Distributed conflict-free $o(\Delta)$-coloring

Noé Vincent

Département d'informatique, École Normale Supérieure de Rennes, Bruz 35170, France

Abstract. This work is the report on the internship Noé Vincent carried out at IRIF[a], Paris, France, under the supervision of Pierre Fraigniaud, CNRS researcher at IRIF.
We studied graph conflict-free coloring under the distributed LOCAL model.
A vertex coloring of a graph $G = (V, E)$ maps a color to each vertex of V. It is called conflict-free if for each vertex $v \in V$ there exists a color associated with a unique vertex in v's closed neighborhood, following the definition used by Abel, Alvarez et al. in [1].
The LOCAL model is a distributed computing model where each node of the graph has a unique ID, infinite computing power, and can communicate any-size messages to its neighbors during communication rounds. The complexity of an algorithm is measured by the number of rounds needed for its execution; this reflects the distance of the furthest information a node has to know to achieve its task.
Using a bound from [3], we show that any graph of maximal degree Δ is conflict-free $O(\log^2 \Delta)$-colorable and Δ-colorable.
We provide a distributed algorithm solving conflict-free $O(\log n \log^2 \Delta)$ coloring in $\tilde{O}(\log^2 n)$ rounds based on network decomposition [5]. Moreover, we provide initial work aiming to provide an algorithm based on fractional assignment rounding [4] [6].
In section 4, we show that conflict-free coloring a graph with strictly less than Δ colors is not greedily completable, ie, some partial coloring cannot be extended without any modification of the partial output. We show that it is sometimes necessary to modify the output of nodes at distance $\Omega(\log n)$ to extend the coloring to a new node. Formally, we prove this problem to be $\Omega(\log n)$-mendable, which helps understand the complexity of the problem and the barriers encountered while building the algorithms.

Keywords: Conflict-free coloring · Distributed algorithms · Graph theory

[a] Institut de Recherche en Informatique Fondamentale (CNRS and Université Paris-Cité joint research unit)

1 Introduction

1.1 The LOCAL MODEL

Distributed computing Distributed computing refers to a computing model where there is not one but a network of computers, and each of those can execute programs in parallel and send messages to each other. In general, when considering graph theory problems, we consider that the communication network is the given graph and that each vertex represents a computer. At the beginning, we assume that each node only knows a limited quantity of information depending on the model.

The LOCAL model The LOCAL Model of distributed computing is a model where each node knows its unique ID (an integer), has unlimited computing power, unlimited storage space, and can send messages of any size to its neighbors. The communication happens in synchronous rounds, ie, messages are sent and received at the same time across the network. The complexity of an algorithm is defined here as the maximum number of communication rounds needed to complete the algorithm.

This model is motivated by the fact that it helps understand how much information a node needs to choose its final state. Indeed, during a communication round, the messages travel through only one edge. Thus, during the execution of an algorithm, a node cannot know information about another node at a distance more than the number of rounds used by the algorithm. So, one can see the LOCAL complexity as the radius of the neighborhood that influences the output of a node.

1.2 Locally checkable labeling

A locally checkable labeling [7] is a labeling of the vertices that can be checked by only looking at the labels of nodes within a constant radius around the node. Formally, a locally checkable labeling of a graph of maximal degree Δ can be defined as a finite set of labels and a finite set of labeled balls ("good balls") of constant radius and degree at most Δ.

Classical LCL One of the most studied LCLs is finding a $(\Delta + 1)$-coloring. It denotes the following problem: find a vertex labeling (using at most $\Delta + 1$ distinct labels (colors)) such that no edge is monochromatic (ie, both endpoints have the same label). This problem is locally checkable; indeed, to verify the validity of a coloring, one only needs to check, for each node, the labeling of nodes at a distance of at most one from the said node.

Another classical LCL is finding a Maximal Independent Set. The goal is to find a labeling (with labels being 1 or 0) of the node such that no neighbors are both labeled with 1, and there are no 0-labeled nodes that can change their color to 1. This problem is also locally checkable; indeed, to verify the validity of an

MIS, one only needs to check, for each node, the labeling of nodes at a distance of at most one from the said node.

The state-of-the-art algorithms for $\Delta + 1$-coloring and MIS [5] solve these problems in $\tilde{O}(\log^{5/3} n)$ [b] rounds in the LOCAL model and are based on a combination of different ingredients, including network decomposition and fractional assignment rounding.

1.3 Network decomposition

Network decomposition refers to the idea of splitting the network into small clusters to help the execution by solving the problem on those parts before turning those partial solutions into a global solution.

Definition 1 ((c, d)-decomposition). *A (c, d)-decomposition of the network is a partitioning of the vertices in c partitions $V_1, ...V_c$ such that $\forall i \in [\![1, c]\!]$ any connected component in V_i has diameter d in G (ie for each two node $u, v \in V_i$ connected in $G[V_i]$, there is $d_G(u, v) \leq d$).*

Theorem 1 (Algorithm for network decomposition th. 1.1 in [5]). *There is a deterministic distributed algorithm that in any n-node graph $G = (V, E)$ computes a $(c \in O(\log n), d \in O(\log n))$-decomposition in $\tilde{O}(\log^2 n)$ rounds of the LOCAL model.*

Once a (c, d)-decomposition is computed. Each partition will execute the following steps: each node gathers the topology of its cluster and executes a centralized algorithm to solve the problem on said cluster. Once done, each node has to compute its final output based on the centralized algorithm's output and on the previously computed outputs from neighboring nodes.

1.4 Fractional assignment rounding

The rounding method for labeling problems [4][6] is a method inspired by basic randomized algorithms for labeling problems. Indeed, those algorithms would pick a label randomly among the available ones and then terminate if the labeling is valid or try again.

Instead of picking a label at random, the rounding method "modifies the probability distribution" over the labels until one of the labels has probability one (and the other labels have probability 0). To do so, it will begin with a fractional assignment, which can be seen as a probability distribution over the labels, and at each round, it will increase or decrease some of the "probabilities" in a way that does not increase the value of a weight function over the graph. For instance, in the traditional graph coloring problem, such weight function can be seen as the expected number of monochromatic edges if the nodes pick their colors according their probability distribution, this value being kept unchanged, at the end we

[b] $f \in \tilde{O}(g) \iff \exists c, k, n_0, \forall n \geq n_0, 0 \leq f(n) \leq c \cdot g(n) \cdot log^k(n)$

have a color assignment with the expected number of monochromatic edges (and thus the expected number of valid edges).

