Properly 2-Colouring Linear Hypergraphs *

Arkadev Chattopadhyay
School of Computer Science
McGill University, Montreal, Canada
achatt3@cs.mcgill.ca

Bruce A. Reed
School of Computer Science
McGill University, Montreal, Canada
&
Projet Mascotte, Laboratoire I3S
CNRS Sophia-Antipolis, France
breed@cs.mcgill.ca

Abstract

Using the symmetric form of the Lovász Local Lemma, one can conclude that a k-uniform hypergraph $\mathcal{H}$ admits a proper 2-colouring if the maximum degree (denoted by Δ) of $\mathcal{H}$ is at most $\frac{2^k}{8k}$ independently of the size of the hypergraph. However, this argument does not give us an algorithm to find a proper 2-colouring of such hypergraphs. We call a hypergraph *linear* if no two hyperedges have more than one vertex in common.

In this paper, we present a deterministic polynomial time algorithm for 2-colouring every k-uniform linear hypergraph with $\Delta \leq 2^{k-k^{\epsilon}}$, where $1/2 < \epsilon < 1$ is any arbitrary constant and k is larger than a certain constant that depends on ϵ. The previous best algorithm for 2-colouring linear hypergraphs is due to Beck and Lodha [4]. They showed that for every $\delta > 0$ there exists a $c > 0$ such that every linear hypergaph with $\Delta \leq 2^{k-\delta k}$ *and* $k > c \log\log(|E(\mathcal{H})|)$, can be properly 2-coloured deterministically in polynomial time.

*Research of the first author is supported by a NSERC graduate scholarship and research grants of Prof. D. Thérien, the second author is supported by a Canada Research Chair in graph theory

1 Introduction

The probabilistic method [2] is widely used in theoretical computer science and discrete mathematics to guarantee the existence of a combinatorial structure with certain desired properties. However, these techniques are often non-constructive. Efficient construction of structures with desired properties is an important research theme in various areas like Ramsey theory, graph colouring and coding theory.

In many applications of the method, there are N events (typically "bad") in some probability space, and one is interested in showing that the probability that none of the events happen is positive. This will be true if the events are *mutually independent* and each event occurs with probability less than one. In practice, however one often finds events that are not indpependent of each other. The Lovász Local Lemma is a powerful sieve method that can be used in this context when the events have limited dependence.

The power of the Lemma comes from the fact that it allows one to conclude a global result by analysing the local property of random combinatorial structures. In particular, the symmetric form of the Lemma implies that a random 2-colouring of the vertices of a k-uniform hypergraph of maximum degree $\frac{2^k}{8k}$, generates no monochromatic edges with non-zero probability. But it provides no clue as to how to find such a structure efficiently. The Local Lemma merely guarantees the existence of, what has been called a "needle in a haystack". It can be easily verified that if the total number of edges m in the hypergraph is much larger than 4^k, then with extremely small probability the colouring is proper. The question that we are interested in is the following :
"Is there an efficient algorithm to find such a 'rare' colouring?"

Beck[3] and Alon[1] developed algorithmic versions of the Lemma which evoked its symmetric form. Beck showed that if the maximum degree Δ of the hypergraph is reduced to $2^{\alpha k}$, where $\alpha = 1/48$, then indeed there is a positive answer to the question above.

Later, Beck and Lodha[4] obtained a deterministic polytime algorithm to find such a 2-colouring for *linear* hypergraphs under the restrictions that $\Delta < 2^{(1-\delta)k}$ and that the total number of hyperedges (denoted by m) satisfies $(2m)^{1+\beta} < 2^{(k-2k')2^{k/4}}$, where $\beta > 0$ is a real constant and $3 \leq k' \leq 2k/3$ is an integer. Their method builds upon the ideas in [3] and uses involved combinatorial analysis that runs into several cases.

1.1 Our result

We build upon the general method outlined in [9] to get an algorithm for 2-colouring k-uniform linear hypergraphs with larger maximum degree. Further, our algorithm does not require any conditions to be imposed on the relationship between total size of the hypergraph and the number of vertices in a hyperedge.

Theorem 1 *There exists a deterministic polytime algorithm that for every constant $\epsilon > 1/2$, finds a proper 2-colouring of any k-uniform linear hypergraph $\mathcal{H}$, whose maximum degree is bounded by 2^{k-k^ϵ}, provided $k \geq k_\epsilon$, where k_ϵ is a positive integer that depends only on ϵ.*

Our deterministic algorithm is constructed in two steps. First, we develop a randomized algorithm that is conceptually fairly simple. The random algorithm genarates the colouring iteratively in phases. It employs novel *freezing* techniques that may be useful for dealing with other problems. In particular, our freezing is guided by the crucial use of the asymmetric form of the Local Lemma as opposed to the method of Beck and Lodha [4] that uses the symmetric form. This allows us to minimize freezing enabling the random colouring to work more effectively. We finish off by derandomizing our algorithm by a simple application of the method of conditional expectations due to Erdös and Selfridge [6].

In Section 2, we introduce the basic terminology and the needed background on the Local Lemma. Section 3 introduces a randomized algorithm, Section 4 provides its analysis and Section 5 derandomizes the algorithm proving Theorem 1. We remark that the technique in this paper can be suitably modified to make it work for the more general case of hypergraphs having constant co-degree (co-degree of any pair of vertices being the number of edges containing that pair). Note that linear hypergraphs have co-degree at most one.

