A $(1+\epsilon)$-Approximation Algorithm for Partitioning Hypergraphs Using a New Algorithmic Version of the Lovász Local Lemma *

Mohammad R. Salavatipour †

Department of Combinatorics and Optimization, University of Waterloo,
Waterloo, ON N2L 3G1, Canada
E-mail: `mreza@math.uwaterloo.ca`

Abstract

In his seminal result, Beck gave the first algorithmic version of the Lovász Local Lemma by giving polynomial time algorithms for 2-coloring and *partitioning* uniform hypergraphs. His work was later generalized by Alon, and Molloy and Reed. Recently, Czumaj and Scheideler gave an efficient algorithm for 2-coloring non-uniform hypergraphs. But the partitioning algorithm obtained based on their second paper only applies to a more limited range of hypergraphs, so much so that their work doesn't imply the result of Beck for the uniform case. Here we give an algorithmic version of the general form of the Local Lemma which captures (almost) all applications of the results of Beck and Czumaj and Scheideler, with an overall simpler proof. In particular, if H is a non-uniform hypergraph in which every edge e_i intersects at most $|e_i|2^{\alpha k}$ other edges of size at most k, for some small constant α, then we can find a *partitioning* of H in expected linear time. This result implies the result of Beck for uniform hypergraphs along with a speedup in his running time.

Keywords: *Probabilistic Method, Lovász Local Lemma, Random trial, Hypergraph coloring.*

1 Introduction

The probabilistic method is used to prove the existence of an object with certain properties by showing that a randomly chosen object in an appropriate probability space has the desired properties with positive probabilities. Some applications, for example, are in proving the existence of efficient routing algorithms [15], disjoint paths in expander graphs [10, 11, 16], and many graph coloring problems [2, 13, 17, 19, 20]. In most applications, the probability that the randomly selected object has the desired property is not only positive, but is actually high and frequently tends to 1 as the parameters of the problem tend to infinity. In these cases, the proof yields a randomized algorithm for constructing an object with the required properties: we simply pick objects at random until we find one. Under fairly general conditions, these algorithms can be derandomized using the method of conditional probabilities due to Erdös and Selfridge [9].

*An earlier version of this paper appeared in the Proceedings of Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) 2003.

†Supported by Research Assistantship, Department of Computer Science, University of Toronto.

On the other hand, there are some tools, by which we can show that a certain event happens with positive, but exponentially small, probability. One of these tools is the Local Lemma, first proved by Erdös and Lovász [8], which turned out to be an extremely powerful tool.

Lemma 1.1 (General Local Lemma) *Let $E_1, \ldots, E_n$ be a set of "bad" events in an arbitrary probability space and let G be the dependency graph for them. That is, for every i, $1 \leq i \leq n$, the event E_i is mutually independent from all E_j with $(i,j) \notin G$. Assume that there exist $x_i \in [0,1)$ for all $1 \leq i \leq n$ with $\Pr[E_i] \leq x_i \prod_{(i,j)\in G}(1-x_j)$, for all i. Then with positive probability no bad event E_i occurs.*

There are many applications of the Local Lemma (see [2, 3, 8, 10, 12, 13, 15, 16, 19, 20]). Perhaps the most typical example of an application of the LLL is the hypergraph 2-coloring problem, which can be formulated as follows:

Corollary 1.2 *[8] Let $A_1, A_2, \ldots$ be n-element subsets of a finite set X, and assume that every A_i intersects at most 2^{n-3} other A_j, $n \geq 2$. Then there is a 2-coloring of X such that no A_i is monochromatic.*

For the Local Lemma, the simple general randomized procedure we mentioned above does not produce a polynomial time algorithm. Therefore, we need some more elegant techniques to get an algorithmic version of this lemma. In a breakthrough paper, Beck [4] developed the first algorithmic version of the LLL by giving a polynomial time algorithm for 2-coloring uniform hypergraphs, provided that the condition 2^{n-3} is replaced by a weaker bound $2^{\alpha n}$, such as $\alpha = \frac{1}{48}$. Alon [1] described a parallel version of this algorithm, for smaller values of α. These methods were further generalized by Molloy and Reed [18] to yield efficient algorithms for a wider range of applications of the LLL. Recently, some other algorithmic versions of the LLL have been presented by various authors [5, 6, 7, 14, 17]. Czumaj and Scheideler [6] extended the result of Beck [4] to non-uniform hypergraphs, by showing that:

Corollary 1.3 *[6] There exist constants α, β, λ, such that for every hypergraph H, if every edge $e \in H$ has size at least λ and intersects at most $\beta|e|2^{\alpha k}$ other edges of size at most k, then there is randomized algorithm that finds a 2-coloring of H in polynomial time w.h.p. The expected running time of the algorithm is linear in $\sum_{e\in H}|e|$.*

