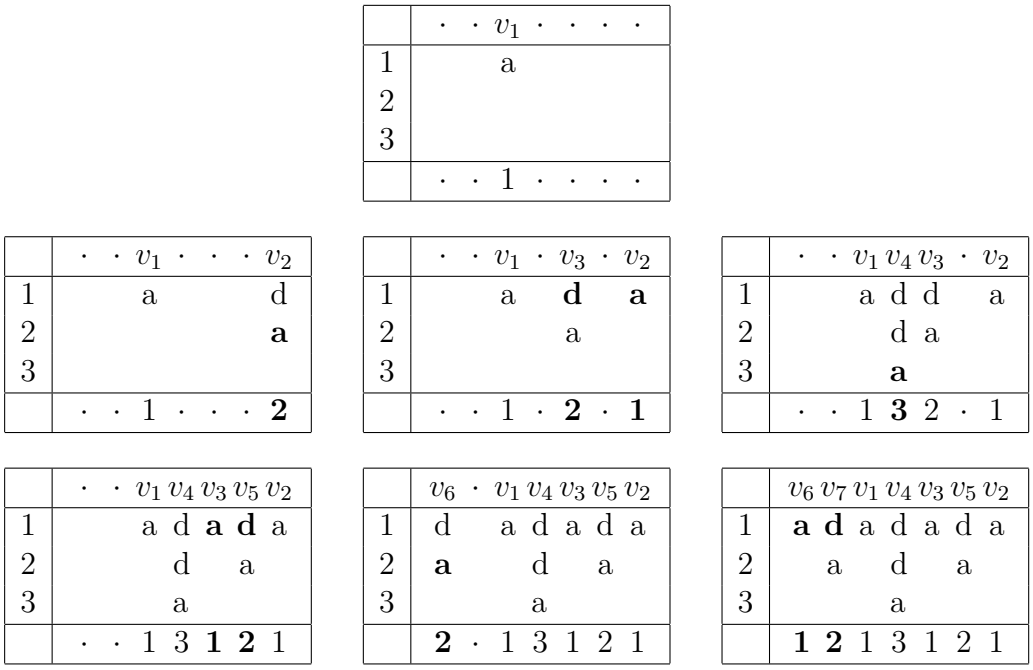

Figure 3.1: An example run of the recoloring algorithm

Proposition 3.11. *The number of recolorings in the above algorithm equals* $n - (\lfloor \lg n \rfloor + 1)$ *in the worst case.*

Proof. An input with n vertices uses at least $\lfloor \lg n \rfloor + 1$ colors (see optimal static coloring of points with respect to intervals in section 2.1). Whenever a new color is introduced during the run of the algorithm, there is no recoloring. Therefore, there are at most $n - (\lfloor \lg n \rfloor + 1)$ recolorings, because in every other insertion at most one old point is recolored.

Now, we are going to show a family of instances for which the above algorithm performs exactly $n - (\lfloor \lg n \rfloor + 1)$ recolorings. We use the relative positions notation for the input, that we introduced in section 2.2 on page 32.

We define, for $k \geq 1$, an instance σ^k of length $n = 2^k - 1$ for which

our recoloring algorithm uses k colors and does $2^k - k - 1$ recolorings. The instance $\sigma^1 = 0$. For $k \geq 1$, the instance σ^{k+1} is defined recursively.

$$\sigma^{k+1} = \sigma^k \circ \underbrace{(2^k - 1, \ldots, 2^k - 1)}_{2^k \text{ times}}$$

where '$\circ$' is the concatenation operation for finite sequences (also mentioned in chapter 2). Since, for every k, σ^k is a prefix of σ^{k+1}, we have in fact provided an unbounded length relative positions input

$$\sigma = \underbrace{2^0 - 1}_{2^0}, \underbrace{2^1 - 1, 2^1 - 1}_{2^1}, \underbrace{2^2 - 1, \ldots, 2^2 - 1}_{2^2}, \ldots, \underbrace{2^k - 1, \ldots, 2^k - 1}_{2^k}, \ldots$$

or

$$\sigma = 0, 1, 1, 3, 3, 3, 3, 7, 7, 7, 7, 7, 7, 7, 7, \ldots$$

The following can be proven by induction and we omit the easy but tedious details. For each σ^k, the recoloring algorithm produces the coloring C^k, defined recursively as $C^1 = 1$ and $C^k = C^{k-1} \circ (k) \circ C^{k-1}$, for $k > 1$. Therefore for $t < 2^k$, input σ is using at most k colors. The point inserted at $t = 2^k$, which is the first point of σ^{k+1} (or σ) that is inserted at relative position $2^k - 1$, is colored with a new color $k + 1$, and therefore no recoloring happens. For all subsequent $2^k - 1$ points inserted at relative position $2^k - 1$, there is a recoloring by the algorithm. For example, the run of the recoloring algorithm on input σ^3 is shown in figure 3.2, where recolorings are shown with bold.

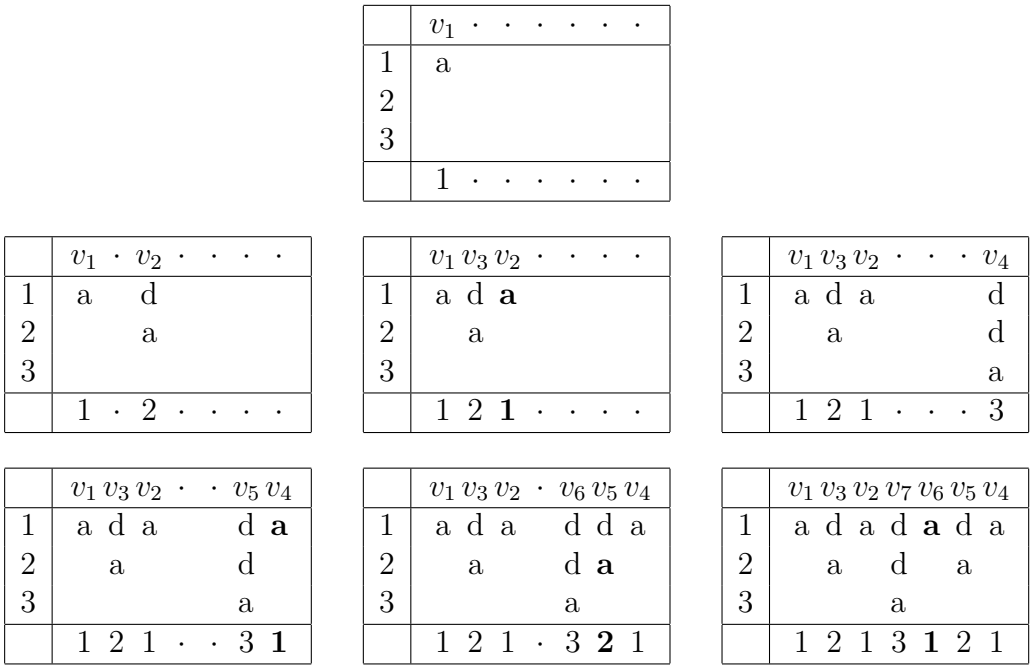

Figure 3.2: The run of the recoloring algorithm on input σ^3

Therefore, for all points, except the ones inserted at $t = 1, 2, 4, \ldots, 2^k, \ldots$ a recoloring happens, and therefore after n insertions, $n - (\lfloor \lg n \rfloor + 1)$ recolorings happen in σ. □

3.5.2 An $O(\log n)$ colors 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 finite 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 is not difficult to show that n points can be optimally conflict-free colored with respect to circular arcs with $\lfloor \lg(n-1) \rfloor + 2$ colors: There must be a point p with unique color in P, and

therefore all circular arcs that include p have the conflict-free property; the remaining $n-1$ points of $P \setminus \{p\}$ and the remaining circular arcs induce the same hypergraph as the set of intervals on $n-1$ points, which is optimally colored with $\lfloor \lg(n-1) \rfloor + 1$ more 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 to 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 3.5.1, 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 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.

3.5.3 An $O(\log n)$ colors 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 3.5.2. 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 *consec-*

utive 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 new 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 3.5.2. For an insertion move of the adversary, it colors the new point like in section 3.5.2. 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:

Lemma 3.12. *After t moves, the above coloring algorithm uses at most $\log_{3/2} t + 1$ colors.*

Proof. During a substitution move we might break the invariant that between any pair of consecutive a's there are at most two d-points. However if we denote in each entry a point colored with a which was substituted by $\bar{a}$, then it can be proven that between any two consecutive points colored with a or $\bar{a}$, there are at most two d-points and thus it implies that at least one third of the points in every level are colored either by that level or have been substituted. We call these points colored with $\bar{a}$ *ghost* points. Moreover, we

assign ghost points to substitution points as follows. If a point p substitutes a point p' colored with a, p' becomes a ghost point and p is assigned the ghost point p'. If a point p substitutes a point q which has some ghost points, p is assigned all ghost points of q. We ignore the trivial substitution of one point colored with a and do not create a ghost point and any assignment in this case. It is not difficult to see that at any point in time each ghost point is assigned to exactly one non-ghost point.

We intend to make the above argument formal as follows. We will prove the stronger result that the number of colors used by the algorithm is at most $\log_{3/2} i + 1$, where i is the number of insertion moves until time t. In order to prove the previous statement it is enough to show that at each entry ℓ, the number of points getting auxiliary color d in entry ℓ is bounded by the number of insertion moves that reached entry ℓ as follows.

$$d_\ell \leq \lceil \tfrac{2}{3} i_\ell \rceil \tag{3.2}$$

where d_ℓ is the number of points getting auxiliary color d in entry ℓ and i_ℓ is the number of insertion moves that reached entry ℓ. The above inequality is true when no points have reached entry ℓ. Moreover, it remains true as long as a substitution move happens, or an insertion move happens in which the point at entry ℓ is colored with a. The number of d's increases only if there is an insertion move where the point at level ℓ is colored with d. We will study further this last case. For a new point p to get auxiliary color d

it must be the case that it is inserted next to a point colored with a and a point q colored with d such that q is adjacent to a point colored with a. In a fixed entry ℓ, we call a maximal set of consecutive points colored with d a *strip*. The length of a strip s is the number of d's in it and is denoted by $\text{len}(s)$. It is not difficult to see that if there is at least one a in entry ℓ, as in our case, the number of strips is the same as the number of a's.

The number of insertion moves that reach entry ℓ satisfies the following equation.

$$i_\ell \geq a_\ell + d_\ell + \bar{a}_\ell \tag{3.3}$$

where a_ℓ is the number of points colored with a, and $\bar{a}_\ell$ the number of ghost points (points substituted that were colored with a). We have an inequality, because we omit the points substituted that were colored with d. If a strip s has length $\text{len}(s) > 2$, it necessarily contains ghost points. In fact if a strip s has length $\text{len}(s)$, one can prove that points in it have been assigned at least $\lceil \frac{1}{2}\text{len}(s) \rceil - 1$ ghost points. We defer the proof of the above fact to lemma 3.13. Because of all the above, inequality (3.3) implies the following.

$$i_\ell \geq a_\ell + \sum_{s\colon \text{strip}} \text{len}(s) + \sum_{s\colon \text{strip}} (\lceil \tfrac{1}{2}\text{len}(s) \rceil - 1) = \sum_{s\colon \text{strip}} \lceil \tfrac{3}{2}\text{len}(s) \rceil$$

The above inequality implies

$$\lfloor \tfrac{2}{3} i_l \rfloor \geq \lfloor \tfrac{2}{3} \sum_{s\colon \text{strip}} \lceil \tfrac{3}{2}\text{len}(s) \rceil \rfloor \geq \lfloor \sum_{s\colon \text{strip}} \text{len}(s) \rfloor = \sum_{s\colon \text{strip}} \text{len}(s) = d_\ell$$

which is inequality (3.2). □

Lemma 3.13. *The points in a strip s have been assigned at least $\lceil \frac{1}{2}\text{len}(s) \rceil - 1$ ghost points.*

Proof. We prove the above fact by induction on t. For $t = 0$ it is trivially true. For length of a strip at most two, again it is trivially true because $\lceil \frac{1}{2}\text{len}(s) \rceil - 1 = 0$. We ignore trivial substitutions of one point colored with a because they do not change the lengths of the strips and do not create ghost points. Assume there is a strip of length greater than two. Necessarily, the last action in the strip was a substitution move, because in an insertion the algorithm never colors with d, if there are already two d points in the strip. There are two possible cases for a substitution move.

In the first case, there is a substitution of only d points as shown in figure 3.3, i.e., the substitution is completely contained in one strip, say of length L', and the new strip created has length $L \leq L'$. In this case,

$$\overbrace{dddd\underbrace{dddd\ldots ddddd}_{\text{substitution}}dd}^{L'}$$

Figure 3.3: A substitution move contained in one strip

the number of ghost points in the new strip is the same as the number of ghost points in the old strip, which is, by the inductive hypothesis, at least $\lceil \frac{1}{2}L' \rceil - 1$, which is at least $\lceil \frac{1}{2}L \rceil - 1$.