2 Basic Notions

A *hypergraph* $\mathcal{H}$ is given by a set of vertices (denoted by V) and a set of non-empty subsets of V (denoted by E). Each subset in E is called a *hyperedge*. In this paper, we will often call a hyperedge simply an edge. The *size* of $\mathcal{H}$ is simply the sum of the cardinalities of V and E. $\mathcal{H}$ is called *k-uniform* if each of its hyperedges has cardinality k. The *degree* of a vertex v in V is the number of hyperedges containing v and is denoted by d_v. We denote the *maximum degree* of $\mathcal{H}$ by Δ i.e $\Delta = max_{v \in V} d_v$. Two hyperdges *intersect* at a vertex v if v is contained in both of them. A hypergraph is called *linear* if every pair of hyperedges intersect at most at one vertex. We associate an undirected graph $G_{\mathcal{H}}$ with every hypergraph $\mathcal{H}$ in the following way: $V(G_{\mathcal{H}}) = V(\mathcal{H}) \cup E(\mathcal{H})$, and $E(G_{\mathcal{H}}) = \{(x,y) | x,y \in V(\mathcal{H}),\ x,y \in e$ for some $e \in E(\mathcal{H})\ OR\ x \in V(\mathcal{H}), y \in E(\mathcal{H}), x \in y\ OR\ x,y \in E(\mathcal{H}), x \cap y \neq \emptyset\}$. A *path* from one vertex/edge x of $\mathcal{H}$ to another vertex/edge y of $\mathcal{H}$ is just a simple path in $G_{\mathcal{H}}$ from x to y. The *distance* from x to y is the length of the shortest path in $G_{\mathcal{H}}$ from x to y. We say that vertex x and y are *reachable* from each other if there exists a path from one to the other. A *(connected) component* of $\mathcal{H}$ is a hypergraph C such that $V(C)$ is a maximal set of vertices of $\mathcal{H}$ that are reachable from each other and $E(C) = \{e \in E(\mathcal{H}) | e \subseteq V(C)\}$.

A 2-*colouring* χ of $\mathcal{H}$ is any assignment of colours from the set of 2-colours (we use red and blue) to its vertices i.e $\chi : V \rightarrow \{red, blue\}$. We say a hyperedge e is *monochromatic* under the colouring χ if every vertex contained in e receives the same colour. χ is called a *proper* 2-*colouring* precisely if it does not generate any monochromatic hyperedge. In this work, we generate a 2-colouring iteratively, and each phase of our algorithm generates a partial colouring of vertices. For such a partial colouring, we call an edge *uni-coloured* if its vertices have not received two different colours. An uncoloured vertex is called *nice* if it lies in exactly one uni-coloured edge. For any uni-coloured edge e, its *restriction* to the uncoloured vertices is $e' = \{v \in e : v \text{ is not coloured}\}$. A partial colouring χ of $\mathcal{H}$ *induces* the hypergraph $\mathcal{H}'$ with its vertex set consisting of all uncoloured vertices and its edge set

consisting of the restriction of all its uni-coloured edges. The following simple remarks will be useful:

Remark 2 *A hypergraph $\mathcal{H}$ has a proper 2-colouring if and only if each of its components has one.*

Remark 3 *A partial colouring χ of $\mathcal{H}$ extends to a proper colouring precisely if the induced hypergraph $\mathcal{H}'$ has a proper colouring.*

Remark 4 *Let χ be a partial colouring of $\mathcal{H}$ and let e be a uni-coloured edge that has a pair of nice vertices v_1 and v_2. Extend χ to χ' by colouring v_1 and v_2 such that $\chi'(v_1) \neq \chi'(v_2)$. Then the hypergraph induced by χ' (denoted by $\mathcal{H}''$) has the set of vertices $V(\mathcal{H}') - v_1 - v_2$ and its set of edges is $E(\mathcal{H}') - e$, where $\mathcal{H}'$ is the hypergraph induced by χ. Further, it has a proper 2-colouring precisely if $\mathcal{H}'$ has one.*

We denote the probability of an event X by $\mathbf{Pr}[X]$ and the expected value of a random variable y by $\mathbf{E}[y]$. We now state the Asymmetric form of the Lovász Local Lemma [5] that is one of the most powerful tools of the probabilistic method.

Lemma 5 *(Asymmetric form of Local Lemma) Let $\mathcal{E} = A_1, \ldots, A_m$ be m (typically bad) events in a probability space, where event A_i occurs with probability p_i and is mutually independent of events in $\mathcal{E} - \mathcal{D}_i$, for some $\mathcal{D}_i \subseteq \mathcal{E}$. Then, the probability that none of these m events occur is positive provided that*

- $\sum_{k:A_k \in \mathcal{D}_i} p_k \leq \frac{1}{4}$. *for all i.*

In the application of the Local Lemma to colouring a k-uniform hypergraph, we consider a random 2-colouring of its vertices. With each hyperedge e, we associate the event (denoted by A_e) that e is monochromatic. One can easily verify that A_e is mutually independent of all other events except those in $\mathcal{D}_e = \{A_f | f \cap e \neq \emptyset\}$. Clearly, $|\mathcal{D}_e| \leq 1 + (\Delta - 1)k$. Since $\mathbf{Pr}[A_e] = 1/2^{k-1}$ for all edges e, the local lemma implies that the probability of not having any monochromatic edge is positive if $\Delta \leq \frac{2^k}{8k}$.