In [4], Beck studied a discrepancy version of the 2-coloring problem, which is also called the *hypergraph partitioning* problem, and gave the first polynomial time algorithm for it. Roughly speaking, this is the problem of 2-coloring the vertices of a hypergraph, such that in each edge the number of nodes of each color are relatively close. More formally:

Theorem 1.4 *[4] Assume that $0 < \epsilon \leq 1$ is a given real number. Let $H(V,E)$ be an n-uniform hypergraph in which each edge intersects at most $2^{\gamma n}$ other edges, where $\gamma = \gamma(\epsilon)$. If n is large enough with respect to ϵ, then we can find a 2-coloring $f : V \longrightarrow \{-1,+1\}$ in polynomial time such that, with $f(e) = \sum_{v\in e} f(v)$: for each $e \in H$, $|f(e)| \leq \epsilon n$.*

Among the known algorithmic versions of the LLL, only the result of Czumaj and Scheideler [7] can be used to extend this result to some non-uniform hypergraphs, but their result is somewhat weaker than Theorem 1.4, as it requires that every edge intersects at most $O(2^{O(k^\alpha)})$ other edges of size at most k, for some small constant α. As a consequence, their result is not strong enough to imply Theorem 1.4 for uniform hypergraphs.

Here we give a new and stronger algorithmic version of the LLL, which implies all the applications of the results of Beck [4] and Czumaj and Scheideler [6, 7]. In particular this result extends Theorem 1.4 to non-uniform hypergraphs. The overall analysis of the algorithm is simpler and under some slightly stronger conditions its expected running time is *linear*. Before stating our main theorem, we need a few definitions and notation.

Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a set of random trials, and $\mathcal{E} = \{E_1, \ldots, E_m\}$ be a set of "bad" events, where each E_i is determined by the outcomes of the trials in $F_i \subseteq \mathcal{F}$. Two events E_i and E_j are called neighbors if $F_i \cap F_j \neq \emptyset$. By $N(E_i)$ we mean the set of neighbors of event E_i. During the course of our algorithm, we will need to break some of the events into at most three smaller sub-events. For this reason, we need to assume that for every event E_i and every set $S \subseteq F_i$, the event E_i *restricted* to S is well-defined and is denoted by $E_i|_S$. For example, consider the problem of partitioning a non-uniform hypergraph $H(V, E)$ into two parts. That is, given H and real number $\epsilon > 0$, we want to find a 2-coloring $f : V \longrightarrow \{-1, 1\}$ such that with $f(e_i) = \sum_{v \in e_i} f(v)$, for each edge $e_i \in E$ we have $|f(e_i)| \leq \epsilon |e_i|$. For each vertex v_i we will have a random trial f_i which determines the color to be assigned to v_i and for each edge $e_j \in E$ the bad event E_j will be the event that $|f(e_j)| > \epsilon |e_j|$. If S_1, S_2, S_3 is a partition of the vertex set of e_j, T_1, T_2, T_3 is the corresponding partition of the trial set of E_j, and if we define $f(S_x) = \sum_{v \in S_x} f(v)$ $(1 \leq x \leq 3)$, then $E_j|_{T_x}$ can be defined as the event that $|f(S_x)| > \frac{\epsilon |S_x|}{3}$. By this definition, if none of $E_j|_{S_x}$ $(1 \leq x \leq 3)$ happens it means that $|f(S_x)| \leq \frac{\epsilon |S_x|}{3}$, for $1 \leq x \leq 3$, and since $f(e_j) = \sum_{1 \leq x \leq 3} f(S_x)$, therefore $|f(e_j)| \leq \sum_{1 \leq x \leq 3} |f(S_x)| \leq \epsilon |e_j|$. Thus E_j does not happen.

Theorem 1.5 *Assume that $\mathcal{F}$ and $\mathcal{E}$ are defined as above. There exists a constant δ such that for any $0 < \epsilon \leq 1$ the following holds:*
Suppose that every trial $f_i \in \mathcal{F}$ has a constant number of outcomes and we can carry out the random trial in time t_1. Let $p_i = e^{-\epsilon |F_i|}$ and suppose that for all $S \subseteq F_i$ it holds that:

- *if $|S| > \epsilon |F_i|$ then $\Pr[E_i|_S] \leq p_i$,*
- *if $|S| \leq \epsilon |F_i|$ then $\Pr[E_i|_S] = 0$, and*
- *knowing the outcomes of the trials in S, we can evaluate whether $E_i|_S$ holds or not in time t_2.*

Furthermore assume that for $x_i = e^{-\delta \epsilon^2 |F_i|}$ it holds that $x_i \leq \frac{1}{e}$ and:

$$p_i^{\epsilon} \leq x_i \prod_{E_j \in N(E_i)} (1 - x_j), \qquad 1 \leq i \leq m. \tag{1}$$

Then there is a randomized algorithm that finds the outcomes of the random trials in time $O((t_1 + t_2) \times \mathrm{Poly}(n+m))$ with high probability, where $\mathrm{Poly}(n+m)$ is a polynomial in $(n+m)$, such that for every event E_i, the set F_i is partitioned into at most 3 subsets $S_{i,1}, S_{i,2}, S_{i,3}$, so that $E_i|_{S_{i,1}}$, $E_i|_{S_{i,2}}$, and $E_i|_{S_{i,3}}$ are all false.