In the second case, the substitution spans more than one strip, i.e., also some (non-ghost) points colored with a. Say that the substitution spans k

a's which are surrounded by $k+1$ strips of lengths $L_1, \ldots, L_{k+1}$, as shown in figure 3.4. The length of the new strip is $L \leq L_1 + L_{k+1} + 1$ if $k \geq 2$,

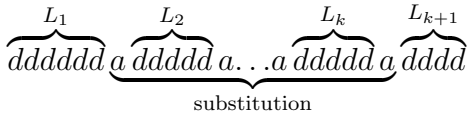

Figure 3.4: A substitution move spanning more than a strip

and $L \leq L_1 + L_2$ if $k = 1$ (this last inequality is true because there can be no trivial substitution). In this case, the number of ghost points in the new strip is the same as the number of ghost points in the $k+1$ strips plus k, which is at least

$$\sum_{i=1}^{k+1}(\lceil \tfrac{1}{2}L_i \rceil - 1) + k \geq \lceil \tfrac{1}{2}\sum_{i=1}^{k+1} L_i \rceil - 1 \geq \lceil \tfrac{1}{2}L \rceil - 1$$

In the above, we used the inductive hypothesis for each of the $k+1$ strips. □

3.5.4 An $O(\log n)$ colors algorithm for halfplanes

In this section we describe a deterministic algorithm for online conflict-free coloring points with respect to halfplanes that uses $O(\log n)$ colors and performs $O(n)$ recolorings. Thus, it can also be modified for conflict-free coloring points in the plane with respect to unit disks as described in section 3.6 (see proof of corollary 3.17). At every time instance t, the algorithm maintains the following invariant (V_t is the set of points that have appeared so far):

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.

Every non-empty 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. If one considers the points of $\mathrm{CH}(V_t)$ in their circular order on the convex hull, it is enough to conflict-free color them with respect to circular arcs. The number of colors used in $\mathrm{CH}(V_t)$ must be logarithmic in t.

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 3.5.3 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 $\mathrm{CH}(V_t)$ to appear in the interior of $\mathrm{CH}(V_{t+1})$, 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 3.5.3).

- 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 3.5.3, 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 '$\star$' keeps that color for the rest of the algorithm run.

3.6 Application to geometry

Our randomized algorithm has applications to conflict-free colorings of certain geometric hypergraphs studied in Chen [2006], Fiat et al. [2005], Kaplan and Sharir [2004], Chen et al. [2007]. We obtain the same asymptotic result as in Chen [2006] and Chen et al. [2007] but with better constant of proportionalities and using much fewer random bits. For example, if the hypergraph H is induced by intervals, it can be proven (with an analysis similar to the one given in section 3.4) 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 $\log_{3/2} n + 1$. 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 $\log_{3/2} n + 1$ (see Bar-Noy et al. [2006] and section 2.3 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 Chen [2006] and Chen et al. [2007] by

$1 + \log_{8/7} n \approx 5.19 \log_2 n$, three times our bound. We provide a run example for the algorithm on intervals.

Example 3.14. Consider the case where the hypergraph is induced by points with respect to intervals. Namely $V = (1, \ldots, n)$ and 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 not difficult to see that the hypergraphs $H(V_t^i)$ can always be properly non-monochromatically online 3-colored (say with auxiliary colors a, b, c) as follows: Each newly inserted point has at most two immediate neighbors and thus even a first-fit coloring suffices. In figure 3.5, 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 bottom 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 non-monochromatic auxiliary coloring of the vertices in the hypergraph V_t^i with three colors a, b or 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

i	$f(i)$	· v_1 · · · ·
1	**b**	a
2	**a**	**a**
3		
4		
5		
6		
		· **2** · · · ·

i	$f(i)$	· v_1 · · v_2 ·
1	b	a **b**
2	a	a
3		
4		
5		
6		
		· 2 · · **1** ·

i	$f(i)$	· v_1 v_3 · v_2 ·
1	b	a c b
2	a	a b
3	**c**	a
4	**a**	**a**
5		
6		
		· 2 **4** · 1 ·

i	$f(i)$	v_4 v_1 v_3 · v_2 ·
1	b	**b** a c b
2	a	a b
3	c	a
4	a	a
5		
6		
		1 2 4 · 1 ·

i	$f(i)$	v_4 v_1 v_3 · v_2 v_5
1	b	b a c b a
2	a	a b **a**
3	c	a
4	a	a
5		
6		
		1 2 4 · 1 **2**

i	$f(i)$	v_4 v_1 v_3 v_6 v_2 v_5
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**
		1 2 4 **6** 1 2

Figure 3.5: A run example of the framework for hypergraphs induced by points with respect to intervals

in the conflict-free 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 conflict-free 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.

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 with high probability. Before proceeding it is necessary to prove a degeneracy result about hypergraphs induced by halfplanes.

Lemma 3.15. *Let V be a finite set of n points in the plane and let E be all subsets of V that can be obtained by intersecting V with a halfplane. Then the hypergraph $H = (V, E)$ is 3-degenerate.*

Proof. We assume that points are in general position, i.e., no three of them are on the same line. We also assume that points are inserted in some order $v_1, v_2, \ldots, v_n$. Following the notation of definition 3.2 on page 59, it is enough to prove that for every t, we have

$$S_t \leq 3t \tag{3.4}$$

(we remark that we have dropped the permutation π, appearing in inequality (3.1) on page 60, because it is implied by the order $v_1, v_2, \ldots, v_n$). We

prove something stronger than inequality (3.4), namely that

$$S_t + C_t \leq 3t, \tag{3.5}$$

by induction, where C_t is the number of points on the boundary of the convex hull at time t, which is always a positive number. It will be helpful to define the following differences:

$$\Delta S_t = S_t - S_{t-1},$$
$$\Delta C_t = C_t - C_{t-1}.$$

The difference ΔS_t is exactly the number of neighbors of v_t in the Delaunay graph $G(H(\{v_1, \ldots, v_t\}))$. For v_t to be a neighbor of $v_{t'}$, with $t' < t$, in the Delaunay graph, there must exist a halfplane at time t which contains exactly v_t and $v_{t'}$.

First, we show that inequality (3.5) is true for small values of t. For $t = \{1, 2, 3\}$, inequality (3.5) is true as exhibited in table 3.1, because the size of the convex hull is the same as the number of points and every two points are neighbors in the Delaunay graph.

Then, for the inductive step, for $t > 3$, it is enough to prove that

$$\Delta S_t + \Delta C_t \leq 3, \tag{3.6}$$

because then the sum $S_t + C_t$ increases at most by 3 at every time step and

t	1	2	3
S_t	0	1	3
C_t	1	2	3
$S_t + C_t$	1	3	6

Table 3.1: Edges in Delaunay graph for halfplanes and size of convex hull for small values of t

therefore always remains bounded by $3t$. Denote the convex hull of points $\{v_1, \ldots, v_t\}$ with CH_t. Consider the following two cases. Either v_t lies outside of the convex hull CH_{t-1} or v_t is inside the convex hull CH_{t-1}.

Assume v_t lies outside of the convex hull CH_{t-1} (see figure 3.6). Then v_t

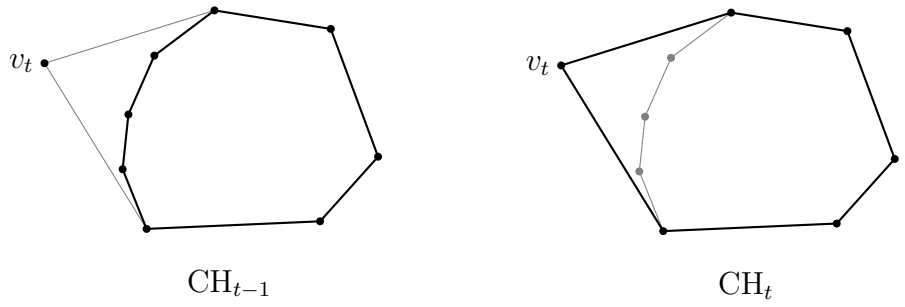

Figure 3.6: The new point is outside the old convex hull

lies on the boundary of the boundary of the convex hull CH_t. Consider the two points u and w that are the neighbors of v_t in the cyclic order of points on the convex hull CH_t (see figure 3.7). Consider the line ℓ passing through u and w. This line partitions points of CH_{t-1} in two types: (a) points on ℓ or in the same halfplane as v_t defined by ℓ (points like u, v', w in figure 3.7) and (b) points on the other halfplane defined by ℓ (points like v'' in figure 3.7). Vertex v_t is adjacent to every vertex v' of type (a) (including u, w) in the

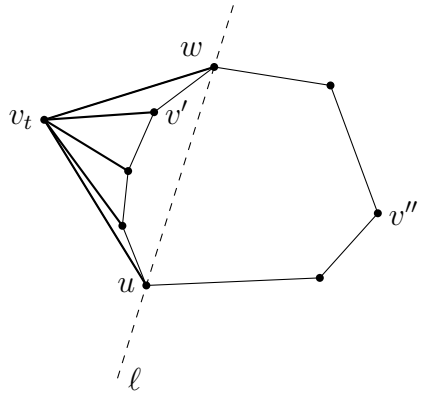

Figure 3.7: Delaunay graph neighbors of a new point outside the old convex hull

Delaunay graph, because one can take a halfplane with defining line passing through v' and slope between the slopes of the incident sides to v' of the convex hull CH_{t-1}, and this halfplane contains only v_t and v'. On the other hand, no vertex v'' of type (b) is a neighbor in the Delaunay graph with v_t, because every halfplane that contains v_t and v'' must contain at least one of u, w. Assume there are d vertices of CH_{t-1} of type (a), with $d \geq 2$. Then, $\Delta S_t = d$ and $d-2$ of them no longer appear on the convex hull, but v_t appears on CH_t, i.e., $\Delta C_t = -(d-2)+1$. Therefore, we have proved that when v_t lies outside CH_{t-1}, $\Delta S_t + \Delta C_t = d + -(d-2) + 1 = 3$, i.e., inequality (3.6) is true.

Assume v_t is inside the convex hull CH_{t-1} (see figure 3.8). Then, consider any triangulation of CH_{t-1}. Point v_t is in exactly one triangle of the triangulation, call it xyz, where x, y, z are points on the convex hull, corresponding to points inserted before v_t. It is not difficult to prove that every halfplane

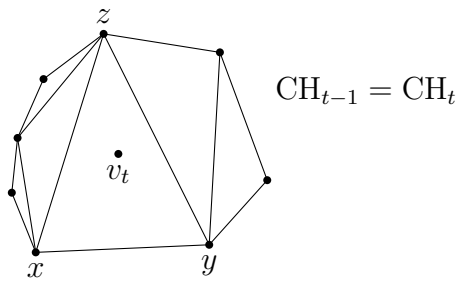

Figure 3.8: A triangulation of the convex hull in case the point v_t is in the convex hull CH_{t-1}

that contains v_t, contains at least one of x, y, z. Therefore v_t can have at most three neighbors in the Delaunay graph. The three neighbors case can be realized when the only points on the convex hull are x, y, z, i.e., when $t = 4$, by taking for every point $p \in \{x, y, z\}$ a halfplane that contains p, and the defining line of the halfplane (a) is passing through v_t and (b) is parallel to the line passing through the other two points in $\{x, y, z\}$. If there are more than three points in CH_{t-1}, we will prove that it is not possible for v_t to have all three neighbors x, y, z in the Delaunay graph. Assume for the sake of contradiction that there is a halfplane h_x containing only v_t and x, a halfplane h_y containing only v_t and y, and a halfplane h_z containing only v_t and z. For every point $p \in \{x, y, z\}$ define the halfline starting at v_t with direction $\overrightarrow{pv_t}$, not containing p. These halflines are shown in figure 3.9. These three halflines partition the plane into three areas, A_x, A_y, A_z, each one containing one of the points x, y, z, respectively. We now consider halfplanes containing at least v_t. It is not difficult to see that such a halfplane containing only x and not y or z must contain all of A_x. Similarly, such a

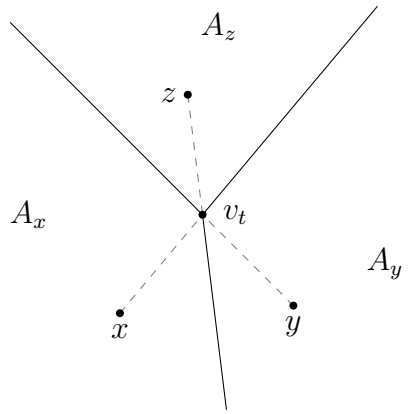

Figure 3.9: A partition of the plane

halfplane containing only y and not x or z must contain all of A_y, and such a halfplane containing only z and not x or y must contain all of A_z. Therefore, any other point contained in CH_{t-1} except x, y, z must be contained in one of h_x, h_y, and h_z, which is a contradiction. Thus, we have proved that when v_t is in CH_{t-1}, $\Delta S_t \leq 3$ and $\Delta C_t = 0$, i.e., inequality (3.6) is true. □

Corollary 3.16. *Let H be a hypergraph as in lemma 3.15. Then, the expected number of colors used by our randomized online conflict-free coloring algorithm applied to H is at most $\log_{\frac{14}{13}} n+1$, in the oblivious adversary model. 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 3.15, lemma 3.8 and theorem 3.4. □

Corollary 3.17. *Let V be a set of n points in the plane and let E be the set of all subsets of V that can be obtained by intersecting V with a unit disk.*

Then, there exists a randomized online algorithm for conflict-free coloring H which uses $O(\log n)$ colors and $O(\log n)$ random bits with high probability, against an oblivious adversary.