We will denote by $\mathcal{BIN}(n, p)$ the sum of n independent Bernoulli random variables, each equal to 1 with probability p and 0 otherwise. The following inequality, called Chernoff's bound (see [2]), is used to bound the probability of the sum deviating from its expected value np:

$$\mathbf{Pr}(|\mathcal{BIN}(n, p) - np| > t) < 2e^{-((1+\frac{t}{np})ln(1+\frac{t}{np})-\frac{t}{np})np} \quad (1)$$

3 Randomized Algorithm

We assume that the vertices are labelled $\{1, \ldots, n\}$. We first provide a randomized algorithm to obtain a proper 2-colouring and then show that it can be easily derandomized. The randomized algorithm works in three phases. In the first phase, we randomly colour a vertex v_i, unless it is frozen using one of the rules described below. These freezing rules are imposed to ensure that the Local Lemma can be applied to prove that the partial colouring obtained in the first phase extends to a proper colouring. We shall also show that with probability near 1, the connected components of the induced hypergraph are small in size. We post-process the random colouring in the following way : identify those uni-coloured edges that contain at least two nice vertices. For every such uni-coloured edge, we choose a pair of nice vertices and assign them different colours. By Remark 4, the hypergraph induced by this partial colouring (denoted by $\mathcal{H}_1$) is still properly 2-colourable and we shall show that it has small components, with high probability.

In the second phase, we reapply the procedure of the first phase to each component of $\mathcal{H}_1$ separately. We use very similar freezing rules and do identical post-processing to that in Phase 1. We show again that the partial colouring extends to proper colouring using the Local Lemma. We denote the induced hypergraph at the end of phase 2 by $\mathcal{H}_2$ and as before, each of its component is very small with high probability.

Finally, in the third phase, we apply brute force to explicitly compute a proper completion for the partial colouring of each component of $\mathcal{H}_2$. This is possible in poly-time as the size of each component is very small.

Forthwith the details. To ensure that we can apply the Local Lemma to show that our partial colouring extends to a proper colouring, we introduce a random variable H_e for each edge e as given below:

$$H_e \;=\; \sum_{f : e \cap f \neq \emptyset} \mathbf{Pr}[A_f | \textit{partial colouring}]$$

Note that if H_e lies below $1/4$ at the end of a phase for every edge e, then the asymmetric form of the Local Lemma can be applied to conclude that the partial colouring extends to a proper 2-colouring. It will be more convenient to think of H_e as being a sum of k random variables - one for each vertex v contained in edge e. Formally, $H_e = \sum_{v \in e} H_v - (|e| - 1)\mathbf{Pr}[A_e | \textit{partial colouring}]$, where H_v is given by

$$H_v \;=\; \sum_{f : v \in f} \mathbf{Pr}[A_f | \textit{partial colouring}]$$

Thus, $H_v \leq 1/4k$ ensures that our partial colouring extends to a proper colouring. The key technique in our procedure is to control H_v, for each vertex v. A vertex v is considered *bad* if H_v exceeds a prescribed upper bound b_i for Phase i. Whenever a vertex v turns bad, we freeze all uncoloured vertices in each edge containing v. If this were the only freezing done, then the probability of v turning bad would be substantial and our freezing would be

widespread. Indeed, the probability that a specific edge e containing v is monochromatic is $1/2^{k-1}$. So, the probability that v turns bad is at least $1/2^{k-1}$. To deal with this, recall that H_v is a sum of conditional probabilities. The intuition is if each of these probabilities in the sum is small, then H_v is concentrated and the probability that it turns bad is extremely small provided the threshold b_i is appropriately chosen. In particular, for each of the first two phases, we specify a lower bound on the number of vertices remaining uncoloured in a uni-coloured edge. If edge e attains this lower bound, we call e *naughty* and freeze all of its uncoloured vertices. This freezing of naughty edges ensures that the conditional probabilities that appear in the sum for H_v remain small, and reduces the probability of a vertex becoming bad to well below 2^{-k}. Unfortunately, there is considerable freezing done due to naughty edges. However, as we shall see, the post-processing step allows us to ignore all but a vanishingly small proportion of this second kind of freezing. This is the basic structure of our random colouring procedure in each phase. We fix constants β, α_1 and α_2 such that $1/2 < \alpha_2 < \beta < \alpha_1 < \epsilon$ for reasons that will become clear from subsequent discussion. For Phase 1, $b_1 = 2^{-k^\beta}$ and for Phase 2 $b_2 = 1/8k$. The lower bound used for freezing uncoloured vertices in naughty edges in Phase i will be k^{α_i} for some α_i and $i = 1, 2$.

With this intuitive description behind us, we are ready to give the formal description of the basic random colouring procedure parameterized with t_1, the threshold for detecting naughty edges and t_2, the threshold for detecting bad vertices:

RANDCOLOUR($\mathcal{F}$, t_1, t_2);
Input: Hypergraph $\mathcal{F}$ s.t. $\forall e \in E(\mathcal{F})\ |e| > t_1$, $\forall v \in V(\mathcal{F})\ H_v \leq t_2$.
Returns a partial colouring and the induced hypergraph $\mathcal{F}'$ such that $\forall e \in E(\mathcal{F}')\ |e| \geq t_1, \forall v \in V(\mathcal{F}')\ H_v \leq 2t_2$.