From this color assignment, one can keep a part of those valid edges and repeat the procedure on the rest of the graph. If the removed part of the graph is of size a constant fraction of the size of the graph, then $O(\log n)$ repetitions are enough to label each node of the graph.

We provide preliminary work to use this method to find conflict-free $o(\Delta)$-coloring, but no complete algorithms because, unlike $(\Delta+1)$-coloring, this problem is not greedily completable (see theorem 9).

1.5 Our results

Using a bound from [3], we prove, in section2, that for any graph G of size $n = |V|$ and maximum degree Δ, there exists a conflict-free k-coloring where $k \in O(\log^2(\Delta))$.

In section 3, we provide a new distributed algorithm for computing a conflict-free $O(\log n \cdot \log^2 \Delta)$-coloring in $\tilde{O}(\log^2 n)$ rounds in the local model. Then we provide preliminary work for using fractional assignment rounding.

In section 4, we prove that conflict-free $(\leq \Delta)$-coloring[c] is not greedily completable, ie $\forall \Delta \in \mathbb{N}$ there exists some partial cf k-coloring of a graph of maximum degree Δ that cannot be extended.

We also provide some mendability [2] results in section 4. Informally, we consider a problem to be r-mendable if one can choose the output of a node v without modifying the output of nodes that are at a distance of more than r from v. We prove that conflict-free $(< \Delta)$-coloring is $\Omega(log(n))$-mendable.

2 Goals of the internship

This internship aimed to produce efficient algorithm(s) based on the methods presented above to address the problem of graph conflict-free coloring in the LOCAL model.

Notations For any $\mathbb{K}$, $\mathbb{K}^* = \mathbb{K} \setminus \{0_\mathbb{K}\}$, $\mathbb{K}^+ = \mathbb{K} \setminus \{x \in \mathbb{K}, x < 0_\mathbb{K}\}$. Where $0_\mathbb{K}$ is the neutral element for the addition in $\mathbb{K}$.

The set of integers ranging from $a \in \mathbb{N}$ to $b \in \mathbb{N}, b \geq a$ is denoted $[\![a, b]\!]$.

Let a graph $G = (V, E)$, $\forall v \in V$,

$$N(v) = \{v\} \cup \{v', \{v, v'\} \in E\}$$

denotes the (closed) neighborhood of v.

Let $S \subseteq V$ be a subset of vertices, $G[S]$ denotes the graph induced by S (ie the graph $(S, E \cap (S \times S))$.

For any $v \in V, r \in \mathbb{N}$, $B_r(v)$ is the r-radius ball around V, formally $B_r(v) = G[\{u \in V | d_G(v, u) \leq r\}]$

[c] any $k \in \mathbb{N}, k \leq \Delta$

Unless otherwise stated, Δ denotes the maximal degree of the considered graph, n denotes the number of vertices of the latter.

Conflict-free coloring Conflict-free coloring can be seen as a generalization of proper coloring to hypergraphs, ie, graphs whose edges are sets of vertices of any size. If each hyper-edge is of size 2, then the two definitions match.

Definition 2 (Conflict-free k-coloring of hypergraph). *Let $H = (V, S)$ be a hypergraph. $\chi : V \to [\![1, k]\!]$ is a cf k-coloring if $\forall s \in S$:*

$$\exists \alpha \in [\![1, k]\!], \exists! v \in s, \chi(v) = \alpha$$

Graph conflict-free coloring is such that each node v has in its neighborhood a "conflict-free neighbor". The latter is a colored vertex whose color is not worn by any other neighbors of v. This follows the definition introduced in [1].

Note that this does not imply that each node is colored.

This is a conflict-free coloring on the hypergraph $H = (V', S)$ where V' is the dominating subset of V containing the colored nodes, $S = \{N(x), \forall x \in V\}$

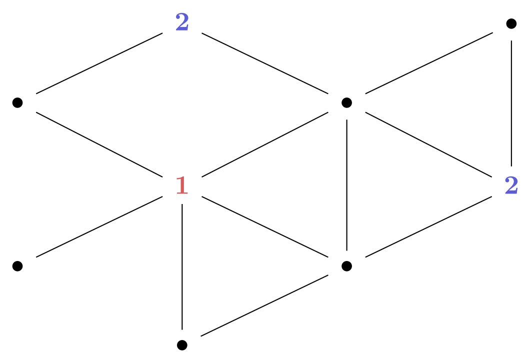
Definition 3 (Graph conflict-free coloring). *Let $G = (V, E)$ be a graph, $\chi : V \to [0, k]$ is a cf k-coloring if :*

$$\forall u \in V, \exists \alpha \in [\![1, k]\!], \exists! v \in N(u), \chi(v) = \alpha$$

We call v a conflict-free neighbor of u. If there are multiple neighbors of unique color, then the cf neighbor is chosen arbitrarily.

Note that the cf-neighbor necessarily wears a non-zero color. The "color" zero is assigned to non-colored vertices.

Here is a conflict-free 2-coloring of a graph G, the nodes labeled with $\bullet$ are the non colored (or zero colored) nodes.



Motivation of the internship Conflict-free coloring is motivated by the frequency assignment problem. Consider a vast factory where each equipment is controlled using wireless technology. In this factory, some antennas are installed, and one wants to know how to assign frequencies to those antennas so that each equipment receives its instructions without interference. Those interferences are produced when equipment receives two signals from distinct antennas on the same frequency. Moreover, the size of the spectrum of frequencies is costly, so the goal is to minimize the number of frequencies used. In such a context, non colored nodes can be seen as turned-off antennas.

This problem was well studied in the centralized model [1],[8],[3], but no known specialized distributed algorithms existed until then. Moreover, a complexity wall for $\Delta + 1$-coloring, MIS etc... was recently broken by the rounding method developped by Ghaffari, Kuhn et al. [6] and Faour, Ghaffari et al. [4] as well as the complexity of graph decomposition by Ghaffari and Grunau [5] which again improved the complexity of MIS and $\Delta + 1$-coloring. So, the initial idea of this internship was to explore whether those efficient methods could be adapted to our problem. In other words, the goal was to understand how the slight difference between this problem and usual vertex coloring impacts its "locality" and whether conflict-free coloring was part of those problems where those standard methods can be applied easily.