Proof. Kaplan and Sharir [2004] observed that by an appropriate partitioning of the plane one can modify any online algorithm for conflict-free coloring points with respect to halfplanes to an online algorithm for conflict-free coloring points with respect to congruent disks. The congruent disks algorithm uses a constant multiple of the colors used by the the halfplanes algorithm. Using the same technique as developed in Kaplan and Sharir [2004] and corollary 3.16 we obtain the desired result. □

3.7 Discussion and open problems

We presented a framework for online conflict-free 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 conflict-free 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 our notion is similar to the classical definition of degeneracy of a graph; see definition 3.1). Specifically, if the hypergraph is k-degenerate then our algorithm uses $O(k \log n)$ colors with

high probability, which is asymptotically optimal for any constant k, and $O(k \log k \log n)$ random bits. This is the first efficient online conflict-free coloring for general hypergraphs and subsumes all the previous randomized algorithmic results of Chen [2006], Fiat et al. [2005], Kaplan and Sharir [2004], Chen et al. [2007]. It also substantially improves the efficiency with respect to the amount of randomness used in the special cases studied in Chen [2006], Fiat et al. [2005], Kaplan and Sharir [2004], Chen et al. [2007].

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 with high probability. Irani [1994] showed that the same bound is achieved deterministically by the first-fit greedy 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 3.14 where the number of neighbors of the Delaunay graph of every induced hypergraph is at most two), the best known deterministic online conflict-free coloring algorithm from Fiat et al. [2005] uses $\Theta(\log^2 n)$ colors. We hope that our technique together with a possible clever derandomization technique can shed light on this problem.

As mentioned already, the framework of section 3.3 can describe some known algorithms such as the Unique Max Greedy of Fiat et al. [2005] for online conflict-free 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 (asymptotically $n/2$ from Bar-Noy et al. [2006, 2008]), 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 3.5 we initiate the study of online conflict-free coloring with recoloring: We provide a deterministic online conflict-free 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 conflict-free coloring for this case that uses $\Theta(\log^2 n)$ colors in the worst case without recoloring, by Fiat et al. [2005]. We also present deterministic online algorithms for conflict-free coloring points with respect to circular arcs and halfplanes (and unit disks) that use $O(\log n)$ colors and $O(n)$ recolorings in the worst case. In the special case of intervals or circular arcs at most one point is recolored per insertion.

It would be interesting to find a deterministic online conflict-free coloring algorithm for points in the plane with respect to halfplanes that uses $\Theta(\log n)$ colors in the worst case and recolors at most a constant number of points per insertion. We leave this as an open problem for further research.

All of our randomized algorithms assume the oblivious adversary model, in which the adversary has to commit to a specific input sequence before revealing the first vertex to the algorithm without knowing the random bits that the algorithm is going to use and the expected number of colors is analyzed. The randomized model can be seen as a relaxation of the strict

deterministic model: some power is taken from the adversary, or equivalently given to the algorithm, in order to achieve just a logarithmic number of colors. Another such relaxation, introduced in chapter 2 and Bar-Noy et al. [2006], is to give extra information to the algorithm about where each requested point will end up in the final coloring (the 'absolute positions' model described in section 2.4). Other such relaxations are given in section 2.5 (coloring with respect to rays) and Schiermeyer et al. [2000] (online ranking of paths). In this chapter we introduced yet another relaxation, the recoloring model, in which the algorithm is allowed to recolor some of the points. An interesting question is to construct $O(\log n)$ colors algorithms that rely as little as possible on their extra power (as few random bits as possible, as few recolorings as possible). Towards that goal, in a unified framework, we provided the best known results: randomized algorithm that use an expected logarithmic number of random bits, and recoloring algorithms that perform at most a linear number of recolorings.

Chapter 4

Variations on conflict-free coloring

In this chapter, we study some variations on conflict-free coloring with respect to intervals. First, we study a variation in which only a given subset of intervals is required to have the conflict-free property. For this problem, we provide a polynomial time 2-approximation algorithm, i.e., an algorithm that for every input computes a coloring which uses at most 2 times the number of colors of an optimal solution, improving on a 4-approximation algorithm by Katz et al. [2007]. Conflict-free coloring with respect to intervals can also be seen as a vertex coloring of a chain or path graph in which there is a unique color in every subpath of the graph. Starting from that point, we generalize to conflict-free coloring the vertices of a general graph with respect to all paths in the graph and present some results. We also consider vertex conflict-free

colorings of graphs with respect to neighborhoods of the graph. We prove, among other things, that for a graph with n vertices $2\sqrt{n}$ colors are enough to conflict-free color with respect to neighborhoods and sometimes $\Omega(\sqrt{n})$ colors are necessary. Finally, we initiate the study of Ramsey-type problems for conflict-free colorings and compute a van der Waerden-like number.

4.1 Conflict-free coloring with respect to a set of intervals

We describe a general algorithm for computing a conflict-free coloring of a hypergraph, which is based on computing minimal hitting sets in hypergraphs. For a hypergraph induced by a set of intervals, we propose a way to compute hitting sets that gives rise to a 2-approximation conflict-free coloring algorithm. We also prove that our analysis of the algorithm is best possible, by providing tight instances.

4.1.1 A hitting set algorithm for conflict-free coloring

In this section, we present an algorithm for conflict-free coloring a hypergraph. It is based on repeatedly computing a minimal hitting set in hypergraphs.

Definition 4.1. A *hitting set* of a hypergraph $H = (V, E)$ is a subset $S \subseteq V$ such that for every $e \in E$ there exists some $v \in S$ with $v \in e$. A hitting set

S is *minimal* if for every $v \in S$, $S \setminus \{v\}$ is not a hitting set.

So far, in the problems we have studied, a conflict-free coloring is an assignment of colors (positive integers) to the vertices of the hypergraph. Here we consider a slight variation of conflict-free coloring, in which, we allow some vertices to not be assigned colors, as long as there is in every hyperedge a vertex with assigned color, that is uniquely occurring in the hyperedge, namely the coloring function $C\colon V \to \mathbb{N}^+$ is a *partial* function. Alternatively, we can use a special color '0' given to vertices that are not assigned any positive color and have a total function $C\colon V \to \mathbb{N}$.

Remark 4.2. We claim that this setting, with the partial coloring function, is interesting from the point of view of applications. As mentioned in section 1.2, vertices model base stations in a cellular network. A vertex with no color assigned to it can model a situation where a base station is not activated at all, and therefore the base station does not consume energy. One can think of a bi-criteria optimization problem where a conflict-free assignment of frequencies has to be found with small number of frequencies (in order to conserve the frequency spectrum) and few activated base stations (in order to conserve energy).

In this setting, we describe an algorithm for conflict-free coloring any hypergraph $H = (V, E)$ in figure 4.1.

Claim 4.3. *The algorithm terminates.*

Proof. At every iteration of the loop, there is some hyperedge $e \in E^\ell$ for

$\ell \leftarrow 0$; $V^0 \leftarrow V$; $E^0 \leftarrow E$
while $E^\ell \neq \emptyset$ do:
    $S^\ell \leftarrow$ a minimal hitting set for (V^ℓ, E^ℓ)
    color every $v \in V^\ell \setminus S^\ell$ with color ℓ
    $V^{\ell+1} \leftarrow S^\ell$
    $E^{\ell+1} \leftarrow \{e \cap S^\ell \mid e \in E^\ell \text{ and } |e \cap S^\ell| > 1\}$
    $\ell \leftarrow \ell + 1$
end of while
if $V^\ell \neq \emptyset$, color every $v \in V^\ell$ with color ℓ

Figure 4.1: A hitting set algorithm for conflict-free coloring

which $|e \cap S^\ell| = 1$, because the hitting set S^ℓ is minimal. Thus, $|E^\ell| > |E^{\ell+1}|$. Therefore, the number of hyperedges decreases at every iteration of the loop, and necessarily reaches zero after a finite number of iterations of the loop. □

Claim 4.4. *The algorithm produces a conflict-free coloring.*

Proof. For every hyperedge $e \in E$, there is some ℓ for which $|e \cap S^\ell| = 1$. Let v be the one element of $e \cap S^\ell$. Vertex v is colored with some color greater than ℓ by the algorithm and all other vertices of e are colored with colors which are at most of value ℓ. Thus, e has the conflict-free property. □

4.1.2 A 2-approximation algorithm for a set of intervals

Consider a hypergraph $H = (V, E)$ with $V = \{1, \ldots, n\}$ and every $e \in E$ is a set of consecutive integers. For every i, j with $1 \leq i \leq j \leq n$, the set of consecutive integers $\{k \mid k \in \mathbb{N} \text{ and } i \leq k \leq j\}$ is denoted by $[i, j]$ and

called an *interval*. The hypergraphs studied in chapter 2 contain *all* possible intervals, in contrast with hypergraphs studied in this section.

We use the algorithm described in the previous section to conflict-free color hypergraphs induced by a set of intervals. It is necessary to specify how to compute the minimal hitting set.

The minimal hitting set S is computed as follows.

> First, we compute a special independent set of intervals $I \subseteq E$ (i.e., in I no two intervals have a common vertex). We compute the independent set of intervals incrementally. Initially, there is nothing in the independent set. We scan vertices from 1 to n and we include in the independent set the interval $[i, j] \in E$ with minimum j such that $[i, j]$ does not intersect anything already in the independent set. After computing I, for every interval $[i, j] \in I$, we take in S the vertex j (i.e., the maximum or rightmost vertex).

It is not difficult to see that that S is a minimal hitting set (in fact, it is a minimum cardinality hitting set), as follows. Set S is a hitting set because no interval is completely contained between two vertices in S, no interval ends before the first interval in I, and no interval starts after the last interval in I; otherwise such intervals would be chosen in the independent set I. Set S is minimal, because removing any element j of it, means that the interval with right endpoint j in I is not hit any more.

Remark 4.5. The computation of the maximum independent set of intervals given above is also known as a solution to the activity selection problem. See for example Cormen et al. [2001, section 16.1] or Zachos [2006].

We intend to compare colorings produced by the above algorithm with optimal colorings. We define recursively the following configuration of intervals.

Definition 4.6. Configuration J_1 is a single interval. For $k > 1$, configuration J_k consists of two instances of configuration J_{k-1} that are disjoint (i.e., no interval from one instance has a common point with an interval in the other instance) and of an interval that contains every interval in the two disjoint J_{k-1} instances.

Lemma 4.7. *Any conflict-free coloring uses at least k colors for a set of intervals I that contains an instance of a J_k configuration.*

Proof. We use induction on k. For $k = 1$, the statement is trivially true. Assume it is true for k, we will prove it for $k + 1$. Assume, for the sake of contradiction, that there is a conflict-free coloring with just k colors of a set of intervals I that contains an instance of a J_{k+1} configuration. Set I contains two independent instances of J_k that each (by the inductive hypothesis) is using all k colors. But then, the interval of the J_{k+1} instance that contains both independent instances of J_k is not conflict-free colored, which is a contradiction. □

We are now ready to bound the approximation ratio of the proposed algorithm.

Theorem 4.8. *The conflict-free coloring algorithm for hypergraphs induced by a set of intervals is a 2-approximation algorithm.*

Proof. It is enough to prove that if some hyperedge (or interval), say ι, reaches iteration with $\ell = k - 1$ of the loop (i.e., the algorithm uses at least k colors), then there is a $J_{\lceil k/2 \rceil}$ configuration in the input and moreover this configuration is entirely contained in ι.

We prove it by induction. For $k = 1, 2$, it is true, because there is at least one interval in the input, and therefore at least one non-zero color is needed in any optimal coloring. For $k > 2$, assume there is a vertex v that gets color k. Then at iteration with $\ell = k - 1$ of the loop there is an interval ι with its rightmost vertex being $v \in S^\ell$ (see figure 4.2).