- **Main loop:** For each vertex $v_i \in V$.
  - If v_i is frozen, skip colouring it and go back to the main loop
    - Colour v_i uniformly at random and then do the following:
      1. If some uni-coloured edge e has only t_1 uncoloured vertices remaining, freeze every uncoloured vertex in e.
      2. If for some vertex v, H_v exceeds t_2, freeze all uncoloured vertices in uni-coloured edges containing v.
- **Post-processing:** For every uni-coloured edge containing two nice vertices, colour one of them red and the other blue.

Note that the main loop of RANDCOLOUR runs $|V(\mathcal{H})|$ times. To determine if an edge is naughty or a vertex is bad, we need to only look at edges containing the vertex that got coloured in that iteration. Thus, each loop takes $O(k.\Delta^2)$ time. Summing this up,

Remark 6 *The running time of* **RANDCOLOUR** *is* $O(k\Delta^2 \cdot |V(\mathcal{H})|)$.

We make another remark that is useful to get the intuition behind our post-processing step.

Remark 7 *Every uni-coloured edge at the end of Phase i contains at least $k^{\alpha_i} > \sqrt{k}$ uncoloured vertices and satisfies at least one of the following conditions:*

1. *it is naughty*
2. *for every uncoloured vertex v in it, $v \in g$ for some edge g that is naughty or contains a bad vertex.*

Let us call a uni-coloured edge *bad* if it intersects with at least $\sqrt{k}$ other naughty edges. Intuitively, we expect bad edges to occur with small probability. The following fact points out that our post-processing step deals effectively with naughty edges that are not bad and are far from both bad edges and vertices.

Proposition 8 *Consider a uni-coloured edge e. If e is at distance at least 4 from every bad edge and bad vertex, then e has two nice vertices, provided k is larger than a certain constant.*

Proof: Consider a uni-coloured edge f that intersects e. Remark 7 implies the following: if f is not naughty then it must be bad or it is at distance at most 2 from a bad vertex. As e is at distance at least 4 from every bad edge/vertex, f must be naughty. This shows that every uni-coloured edge that intersects e is naughty. Since e can not be bad, there are at most $\sqrt{k}$ uni-coloured edges intersecting e. Hence for large k satisfying $k^{\alpha_i} - \sqrt{k} \geq 2$, we conclude that e has 2 nice vertices. ∎

Thus, Remark 4 implies that all uni-coloured edges that are not within distance 3 of any bad vertex or bad edge, do not remain uni-coloured after post-processing and hence are not part of the induced hypergraph returned by RANDCOLOUR. We will see later in the analysis of our algorithm in Section 4 that this observation is very helpful in bounding the size of components of the hypergraph returned by RANDCOLOUR. Lemma 13 and 15 in the next section imply that the component size at the end of Phase i is s_i with high probability, where $s_1 = 2^{6k}(c_\epsilon/k^{\epsilon+1/2}) \cdot \log(n+m)$ and $s_2 = 2^{6k}(c_\epsilon/k^{\beta+1/2}) \cdot \log(s_1)$. The constant c_ϵ depends on ϵ and is chosen according to Lemma 15.

Using RANDCOLOUR as our key sub-routine, the entire 3-phase algorithm is described below.

2-COLOUR$(\mathcal{H},\ \alpha_1,\ \alpha_2,\ \beta, \epsilon)$

- *Phase 1*: $\mathcal{H}_1 =$RANDCOLOUR$(\mathcal{H},\ k^{\alpha_1},\ 2^{-k^\beta - 1})$. If the largest component of $\mathcal{H}_1$ is larger than s_1 then repeat.

- *Phase 2*: Enumerate the components of $\mathcal{H}_1$ as $C_1, \ldots, C_j$.
  - For every connected component C_i
    - $\mathcal{H}_2^i$ =RANDCOLOUR(C_i, k^{α_2}, $1/16k$).
      - *Case 1.* $2^{6k} \leq \log(n+m)/\log\ \log(n+m)$. If the largest component of $\mathcal{H}_2^i$ is larger than s_2, then repeat call to RANDCOLOUR.
      - *Case 2.* $2^{6k} > \log(n+m)/\log\ \log(n+m)$. If there are any uni-coloured edges remaining, repeat call to RANDCOLOUR.
- *Phase 3* Let $\mathcal{H}_2 = \cup_{i=1}^{j} \mathcal{H}_2^i$.
  - If there are uni-coloured edges left in $\mathcal{H}_2$, then
    - Colour each connected component C of $\mathcal{H}_2$ in turn, by considering all $2^{|V(C)|}$ colourings.

Proposition 9 *Every partial colouring produced at the end of Phase 1 or Phase 2 can be completed to a proper 2-colouring.*

Proof: Let $\mathcal{H}_1'$ be the hypergraph induced at the end of random colouring in Phase 1 just before the post-processing step. Consider any edge f and any vertex u in f. It is simple to verify that colouring u at most doubles the probability that f becomes monochromatic. Hence, freezing done due to bad vertices ensures that for every vertex v, H_v is at most 2^{-k^β} which is less than $1/4k$ for large k. Hence, $H_e < 1/4$ for all edges e in $\mathcal{H}_1'$. This ensures that the second condition of the Local Lemma is satisfied for each event A_e. Thus $\mathcal{H}_1'$ has a proper colouring. Recalling Remark 4, we see colouring a pair of nice vertices in a uni-coloured edge differently, does not increase the probability of any event A_e. Thus, $\mathcal{H}_1$ has a proper colouring. Using Remark 3, the partial colouring at the end of Phase 1 extends to a proper 2-colouring. A very similar argument applied to each connected component of $\mathcal{H}_1$ shows that each partial colouring obtained at the end of Phase-2 also has a proper completion. ∎