Remark 1: If we have the stronger assumption that for $x'_i = e^{-\delta \epsilon^3 |F_i|}$: $x'_i \leq e^{-1}$ and

$$p_i^{\epsilon^2} \leq x'_i \prod_{E_j \in N(E_i)} (1 - x'_j), \qquad 1 \leq i \leq m, \tag{2}$$

Then:

(i) We can still get polynomial running time even if the number of outcomes of each random trial $f_i \in \mathcal{F}$ is polylogarithmic in $n+m$.

(ii) Under some reasonably general assumptions, the expected running time of the algorithm is linear in $\sum_{i=1}^{m} |F_i|$. As an example, this applies to the hypergraph partitioning problem. We talk about this in more detail in Section 4.

It is straightforward to check that (2) implies (1).

Remark 2: Throughout the paper, we do not attempt to find the optimal values of the constants; rather we give the proofs based on their existence. As an example, it can be verified that $\delta \leq 1/600$ is enough for the theorem to hold.

Theorem 2.1 in Czumaj and Scheideler [7] needs the stronger requirements $p_i = e^{-|F_i|^{\epsilon}}$ and $x_i = e^{-\delta |F_i|^{\epsilon^2}}$ (which are even stronger than those in Remark 1). This is why their theorem does not match Theorem 1.4 for non-uniform hypergraphs. Also, the requirement of their theorem corresponding to Inequality (1) is stronger than Inequality (1), and they don't specify into how many sets each event E_i might be partitioned.

More importantly, there is an error in their theorem in that it only guarantees polynomial running time if the number of outcomes of each trial is constant (rather than polylogarithmic in $n+m$ as they claim) unless an inequality stronger than the one in their theorem is satisfied.[1] Our proof is similar to that of [7] with an overall simpler analysis. Using Theorem 1.5 we can show that:

Theorem 1.6 *Let $H(V,E)$ be a hypergraph with vertices $v_1,\ldots,v_n$ and edges $e_1,\ldots,e_m$. Suppose that a real $0<\alpha<1$ and an integer C are given. There exist constants $0<\beta,\gamma,\lambda<1$, such that if every edge e_j has size at least $1/\lambda$ and intersects at most $\beta|e_j|e^{\gamma k}$ other edges of size at most k, then we can find a C-coloring $f: V \longrightarrow \{1,2,\ldots,C\}$ in expected linear time, such that if $d_i(e_j) = |\{v \in e_j : f(v) = i\}|$ then for each $1 \leq i \leq C$ and each $1 \leq j \leq m$:*

$$|d_i(e_j) - \frac{|e_i|}{C}| \leq \frac{\alpha |e_i|}{C}.$$

One application of the hypergraph partitioning problem is in splitting expander graphs [12]. Frieze and Molloy [12] use an asymmetric version of the LLL to prove the existence of good splittings for expander graphs. The result of [7] is not general enough to replace the non-constructive version of the LLL in the proof of [12], but it is not very difficult to check that we can use Theorem 1.6 for this application. However, this doesn't yield a polynomial time algorithm, due to the fact that the number of events that Molloy and Frieze [12] define to apply the LLL, is exponential in the size of input.

The organization of the paper is as follows. In the next section we present the polynomial time algorithm for Theorem 1.5. Section 3 contains the analysis of the algorithm and the proof of correctness for it. More details of the proof are explained in Section 6. In Section 4 we show the conditions under which the expected running time of the algorithm would be linear. Finally, in Section 5 we explain how to use Theorem 1.5 to prove Theorem 1.6.

[1] The reason is that Step 3 of their algorithm cannot necessarily be performed, unless it is proved that the requirements of their theorem holds for the new set of (possibly reduced) events that have to be considered for each 2-component. This, of course, does not necessarily hold unless we start with an stronger inequality.