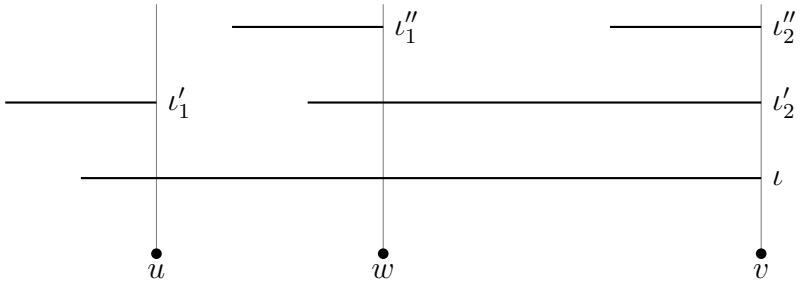

Figure 4.2: Intervals in an input using k colors

Since ι was not removed in the previous iteration $\ell - 1$, there were two vertices of ι in $S^{\ell-1}$, say u and v, with $u < v$. Also, since u and v are in $S^{\ell-1}$ there are two intervals with them as right endpoints in the independent

set computed at iteration $\ell - 1$, say ι'_1 and ι'_2. Since ι'_2 was not removed in the iteration $\ell - 2$, there were two vertices of ι'_2 in $S^{\ell-2}$, say w and v, with $u < w < v$. Also, since u, w, and v are in $S^{\ell-2}$ there are three intervals with them as right endpoints in the independent set computed at iteration $\ell - 2$; call ι''_1 the one ending at w and ι''_2 the one ending at v. Since the three intervals are independent, ι''_1 and ι''_2 start after u, therefore they are fully contained in ι (which contains u). By the inductive hypothesis, since each of ι''_1, ι''_2 reach iteration $\ell - 2$, each of them contains a $J_{\lceil (k-2)/2 \rceil}$ configuration, and, since ι''_1 and ι''_2 are disjoint, together with ι they constitute a $J_{\lceil k/2 \rceil}$ configuration. □

4.1.3 A tight instance for the 2-approximation algorithm

For $k \geq 2$, we intend to define an input I_k that is a tight instance for the approximation algorithm, i.e., an instance that forces the algorithm to use at least twice the number of colors in an optimal coloring. Before doing that, we define some notation that will prove useful.

Definition 4.9. Given a set of intervals I and a natural number d, we define I^{+d} to be the set of intervals, where all intervals of I are shifted d to the right, i.e., for every $[i, j] \in I$, there is $[i + d, j + d] \in I^{+d}$, and there are no other intervals in I^{+d}.

Definition 4.10. Given a set of intervals I, we define the *length* of I, denoted

$\text{len}(I)$ to be the rightmost point occurring in any of the intervals of I minus the leftmost point occurring in any of the intervals of I plus one.

Now, we are ready to proceed with the definition of the tight instance.

Definition 4.11. For $k = 2$ the input I_2 has length equal to four and consists of three intervals.

$$I_2 = \{[1, 2], [3, 3], [2, 4]\}$$

For $k > 2$ the input is defined recursively as follows.

$$I_{k+1} = I_k \cup I_k^{+\text{len}(I_k)} \cup \{[\text{len}(I_k) - k + 1, 2\text{len}(I_k) + 1]\}$$

Abusing notation, we call the I_k component the *left* I_k part of I_{k+1} and the $I_k^{+\text{len}(I_k)}$ component the *right* I_k part of I_{k+1}. These left and right parts are disjoint. Inputs I_2, I_3, I_4, I_5 are shown in figures 4.3, 4.4, 4.5, 4.6, respectively. Moreover, in the figures, under the vertices of each input we give the coloring produced by the 2-approximation algorithm and then an optimal conflict-free coloring.

Figure 4.3: Input I_2 and conflict-free colorings

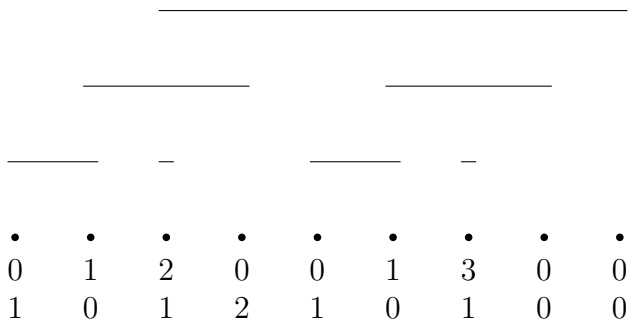

Figure 4.4: Input I_3 and conflict-free colorings

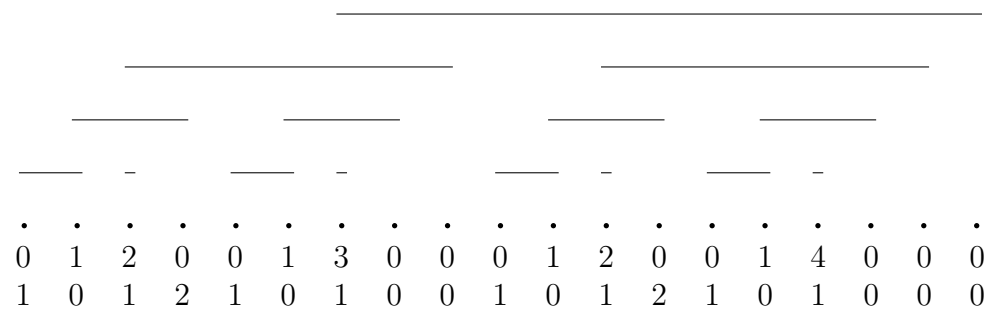

Figure 4.5: Input I_4 and conflict-free colorings

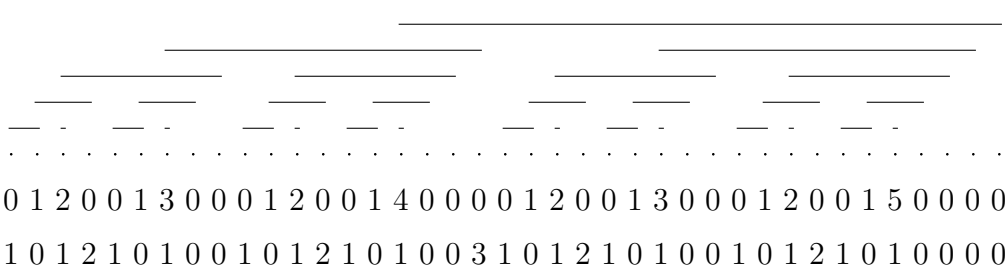

Figure 4.6: Input I_5 and conflict-free colorings

It is not difficult to see that the length of the instance satisfies the recurrence relation

$$\text{len}(I_{k+1}) = 2\text{len}(I_k) + 1 \tag{4.1}$$

which implies, since $\text{len}(I_2) = 4$, that

$$\text{len}(I_k) = 5 \cdot 2^{k-2} - 1$$

Another notion that will prove useful is the *level* of each interval in the above instance that we define in the following.

Definition 4.12. In input I_2, intervals $[1, 2]$ and $[3, 3]$ are of level 1 and interval $[2, 4]$ is of level 2. In the recursively defined instance

$$I_{k+1} = I_k \cup I_k^{+\text{len}(I_k)} \cup \{[\text{len}(I_k) - k + 1, 2\text{len}(I_k) + 1]\}$$

the intervals of the I_k part have the same levels as the corresponding intervals in the I_k instance, the intervals of the $I_k^{+\text{len}(I_k)}$ part have the same levels as the corresponding intervals of the I_k instance before the '$+\text{len}(I_k)$' operation, and interval $[\text{len}(I_k) - k + 1, 2\text{len}(I_k) + 1]$ has level $k + 1$.

In fact, in figures 4.3, 4.4, 4.5, and 4.6 the vertical coordinate of each interval signifies its level, with higher intervals having higher level.

Lemma 4.13. *For $k \geq 3$, in I_k, the leftmost point of the level k interval is the same as the rightmost level 1 interval in the left I_{k-1} part of I_k.*

Proof. We prove by induction that the rightmost level 1 interval of the left I_{k-1} part of I_k is at position $\text{len}(I_{k-1}) - (k-1) + 1$. For I_3, the rightmost level 1 interval of the left I_2 part of I_3 consists of point $4 - (3-1) + 1 = 3$. By the inductive hypothesis, the rightmost level 1 interval of the left I_{k-1} part of I_k is at $\text{len}(I_{k-1}) - (k-1) + 1$. Then for I_{k+1}, the rightmost level 1 interval of its left I_k part is at

$$\text{len}(I_{k-1}) - (k-1) + 1 + \text{len}(I_{k-1}) = (2\text{len}(I_{k-1}) + 1) + k - 1 = \text{len}(I_k) + k - 1.$$

The last equality is implied by equation (4.1). □

Lemma 4.14. *Instance I_k contains a $J_{\lceil k/2 \rceil}$ configuration.*

Proof. By induction. It is true for $k = 2$ and $k = 3$, because I_2 contains a J_1 configuration and I_3 contains a J_2 configuration. For $k > 3$, in instance I_k, the interval of level k contains completely a copy of I_{k-1}, in which two disjoint copies of I_{k-2} are contained. By the inductive hypothesis, in each copy of I_{k-2}, a $J_{\lceil (k-2)/2 \rceil}$ configuration is contained. These two disjoint $J_{\lceil (k-2)/2 \rceil}$ configurations, together with the level k interval constitute a $J_{\lceil k/2 \rceil}$ configuration in I_k. □

Lemma 4.15. *There is a conflict-free coloring of I_k with $\lceil k/2 \rceil$ colors.*

Proof. We define recursively a coloring of I_k that uses $\lceil k/2 \rceil$ colors and we prove by induction that it is conflict-free.

For $k = 2$ the coloring is 1010, which can be easily checked to be conflict-free.

If k is odd, take a coloring of I_{k-1} and in its rightmost position use color $\lceil k/2 \rceil$, concatenate a coloring of I_{k-1}, and then concatenate color '0'. By induction, the left I_{k-1} part is conflict-free because we started with a conflict-free coloring and we introduced a new color $\lceil k/2 \rceil$, the right I_{k-1} part is conflict-free because it is colored with a conflict-free coloring. The level k interval is conflict-free because of color $\lceil k/2 \rceil$ that occurs uniquely.

If k is even, with $k > 2$, take a coloring of I_{k-1}, concatenate a coloring of I_{k-1}, and then concatenate color '0'. By induction, the left I_{k-1} part is conflict-free because it is colored with a conflict-free coloring, the right I_{k-1} part is conflict-free because it is colored with a conflict-free coloring. The level k interval is conflict-free because of color $\lceil k/2 \rceil$ that occurs in the right I_{k-1} part and because its leftmost point, by lemma 4.13, is to the right of the $\lceil k/2 \rceil$ color occurring in the left I_{k-2} part of the left I_{k-1} part. □

Corollary 4.16. *An optimal coloring of I_k uses $\lceil k/2 \rceil$ colors.*

We now describe a family of hypergraphs that arise after the first iteration of the while loop of the 2-approximation algorithm, if the initial input is I_k.

Definition 4.17. The instance L_0 is on one vertex, namely the vertex set is $\{1\}$, and contains no interval, i.e, $L_0 = \{\}$. The length of instance L_0 is defined to be 1. For $k > 0$, L_{k+1} is defined recursively.

$$L_{k+1} = L_k \cup L_k^{+\text{len}(L_k)} \cup \{[\text{len}(L_k), 2\text{len}(L_k)]\}$$

It is not difficult to see that the length satisfies the recurrence relation

$\text{len}(L_{k+1}) = 2\text{len}(L_k)$, which implies $\text{len}(L_k) = 2^k$. We say that L_{k+1} consists of a left L_k part, a right L_k part, and the interval $[2^k, 2^{k+1}]$.

Proposition 4.18. *The 2-approximation algorithm colors I_k with k colors.*

Proof. Assume input I_k is given to the 2-approximation algorithm. In the iteration of the while loop where the algorithm colors points with color ℓ ($\ell = 0, 1, \dots$), the algorithm considers a hypergraph H_ℓ. We will prove that the algorithm considers the hypergraphs

$$H_0 = I_k, H_1 = L_{k-1}, \dots, H_{k-1} = L_1, H_k = L_0,$$

and then it terminates, i.e., it uses k colors. We say that H_i is *followed* by H_{i+1}, to show that two hypergraphs H_i, H_{i+1} are considered successively by the algorithm, in that order.