Our aim in the next section is to show the result given below:

Theorem 10 *With high probability,* **2-COLOUR** *produces a proper 2-colouring of hypergraph $\mathcal{H}$ in time polynomial in the size of $\mathcal{H}$.*

4 Analysis of 2-COLOUR

We have already noted in Remark 6 that RANDCOLOUR runs in poly-time. Further Case 1 and Case 2 handled in Phase 2 of 2-COLOUR ensures that either the size of every component of $\mathcal{H}_2$ is at most s_2 or no uni-coloured edge is left at end of Phase 2. Thus Phase 3 of 2-COLOUR also runs in poly-time. Hence, to prove Theorem 10 it is sufficient to show the following:

Lemma 11 *Every call from 2-COLOUR to RANDCOLOUR succeeds with probability at least* $1/2$.

To prove Lemma 11, we simply need to show that the probability that the induced hypergraph returned by RANDCOLOUR has large components is less than a half. One way to do this would be to establish the following:

Claim 12 $\mathbf{E}[X_i] < 1/2$, *for* $i = 1, 2$.

where, X_i denotes the number of components of size at least s_i of the hypergraph $\mathcal{H}_i$. Lemma 11 follows from the above claim by a simple application of Markov's inequality.

Instead of bounding directly the expected value of X_i, we will define another integer valued random variable Y_i, such that $X_i \leq Y_i$. Bounding the expected value of Y_i turns out to be more convenient. But we need to introduce a few more notions before we can define Y_i.

Consider an auxiliary graph G_i associated with $\mathcal{H}_i$ in the following way. Each node of G_i corresponds to either a vertex or a hyperedge of hypergraph $\mathcal{H}_i$. Two nodes of G_i are connected by an arc if their distance from each other in $\mathcal{H}_i$ is at least 4 and at most 10. Every rooted tree T_i of G_i is called a $(4, 10)$-tree of $\mathcal{H}_i$. We call a $(4, 10)$-tree *bad* if every node of the tree corresponds to either a bad edge or a bad vertex generated in Phase i. Let Y_i be the number of bad $(4, 10)$ trees of $\mathcal{H}_i$ of size $s_i \cdot 2^{-6k}$. The following lemma establishes the desired relationship between X_i and Y_i (i.e. $X_i \leq Y_i$).

Lemma 13 *If* $\mathcal{H}_i$ *has a component* C *of size* s, *then there is a bad* $(4, 10)$*-tree of size at least* $s/2^{6k}$, *whose set of nodes is contained in* $V(C) \cup E(C)$.

Proof: Let T be a maximal bad $(4, 10)$-tree formed from vertices and edges of C. Let C_1 be the sub-hypergraph in the component that includes all edges that are within distance 6 from T. We show that there cannot be any vertex or edge outside of C_1 in C and the result follows from that by recalling that the degree of any vertex is at most 2^{k-k^ϵ}.

Suppose the contrary is true. Then, let b be a edge that is at minimal distance (at least 7) in C from T. Consider a minimal path P in C from b to T. Consider a node u in P at distance 7 from T. We have following possibilities:

- u is a bad edge, in which case one can add u to T.
- u is not bad. Recalling Proposition 8 and our post-processing step in RANDCOLOUR, there is a node u' that corresponds to either a bad vertex or bad edge at distance at most 3 from u. Hence u' is at distance at least 4 and at most 10 from T. We can grow T by adding u'.

Thus, in each case our assumption that T is maximal is contradicted if there exists any edge outside of C_1. ∎

Now, to estimate the expected value of Y_i we need to calculate the probabilites of certain events. We denote the events of any edge e becoming naughty and bad in phase i by N_e^i and B_e^i respectively. Let the event that a vertex v becomes bad in phase i be denoted by B_v^i.

Proposition 14 *Let $1/2 < \alpha_2 < \beta < \alpha_1 < \epsilon < 1$. Then for sufficiently large k, the following probability bounds hold:*

1. $\mathbf{Pr}[B_e^1] \leq 2^{-k^{\epsilon+1/2}+k^{\alpha_1+1/2}+\sqrt{k}\log\ k+1}$ *for any edge e.*
2. $\mathbf{Pr}[B_v^1] \leq 2e^{-2^{k^{\alpha_1}-k^{\beta}-\log\ k-1}} \leq 2^{-2^{0.5k^{\alpha_1}}}$, *for any vertex v.*
3. $\mathbf{Pr}[B_e^2] \leq 2^{-k^{\beta+1/2}+k^{\alpha_2+1/2}+\sqrt{k}\log\ k+1}$ *for any edge e.*
4. $\mathbf{Pr}[B_v^2] \leq 2e^{-2^{k^{\alpha_2}-3\log\ k-6}} \leq 2^{-2^{0.5k^{\alpha_2}}}$, *for any vertex v.*