2 Description of the Algorithm

2.1 Overview of the Algorithm

The goal is to find a set of outcomes for the trials in $\mathcal{F}$ such that each trial set F_i (for event $E_i \in \mathcal{E}$) is partitioned into at most three subsets S_1, S_2, S_3 where all $E_i|_{S_1}$, $E_i|_{S_2}$, and $E_i|_{S_3}$ are false. The main idea of the algorithm is essentially the same as the ones in Beck [4] and Czumaj and Scheideler [6, 7]. There are two main steps in the algorithm:

Step 1: Choose an outcome for each trial uniformly at random.
Step 2: Select a (possibly empty) subset of trials of each event and redo them.

Of course, after Step 1, there might be many bad events that hold. Therefore, we must redo the trials in them. The general idea of this part of the algorithm is similar to the other known algorithmic versions of the Local Lemma, which is to show that the connected components constructed by these bad events are disjoint, and then to try to handle each one separately in later steps. To achieve this, we might break some of the basic events into some smaller events. For example, consider the situation in which an event E_i is true (and therefore its trials have to be redone). If there is another event $E_j \in N(E_i)$ which is also true then of course E_i and E_j will be in the same component and we cannot redo their trials independently. What if E_j and the restricted event $E_j|_{F_j - F_i}$ are both false, but E_j has more than $\epsilon|F_j|$ trials in common with E_i? In this case, although we don't have to redo the trials of E_j (because it is false), there is this possibility that after redoing the trials of E_i, E_j becomes true, although a large portion of it, i.e. $E_j|_{F_j - F_i}$ remains unchanged. This might happen, for instance, if $E_j|_{F_j \cap F_i}$ becomes true. For this reason, we break event E_j into two parts; one event is $E_j^1 = E_j|_{F_j \cap F_i}$ and the other is $E_j^2 = E_j|_{F_j - F_i}$. Each of these is called a *reduced* event. Now we consider the event E_j^1 to be in the same component as E_i (since it has common trials with E_i). If we find a set of outcomes for the trials in this component that makes all the (possibly reduced) events in the component false, then E_j^1 will become false. Since the trials of E_j^2 are disjoint form the component and therefore are not affected by this process of redoing the trials of the events of the component, E_j^2 remains false. Hence, the trial set of E_j is partitioned into two sets, one defining E_j^1 and one defining E_j^2, where both of these events are false at the end.

So our first goal after Step 1 is to find connected components of bad (possibly reduced) events. However, we cannot only focus on these components and redo their trials independently. The reason is that there might be an event, such as E_k, which is false even restricted to any subset $S \subseteq F_k$ and E_k does not intersect any other event in more than $\epsilon|F_k|$ trials. Therefore E_k cannot become true if we redo the trials of any single event in $N(E_k)$. So it does not belong to any component even as a reduced event. But it is possible that E_k is intersecting too many bad events of different components that are true, so many so that the that the total number of trials that E_k has in common with those events is more than $\epsilon|F_k|$. In this case, if we are not careful enough when we redo the trials of those components, then E_k may end-up being true after all. So, although E_k (even restricted to any subset of its trial set) is not true, it is "dangerous" as it is intersecting too many events that are true. So we must consider the trials of the (possibly reduced) events of the components that are connected by dangerous events together. This will result in larger connected components.

However, if an event E_x is not true (even when restricted to any subset of F_x) and it is not dangerous, then by redoing the trials of these larger components, E_x cannot become true, since the subset S of trials of E_x that may have been redone in the second step has size at most $\epsilon|F_x|$.

Therefore $\Pr[E_x|_S] = 0$. Also $E_x|_{F_x - S}$ is not true either by our assumption. So E_x remains false after Step 2.

It can be shown that there is a suitable set of outcomes of the random trials of these components such that every (reduced) event is false and the dangerous events are not true either. So if the size of each of these components is small enough, then we can find these suitable outcomes by exhaustive search in Step 2! We show that in fact this is the case. That is, with high probability, there is no components which has more than $O(\ln m)$ trials. So exhaustive search yields polynomial running time.

2.2 Details of the Algorithm

In this subsection, we describe how to perform each step of the algorithm in more details. We also explain the extra steps for the cases that we have the stronger assumptions of Remark 1.

Step 1: Choose an outcome for each trial $f_i \in \mathcal{F}$ uniformly at random.