First, we prove that for every $k \geq 2$, I_k is followed by L_{k-1}, by induction on k. It is not difficult to see that, when I_k is considered, the independent set of intervals chosen consists of all level 1 intervals of I_k and the hitting set that is chosen consists of the right endpoints of all level 1 intervals of I_k (a formal proof can be carried out by induction on k). For $k = 2$ it is not difficult to check that I_2 is followed by L_1. For $k > 2$, I_k consists of a left I_{k-1} part which induces a left L_{k-2} part and a right I_k part, which induces a right L_{k-2} part (we use the inductive hypothesis). From lemma 4.13, the leftmost point of the level k interval is the same as the rightmost level 1 interval in the left I_{k-1} part of I_k, and therefore the level k interval induces an interval

that starts from the last point of the left L_{k-2} part of the hypergraph that follows I_k and ends at the last point of the right L_{k-2} part of the hypergraph that follows I_k. To summarize, the I_k is followed by a left L_{k-2} part, a right L_{k-2} part and interval $[2^{k-2}, 2^{k-1}]$, i.e., it is L_{k-1}.

Then, we prove that for $k > 0$, L_k is followed by L_{k-1}, by induction on k. For $k = 1$, it is not difficult to see that for L_1 the interval $[1, 2]$ is chosen and its right endpoint, i.e., 2, makes up the hitting set. Then, easily, L_1 is followed by L_0. For $k > 1$, when L_k is considered, the independent set of intervals that is chosen consists of the intervals of length two of the left L_{k-1} part

$$\{[1, 2], [3, 4], \ldots, [2^{k-1} - 1, 2^{k-1}]\}$$

and the intervals of length two of the right L_{k-1} part

$$\{[2^{k-1} + 1, 2^{k-1} + 2], [2^{k-1} + 3, 2^{k-1} + 4], \ldots, [2^k - 1, 2^k]\}.$$

Therefore the hitting set is

$$\{2, 4, \ldots 2^{k-1}\} \cup \{2^{k-1} + 2, 2^{k-1} + 4, \ldots, 2^k\} = \{i \colon \text{odd} \mid 2 \leq i \leq 2^k\}$$

and consists of 2^{k-1} elements. By induction, after removal of the points of the hitting set, the left L_{k-1} part induces a L_{k-2} part, and the right L_{k-1} part induces a L_{k-2} part. The interval $[2^{k-1}, 2^k]$ of L_k contains all points in $\{2^{k-1} + 2, 2^{k-1} + 4, \ldots, 2^k\}$ of the right L_{k-1} part and just point 2^{k-1} of

the left L_{k-1} part, and therefore induces $[2^{k-2}, 2^{k-1}]$ in the hypergraph that follows L_k. To summarize, the L_k is followed by a left L_{k-2} part, a right L_{k-2} part and interval $[2^{k-2}, 2^{k-1}]$, i.e., it is L_{k-1}.

Finally, we prove that when L_0 is reached, no hypergraph follows, and the algorithm terminates. This is true, because L_0 contains no interval (hyperedge). □

Remark 4.19. From the above proof of proposition 4.18, it is immediate that if L_k is given as an input to the 2-approximation algorithm, the following sequence of hypergraphs

$$H_0 = L_k, H_1 = L_{k-1}, \ldots, H_{k-1} = L_1, H_k = L_0$$

is considered in the iterations of the while loop. Moreover, it can also be proven, with a proof similar to those of lemmata 4.14 and 4.15, that an optimal coloring for L_k uses $\lceil k/2 \rceil$ colors. Therefore, the family of instances L_k is also a family of tight instances for the 2-approximation algorithm. However, the family of instances I_k has the additional property that no two intervals in it share a common right endpoint.

4.2 Conflict-free coloring for paths

A conflict-free coloring of n points with respect to intervals can be seen as a conflict-free coloring of the vertices of the path graph P_n with respect to all

paths in the graph. We generalize to arbitrary graphs.

Definition 4.20. A conflict-free coloring of graph G is a function $C\colon V \to \mathbb{N}^+$ such that in every path of G there is a vertex with a uniquely occurring color in the path. The minimum k for which there is a conflict-free coloring of graph G is called the conflict-free chromatic number of G, denoted by $\chi_{\mathrm{cf}}^{\mathrm{path}}(G)$, or simply $\chi_{\mathrm{cf}}(G)$ (when there is no danger of confusion with the same notation for conflict-free coloring arbitrary hypergraphs).

As we have mentioned, an ordered coloring of a graph is a unique maximum conflict-free coloring of a graph; see definition 1.15 on page 15. We show a simple relation between the conflict-free chromatic number, the ordered chromatic number, and the traditional chromatic number of a graph.

Proposition 4.21. *For every graph G, $\chi(G) \leq \chi_{\mathrm{cf}}(G) \leq \chi_{\mathrm{o}}(G)$.*

Proof. Since every ordered coloring is a also a conflict-free coloring, we have $\chi_{\mathrm{cf}}(G) \leq \chi_{\mathrm{o}}(G)$. A traditional coloring can be defined as a coloring in which paths of length one are conflict-free. Therefore every conflict-free coloring is also a traditional coloring and thus $\chi(G) \leq \chi_{\mathrm{cf}}(G)$. □

Moreover, we prove that both conflict-free and ordered chromatic numbers are monotone under taking subgraphs.

Proposition 4.22. *If $X \subseteq Y$, then $\chi_{\mathrm{cf}}(X) \leq \chi_{\mathrm{cf}}(Y)$ and $\chi_{\mathrm{o}}(X) \leq \chi_{\mathrm{o}}(Y)$.*

Proof. Take the restriction of any conflict-free or ordered coloring of graph Y to the vertex set $V(X)$. This is a conflict-free or ordered coloring of graph X because the set of paths of graph X is a subset of all paths of Y. □

We now define a family of graphs where the two numbers, $\chi_{\text{o}}(G)$ and $\chi_{\text{cf}}(G)$, differ by one.

Definition 4.23. The hedgehog graph H_k consists of a complete graph K_k, a null graph $\overline{K_k}$, and a matching of k edges, where each vertex of K_k is matched with a different vertex of $\overline{K_k}$.

In figure 4.7, we show some small hedgehog graphs.

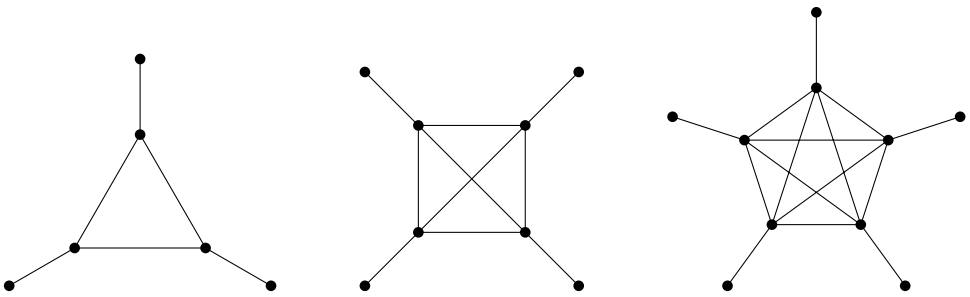

Figure 4.7: Hedgehog graphs for $k = 3, 4, 5$

Lemma 4.24. *For all $k \geq 3$, $\chi_{\text{cf}}(H_k) \geq k$.*

Proof. Since $K_k \subseteq H_k$, we have $\chi_{\text{cf}}(H_k) \geq \chi_{\text{cf}}(K_k) \geq \chi(K_k) = k$, by propositions 4.21 and 4.22. □

Lemma 4.25. *For all $k \geq 3$, $\chi_{\text{cf}}(H_k) \leq k$.*

Proof. We exhibit, for $k \geq 3$, a conflict-free coloring of H_k with k colors. First we show how to color vertices in the K_k part and then in the $\overline{K_k}$ part. Color the k vertices of the K_k part with k different colors from $\{1, \ldots, k\}$. For every vertex v in the $\overline{K_k}$ part, if its neighbor in the K_k part is colored

with color c, then color v with $(c+1) \bmod_1 k$, where '$\bmod_1$' is a variation of the modulo binary operator that returns k instead of 0. Now, we check that the above coloring is indeed conflict-free. Every trivial path, i.e., a path of one vertex, always has the conflict-free property, so we consider only paths of length at least one in what follows. Every path contained in the K_k part has the conflict-free property, because no two vertices in the K_k part share the same color. Every non-trivial path that contains exactly one vertex v of the $\overline{K_k}$ part has the conflict-free property because the color of the neighbor of v in the K_k part is unique in the path. What remains is to consider paths in which the initial and the last vertex in the path are in part $\overline{K_k}$. In that case, the remaining vertices of the path are in K_k because there is no (simple) path in the graph that contains more than two vertices of $\overline{K_k}$ and the degree of vertices in $\overline{K_k}$ is one. Assume such a path looks like $vu \dots u'v'$, where v and v' are in the $\overline{K_k}$ part, and u and u' are in the K_k part (see figure 4.8). Assume

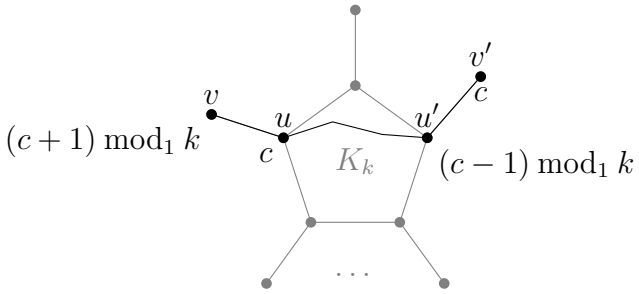

Figure 4.8: A path with two vertices of degree one, v and v'

$C(u) = c$. If c is not a unique color in the path, it has to be the case that $C(v') = C(u) = c$ (see figure 4.8), but in that case $C(u') = (c-1) \bmod_1 k$,

which is a unique color in the path, because it is different from all other colors in the path, even $C(v) = (c+1) \bmod_1 k$ (because $k \geq 3$). □

Lemma 4.26. *For all $k \geq 3$, $\chi_o(H_k) > k$.*

Proof. Assume for the sake of contradiction that there is an ordered coloring C of H_k with colors in the set $\{1, \ldots, k\}$. It is not possible for a color to be shared by two vertices of the K_k part (because there is an edge between any two such vertices). Therefore, all colors are used in the K_k part, a different one for every vertex in K_k. Consider the vertex u of K_k with $C(u) = 1$. Its neighbor v in the $\overline{K_k}$ part has a different color $C(v) = c > 1$. But there is also a neighbor of u, call it u', in the K_k part, with $C(u') = c$. But then the path vuu' does not have the unique maximum property, because the maximum color occurring in it, namely $c > 1$, is not unique. Therefore, we have a contradiction to C being an ordered coloring of H_k. □

Lemma 4.27. *For all $k \geq 3$, $\chi_o(H_k) \leq k + 1$.*

Proof. We exhibit, for $k \geq 3$, an ordered coloring of H_k with $k+1$ colors. Color all vertices in the $\overline{K_k}$ part with the same color 1. Color each vertex in the K_k part with a different color from the set $\{2, \ldots, k+1\}$. Every non-trivial path contains at least a vertex of the K_k part. Since the colors of the K_k part are unique in the whole graph and greater than the color 1, every path has the unique maximum property. □

Therefore, by combining lemmata 4.24, 4.25, 4.26, and 4.27, we have the following.

Corollary 4.28. *For $k \geq 3$, $\chi_{\mathrm{cf}}(H_k) = k$ and $\chi_{\mathrm{o}}(H_k) = k + 1$.*

4.3 Conflict-free coloring for neighborhoods

In this section, we consider conflict-free coloring hypergraphs induced by neighborhoods of vertices in a graph. We give formal definitions in the following.

Definition 4.29. The *open neighborhood* of a vertex v in graph G is the set of vertices adjacent to v in G and is denoted by $\mathit{\Gamma}_G(v)$ (or just $\mathit{\Gamma}(v)$ if the graph G is clear from context) The *closed neighborhood* of a vertex v in graph G is the open neighborhood together with the vertex v and is denoted by $\bar{\mathit{\Gamma}}_G(v)$ (or just $\bar{\mathit{\Gamma}}(v)$ if the graph G is clear from context).

We remark that for a simple graph, we have $v \notin \mathit{\Gamma}(v)$, but $v \in \bar{\mathit{\Gamma}}(v)$.

Definition 4.30. A function $C\colon V(G) \to \{1, \dots, k\}$ is a *conflict-free k-coloring with respect to open neighborhoods* of G if for every $v \in V(G)$ there is a vertex with unique color in $\mathit{\Gamma}(v)$. The *open neighborhood conflict-free chromatic number* of G is denoted by $\chi_{\mathrm{cf}}^{\mathrm{on}}(G)$ and is the smallest number k such that G has a conflict-free k-coloring with respect to open neighborhoods.