Known centralized algorithms Some centralized algorithms solving conflict-free $O(poly(\log \Delta))$-coloring were presented in previous works ([8],[3]). In [8], is presented a centralized algorithm solving conflict-free $O(\log^{2+\epsilon} n)$-coloring.

The output of this algorithm is a variant of conflict-free coloring where each node is colored (ie, no color 0). This is also a conflict-free coloring, so it solves the version studied here.

Theorem 2 (Lovasz Local Lemma). *Let X be a set of mutually independent variables. Let $B_0, ..., B_n$ be a set of bad events, each is defined on $A_i \subset X$. We consider B_i and B_j to be dependent if $A_i \cap A_j \neq \emptyset$.*

If each bad event is dependent on at most $d \in \mathbb{N}$ other bad events and the probability of any bad event happening is bounded by $p \in [0,1]$, then the Lovasz Local Lemma states that:

$e.d.p < 1 \Rightarrow$ there is a valuation of the variables so that no bad event happens.

From the bad events, one can produce a dependency graph where bad events share an edge if they are dependent.

In the case of conflict-free coloring of a graph $G = (V, E)$. The graph $G^2 = (V, E \cup \{\{i,j\}, d_G(i,j) < 2\}$ is the dependency graph of the bad events: $(B_v)_{v \in V}$ where B_v is true when the neighborhood of v is not conflict-free colored. The set of independent variables is the set of $(x_v)_{v \in V}$ where x_v refers to the color assigned to v.

To find the value of $(x_v)_{v \in V}$, there are some distributed algorithms, and most of them use random sampling (ie, picking at random a possible value and evaluating the bad events). The deterministic algorithms use a heuristic to choose among them.

The centralized algorithm from Pach and Tardos in [8] uses, at some point, a deterministic Algorithmic Local Lemma. Even though there exist distributed deterministic Lovász Local Lemma algorithms, the choice of the color is made so that the number of "dangerous (2,3)-trees" over the graph does not increase. Following such information would require knowing the whole graph and thus is not suitable in distributed computing when the desired complexity is lower than the diameter.

In [3], a trickier version of the algorithm from [8] is presented. This algorithm uses a very similar approach but manages to reduce the number of colors to $O(\log^2 \Delta)$.

Theorem 3 (Centralized algorithm for conflict-free coloring). *There is a deterministic centralized algorithm that for any graph $G = (V, E)$ computes a conflict-free k-coloring with $k \in O(\log^2 \Delta)$.*

Proof. In the proof of th. 3 of [3], there is a deterministic algorithm for computing a variant of conflict-free coloring where each node is colored (no color 0).

The output of this algorithm is a conflict-free coloring, then it can be used as it is presented in [3]

However, this algorithm assigns conflict-free neighbors to each node and then colors every remaining uncolored node with a new color. This color can be renamed to 0, and we obtain an output coherent with the definition studied in this work.

Conflict-free chromatic number This denotes the number of colors needed to create a conflict-free coloring.

Theorem 4 (Conflict-free Brooks' Theorem). *If G=(V,E) is a connected graph of degree at most Δ. Then one needs at most Δ colors to produce a CF-coloring of G.*

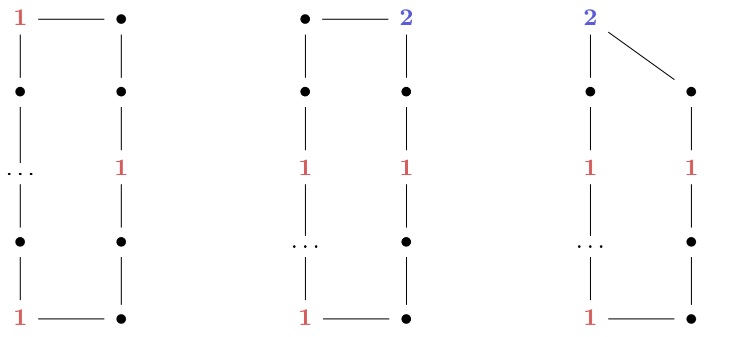
Proof. If G is not complete nor an odd cycle, then the usual Brooks' theorem provides a Δ-coloring which is a CF Δ-coloring (each vertex is its own CF-neighbor).

If G is complete, one just needs to color with color **1** one of the nodes and let every other node uncolored. This is a valid CF-coloring. Indeed, the graph being complete, each node has the colored node as its conflict-free neighbor.

If G is an odd cycle (let $n = |V|$):

- If $\exists k'$ such that $n = 3 * k'$, one can assign color **1** on the second of every three consecutive vertices. The colored vertex will then become the CF-neighbor of those three vertices. Such a procedure ensures that each node will only have one colored vertex in its neighborhood. This is then a valid CF 1-coloring of the cycle. (see Fig. 1a)

- If $\exists k'$ such that $n = 3 * k' - 1$, one can follow the same procedure as above on the $3(k' - 1)$ first nodes. There remain two nodes that are not colored. One



(a) $n = 3k'$ (b) $n = 3 * k' - 1$ (c) $n = 3 * k' - 2$

Fig. 1: CF-coloring odd cycles

can color the second of them with **2**. This will provide a conflict-free neighbor to those two remaining nodes, and the very first vertex of the cycle will keep its CF-neighbor of color **1**. (see Fig. 1b)

- If $\exists k'$ such that $n = 3 * k' - 2$, one can follow the same procedure as above to color the $3(k' - 1)$ first nodes and assign **2** to the remaining node. (see Fig. 1c)

Theorem 5 (Asymptotic bound on the CF chromatic number). *Let G be a graph with maximum degree Δ. Then it is conflict-free k-colorable with $k \in O(\log^2 \Delta)$.*

Proof. In th. 3 from [3], the same bound is obtained on a variation of conflict-free coloring, where each node is colored (ie, no color 0). Though a solution to this variant is a solution to the version studied here. Hence, we have that any graph with maximum degree Δ is cf k-colorable with $k \in O(\log^2 \Delta)$.

Known distributed algorithms As said earlier, no known distributed algorithms addressed this problem, especially. But one can easily understand that a traditional graph coloring is a conflict-free coloring. Indeed, in a usual coloring, each node is its conflict-free neighbor. Thus, a distributed coloring algorithm is a distributed conflict-free coloring algorithm.

In th. 4.1 from [6], there is a deterministic distributed algorithm that solves $(\Delta + 1)$-coloring in $O(\log^2 \Delta \cdot \log n)$ rounds.