We call events $E_1, \ldots, E_m$ *basic* events and any event defined by a basic event E_i restricted to a subset $S \subseteq F_i$ is called a *reduced* event and is denoted by $E_i|_S$. Now we find the connected components of bad events using a Breadth First Search (BFS) algorithm:

Set $R = \mathcal{F}$ /* R is going to be the set of remaining trials of $\mathcal{F}$ */
for i=1 **to** m **do**
  if $E_i|_{F_i \cap R}$ is true **then**
    $R = R - \{F_i \cap R\}$
    BFS $(E_i|_{F_i \cap R})$

Procedure BFS (E: event)
$Fathers = \{E\}$;
$CE = \{E\}$ /* CE will be the set of core events of the 1-component */
repeat
  $Children = \emptyset$
  for all $E_j \in N(Fathers)$ in increasing j **do**
    $F'_j = F_j \cap R$
    if $E_j|_{F'_j}$ is true **then** /* So $|F'_j| \geq \epsilon |F_j|$*/
      $R = R - F'_j$
      $Children = Children \cup \{E_j|_{F'_j}\}$
      $CE = CE \cup \{E_j|_{F'_j}\}$
      endif
  $Fathers = Children$
  endfor
until $Children = \emptyset$

We call each connected component found by the BFS algorithm a *1-component* and CE the set of *core* events of the 1-component. Every basic event E_j that has at least $\epsilon|F_j|$ trials in common with CE is called a *participating* event of the 1-component. For every participating event E_j let $F'_j \subseteq F_j$

be the set of trials of E_j that are *not* in CE. It can be seen that if we redo the trials in the core events, $E_j|_{F'_j}$, the reduced event of E_j induced by its trials which are *not* in CE, cannot be true, otherwise $E_j|_{F'_j}$ would have been added to the 1-component as a core event. Another fact is that by this algorithm, every basic event is reduced to at most one reduced (core) event.

After finding 1-components, it might happen that some basic events are not core events of any 1-component but are intersecting (or maybe participating in) "too many" 1-components and so after redoing the trials in each of the 1-components, these basic events become true. Such a basic event is called *dangerous*. More formally, a basic event E_i, that is not a core event, is dangerous if more than $\epsilon|F_i|$ of its trials belong to (different) 1-components. So we introduce the *2-component* structures. A 2-component is basically a maximal set of 1-components that are connected by dangerous events. The core set of a 2-component C is the union of the core sets of its 1-components and is denoted by CE_C. An event E_i is participating in a 2-component if at least $\epsilon|F_i|$ trials of it are covered by the core events of the 2-component. So every dangerous event is also a participating event. For every basic event E_i that is not participating in any 2-component let $F'_i \subseteq F_i$ be the (possibly empty) set of trials of E_i that are in the core events of the 2-components and let $F''_i = F_i - F'_i$. By definition, $|F'_i| < \epsilon|F_i|$ and therefore, even after redoing the trials of the core events of the 2-components, $E_i|_{F'_i}$ cannot become true. Since $E_i|_{F''_i}$ is not true either (otherwise it would have been added as a core event to a 1-component), therefore F_i is partitioned into at most two subsets F'_i and F''_i, such that even after redoing the trials of the core events of the 2-component $E_i|_{F'_i}$ and $E_i|_{F''_i}$ are both false.

Lemma 2.1 *The following statements are true:*

(i) Every basic event is participating in at most one 2-component.

(ii) For every 2-component C, there is a set of outcomes of the trials of the core set of C such that each participating event is partitioned into at most two subsets, each of which is false.

Proof: (i) If an event E_i is participating in two different 2-components then it has at least $\epsilon|F_i|$ trials in common with the core events of each of them. In that case, it would be a dangerous event and those two 2-components would have been merged into one 2-component, a contradiction.

(ii) Every basic event participates as at most one core event in CE_C. For every participating basic event E_i, we can consider the union of the trials of E_i that are covered by the 1-components of C, as one reduced event E'_i. Let us denote the trial set of E'_i by F'_i. By definition of a participating event, the size of F'_i is at least $\epsilon|F_i|$. So every participating event E_i is divided into at most two parts: one reduced event whose trial set is F'_i and is a subset of CE_C, and another part which has no intersection with CE_C. Note that $F_i - F'_i$ corresponds to $S_{i,1}$ defined in Theorem 1.5, and F'_i corresponds to $S_{i,2}$ for now (but may actually be divided later into at most two subsets, corresponding to $S_{i,2}$ and $S_{i,3}$). The event $E_i|_{F_i - F'_i}$ is false, even if we redo the trials of CE_C. So if we prove the existence of a set of outcomes of the trials of CE_C that makes every core event and $E_i|_{F'_i}$ false, then we are done. The existence of this set of outcomes can be proved by the Local Lemma. Each basic event appears as at most one (possibly reduced) event and the probability of each reduced event E'_i satisfies $\Pr[E'_i] \le p_i$. So the conditions of the Local Lemma in the statement of the theorem hold. ∎

Therefore, we can consider each 2-component independently. The main lemma (Lemma 2.2, part (i)) shows that with high probability, the number of trials in the set of core events of any 2-component, which we call the *size* of the 2-component, will be at most $O(\ln m)$. After the first step, if there are any 2-components of size greater than $O(\ln m)$ we redo the first step. The expected number of times we have to redo it is at most a constant. Thus, if the number of outcomes of the random trials in $\mathcal{F}$ is constant, then we can use exhaustive search in the next step:

Step 2: If the number of outcomes of the random trials is $O(1)$ and we don't require expected linear time then using exhaustive search on each 2-component find a suitable set of outcome for the random trials of the core events such that no core event is true. The algorithm will stop at this point.