Proof: We consider a set S_e of $\sqrt{k}$ edges $\{e_1, \ldots, e_{\sqrt{k}}\}$ that intersect an edge e. Let V_{e_i} be the set of those vertices of e_i that are not contained in any edge in $S_e \backslash \{e_i\}$. Since $\mathcal{H}$ is linear, $|V_{e_i}| \geq k - \sqrt{k} + 1$, for all i. If e_i is to become naughty, the first $k - k^{\alpha_i} - \sqrt{k}$ vertices in V_{e_i} that get coloured, receive the same colour. The actual vertices in V_{e_i} that get coloured in any instance by RANDCOLOUR may depend on the freezing caused by colouring in the rest of the hypergraph. Nevertheless, every instance of RANDCOLOUR that makes e_i naughty, makes a sequence L_i of $k - k^{\alpha_i} - \sqrt{k}$ independent colour assignments to some vertices in V_{e_i}. Since $V_{e_i} \cap V_{e_j} = \emptyset$ for $i \neq j$, every call to RANDCOLOUR makes all edges in S_e naughty with probability at most $2^{(-k+k^{\alpha_1}+\sqrt{k})\sqrt{k}}$. We have at most $(\Delta \cdot k)^{\sqrt{k}} = (2^{k-k^{\epsilon}} \cdot k)^{\sqrt{k}}$-many choices for S_e. Combining everything yields the first bound of our proposition.

We now compute $\mathbf{Pr}[B_v^1]$. Let $n_{v,i}$ represent the number of edges containing v and having i vertices uncoloured with $k - i$ vertices coloured the same. Let $n'_{v,i}$ be a random variable that is defined in the following way:

$$n'_{v,i} = \sum_{e: v \in e} x_e^i$$

where, for every edge e, x_e^i is a Bernoulli random variable that takes value 1 with probability 2^{-k+i+1}. Since our hypergraph is linear, the edges containing v only intersect each other at v. Thus, using sequences of colour assignments for each edge as before in the first part, one can show that $\mathbf{Pr}[n_{v,i} \geq N] \leq \mathbf{Pr}[n'_{v,i} \geq N]$ for every number N. Further, one can easily verify $\mathbf{E}[n'_{v,i}] = \Delta \cdot 2^{-k+i+1} = 2^{i+1-k^{\epsilon}}$. Since $n'_{v,i}$ is the sum of independent and identically distributed random variables, the classical result of Chernoff says that it is concentrated around its mean value. We re-write H_v in the following way:

$$H_v = \sum_{i: k^{\alpha_1} \leq i \leq k} H_{v,i}$$

where

$$H_{v,i} \;=\; 2^{-i+1} \times n_{v,i}$$

If $H_v \geq 2^{-k^\beta}$, then for some i, we have $H_{v,i} \geq \frac{2^{-k^\beta}}{k}$ and so

$$n_{v,i} \;\geq\; \frac{2^{i-1-k^\beta}}{k} \;=\; \mathbf{E}[n'_{v,i}] \cdot 2^{k^\epsilon - k^\beta - \log\, k - 2}$$

Since, $n'_{v,i} = \mathcal{BIN}(\Delta, p)$ for $p = 2^{-k+i+1}$, we get

$$\mathbf{Pr}\big[n_{v,i} \geq \frac{2^{i-1-k^\beta}}{k}\big] \;\leq\; \mathbf{Pr}\Big[\,|\mathcal{BIN}(\Delta,p) - \Delta p| \geq \Delta p \cdot \big(2^{k^\epsilon - k^\beta - \log\, k - 2} - 1\big)\Big] \tag{2}$$

Recalling that $\Delta p = 2^{i+1-k^\epsilon}$, we apply Chernoff's bound from (1) to the RHS of (2)

$$\mathbf{Pr}\big[n_{v,i} \geq \frac{2^{i-1-k^\beta}}{k}\big] \;\leq\; 2\exp\Big(-2^{k^\epsilon - k^\beta - \log k - 2}(k^\epsilon - k^\beta - 2)\big(2^{i+1-k^\epsilon}\big)\Big) \tag{3}$$

Noting that our freezing ensures that $i \geq k^{\alpha_1}$, we get finally

$$\text{RHS of (3)} \leq 2\exp\big(-(k^\epsilon - k^\beta - 2)2^{k^{\alpha_1} - k^\beta - \log k - 2}\big) \tag{4}$$

Recalling our assumption that $\beta < \alpha_1 < \epsilon$, we see that (4) directly yields the second bound of our Proposition for large k.

To obtain the probability bounds for events in the second phase of the algorithm, we introduce another notation. Let $\mathcal{H}_i$ represent the subgraph induced by those monochromatic edges at the end of Phase 1 that have exactly i uncoloured vertices left. Note that the degree of $\mathcal{H}_i$ denoted by Δ_i is at most 2^{i-k^β}, since $H_v \leq 2^{-k^\beta}$ for all vertices v at the end of phase 1.

Consider $\sqrt{k}$ second phase naughty edges $e_1, \ldots, e_{\sqrt{k}}$ intersecting edge e, where edge e_j had i_j uncoloured vertices at the beginning of phase 2. Applying a similar argument as before for computing $\mathbf{Pr}[B_e^1]$ and observing that there are at most $k^{\sqrt{k}}$ ways of choosing the $\sqrt{k}$-tuple $(i_1, \ldots, i_{\sqrt{k}})$ where $k^{\alpha_1} \leq i_j \leq k$ for each j, we get

$$\mathbf{Pr}[B_e^2] \;\leq\; k^{\sqrt{k}} \times \prod_{j=1}^{\sqrt{k}} 2^{-i_j + k^{\alpha_2} + \sqrt{k}} \cdot (\Delta_{i_j} k) \;\leq\; 2^{(-k^\beta + k^{\alpha_2} + 2\log\, k + 1)\sqrt{k}}$$

giving us our third bound.