Unfortunately, $\Delta + 1$ being the smallest number of colors (depending on Δ) that allows for coloring any graph, such a method would use too many colors compared to the conflict-free chromatic number defined earlier. So, using usual coloring algorithms does not seem satisfactory.

3 Distributed algorithms

3.1 Rounding method

Distributed rounding algorithm The following description is greatly inspired by section 2.1 from [4].

Let $G = (V, E)$ be a graph, let k be the size of the palette (ie, the maximal number of labels the algorithm is allowed to use). A fractional label assignment $\lambda : V \to [0,1]^k$ is an assignment of a probability distribution over the labels to each node $v \in V$. For an integer $K \geq 1$ a fractional assignment is called $1/K$-integral if $\forall \alpha \in [\![0, k]\!], \forall v \in V, \lambda_\alpha(v) = a/K$ with $a \in \mathbb{N}$.

For each node $v \in V$, we use $\mathcal{L}(v)$ (and $\Lambda(v)$) to denote the set of possible (fractional) assignments to v. This notation extends trivially to vertex sets and edges.

We define a utility function $u : E \times \mathcal{L}(V) \to \mathbb{R}^{+*}$ and a cost function $c : E \times \mathcal{L}(V) \to \mathbb{R}^{+*}$. That is, for a given color assignment $l \in \mathcal{L}(V)$, u and c assign non-negative utility and cost values $u(e,l)$ and $c(e,l)$ to every edge $e \in E$. We slightly overload the notation and also define u and c for a fractional color assignment $\lambda \in \Lambda(V)$. In this case, utility and cost of e are defined as the expected values of u and c if the colors of the two nodes $V(e)$ of e are chosen independently at random from the distributions given by the fractional color assignment.

Finally for a set of edge $F \subseteq E$, we define $u(F,l) = \sum_{e \in F} u(e,l)$ and $c(F,l) = \sum_{e \in F} c(e,l)$.

Theorem 6 (Rounding of fractional assignment, lemma 2.2 in [4]). *Let $G = (V, E)$ be a multi-graph, which is equipped with utility and cost functions u and c and with an $1/2^k$-integral fractional assignment λ for a given integer $k \geq 1$. Let $\epsilon \in [0,1]$ and $\mu \in]O,1]$ be two parameters. If $u(\lambda) - c(\lambda) \geq \mu.u(\lambda)$. There is an algorithm that returns an integral color assignment l with*

$$u(l) - c(l) \geq (1 - \epsilon) \cdot (u(\lambda) - c(\lambda))$$

Once the integral color assignment l is created, the idea is that some nodes that have a valid color assignment terminate. To do so efficiently, u and c have to be chosen wisely.

For instance, when considering usual $(\Delta + 1)$-coloring, one can choose them so that $u(\lambda) - c(\lambda)$ lower-bounds the expected number of edges that are not monochromatic. Then, using some independent subset of the nodes that have a limited monochromatic degree, one can select a constant part of the nodes and make them terminate. If active nodes are remaining, the algorithm is executed again on those nodes to extend the coloring.

A useful property to have on the utility and cost function is that it enables the terminating part of the graph, at each round, to be of size a constant fraction of the size of the graph. Such a property ensures that $O(\log n)$ repeated executions of the algorithm are enough for the whole graph to be colored.

An algorithm for conflict-free coloring

Theorem 7. *There is an initial assignment λ, a cost function c, and a utility function u so that at the end of the rounding procedure, a constant fraction of the neighborhoods are conflict-free $O(\sqrt{\Delta})$-colored.*

Proof. See appendix 6.1

This last theorem is interesting, but it is not sufficient. Indeed, among the nodes with a conflict-free colored neighborhood, it is not possible for each of those to terminate. Different barriers exist:

- Some may have their CF-neighbor that does not terminate and then changes color. To solve this problem, we could consider that those nodes do not terminate. However, this would imply that the number of terminating nodes might not be a constant fraction of the graph.
- Even without such a situation, it is possible that some nodes create a labeling that forbids every available color to a single non-terminated node (see fig. 2). This makes it impossible for the latter to be colored during the next phase.
- etc

Those barriers are formalized in section 4 by the fact that conflict-free ($< \Delta$)-coloring is not greedily completable. In other words, some partial coloring cannot be extended using the same palette. This makes it difficult to find a way of choosing the nodes that should terminate while still keeping it a constant fraction of the graph.

Note that in the above theorem, $O(\sqrt{\Delta})$ colors are used, but it seems that it could also work with $O(\log^2 \Delta)$ colors.

3.2 Network decomposition

With the Network decomposition, we managed to avoid the barriers described above by using a larger number of colors.

Theorem 8 (Distributed algorithm for CF $O(\log n \cdot \log^2 \Delta)$-coloring). *There is a deterministic distributed algorithm that in any n-node graph $G = (V, E)$ computes a conflict-free k-coloring with $k \in O(\log^2 \Delta \cdot \log n)$ in $\tilde{O}(\log^2(n))$ rounds of communication in the LOCAL model.*

Proof. This algorithm contains two main steps.

First, the algorithm from Theorem 1 is executed. Producing a $(c \in O(\log n), d \in (\log n))$-decomposition of the network.

Then, a series of c stages begin. On stage i, the nodes from V_i will gather the topology of their connected component and the highest color used by neighboring nodes from $V_1 \cup ... \cup V_{i-1}$. To do so, each node will share with its neighbors what it knows from its environment until each node learned the topology of its component. Having that $d \in O(\log n)$, this gathering procedure will take $O(\log n)$ rounds.

Once each node knows its component. They will execute the algorithm from Theorem 3. This algorithm, being deterministic, each node in the component will then obtain the same conflict-free coloring. To ensure compatibility with the coloring of other components, each will offset its colors by the value of the highest previously used color.

At the end of the c stages, each node will be colored, and this coloring will be valid because for each $i \in [\![1, c]\!]$, $G[V_i]$ wears a valid coloring and there are no conflicts between neighboring partitions because each used a distinct palette. This latter property implies that it uses $O(\log n \cdot \log^2 \Delta)$ colors.

The network decomposition has complexity in $\tilde{O}(\log^2 n)$ rounds, the coloring part of the algorithm has complexity in $O(\log^2 n)$ rounds, as the $c \in O(\log n)$ stages each use $O(\log n)$ rounds.