If the number of outcomes of the random trials is polylogarithmic in $n + m$ (and therefore exhaustive search on 2-components of size $O(\ln m)$ does not yield polynomial running time) or if we want speed up in the algorithm then, instead of doing exhaustive search at this point, we may run Step 1 on each of the created 2-components independently, to obtain sufficiently small 2-components. To be able to do this we need the stronger assumptions explained in Remark 1.

More specifically, for each 2-component C and each event E_i which is a core event or a participating event in CE_C, let $F_i' \subseteq F_i$ be the set of trials of E_i that are in CE_C and let $E_i' = E_i|_{F_i'}$.

Step 3: If the stronger conditions of Remark 1 hold then for each 2-component of size $O(\ln m)$ obtained by Step 1, independently, consider the set of reduced events E_i' defined above as the new set of *basic* events and apply Step 1 to them, to find sufficiently small 2-components.

Lemma 2.2 (Main Lemma) *For any constant $\alpha > 0$:*

(i) After the first step, with probability at least $1 - \frac{1}{m^\alpha}$, there is no 2-component of size more than $O(\frac{\alpha}{\epsilon}\ln m)$.

(ii) If the assumptions of Remark 1 hold we can do Step 3. Furthermore, after Step 3, with probability at least $1 - \frac{1}{(\ln m)^\alpha}$, there is no 2-component of size more than $O(\frac{\alpha}{\epsilon}\ln\ln m)$.

If we get to run Step 3 then, by the second part of Lemma 2.2, with high probability we will get 2-components each of which has size at most $O(\ln\ln m)$. Now we can find the required set of outcomes of the trials of the core sets of these small 2-components using exhaustive search.

Step 4: Using exhaustive search find a suitable set of outcomes for the random trials of core events of the new 2-components such that no core event is true anymore.

Recall from the proof of Lemma 2.1 that after Step 1 of the algorithm, each event E_i is partitioned into at most two sets, one of which corresponds to $S_{i,1}$, and the other one is F_i'. After Step 3 of the algorithm, the event induced by F_i' might be divided into two smaller sets. These two sets correspond to $S_{i,2}$ and $S_{i,3}$. Therefore the total number of sets to which an event E_i might be partitioned is at most three.

3 Correctness of the Algorithm

In this section we prove Lemma 2.2. Our proof uses some key ideas from both [4] and [7]. In particular, we incorporate the "sum over all trees" approach from [4] into the "levels analysis" from [7].

The main idea of the proof is to associate tree-like structures to 1- and 2-components that can be created by the algorithm. These structures are purely combinatorial and are introduced to find an upper bound on the expected number of 2-components of a certain size created by the algorithm. The structure associated to a 2-component will be called a (1,2)-tree. We show how to construct a (1,2)-tree from a 2-component and then prove that the expected number of (1,2)-trees is exponentially small (in terms of their sizes). Then this will be used to show that with high probability there is no 2-components of size larger than $O(\ln m)$ after step 1 of the algorithm, which is what we need in the main lemma.

More specifically, to a 1-component C^1 we associate a directed tree $T^1 = (V_C, E_C)$ as follows: create a vertex v_i for each basic event E_i that has a core event in C^1 and create edge (v_i, v_j) if E_i was a father of E_j during the BFS procedure and i is the smallest index among the fathers of E_j. We call this structure a *1-tree*. The *index* of the vertex associated to event E_i is the same as the index of E_i, which is i. Note that the vertex corresponding to the initial event of C^1 will be the root of T^1.

To a 2-component C^2 we associate a tree-like structure $T^2 = (V_C \cup V_D, E_C \cup E_D)$, where V_C and E_C are the unions of the vertex sets and the edge sets of all the 1-trees corresponding to the 1-components of C^2, respectively. Let $D \in C^2$ be a dangerous event. Assume that there are k 1-components in C^2 intersecting D, called $C^1_1, \ldots, C^1_k$. We create a vertex v_D in V_D. For each C^1_i let v^1_i be the vertex corresponding to a core event intersecting D with the smallest index in C^1_i. We create the (undirected) edge (v^1_i, v_D) in E_D for every $1 \leq i \leq k$. This tree is called a *(1,2)-tree*. For consistency, we call each edge of E_C a *1-tree edge* and each edge of E_D a *2-tree edge*. For a (1,2)-tree, the node in V_C that has only outgoing edges and has the minimum index corresponds to the initial event of the first 1-component of the corresponding 2-component.