Definition 4.31. A function $C\colon V(G) \to \{1, \dots, k\}$ is a *conflict-free k-coloring with respect to closed neighborhoods* of G if for every $v \in V(G)$ there is a vertex with unique color in $\bar{\mathit{\Gamma}}(v)$. The *closed neighborhood conflict-free chromatic number* of G is denoted by $\chi_{\mathrm{cf}}^{\mathrm{cn}}(G)$ and is the smallest number k

such that G has a conflict-free k-coloring with respect to closed neighborhoods.

Definition 4.32. A function $C\colon V(G) \to \{1, \dots, k\}$ is a *unique maximum k-coloring with respect to open neighborhoods* of G if for every $v \in V(G)$ the largest color occurring in $\Gamma(v)$ is a unique color in $\Gamma(v)$. The *open neighborhood unique maximum chromatic number* of G is denoted by $\chi^{\text{on}}_{\text{um}}(G)$ and is the smallest number k such that G has a unique maximum k-coloring with respect to open neighborhoods.

Definition 4.33. A function $C\colon V(G) \to \{1, \dots, k\}$ is a *unique maximum k-coloring with respect to closed neighborhoods* of G if for every $v \in V(G)$ the largest color occurring in $\bar{\Gamma}(v)$ is a unique color in $\bar{\Gamma}(v)$. The *closed neighborhood unique maximum chromatic number* of G is denoted by $\chi^{\text{cn}}_{\text{um}}(G)$ and is the smallest number k such that G has a unique maximum k-coloring with respect to closed neighborhoods.

Every unique maximum coloring is a conflict-free coloring and therefore $\chi^{\text{on}}_{\text{cf}}(G) \leq \chi^{\text{on}}_{\text{um}}(G)$ and $\chi^{\text{cn}}_{\text{cf}}(G) \leq \chi^{\text{cn}}_{\text{um}}(G)$.

4.3.1 Upper bounds on closed neighborhoods conflict free colorings

Proposition 4.34. *For every graph G, $\chi^{\text{cn}}_{\text{cf}}(G) \leq \chi(G)$.*

Proof. A traditional vertex coloring is a closed neighborhood conflict-free

coloring, because in every $\bar{\Gamma}(v)$, vertex v has a unique color among its neighbors. □

The *degree* of a vertex v in a graph is the cardinality of its open neighborhood $\Gamma(v)$. We denote by $\Delta(G)$ the *maximum degree* of a vertex in graph G.

We can prove a result analogous to Brooks' theorem for the closed neighborhood conflict-free chromatic number. The theorem of Brooks [1941] states that $\chi(G) \leq \Delta(G)$, unless the graph is complete or an odd cycle.

Proposition 4.35. *For every graph G, except K_1 and K_2, $\chi_{\text{cf}}^{\text{cn}}(G) \leq \Delta(G)$. For K_1, $\chi_{\text{cf}}^{\text{cn}}(K_1) = \Delta(K_1) + 1 = 1$; for K_2, $\chi_{\text{cf}}^{\text{cn}}(K_2) = \Delta(K_2) + 1 = 2$.*

Proof. By proposition 4.34 and Brooks' theorem we have $\chi_{\text{cf}}^{\text{cn}}(G) \leq \Delta(G)$, except possibly for complete graphs and odd cycles. However, from proposition 4.49, for $n \geq 2$, $\chi_{\text{cf}}^{\text{cn}}(K_n) = 2$, and from proposition 4.47, for $n \geq 3$, $\chi_{\text{cf}}^{\text{cn}}(C_n) = 2$. □

A *dominating set* D in graph G is a subset of $V(G)$ such that every vertex in $V(G) - D$ has a neighbor in D. The cardinality of the smallest dominating set in G is denoted by $\gamma(G)$.

Proposition 4.36. *For every graph G, $\chi_{\text{cf}}^{\text{cn}}(G) \leq \chi_{\text{um}}^{\text{cn}}(G) \leq \gamma(G) + 1$.*

Proof. Color vertices in a smallest size dominating set D with colors 2 to $\gamma(G) + 1$. Color vertices in $V(G) - D$ with color 1. □

Proposition 4.37. *For every graph G, $\chi_{\text{cf}}^{\text{cn}}(G) \leq 2\sqrt{n}$.*

Proof. Use the following coloring algorithm: Repeatedly remove the vertex v in the graph that dominates most other vertices that have not yet been dominated. Color v with a unique color and vertices in $\Gamma(v)$ with a dummy color (say color 1). Repeat until no vertex that dominates $\sqrt{n}$ or more vertices remains, and color the remaining graph (the uncolored vertices) with a traditional coloring, greedily, using a new set of colors. The first coloring process can not go on for more than $\sqrt{n}$ steps because for every step, where v is chosen, at least $\sqrt{n}$ new vertices are colored with color 1. The second coloring process uses at most $\sqrt{n}$ colors, because the remaining graph has maximum degree less than $\sqrt{n}$. Therefore, $\chi_{\text{cf}}^{\text{cn}}(G) < \sqrt{n} + 1 + \sqrt{n}$, or equivalently $\chi_{\text{cf}}^{\text{cn}}(G) \leq 2\sqrt{n}$. □

Remark 4.38. Recently, with the help of a randomized coloring and an analysis based on the Lovász local lemma, Pach and Tardos [2008] improved on the above result and showed that the closed neighborhood conflict-free chromatic number is bounded by $O(\log^{2+\varepsilon} n)$, for every $\varepsilon > 0$. They also proved a lower bound of $\Omega(\log n)$ on $\chi_{\text{cf}}^{\text{cn}}$, which holds almost surely for the random graph on n vertices with edge probability $1/2$.

4.3.2 Upper bounds on open neighborhood colorings

In order to upper bound the open neighborhood conflict-free chromatic numbers, we use the notion of total dominating set which is related to the notion of dominating set. A *total dominating set* D in graph G is a subset of $V(G)$

such that every vertex in $V(G)$ has a neighbor in D. Observe that a graph which has at least one isolated vertex has no total dominating set. In the following we only consider graphs with no isolated vertices. The cardinality of the smallest total dominating set in G is denoted by $\gamma_{\mathrm{t}}(G)$.

Proposition 4.39. *For every graph G, $\chi_{\mathrm{cf}}^{\mathrm{on}}(G) \leq \chi_{\mathrm{um}}^{\mathrm{on}}(G) \leq \gamma_{\mathrm{t}}(G) + 1$.*

Proof. Color vertices in a smallest size total dominating set D with colors 2 to $\gamma_{\mathrm{t}}(G) + 1$. Color vertices in $V(G) - D$ with color 1. □

We also prove a bound similar to the one given in proposition 4.37. The proof is also similar, with some slight adjustments.

Proposition 4.40. *For every graph G, $\chi_{\mathrm{cf}}^{\mathrm{on}}(G) \leq 2\sqrt{n}$.*

Proof. Use the following coloring algorithm.

In the first phase, as long as there is a vertex v in at least $\sqrt{n}$ neighborhoods, color it with a unique color and remove v along with all neighborhoods to which it belongs. This coloring process can not go on for more than $\sqrt{n}$ steps because in every round at least $\sqrt{n}$ new open neighborhoods are removed and the graph has exactly n open neighborhoods.

We now proceed to the second phase of the algorithm, in what remains in the graph. Each remaining vertex is in fewer than $\sqrt{n}$ remaining open neighborhoods. From now on we use a new set of at most $\sqrt{n}$ colors. We consider an arbitrary order of the remaining vertices. We consider vertices one by one in this order. We intend to color the vertices so that for each

remaining open neighborhood X, the first vertex of X in the order has a uniquely occurring color in X. For each vertex v in the order, we define the set S_v of vertices containing already colored vertices (i.e., vertices before v in the order) which have the additional two properties: (a) they share some hyperedge with v and (b) they were the first vertex colored in some hyperedge. The algorithm colors v with the smallest color that does not occur in any vertex of S_v. This coloring process uses at most $\sqrt{n}$ colors, because each vertex is in less than $\sqrt{n}$ remaining open neighborhoods, and therefore $|S_v| < \sqrt{n}$.

We argue that the coloring is conflict-free. Open neighborhoods that are removed in the first phase of the algorithm have the conflict-free property because each of the colors used in the first phase is unique. Every open neighborhood X that is considered in the second phase has the conflict-free property because of the color c of the first vertex v that is considered for coloring in X. In fact, every other vertex that is subsequently colored in X is going to have a color different from c.

The first coloring process uses at most $\sqrt{n}$ colors. The second coloring process uses at most $\sqrt{n}$ colors. Therefore, $\chi_{\text{cf}}^{\text{on}}(G) \leq 2\sqrt{n}$. □

Remark 4.41. The coloring algorithm described in the proof of proposition 4.40 can be used with slight adaptations to conflict-free color also with respect to closed neighborhoods. However, we still prefer to also give the more direct algorithm for conflict-free coloring closed neighborhoods of graphs in the proof of proposition 4.37, because we think it is simpler and more intu-

itive. Moreover, as shown by Pach and Tardos [2008], the aforementioned algorithm can be used to conflict-free color every hypergraph having at most m hyperedges with at most $2\sqrt{m}$ colors. A more careful switch from the first to the second phase in the algorithm, also due to Pach and Tardos [2008], can even give an asymptotic upper bound of $\sqrt{2}\sqrt{m}$ colors, which is tight (see proposition 4.43).

4.3.3 A lower bound for open neighborhood conflict-free colorings

We have seen in the previous section that for a graph G with n vertices, $\chi_{\text{cf}}^{\text{on}}(G) = O(\sqrt{n})$. We will prove that the above result is tight in the sense that there is a family of graphs with n vertices and $\chi_{\text{cf}}^{\text{on}}(G) = \Omega(\sqrt{n})$.

Definition 4.42. For $k \geq 2$, define the bipartite graph B_k as follows. Consider the set of elements $X_k = \{1, \ldots, k\}$ and the set S_k of subsets of cardinality two of X_k. The vertex set of the bipartite graph B_k consists of the two parts X_k and S_k and there is an edge between $x \in X_k$ and $y \in S_k$ if and only if $x \in y$. The graph has $n = k + k(k-1)/2 = k(k+1)/2$ vertices.

Since $n = k(k+1)/2$, we have $k \approx \sqrt{2}\sqrt{n}$, and in order to show the aforementioned lower bound it is enough to prove the following proposition.

Proposition 4.43. *For $k \geq 2$, $\chi_{\text{cf}}^{\text{on}}(B_k) \geq k$.*

Proof. Assume there is an open-neighborhood conflict-free coloring of B_k with fewer than k colors. Then, by the principle of forced coincidence (pi-

geonhole principle), two vertices, say u and v in X_k have the same color. But this implies that the vertex corresponding to set $\{u, v\}$ in S_k does not have a conflict-free colored open neighborhood. □

4.3.4 Some (almost) optimal colorings

In this section, we exhibit optimal or almost optimal conflict-free colorings with respect to neighborhoods for some families of graphs.

Paths

Proposition 4.44. *For $n > 2$,*

$$\chi_{\mathrm{cf}}^{\mathrm{on}}(P_n) = \chi_{\mathrm{um}}^{\mathrm{on}}(P_n) = \chi_{\mathrm{cf}}^{\mathrm{cn}}(P_n) = \chi_{\mathrm{um}}^{\mathrm{cn}}(P_n) = 2.$$

Any substring of the infinite string $(1122)^\infty$ is a legal conflict-free or unique maximum coloring with respect to open neighborhoods.

For closed neighborhoods, the only unique maximum colorings are

- $(121)^{k-1}12$ (and its reversal; one leaf of the path is colored with 1 and the other with 2) for P_{3k-1},
- $(121)^k$ for P_{3k}, and
- $(211)^k2$ for P_{3k+1}.

Remark 4.45. The above conflict-free colorings with respect to closed neighborhoods are special substrings of the periodic infinite string

$$\ldots 121121121 \ldots$$

Cycles

Proposition 4.46. *If* $4|n$, $\chi_{\mathrm{cf}}^{\mathrm{on}}(C_n) = \chi_{\mathrm{um}}^{\mathrm{on}}(C_n) = 2$, *otherwise* $\chi_{\mathrm{cf}}^{\mathrm{on}}(C_n) = \chi_{\mathrm{um}}^{\mathrm{on}}(C_n) = 3$.

We exhibit optimal unique maximum open neighborhoods colorings:

- for C_{4k}, $(1122)^k$,
- for C_{4k+1}, $(1122)^k 3$,
- for C_{4k+2}, $(1122)^k 33$,
- for C_{4k+3}, $(1122)^k 133$.

Proposition 4.47. *For* $n \geq 3$, $\chi_{\mathrm{cf}}^{\mathrm{cn}}(C_n) = 2$.