Thus, the complexity of the whole algorithm is in $\tilde{O}(\log^2 n)$.

4 Mendability results

Here we detail and prove some mendability results. Mendability is defined by Balliu, Hirvonen et al in [2]. One can see mendability as the size of the patch needed to fix a hole in a partial solution. In other words, consider that a partial solution makes it impossible to choose the output for a given node; mendability measures the radius of the smallest ball around the given node containing the furthest node one would have to relabel to solve this gridlock. For instance, $\Delta+1$-coloring is 0-mendable because there is always an available color to choose, and 4-coloring a grid is 2-mendable (see [2]). We redefine here the formal objects used to work on mendability for the sake of clarity.

4.1 Locally verifiable Problems

Definition 4. *Locally verifiable problems [2]. A locally verifiable problem Π is defined by some input labels Σ, some output labels Γ, and a verifier ψ. In Π we are given a graph $G = (V, E)$ and an input labeling $\sigma : V \to \Sigma$. The task is to find a solution $\lambda : V \to \Gamma$ that makes the verifier "happy" on each node. In general ψ maps (G, λ, v) to "happy" or "unhappy" and we consider that ψ accepts λ on $G = (V, E)$ if for each $v \in V$, $\psi(G, \lambda, v) = happy$.*

Moreover, this verifier has to be "local", ie, there must exist $r \in \mathbb{N}$ so that $\psi(G; \lambda, v)$ only depends on $B_r(v)$, ie, the r-radius neighborhood of v in G.

Conflict-free k-coloring is locally verifiable. Indeed, it is defined by Σ, the set of IDs, $\Gamma = [\![0, k]\!]$ and ψ the verifier, outputs "happy" for (G, λ, v) if λ provides a conflict-free neighbor to v. ψ is local because it only needs to know $\lambda(N(v))$.

4.2 Greedy completability and local mendability

Definition 5 (Partial solutions of conflict-free coloring [2]). *Let's introduce a new label $\perp$ to denote non-terminated nodes, ie, nodes which have not*

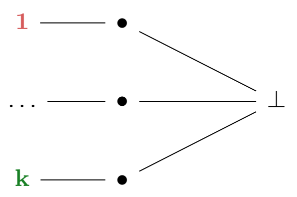


Fig. 2: $G_{k,\Delta}$

chosen their output yet. Let $\Gamma' = \Gamma \cup \{\perp\}$, with Γ being the palette one wants to conflict-free color the graph with. Let ψ' be a verifier such that:

$$\psi'(G,\lambda,v) = \begin{cases} \textit{happy}, & \textit{if } \perp \in \lambda(N(v)) \\ \textit{happy}, & \textit{if } \perp \notin \lambda(N(v)) \wedge \exists u \textit{ a conflict-free neighbor of } v \\ \textit{unhappy}, & \textit{otherwise} \end{cases}$$

Then one can define partial solutions of conflict-free k-coloring as solutions of the locally verifiable problem Π defined by Σ, Γ', ψ'.

Definition 6 (Greedily completable problems [2]). *These are problems such that any partial solution of said problem can be extended without any modification of the partial output. (Formally, they are 0-mendable problems, see definition 7)*

Theorem 9 (($\leq \Delta$)-coloring is not greedily completable.). $\forall \Delta \in \mathbb{N}, \forall k \leq \Delta$ *there exists a graph $G_{k,\Delta} = (V_{k,\Delta}, E_{k,\Delta})$ with maximum degree Δ and a partial CF k-coloring $\lambda'_{k,\Delta}$ of $G_{k,\Delta}$ that cannot be extended ie, it is impossible to color every nodes with the palette.*

Proof. Let's consider the following colored graph (see Fig.2). On the left side, each vertex has a color from $[\![1,k]\!], k \leq \Delta$, and they are connected to nodes marked with $\bullet$, meaning that they are not colored. Those uncolored nodes are connected to the only still alive node on the right side (labeled with $\perp$).

One can see that the maximal degree, being k, is indeed lower than Δ. Note that the node on the right has no cf-neighbor; thus, it must choose a color from $[\![1,k]\!]$.

Let's consider that it chooses color $i \in [\![1,k]\!]$. Then the coloring becomes illegal. Indeed, there is a node on the left side with color i that is connected to an uncolored node. The latter now has two neighbors with color i and thus no conflict-free neighbor. This is not a valid conflict-free coloring.

This property being true for any $i \in [\![1,k]\!]$, there is no legal choice to color this node.

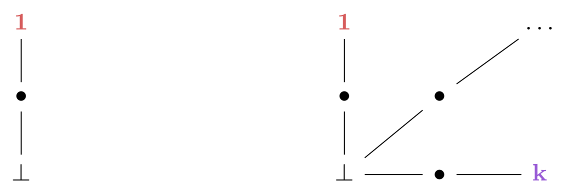
Definition 7 (T-mend [2]). *Let $\lambda : V \to \Gamma'$ be a partial solution such that ψ' accepts it, where ψ' is the relaxation of a verifier on partial solutions. $\mu : V \to \Gamma'$ is a T-mend of λ at node v if:*

1. ψ' accepts it
2. $\mu(v) \neq \perp$
3. $\mu(u) = \perp$ implies $\lambda(u) = \perp$
4. $\mu(u) \neq \lambda(u)$ implies u is within distance T of v.

Definition 8 (T-mendable problems [2]). *Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A verifier ψ' is T-mendable if for all n-node graph and all partial colorings λ accepted by ψ', there exists a $T(n)$-mend of λ at v for any $v \in V$.*

A problem Π is T-mendable if there exists a T-mendable local verifier for the problem Π. The mending radius of a problem Π is T if Π is T-mendable but not T'-mendable for any $T' \neq T$ such that $T'(n) \leq T(n)$.