It is easy to see that for each 2-component there is a unique (1,2)-tree. Also, given a (1,2)-tree T^2 we can uniquely determine the 1-components that correspond to the 1-trees of T^2, and the order in which the events were added to each 1-component, by looking at the direction of the edges of the 1-trees. In particular, the first 1-component is the one that corresponds to the 1-tree whose root has the smallest index amongst the roots of the 1-trees of T^2. But we cannot uniquely specify the trial sets of the core events of a 2-component. The reason is that it can happen that a basic event has a core (reduced) event in one 2-component, say C, and overlaps with the trials of another 2-component, say C'. This can happen if C' is constructed before C. But given only the (1,2)-tree corresponding to C we cannot determine that some of the trials of this basic event are not in C. This is the problem for which Czumaj and Scheideler [7] introduced a more complicated structure, namely 3-components and the related tree structures, in order to make a one to one mapping. But this problem can be solved in a significantly easier way as we don't require a one to one mapping between 2-components and (1,2)-trees. If we show that with very high probability there is no (1,2)-tree of a certain size then it definitely shows that with at least the same probability there is no 2-component of that size. So to prove the main lemma, it is enough to show that with probability at least $1 - \frac{1}{m^\alpha}$ there is no (1,2)-tree of size greater than $O(\frac{\alpha}{\epsilon} \ln m)$, after Step 1 of the algorithm. So from now on, our goal is to show this statement, which will easily imply the main lemma.

In order to count the number of (1,2)-trees of a certain size we need to give a precise definition for the size of a (1,2)-tree. Let's define the order of a basic event E_i (or any reduced event of it), denoted by O_{E_i}, to be $\epsilon|E_i|$. For a set P of (possibly reduced) events, we define the order of P, denoted by O_P, to be the sum of the orders of the events in P. The order of a (1,2)-tree T, denoted by O_T, is the sum of the orders of the events whose corresponding vertices are in $V_C(T)$.

A set P of reduced events is a "possible set of core events" if there is a set of outcomes of the trials in $\mathcal{F}$ and a run of Step 1 of the algorithm such that all the events in P are in the set of core events of (possibly different) 2-components produced. Similarly, a (1,2)-tree T is a "possible (1,2)-tree" if there is a set of outcomes of the trials in $\mathcal{F}$ and a run of Step 1 of the algorithm that produces a 2-component corresponding to T.

Consider any fixed possible set P of core events. To find an upper bound on the probability that all the events in P become core after Step 1 of the algorithm, an important point to note is that these events are disjoint. Since each (possible) core event in P that is a reduced event of a basic event E_i has size at least $\epsilon|F_i|$, by the conditions of Theorem 1.5, the probability of it to become true is at most $p_i = e^{-\epsilon|F_i|}$, at the beginning of the algorithm and independently from other events. Therefore, if Z_P denotes the event that the events of P actually become core after Step 1 of the algorithm, then:

$$\Pr[Z_P] \leq \prod_{E_i \in P} p_i. \tag{3}$$

For a possible (1,2)-tree T, let Z_T denote the event that T becomes a (1,2)-tree after Step 1 of the algorithm. Using (3) it is straightforward (but subtle) to show that:

$$\Pr[Z_T] \leq \prod_{E_j : v_j \in V_C(T)} p_j. \tag{4}$$

The proof of (4) is given in Section 6.

Now our goal is to prove the following lemma, by which the main lemma can be proved easily.

Lemma 3.1 *The expected number of (1,2)-trees T with order at least Ψ is at most $21me^{-\Psi/20}$.*

For a possible (1,2)-tree T, we say T starts from E_0, if E_0 is the initial event of the first 1-component of T. Define

$$\mathcal{T} = \{\text{possible (1,2)-trees } T \text{ with order } O_T = \Psi \text{ that start at } E_0\}.$$

Now let $\mathcal{T}' \subseteq \mathcal{T}$ be the set of (1,2)-trees obtained after Step 1 of the algorithm that are also in $\mathcal{T}$. By this definition and (4):

$$E[|\mathcal{T}'|] = \sum_{T \in \mathcal{T}} \Pr[Z_T] \leq \sum_{T \in \mathcal{T}} \prod_{E_j : v_j \in V_C(T)} p_j. \tag{5}$$