We exhibit optimal conflict-free colorings with respect to closed neighborhoods:

- for C_{3k}, $(112)^k$,
- for C_{3k+1}, $(112)^k 2$,
- for C_{3k+2}, $(112)^k 12$.

Proposition 4.48. *If* $3|n$, $\chi^{\text{cn}}_{\text{um}}(C_n) = 2$, *otherwise* $\chi^{\text{cn}}_{\text{um}}(C_n) = 3$.

We exhibit optimal unique maximum closed neighborhoods colorings:

- for C_{3k}, $(112)^k$,
- for C_{3k+1}, $(112)^k 3$,
- for C_{3k+2}, $(112)^k 13$.

Grids

The *product* of graphs X and Y, denoted by $X \times Y$, is the graph with vertex set $V(X \times Y) = V(X) \times V(Y)$ and vertices (x, y) and (x', y') are adjacent in $E(X \times Y)$ if and only if:

$$(x = x' \text{ and } \{y, y'\} \in E(Y)) \text{ or } (\{x, x'\} \in E(X) \text{ and } y = y').$$

The *grid* $G_{r\times c}$ is the graph product of two paths, $P_r \times P_c$. The standard drawing of the grid $G_{r\times c}$, is one where its vertices are put in r rows and c columns. For example, see figure 4.9 for the 4×5 grid graph.

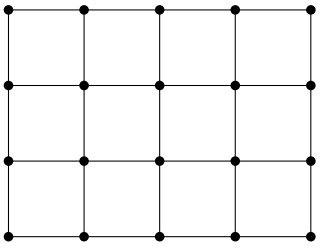

Figure 4.9: A standard drawing of the 4×5 grid

Grids are bipartite graphs, and therefore $\chi_{\text{cf}}^{\text{cn}}(G_{r\times c}) \leq \chi(G_{r\times c}) \leq 2$. For grids, we can show that three colors are enough in all cases, by exhibiting unique maximum colorings with respect to neighborhoods with the aforementioned number of colors.

For open neighborhood unique maximum coloring, we are going to use for each row of $G_{r\times c}$ one of the following two colorings:

$$t = \text{prefix of length } c \text{ of } (2233)^{\infty} \qquad \text{and} \qquad o = 1^c.$$

If $r \equiv 0 \pmod 3$, then color rows as $(oto)^{r/3}$. If $r \equiv 1 \pmod 3$, then color rows as $(too)^{\lfloor r/3 \rfloor}t$. If $r \equiv 2 \pmod 3$, then color rows as $(too)^{\lfloor r/3 \rfloor}to$. An open neighborhood unique maximum coloring of $G_{4\times 5}$, where the *toot* pattern is used, is shown in figure 4.10.

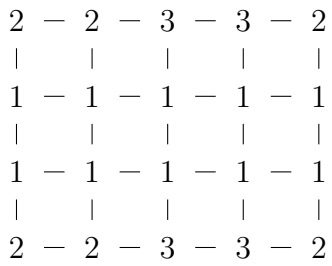

Figure 4.10: An open neighborhood unique maximum coloring of the 4×5 grid

For closed neighborhood unique maximum coloring, depending on the value of c, we use for vertices in each row of $G_{r\times c}$ one of the following two

colorings:

$$t = \begin{cases} (232)^{c/3}, & \text{if } c \equiv 0 \pmod 3 \\ (322)^{\lfloor c/3 \rfloor}3, & \text{if } c \equiv 1 \pmod 3 \\ (322)^{\lfloor c/3 \rfloor}32, & \text{if } c \equiv 2 \pmod 3 \end{cases} \qquad \text{and} \qquad o = 1^c.$$

If $r \equiv 0 \pmod 3$, then color rows as $(oto)^{r/3}$. If $r \equiv 1 \pmod 3$, then color rows as $(too)^{\lfloor r/3 \rfloor}t$. If $r \equiv 2 \pmod 3$, then color rows as $(too)^{\lfloor r/3 \rfloor}to$. A closed neighborhood unique maximum coloring of $G_{4\times 5}$, where the *toot* pattern is used, is shown in figure 4.11.

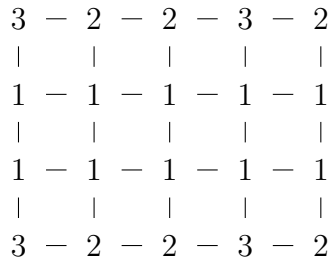

Figure 4.11: A closed neighborhood unique maximum coloring of the 4×5 grid

Complete graphs

Proposition 4.49. *For $n \geq 2$, $\chi_{\text{cf}}^{\text{cn}}(K_n) = \chi_{\text{um}}^{\text{cn}}(K_n) = 2$.*

For an optimal unique maximum neighborhood coloring, color a vertex v with color 2 and every other vertex with color 1.

Proposition 4.50. *For $n \geq 3$, $\chi_{\text{cf}}^{\text{on}}(K_n) = \chi_{\text{um}}^{\text{on}}(K_n) = 2$.*

For an optimal unique maximum neighborhood coloring, color two vertices with colors 2 and 3 and every other vertex with color 1. However, $\chi_{\mathrm{cf}}^{\mathrm{on}}(K_2) = \chi_{\mathrm{um}}^{\mathrm{on}}(K_2) = 1$.

Complete multipartite graphs

For multipartite graphs, three colors are enough as exhibited by a coloring based on a dominating set or total dominating set: With two vertices, each in different parts of the graph, you can totally dominate every vertex.

4.4 Ramsey-type results for conflict-free coloring

In this section, we study a Ramsey-type problem related to conflict-free coloring.

4.4.1 Conflict-free van der Waerden numbers

An *r-coloring* of $\{1, \ldots, n\}$ is a function $C\colon \{1, \ldots, n\} \to \{1, \ldots, r\}$. We consider properties of *arithmetic progressions* of length k of $\{1, \ldots, n\}$, that are induced by a coloring of $\{1, \ldots, n\}$. To avoid degenerate cases, we assume the length k is strictly greater than 1. We say that an arithmetic progression is *monochromatic* if all its elements have the same color. We say that an arithmetic progression is *conflict-free* if there is an element of it with a unique

color in the arithmetic progression. We say that an arithmetic progression is *unique-max* if the maximum color occurring in the arithmetic progression is unique in the arithmetic progression. Colorings of the first n positive integers can also be viewed as strings over an alphabet of r symbols.

The classic van der Waerden numbers are defined as follows.

Definition 4.51. $W(r, k)$ is the smallest number n such that any r-coloring of $\{1, \ldots, n\}$ necessarily has an arithmetic progression of length k which is monochromatic.

The only exactly known non-trivial classic van der Waerden numbers are shown in table 4.1, from Landman and Robertson [2004], Kouril and Paul [2008].

$r \setminus k$	3	4	5	6
2	9	35	178	1132
3	27			
4	76			

Table 4.1: Classic van der Waerden numbers

We consider variations of the van der Waerden numbers as follows.

Definition 4.52. $W_{\mathrm{cf}}(r, k)$ is the smallest number n such that any r-coloring of $\{1, \ldots, n\}$ necessarily has an arithmetic progression of length k which is non-conflict-free.

Definition 4.53. $W_{\mathrm{um}}(r, k)$ is the smallest number n such that any r-coloring of $\{1, \ldots, n\}$ necessarily has an arithmetic progression of length k which is non-unique-max.

Observe that $W(r,2) = W_{\text{cf}}(r,2) = W_{\text{um}}(r,2) = r+1$.

Definition 4.54. We denote the minimum prime factor of k by $p_{\min}(k)$.

We are now ready to prove the main result of this section.

Proposition 4.55. *For every k, $W_{\text{um}}(2,k) = (k-1)p_{\min}(k)+1$.*

Proof. First, we prove that for $n = (k-1)p_{\min}(k)$, the following coloring has the property that every arithmetic progression of length k is unique max:

$$C(i) = \begin{cases} 2, & \text{if } i = \lambda k, \text{ where } \lambda \in \{1, \dots, p_{\min}(k)-1\} \\ 1, & \text{otherwise} \end{cases}$$

In fact any coloring which is a substring of length $n = (k-1)p_{\min}(k)$ of the infinite string

$$\sigma = \underbrace{11\dots12}_{k}\underbrace{11\dots12}_{k}\dots$$

is a legal coloring. Moreover, these are the only possible legal colorings, because if a coloring is legal for the step 1 arithmetic progressions, it has to be the case that exactly one of every k consecutive numbers is colored with 2, and also that any two consecutive occurrences of color '2' in the coloring are exactly at distance k (i.e, have $k-1$ 1's between). However, for simplicity, we will argue about the specific coloring C given above. It is easy to see that for $n = (k-1)p_{\min}(k)$, only arithmetic progressions with steps $s \in \{1, \dots, p_{\min}(k)-1\}$ are defined. We will prove that every arithmetic

progression of step s:

$$A = \{a, a+s, \ldots, a+is, \ldots, a+(k-1)s\}$$

contains exactly one number colored with color 2.

For the sake of contradiction, assume A contains at least two occurrences of color 2, for indices i_1 and i_2, i.e., elements $a+i_1 s$ and $a+i_2 s$ are colored with color 2. Therefore, $\lambda_1 k = a+i_1 s$ and $\lambda_2 k = a+i_2 s$ for some λ_1, λ_2 in $\{1, \ldots, p_{\min}(k)-1\}$. By combining the previous two equations, we get $(\lambda_2 - \lambda_1)k = (i_2 - i_1)s$, which is impossible because $s < p_{\min}(k)$ and therefore $(i_2 - i_1)$ must be divisible by k, but this is impossible because $i_2 - i_1 < k$.

For the sake of contradiction, assume A contains no occurrence of color 2. Then for every $i \in \{0, \ldots, k-1\}$, element $a+is \not\equiv 0 \pmod{k}$. Therefore, by the pigeonhole principle, there are two indices i_1 and i_2, such that elements $a+i_1 s$ and $a+i_2 s$ are congruent modulo k, i.e., $a+i_1 s = \lambda_1 k + b$ and $a+i_2 s = \lambda_2 k + b$ for some integers λ_1, λ_2. By combining the two equations, we get $(\lambda_2 - \lambda_1)k = (i_2 - i_1)s$, which is a contradiction as above.

Then, we prove that any 2-coloring for $n' = (k-1)p_{\min}(k)+1$ contains an arithmetic progression of length k which is not unique max. Assume for the sake of contradiction that there is a legal coloring. As before, it has to be the case that the coloring is a substring of σ, because of the step 1 arithmetic progressions. Therefore, occurrences of color 2 are at distance k. Consider

the step $p_{\min}(k)$ length k arithmetic progression:

$$X = \{1, 1 + p_{\min}(k), \ldots, 1 + ip_{\min}(k), \ldots, 1 + (k-1)p_{\min}(k)\}$$

For X to be unique max, there must be an $i \in \{0, \ldots, k-1\}$ for which element $1 + ip_{\min}(k)$ of X is colored with color 2. For X to be unique max, it has to also be the case that no other element of X is at distance k from $1 + ip_{\min}(k)$. Therefore, it must be the case that:

$$\text{not} \quad (i - k/p_{\min}(k) \geq 0 \quad \text{or} \quad i + k/p_{\min}(k) \leq k - 1)$$

or equivalently:

$$i - k/p_{\min}(k) \leq -1 \quad \text{and} \quad i + k/p_{\min}(k) \geq k$$

By combining the above inequalities, we get $2k/p_{\min}(k) \geq (k+1)$ which is false because $p_{\min}(k) \geq 2$. □

4.5 Discussion and open problems

In this chapter, we collected several results on conflict-free coloring. We considered a variation of conflict-free coloring where only a given subset of intervals are required to have the conflict-free property and provided a polynomial time 2-approximation algorithm, improving on a 4-approximation algorithm

by Katz et al. [2007]. It still remains open whether the problem of computing an optimal coloring in this setting is NP-hard.

Then we studied vertex conflict-free colorings of graphs with respect to all paths in graphs. A closely related vertex graph coloring is ordered coloring. We constructed graphs for which the two chromatic numbers, conflict-free chromatic number $\chi_{\mathrm{cf}}(G)$ and ordered chromatic number $\chi_{\mathrm{o}}(G)$, are different. It would be interesting to characterize the graphs for which the two numbers are the same. It would also be interesting to prove an upper bound on the difference of the two chromatic numbers, $\chi_{\mathrm{cf}}(G)$ and $\chi_{\mathrm{o}}(G)$. Another open problem is determining the computational complexity of computing $\chi_{\mathrm{cf}}(G)$.

We also considered vertex conflict-free colorings of graphs with respect to neighborhoods of the graph. Our main result was that for a graph with n vertices $2\sqrt{n}$ colors are enough to conflict-free color with respect to neighborhoods. We proved that the aforementioned result is order tight for closed neighborhoods. For open neighborhoods, Pach and Tardos [2008] provide an improved upper bound of $O(\log^{2+\varepsilon} n)$ and a lower bound of $\Omega(\log n)$; it remains an interesting open problem to close this gap. Another open problem is the computational complexity of computing optimal conflict-free colorings with respect to neighborhoods. We conjecture that it is NP-complete.