Let $n^i_{v,j}$ be the number of monochromatic edges containing vertex v that have i uncoloured vertices at the end of Phase 1 and have $j \leq i$ uncoloured vertices at the end of Phase 2. As in the argument for bounding $\mathbf{Pr}[B_v^1]$, we introduce a random variable $n'^{,i}_{v,j}$ for each $n^i_{v,j}$

such that $\mathbf{Pr}[n^i_{v,j} \geq N] \leq \mathbf{Pr}[n^{'i}_{v,j} \geq N]$ for every number N. Like before, each $n^{'i}_{v,j}$ is a sum of identical and independent Bernoulli random variables and hence, is concentrated around its mean. Clearly, $\mathbf{E}[n^{'i}_{v,j}] = \Delta_i \cdot 2^{-i+j} = 2^{j-k^\beta}$. Let $n_{v,j,2} = \sum_{i=k^{\alpha_1}}^{k} n^i_{v,j}$ i.e. the number of edges through v that have j uncoloured vertices at end of phase 2. As before, define $H^2_{v,j} = 2^{-j+1} \cdot n_{v,j,2}$, so that H_v at end of phase 2 is simply $\sum_{j=k^{\alpha_2}}^{k} H^2_{v,j}$. Thus, $H_v \geq 1/8k$ implies that for some i, j,

$$n^i_{v,j} \geq \frac{2^{j-1}}{8k^3} = \mathbf{E}[n^{'i}_{v,j}] \cdot \frac{2^{k^\beta-3}}{8k^3}$$

Applying Chernoff's bound as we did for computing the second bound of the proposition,

$$\mathbf{Pr}[n^j_{v,i} \geq \frac{2^{j-1}}{8k^3}] \leq 2e^{-2^{k^\beta-3\log\ k-6}.\mathbf{E}[n^{'j}_{v,i}]}$$

Plugging $\mathbf{E}[n^{'j}_{v,i}] \geq 2^{k^{\alpha_2}-k^\beta}$, we get our desired bound. ∎

With the bounds on probability of relevant events, we are ready to calculate the expected number of bad $(4, 10)$-trees in each phase of 2-COLOUR.

Lemma 15 *There exists a constant c_ϵ for $i = \{1, 2\}$ such that the expected number of bad $(4, 10)$-trees in $\mathcal{H}_i$ of size $(c_\epsilon/k^{\delta_i+1/2}) \cdot \log(n_i + m_i)$ is less than $\frac{1}{2}$, where $\delta_1 = \epsilon$ and $\delta_2 = \beta$ and n_i, m_i are respectively the number of vertices and edges in the largest component of $\mathcal{H}_i$.*

Proof: Let X^i_ℓ denote the random variable that is equal to the number of bad $(4, 10)$-trees of size ℓ at end of Phase i. In the following discussion, we shall drop the superscript i denoting the Phase in which events occur. Consider T to be a $(4, 10)$-tree of size ℓ with nodes $n_1, \ldots, n_\ell$. Further, let B_T denote the event that T turns bad at the end of Phase i. We first want to compute $\mathbf{Pr}[B_T]$. For T to turn bad, each edge and vertex of the hypergraph comprising it has to become bad. These events are not independent as freezing done due to a vertex (an edge) becoming bad (naughty) affects the probabilities of other vertex/edge becoming bad/naughty. But event B_v (B_e) is determined by just exposing colours assigned to a set of vertices at distance 1 (2) from v (e) in $\mathcal{H}_i$. Let this set be denoted by V_v (V_e). For two edges e and f, it is easily verified that sets $V_e \cap V_f \neq \emptyset$ implies that the distance between e and f is less than 4. Similarly one can verify that sets V_{n_j} and V_{n_k} are disjoint for every pair of nodes n_j, n_k in T as distance between n_j and n_k is at least 4 in $\mathcal{H}_i$. For each node n in T, let B'_n denote the auxiliary event that n turns bad when just vertices in V_n are randomly coloured. Our previous observation implies that the auxiliary events associated with nodes of a $(4, 10)$-tree are independent of each other. Further, $\mathbf{Pr}[B_T] \leq \mathbf{Pr}[B'_{n_1} \cap B'_{n_2} \cap \cdots \cap B'_{n_\ell}]$. Thus, using the bounds obtained for probabilities in Proposition 14, one gets the following:

$$\mathbf{Pr}[B^i_T] \leq 2^{-0.5\ell(k^{\delta_i+1/2})} \tag{5}$$

where $\delta_1 = \epsilon$ and $\delta_2 = \beta$. We find out an upper bound on N_ℓ i.e. the possible number of such trees of size ℓ. There are 4^ℓ ways of choosing an unlabelled tree of size ℓ (see [7]). The root of our bad tree can be chosen in $n_i + m_i$ ways. We fix an unlabelled tree and the root of our bad tree and then choose the remaining nodes of the tree in a breadth-first fashion. Each node can then be chosen in at most $(2^{k-k^\epsilon})^{10} \cdot k$ ways. Thus, the total number of choices for a $(4, 10)$-bad tree of size ℓ is at most

$$N_\ell \;=\; (n_i + m_i) \times 4^\ell \times (2^{10(k-k^\epsilon)} \cdot k^{10})^{\ell-1} \tag{6}$$

Combining (5) and (6) for sufficiently large k we have,

$$\mathbf{E}[X_\ell^i] \;\leq\; (n_i + m_i) \cdot 2^{(-0.5k^{\delta_i+1/2}+10k+2)\ell} \tag{7}$$

Taking $\ell = c_\epsilon \cdot \log(n_i + m_i))$, for large k one gets that $\mathbf{E}[X_\ell^i] < 1/2$, since $\delta_i + 1/2 > 1$ and c_ϵ is a constant that just depends on ϵ.