Theorem 10. *CF ($< \Delta$)-coloring has $\Omega(\log n)$ mending radius. In other words, $\forall \Delta > 2, \forall k < \Delta, \forall i \in \mathbb{N}$ there exists $G_{k,i} = (V_{k,i}, E_{k,i}, \lambda_{k,1})$ a partially colored graph of maximal degree at most Δ such that mending any $\perp$-nodes requires to modify nodes at distance i from those nodes and $|V_{k,i}| \in O(k^i)$*



(a) H_1 (b) $G_{k,1}$

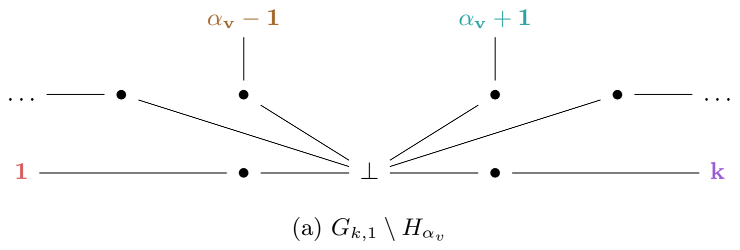
Fig. 3: Forbidding colored graphs

Proof. The colored graph in Fig. 3a forbids the $\perp$-node to choose the color 1. Further, H_i will denote the same graph as H_1 but with the leftmost node colored with i, thus forbidding the $\perp$-node to wear the color i.

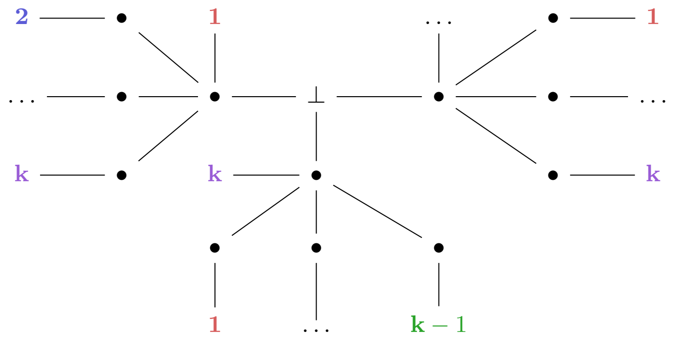
Let $\Delta \in \mathbb{N}, k < \Delta$. Let's first build $G_{k,1}$ and $G_{k,2}$ before recursively building $(G_{k,i})_{i \in \mathbb{N}}$.

One can then build $G_k, 1$ (see Fig. 3b), by plugging every $H_i, \forall i \in [\![1, k]\!]$ around the same $\perp$-node. This is a graph of maximal degree $k < \Delta$ partially colored in a way that it is impossible to color directly the $\perp$-node, because every color is forbidden and the color 0 is not suitable. But if a node at distance 1 from the $\perp$-node is colored, then a valid CF k-coloring can be produced. Note that $|V_{k,1}| = 2 * k + 1 \in O(k)$.

To produce $G_{k,2}$, the idea is to forbid each node at distance 1 from the $\perp$-node to change its color, to do so, we will use $G_{k,1}$. Note that if a node v has a cf-neighbor with color α_v, then it is unnecessary to forbid v to wear α_v as this would either be non-valid if v is its cf-neighbor or useless as each node in $G_{k,1}$ only has a unique cf-neighbor, wearing its color would then make the cf-coloring non-valid.



(a) $G_{k,1} \setminus H_{\alpha_v}$



(b) $G_{k,2}$

Fig. 4: Building $G_{k,2}$

So, for each $v \in V_{i,k}$ at distance 1 from the $\perp$-node, let α_v be the color of their cf-neighbor. $G_{k,2}$ will then be a copy of $G_{k,1}$ where each node $v \in V_{i,k}$ at distance 1 from the $\perp$-node is connected to a copy of $G_{k,1} \setminus H_{\alpha_v}$ (see Fig. 4).

Then, we obtain $G_{k,2}$ a colored graph of maximal degree $k + 1 \leq \Delta$. As we already know from the analysis of $G_{k,1}$, the $\perp$-node can't choose a color. Moreover, every node at distance 1 from the latter can't change its color as they are forbidden to do so by its neighbors. Though the graph of $G_{k,2}$ is a tree, so it is 1-colorable (color each node of depth $3k, k \in \mathbb{N}$), hence it is colorable. So, to reach a valid coloring, one would need to modify the partial coloring, and this is only possible for nodes at distance of at least 2 from the $\perp$-node. Note that $|V_{k,2}| = |V_{k,1}| + k * 2 * (k - 1) \in O(k^2)$.

Then, $G_{k,i+1}$ is made by connecting each node v at distance i from the $\perp$-node in $G_{k,i}$ with a copy of $G_{k,1} \setminus H_{\alpha_v}$.

The colored graph obtained has maximal degree of $k + 1 \leq \Delta$ (the only change to $G_{k,i}$ is that $k - 1$ edges were added to vertices of degree 2) and $|V_{k,i+1}| = |V_{k,i}| + 2 * k^i * (k - 1) \in O(k^{i+1})$

Therefore, for any $i \in \mathbb{N}$, mending the $\perp$-node would require changing the color of nodes at a distance of at least i. Given that $|V_{k,i}| \in O(k^i)$, we have that $i \in \Omega(\log |V_{i,k}|)$.

Thus, CF $(< \Delta)$-coloring is $\Omega(\log n)$-mendable with n being the size of the graph.

These last results formally explain the difficulties encountered when building the algorithms. Indeed, those methods work well on 0-mendable problems like MIS or $\Delta+1$-coloring. But CF-coloring being not greedily completable (in fact $\Omega(\log n)$-mendable), a partial solution cannot always be extended.

5 Conclusion

Wrap-up This work shows how conflict-free $o(\Delta)$-coloring stands out among LCLs by its complexity while being a very natural and straightforward problem. Indeed, we showed how it is not greedily completable (Theorem 9) while other LCLs are (it even has $\Omega(\log n)$-mending radius (see Theorem 10)). This makes it harder to build efficient algorithms to solve it. Still, we managed to provide an algorithm based on Network Decomposition (Theorem 8) and the use of multiple palettes, and produced preliminary works (see subsection 3.1) to build another algorithm based on the rounding method, which could hopefully use a smaller palette.

Open questions This work led to the following questions and ideas for future work on this topic :

1. Can the $\Omega(\log n)$ mending radius be used to reduce the number of colors used by the Network decomposition method?
2. Is it possible, by any means, to reduce the number of colors used by the algorithm using network decomposition?
3. Is it possible to select a constant share of the colored node when using the rounding method?
4. If not, is it possible to use a similar idea of "multiple palettes" for each rounding phase? Constraining the set of colored nodes to an independent set seems interesting.
5. Is it possible to provide a distributed version of the algorithm of Theorem 3, or using more local heuristics, thus turning the algorithm from [8] into a distributed one?