Finally, we initiated the study of Ramsey-type results for conflict-free coloring, by considering conflict-free colorings of $\{1, \ldots, n\}$ with respect to arithmetic progressions of length k. In the traditional Ramsey theory setting this problem is related to van der Waerden numbers. We computed

exactly the unique maximum van der Waerden numbers for two colors. We believe that it would be interesting to compute such numbers (i.e., where conflicts start occurring) for several kinds of problems, because they will be smaller than their monochromatic Ramsey-type number counterparts, and they might be easier to compute.

Bibliography

Aardal, Karen I., Stan P.M. van Hoesel, Arie M.C.A Koster, Carlo Mannino, and Antonio Sassano. Models and solutions techniques for frequency assignment problems. Technical Report ZIB Report 01-40, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Germany, 2001.

Ajwani, Deepak, Khaled Elbassioni, Sathish Govindarajan, and Saurabh Ray. Conflict-free coloring for rectangle ranges using $O(n^{.382})$ colors. In *Proceedings of the 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 181–187, 2007.

Allouche, Jean-Paul and Jeffrey Shallit. *Automatic sequences: theory, applications, generalizations*. Cambridge University Press, 2003.

Alon, Noga and Shakhar Smorodinsky. Conflict-free colorings of shallow discs. In *Proceedings of the 22nd Annual ACM Symposium on Computational Geometry (SoCG)*, pages 41–43, 2006.

Alon, Noga and Joel Spencer. *The probabilistic method*. Wiley, 2nd edition, 2000.

Alon, Noga, Jarosław Grytczuk, Mariusz Hałuszczak, and Oliver Riordan. Nonrepetitive colorings of graphs. *Random Structures and Algorithms*, 21 (3-4):336–346, 2002.

Bar-Noy, Amotz, Panagiotis Cheilaris, and Shakhar Smorodinsky. Conflict-free coloring for intervals: from offline to online. In *Proceedings of the 18th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 128–137, 2006.

Bar-Noy, Amotz, Panagiotis Cheilaris, Michael Lampis, and Stathis Zachos. Conflict-free coloring graphs and other related problems. Manuscript, 2007a.

Bar-Noy, Amotz, Panagiotis Cheilaris, Svetlana Olonetsky, and Shakhar Smorodinsky. Online conflict-free coloring for geometric hypergraphs. In *Proceedings of the 23rd European Workshop on Computational Geometry (EWCG)*, pages 106–109, 2007b.

Bar-Noy, Amotz, Panagiotis Cheilaris, Svetlana Olonetsky, and Shakhar Smorodinsky. Online conflict-free colorings for hypergraphs. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 219–230, 2007c.

Bar-Noy, Amotz, Panagiotis Cheilaris, Svetlana Olonetsky, and Shakhar Smorodinsky. Weakening the online adversary just enough to get optimal conflict-free colorings for intervals. In *Proceedings of the 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 194–195, 2007d.

Bar-Noy, Amotz, Panagiotis Cheilaris, and Shakhar Smorodinsky. Deterministic conflict-free coloring for intervals: from offline to online. *ACM Transactions on Algorithms*, 4(4):44:1–44:18, 2008.

Barát, János and David R. Wood. Notes on nonrepetitive graph colouring. arXiv math 0509608, 2005.

Bodlaender, Hans L., John R. Gilbert, Hjálmtyr Hafsteinsson, and Ton Kloks. Approximating treewidth, pathwidth, frontsize, and shortest elimination tree. *Journal of Algorithms*, 18(2):238–255, 1995.

Bollobás, Béla. *Modern Graph Theory*. Springer, 1998.

Borodin, Allan and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.

Brešar, Boštjan, Jarosław Grytczuk, Sandi Klavžar, Stanisław Niwczyk, and Iztok Peterin. Nonrepetitive colorings of trees. *Discrete Mathematics*, 307: 163–172, 2007.

Brooks, R. L. On colouring the nodes of a network. *Proceedings of the Cambridge Philosophical Society*, 37:194–197, 1941.

Chan, Joseph, Francis Chin, Xiangyu Hong, and Hingfung Ting. Dynamic offline conflict-free coloring for unit disks. In *Proceedings of the 6th Workshop on Approximation and Online Algorithms (WAOA)*, 2008.

Cheilaris, Panagiotis. *Algorithms and Complexity: graph and hypergraph colorings*. Thesis, National Technical University of Athens, 2006.

Cheilaris, Panagiotis, Ernst Specker, and Stathis Zachos. Neochromatica. Submitted, 2006.

Chen, Ke. How to play a coloring game against a color-blind adversary. In *Proceedings of the 22nd Annual ACM Symposium on Computational Geometry (SoCG)*, pages 44–51, 2006.

Chen, Ke, Haim Kaplan, and Micha Sharir. Online conflict free coloring for halfplanes, congruent disks, and axis-parallel rectangles. Manuscript, 2006.

Chen, Ke, Amos Fiat, Haim Kaplan, Meital Levy, Jiří Matoušek, Elchanan Mossel, János Pach, Micha Sharir, Shakhar Smorodinsky, Uli Wagner, and Emo Welzl. Online conflict-free coloring for intervals. *SIAM Journal on Computing*, 36(5):1342–1359, 2007.

Chen, Xiaomin, János Pach, Mario Szegedy, and Gábor Tardos. Delaunay graphs of point sets in the plane with respect to axis-parallel rectangles. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 94–101, 2008.

Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Cliff Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001.

Diestel, Reinhard. *Graph Theory*. Springer, 3rd edition, 2005.

Djidjev, Hristo Nicolov. On the problem of partitioning planar graphs. *SIAM Journal on Algebraic and Discrete Methods*, 3:229–240, 1982.

Duff, Iain S. Parallel implementation of multifrontal schemes. *Parallel Computing*, 3(3):193–204, 1986.

Elbassioni, Khaled and Nabil H. Mustafa. Conflict-free colorings of rectangles ranges. In *Proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 254–263, 2006.

Erdős, Pál and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In Hajnal, András, Richard Rado, and Vera T. Sós, editors, *Infinite and Finite Sets*, pages 609–627. North Holland, 1975.

Erdős, Pál and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

Erlebach, Thomas, Aris Pagourtzis, Katerina Potika, and Stamatis Stefanakos. Resource allocation problems in multifiber WDM tree networks. In *Proceedings of 29th Workshop on Graph Theoretic Concepts in Computer Science (WG 2003), LNCS 2880*, pages 218–229, 2003.

Even, Guy, Zvi Lotker, Dana Ron, and Shakhar Smorodinsky. Conflict-free colorings of simple geometric regions with applications to frequency assignment in cellular networks. *SIAM Journal on Computing*, 33:94–136, 2003.

Fiat, Amos, Meital Levy, Jiří Matoušek, Elchanan Mossel, János Pach, Micha Sharir, Shakhar Smorodinsky, Uli Wagner, and Emo Welzl. Online conflict-free coloring for intervals. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 545–554, 2005.

George, Alan, Michael T. Heath, Esmond G. Ng, and Joseph W. H. Liu. Symbolic cholesky factorization on a local-memory multiprocessor. *Parallel Computing*, 5(1-2):85–95, 1987.

Graham, Ronald L., Bruce L. Rothschild, and Joel H. Spencer. *Ramsey Theory*. Wiley, 2nd edition, 1990.

Gyárfás, András and Jenö Lehel. On-line and first fit coloring of graphs. *Journal of Graph Theory*, 12(2):217–227, 1988.

Har-Peled, Sariel and Shakhar Smorodinsky. Conflict-free coloring of points and simple regions in the plane. *Discrete and Computational Geometry*, 34:47–70, 2005.

Hilbert, David. Über die Irreduzibilität ganzer rationaler Functionen mit ganzzahligen Coefficienten. *Journal für die Reine und Angewandte Mathematik*, 110:104–129, 1892.

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

Iyer, Ananth V., H. Ronald Ratliff, and Gopalakrishanan Vijayan. Optimal node ranking of trees. *Information Processing Letters*, 28:225–229, 1988.

Jordan, Camille. Sur les assemblages de lignes. *Journal für die Reine und Angewandte Mathematik*, 70:185–190, 1869.

Kaplan, Haim and Micha Sharir. Online CF coloring for halfplanes, congruent disks, and axis-parallel rectangles. Manuscript, 2004.

Katchalski, Meir, William McCuaig, and Suzanne Seager. Ordered colourings. *Discrete Mathematics*, 142:141–154, 1995.

Katz, Matthew J., Nissan Lev-Tov, and Gila Morgenstern. Conflict-free coloring of points on a line with respect to a set of intervals. In *Proceedings of the 19th Canadian Conference on Computational Geometry (CCCG)*, pages 93–96, 2007.

Keszegh, Balázs. Weak conflict-free colorings of point sets and simple regions. In *Proceedings of the 19th Canadian Conference on Computational Geometry (CCCG)*, pages 97–100, 2007.

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

Kouril, Michal and Jerome L. Paul. The Van der Waerden number $W(2, 6)$ is 1132. *Experimental Mathematics*, 17:53–61, 2008.

Landman, Bruce M. and Aaron Robertson. *Ramsey Theory on the integers*. American Mathematical Society, 2004.

Leiserson, Charles E. Area-efficient graph layouts (for VLSI). In *Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 270–281, 1980.

Lewis II, Philip M., Richard Edwin Stearns, and Juris Hartmanis. Memory bounds for recognition of context-free and context-sensitive languages. In *Proceedings of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 191–202, 1965.

Lipton, Richard J. and Robert E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

Liu, Joseph W. H. Computational models and task scheduling for parallel sparse cholesky factorization. *Parallel Computing*, 3(4):327–342, 1986.

Liu, Joseph W. H. Reordering sparse matrices for parallel elimination. *Parallel Computing*, 11(1):73–91, 1989.

Llewellyn, Donna Crystal, Craig A. Tovey, and Michael A. Trick. Local optimization on graphs. *Discrete Applied Mathematics*, 23(2):157–178, 1989.

Llewellyn, Donna Crystal, Craig A. Tovey, and Michael A. Trick. Erratum: Local optimization on graphs. *Discrete Applied Mathematics*, 46(1):93–94, 1993.

Pach, János and Gábor Tardos. Conflict-free coloring of graphs and hypergraphs. Manuscript, 2008.

Pach, János and Géza Tóth. Conflict free colorings. In *Discrete and Computational Geometry, The Goodman-Pollack Festschrift*, pages 665–671. Springer Verlag, 2003.

Prouhet, E. Mémoire sur quelques relations entre les puissances des nombres. *Comptes Rendus de l'Académie des Sciences, Paris, Série I*, 33:225, 1851.

Raghavan, Prabhakar and Marc Snir. Memory versus randomization in online algorithms. In *Proceedings of the 16th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 687–703, 1989.

Ramsey, Frank P. On a problem of formal logic. *Proceedings of the London Mathematical Society*, 30:264–286, 1930.

Schäffer, Alejandro A. Optimal node ranking of trees in linear time. *Information Processing Letters*, 33(2):91–96, 1989.

Schiermeyer, Ingo, Zsolt Tuza, and Margit Voigt. On-line rankings of graphs. *Discrete Mathematics*, 212(1–2):141–147, 2000.

Schur, Issai. Über die Kongruenz $x^m + y^m \equiv z^m \pmod{p}$. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 25:114–116, 1916.

Sen, Arunabha, Haiyong Deng, and Sumanta Guha. On a graph partition problem with application to VLSI layout. *Information Processing Letters*, 43(2):87–94, 1992.

Smorodinsky, Shakhar. A note on the online first-fit algorithm for coloring k-inductive graphs. To appear in *Information Processing Letters*, 2008.

Smorodinsky, Shakhar. *Combinatorial Problems in Computational Geometry.* PhD thesis, School of Computer Science, Tel-Aviv University, 2003.

Smorodinsky, Shakhar. On the chromatic number of some geometric hypergraphs. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2006.

Smorodinsky, Shakhar. On the chromatic number of some geometric hypergraphs. *SIAM Journal on Discrete Mathematics*, 21(3):676–687, 2007.

Thue, Axel. Über unendliche Zeichenreihen. *Norske vid. Selsk. Skr. Mat. Nat. Kl*, 7:1–22, 1906.

van der Waerden, Bartel L. Beweis einer Baudetschen Vermutung. *Nieuw Archief voor Wiskunde*, 15:212–216, 1927.

Vazirani, Vijay V. *Approximation Algorithms.* Springer, 2001.

Zachos, Stathis. Notes on Algorithms and Complexity, 2006.