■

Proof:[of Lemma 11] Combining Lemma 13 and Lemma 15 with Markov's inequality yields the bound on the size of the components of the hypergraph induced at the end of Phase 1 (i.e. $\mathcal{H}_1$). The argument for analyzing Phase 2 when *Case 1* holds is very similar, just noting that each component C_i of $\mathcal{H}_1$ now has size at most $2^{6k} c_\epsilon \log(n+m)$. The remaining case is when the following is true:

$$2^{6k} > \log(n+m)/\log\,\log(n+m) \tag{8}$$

Consider any component C of $\mathcal{H}_1$. Let X_B be the random variable representing the total number of bad vertices and bad edges generated by phase 2 for C. Clearly, (8) and bounds from proposition 14 imply

$$\mathbf{E}[X_B] \;\leq\; |C| \cdot 2^{-0.5k^{\beta+1/2}} \;\leq\; 2^{6k} c_\epsilon \log(n+m) \times 2^{-0.5k^{\beta+1/2}} = o(1) \tag{9}$$

since $\beta + 1/2 > 1$. Using Markov's inequality, with probability at most $\mathbf{E}[X_B] \to 0$, there exists a bad vertex or a bad edge. In otherwords, almost surely no bad edges or vertices are present, whence at the end of Phase 2 there are no uni-coloured edges remaining with very high probability. ■

Now we are ready to prove Theorem 10.

Proof:[of Theorem 10] Using Lemma 11, it is easy to see that the expected number of times that Phase 1 is run is at most 2 and that phase 2 is run is at most $2(n+m)$. It is straight-forward to see that each run of Phase 1 and Phase 2 takes poly-time. If $2^{6k} \leq \log(n+m)/\log\,\log(n+m)$, then clearly the brute force method of Phase 3 will take polytime. In the other case, as there are no uni-coloured edges at end of Phase 2, any completion of the partial colouring is proper.

■

5 Derandomization

We will use the method of conditional expectations (due to Erdös and Selfridge [6]) to derandomize our algorithm from the previous section which would prove theorem 1 (see also [9] for an exposition of this technique).

Proof:[of Theorem 1] In Phase 1 of 2-COLOUR, when we consider assigning a colour to a vertex v, we compute the colour that minimizes the expected number of bad $(4, 10)$-trees of size $\ell_1 = (c_\epsilon/k^{\epsilon+1/2}) \log(n+m)$ that would arise if the colouring is randomly completed as prescribed in Phase 1. This colour is then assigned to v. The number of such bad trees that we need to take into consideration is bounded from above by $(n+m) \cdot 2^{10(k-k^\epsilon)\ell_1} \cdot 4^{\ell_1}$. Since $\epsilon > 1/2$, it gets easily verified that the number of such bad trees is $O((n+m)^{1+\delta})$ for every $\delta < 1$ when k is growing function of n. Otherwise it is $O(n^c)$ for some constant c. As shown in Lemma 15, the expected number of such bad $(4, 10)$-trees at the beginning of Phase 1 is less than a half. The deterministic variant of Phase 1 hence produces no bad trees and so $\mathcal{H}_1$ has no large components.

If $2^{6k} \leq \log(n+m)/\log\ \log(n+m)$, we apply a similar procedure to derandomize Phase 2 by considering trees of size $\ell_2 = (c_\epsilon/k^{\beta+1/2}) \log\ \log(n+m)$. The rest of the method and argument is same as above. For larger k, when colouring a vertex v in a component C of $\mathcal{H}_1$, we assign v the colour that minimizes the expected number of bad vertices and edges of C. Using the argument given in the proof of Lemma 11, we conclude that Phase 2 generates no bad vertices or edges. Thus, no edge at the end of Phase 2 is uni-coloured. ∎

References

[1] N. Alon. A parallel algorithmic version of the Local Lemma. *Random Structures and Algorithms*, 2:367–379, 1991.

[2] N. Alon and J. Spencer. *The Probabilistic Method*. Wiley, New York, 1992.

[3] J. Beck. An algorithmic approach to the Lovász Local Lemma. *Random Structures and Algorithms*, 2(4):343–365, 1991.

[4] J. Beck and S. Lodha. Efficient proper 2-coloring of almost disjoint hypergraphs. In *SODA*, pages 598–605, 2002.

[5] P. Erdös and L. Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In A. H. et. al., editor, *Infinite and Finite sets*, volume 11, pages 609–627. Colloq. Math. Soc. J. Bolyai, 1975.

[6] P. Erdös and J. Selfridge. On a combinatorial game. *Journal of Combinatorial Theory (A)*, 14:298–301, 1973.

[7] F. Harary and E. Palmer. *Graphical Enumeration*. Academic Press, 1st edition, 1973.

[8] M. Molloy and B. Reed. Further algorithmic aspects of the Local Lemma. In *STOC*, pages 524–529, 1998.

[9] M. Molloy and B. Reed. *Graph Colouring and the Probabilistic Method*. Springer, 2002.