6 Appendices

6.1 Proof of theorem 3.1

Proof (of theorem 3.1). Let us consider $\forall i \in V, d_i$, the degree of node i. Let $p_i = \frac{1}{5d_i}$, $k_i = 2 * \lceil \sqrt{d_i} \rceil$, let $k^* = 2\lceil \sqrt{\Delta} \rceil + 1$. Now, let

$$\begin{aligned} \lambda \colon V &\to [0,1]^{k^*} \\ i &\mapsto (1 - p_i, \frac{p_i}{k_i}, ..., \frac{p_i}{k_i}, 0, ..., 0). \end{aligned}$$

be the initial assignment.

Let the utility function be

$$\begin{aligned} u \colon E \times \mathcal{L}(V) &\to \mathbb{R}^+ \\ (i,j), l &\mapsto \frac{1}{d_i} \sum_{\alpha \in [\![1,k_i]\!]} \mathbb{1}[n_{i,\alpha} = 1] + \frac{1}{d_j} \sum_{\alpha \in [\![1,k_j]\!]} \mathbb{1}[n_{j,\alpha} = 1] \end{aligned}$$

Its version on fractional assignments is then

$$\begin{aligned} u \colon E \times \Lambda(V) &\to \mathbb{R}^+ \\ (i,j), \lambda &\mapsto \frac{1}{d_i} \sum_{\alpha \in [\![1,k_i]\!]} P_\lambda[n_{i,\alpha} = 1] + \frac{1}{d_j} \sum_{\alpha \in [\![1,k_j]\!]} P_\lambda[n_{j,\alpha} = 1] \end{aligned}$$

Where $n_{i,\alpha} = k$ is the event that i has exactly k neighbors of color α in its neighborhood. The probability P_λ is defined on the fractional assignment λ seen as a probability distribution.

Let the cost function be

$$\begin{aligned} c \colon E \times \mathcal{L}(V) &\to \mathbb{R}^+ \\ (i,j), l &\mapsto \qquad \frac{1}{d_i} \sum_{\alpha=1}^{k_i - 1} \sum_{\beta=\alpha+1}^{k_i} \mathbb{1}[n_{i,\alpha} = 1 \cap n_{i,\beta} = 1] \\ &\qquad + \frac{1}{d_j} \sum_{\alpha=1}^{k_j - 1} \sum_{\beta=\alpha+1}^{k_j} \mathbb{1}[n_{j,\alpha} = 1 \cap n_{j,\beta} = 1] \end{aligned}$$

with the following fractional version

$$\begin{aligned} c \colon E \times \Lambda(V) &\to \mathbb{R}^+ \\ (i,j), \lambda &\mapsto \qquad \frac{1}{d_i} \sum_{\alpha=1}^{k_i - 1} \sum_{\beta=\alpha+1}^{k_i} P_\lambda[n_{i,\alpha} = 1 \cap n_{i,\beta} = 1] \\ &\qquad + \frac{1}{d_j} \sum_{\alpha=1}^{k_j - 1} \sum_{\beta=\alpha+1}^{k_j} P_\lambda[n_{j,\alpha} = 1 \cap n_{j,\beta} = 1] \end{aligned}$$

Note that, according to the inclusion-exclusion principle, we have:

$$\frac{1}{d_i}P_\lambda\left[\bigcup_{\alpha=1}^{k_i} n_{i,\alpha}=1\right]+\frac{1}{d_j}P_\lambda\left[\bigcup_{\alpha=1}^{k_j} n_{j,\alpha}=1\right]\geq u(i,j)-c(i,j)$$

Let A be the number of neighborhoods wearing a valid conflict-free coloring. We have, following the fractional assignment as a probability distribution:

$$E_\lambda[A]=\sum_{i\in V}P_\lambda\left[\bigcup_{\alpha=1}^{k_i} n_{i,\alpha}=1\right]\geq u(\lambda)-c(\lambda)=\sum_{e\in E}u(e,\lambda)-c(e,\lambda)$$

Property 1. $\sum_{\alpha\in[\![1,k_i]\!]}P_\lambda[n_{i,\alpha}=1]-\sum_{\alpha=1}^{k_i-1}\sum_{\beta=\alpha+1}^{k_i}P_\lambda[n_{i,\alpha}=1\cap n_{i,\beta}=1]\geq\frac{4}{25}$

Proof. see Appendix 6.2

Also, we have the following.

$$\sum_{e\in E}u(e)-c(e)=\sum_{i\in V}\left(\sum_{\alpha\in[\![1,k_i]\!]}P_\lambda[n_{i,\alpha}=1]-\sum_{\alpha=1}^{k_i-1}\sum_{\beta=\alpha+1}^{k_i}P_\lambda[n_{i,\alpha}=1\cap n_{i,\beta}=1]\right)$$

Using property 1, we obtain the following:

$$u(\lambda)-c(\lambda)\geq\frac{4}{25}n$$

Moreover $u(\lambda)=\sum_{i\in V}\sum_{\alpha=1}^{k_i}P[n_{i,\alpha}=1]$.

Property 2. $\sum_{\alpha=1}^{k_i}P[n_{i,\alpha}=1]\leq 1$

Proof. see Appendix 6.2

Using property 1, we obtain $u(\lambda)\leq n$. So we have :

$$u(\lambda)-c(\lambda)\geq\frac{4}{25}u(\lambda)$$

To use Theorem 6, the initial assignment must be $\frac{1}{2^K}$-integral for some constant K. To do so, we can use Lemma 2.3 from [4] with λ_{min} being the smallest non-zero probability among $\lambda(V)$. Each node will then produce a new initial assignment λ' with the following properties:

- λ' is $1/2^K$ integral with $2^k\in O(\frac{1}{\epsilon\mu\lambda_{min}})$
- $u(\lambda')-c(\lambda')\geq(1-\epsilon)\cdot(u(\lambda)-c(\lambda))$, with $\epsilon\in[0,1]$
- $u(\lambda')-c(\lambda')\geq\frac{\mu}{2}u(\lambda')$

Using theorem 6 with $\epsilon = 1/2, \mu = \frac{4}{25}$ and λ' the nodes obtain an integral assignment l with

$$u(l) - c(l) \geq \frac{1}{2} \cdot (u(\lambda') - c(\lambda')) \geq \frac{1}{4}(u(\lambda) - c(\lambda)) \geq \frac{1}{25}n$$

But also have that $E_l[A] \geq u(l) - c(l)$ if we see the integral assignment as a trivial probability distribution.

So we obtain that $E_l[A] \geq \frac{1}{25}n$. Hence, we obtain that at least one 25th of the neighborhoods are conflict-free colored.

6.2 Proof of property 1 and 2

Proof (of property 1). First, let's use the multinomial probability law to reach some useful form of the equation.

$$\begin{aligned}
&\sum_{\alpha \in [\![1,k_i]\!]} P_\lambda[n_{i,\alpha} = 1] && - \sum_{\alpha=1}^{k_i-1} \sum_{\beta=\alpha+1}^{k_i} P_\lambda[n_{i,\alpha} = 1 \cap n_{i,\beta} = 1] \\
&= k_i \binom{d_i+1}{1, d_i} \frac{p_i}{k_i} \left(1 - \frac{p_i}{k_i}\right)^{d_i} && - \frac{k_i(k_i-1)}{2} \binom{d_i+1}{1,1,d_i-1} \left(\frac{p_i}{k_i}\right)^2 \left(1 - \frac{2p_i}{k_i}\right)^{d_i-1} \\
&= (d_i+1)p_i \left(1 - \frac{p_i}{k_i}\right)^{d_i} && - \frac{d_i(d_i+1)}{2} p_i^2 \frac{k_i-1}{k_i} \left(1 - \frac{2p_i}{k_i}\right)^{d_i-1} \\
&\qquad \geq (d_i+1)p_i \left(\left(1 - \frac{p_i}{k_i}\right)^{d_i} - \frac{d_i p_i}{2}\right)
\end{aligned}$$

This inequality stands because $\frac{k_i-1}{k_i}\left(1 - \frac{2p_i}{k_i}\right)^{d_i-1} \leq 1$. Then, note the following :

$$\left(1 - \frac{p_i}{k_i}\right)^{d_i} = \left(1 - \frac{1}{10d_i\lceil\sqrt{d_i}\rceil}\right)^{d_i} = e^{d_i \ln\left(1 - \frac{1}{10d_i\lceil\sqrt{d_i}\rceil}\right)}$$

Given that $d_i \geq 1$

$$d_i \ln\left(1 - \frac{1}{10d_i\lceil\sqrt{d_i}\rceil}\right) \geq \ln\left(1 - \frac{1}{10d_i\lceil\sqrt{d_i}\rceil}\right) \geq \ln\left(1 - \frac{1}{10}\right) = \ln(0,9)$$

Then $\left(1 - \frac{p_i}{k_i}\right)^{d_i} \geq 0.9$.

So, we obtain the following bound :

$$\begin{aligned}
&\sum_{\alpha \in [\![1,k_i]\!]} P_\lambda[n_{i,\alpha} = 1] - \sum_{\alpha=1}^{k_i-1} \sum_{\beta=\alpha+1}^{k_i} P_\lambda[n_{i,\alpha} = 1 \cap n_{i,\beta} = 1] \\
&\geq \frac{d_i+1}{5d_i}(0.9 - 0.1) = \frac{d_i+1}{5d_i}\left(\frac{4}{5}\right) \\
&\geq \frac{4}{25}
\end{aligned}$$

Proof (of property 2). From 6.2 we have that :

$$\sum_{\alpha \in [\![1,k_i]\!]} P_\lambda[n_{i,\alpha} = 1] = (d_i + 1)p_i \left(1 - \frac{p_i}{k_i}\right)^{d_i} = \frac{d_i + 1}{5d_i} \left(1 - \frac{p_i}{k_i}\right)^{d_i}$$

Given that $\left(1 - \frac{p_i}{k_i}\right)^{d_i} \leq 1$. We obtain the following bound :

$$\sum_{\alpha \in [\![1,k_i]\!]} P_\lambda[n_{i,\alpha} = 1] \leq 1$$

References

1. Abel, Z., Alvarez, V., Demaine, E.D., Fekete, S.P., Gour, A., Hesterberg, A., Keldenich, P., Scheffer, C.: Conflict-free coloring of graphs. SIAM Journal on Discrete Mathematics **32**(4), 2675–2702 (2018). https://doi.org/10.1137/17M1146579, https://doi.org/10.1137/17M1146579
2. Balliu, A., Hirvonen, J., Melnyk, D., Olivetti, D., Rybicki, J., Suomela, J.: Local mending. In: Parter, M. (ed.) Structural Information and Communication Complexity. pp. 1–20. Springer International Publishing, Cham (2022)
3. Bhyravarapu, S., Kalyanasundaram, S., Mathew, R.: A short note on conflict-free coloring on closed neighborhoods of bounded degree graphs. Journal of Graph Theory **97**(4), 553–556 (2021). https://doi.org/https://doi.org/10.1002/jgt.22670, https://onlinelibrary.wiley.com/doi/abs/10.1002/jgt.22670
4. Faour, S., Ghaffari, M., Grunau, C., Kuhn, F., Rozhoň, V.: Local distributed rounding: Generalized to mis, matching, set cover, and beyond. ACM Trans. Algorithms (Jun 2025). https://doi.org/10.1145/3742476, https://doi.org/10.1145/3742476, just Accepted
5. Ghaffari, M., Grunau, C.: Near-Optimal Deterministic Network Decomposition and Ruling Set, and Improved MIS . In: 2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS). pp. 2148–2179. IEEE Computer Society, Los Alamitos, CA, USA (Oct 2024). https://doi.org/10.1109/FOCS61266.2024.00007, https://doi.ieeecomputersociety.org/10.1109/FOCS61266.2024.00007
6. Ghaffari, M., Kuhn, F.: Deterministic distributed vertex coloring: Simpler, faster, and without network decomposition. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS). pp. 1009 – 1020. IEEE, Piscataway, NJ (2022). https://doi.org/10.1109/FOCS52979.2021.00101, 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2021); Conference Location: Online; Conference Date: February 7-10, 2022; Conference lecture held on February 10, 2022.
7. Naor, M., Stockmeyer, L.: What can be computed locally? SIAM Journal on Computing **24**(6), 1259–1277 (1995). https://doi.org/10.1137/S0097539793254571, https://doi.org/10.1137/S0097539793254571
8. PACH, J., TARDOS, G.: Conflict-free colourings of graphs and hypergraphs. Combinatorics, Probability and Computing **18**(5), 819–834 (2009). https://doi.org/10.1017/S0963548309990290