Computing a good upper bound for $E[|\mathcal{T}'|]$ by bounding the right-hand-side of (5) directly, is very complicated and involves dealing with dependencies of probabilities of neighboring events of the trees in $\mathcal{T}'$. Instead, we proceed indirectly by defining a whole new experiment and a new set, which will be called $\mathcal{T}''$. We show that $E[|\mathcal{T}''|]$ is equal to the right-hand-side of Inequality (5), then we bound $E[|\mathcal{T}''|]$ and this bound combined with (5) will give us an upper bound for $E[|\mathcal{T}'|]$. The new experiment, called the *helper experiment*, is as follows: for every basic event $E_i \in \mathcal{E}$ we flip a coin which comes up heads with probability p_i. We assign a *tag* to event E_i based on the result of the coin flip. The tag is either a *heads tag* or a *tails tag*. The dependency graph of the events is defined in a natural way: create a vertex for each event and two vertices are adjacent iff the corresponding events intersect. Consider the connected components of the events that have a heads tag in the corresponding dependency graph, and call each of them a *pseudo 1-component*. Every event E_j having a tails tag, whose trial set is intersecting at least two different pseudo 1-components and at least $\epsilon|F_j|$ of its trials are part of pseudo 1-components, is a *pseudo dangerous* event. A maximal set of pseudo 1-components connected by pseudo dangerous events is a *pseudo 2-component*.

By these definitions, each possible 2-component C^2 can be associated to a unique possible pseudo 2-component PC^2 in a natural way: the events of PC^2 having heads tags are precisely the basic events of the core events of C^2 and the pseudo dangerous events of PC^2 are precisely the dangerous events of C^2. This mapping is valid since the union of the trial sets of the core events of C^2 is a subset of the union of the trial sets of the basic events of C^2, which is the same as the union of the trial sets of events of PC^2 with heads tags. Therefore, if a basic event is a dangerous event in C^2 it will be a pseudo dangerous event in PC^2. Consequently, any possible (1,2)-tree can be associated to a unique pseudo 2-component. Note that neither of these correspondences is a one to one mapping.

For a fixed set of events P, let Z'_P denote the event that after performing the helper experiment all the events in P get heads tags. So by this definition:

$$\Pr[Z'_P] = \prod_{E_i \in P} p_i. \tag{6}$$

Also, for a fixed possible (1,2)-tree T define the event Z'_T to be the event that we get the corresponding pseudo 2-component after performing the helper experiment. Then:

$$\Pr[Z'_T] = \prod_{E_j : v_j \in V_C(T)} p_j. \tag{7}$$

Define $\mathcal{T}'' \subseteq \mathcal{T}$ to be the set of (1,2)-trees whose corresponding pseudo 2-component is created after performing the helper experiment. By this definition:

$$E[|\mathcal{T}''|] = \sum_{T \in \mathcal{T}} \Pr[Z'_T] = \sum_{T \in \mathcal{T}} \prod_{E_j : v_j \in V_C(T)} p_j. \tag{8}$$

Since the right-most expressions in (5) and (8) are the same, therefore:

$$E[|\mathcal{T}'|] \leq E[|\mathcal{T}''|]. \tag{9}$$

This simplifies our analysis significantly, since we can now continue the analysis based on the helper experiment, and bound $E[|\mathcal{T}''|]$ to get a bound for $E[|\mathcal{T}'|]$.

For each (1,2)-tree $T \in \mathcal{T}''$, we consider the vertices of $V_C(T)$ level-wise. The only vertex in level zero is the one corresponding to E_0. All the vertices in $V_C(T)$ connected to it by 1-tree edges, and those that are in $V_C(T)$ but at distance 2 from it, using only 2-tree edges, are in the second level. Everything that is connected to a node in the second level using a 1-tree edge, or by a path of length two of 2-tree edges, (regardless of the directions of the edges) is in the third level, and so on. Note that the edges between level i and $i+1$ may have different directions. So the levels we consider for T do not necessarily follow the directions of the edges in the 1-trees of T.

Remark 3: A careful reader might have noticed the huge difference that we would have had if we were to analyze $\mathcal{T}'$, rather than $\mathcal{T}''$. Namely, the direction of the edges between different levels would play a crucial role in computing the probabilities of the events, which would involve calculating very complicated conditional probabilities. We get rid of these complications by switching to the helper experiment.

Define the order of a level to be the sum of the orders of the events that correspond to the vertices in that level. We will show in Section 6, Lemma 6.3, that the order of each level should be at least $1/\delta$. For any two sets Q and R of events, we say R is an *extension* of Q if there exists a (1,2)-tree $T \in \mathcal{T}$ and an integer i, such that Q is the set of events corresponding to the vertices in level i and R is the set of events corresponding to the vertices in level $i+1$ of T. For the moment just consider those (1,2)-trees in $\mathcal{T}''$ with k levels, whose level i has order S_i, for a fixed set of $S_1, \ldots, S_k$. Denote these (1,2)-trees by $\mathcal{T}''_{S_1,\ldots,S_k}$. If we denote the order of T, excluding the initial event E_0, by ψ, that is $\psi = \Psi - S_0$, and we let $l_1,\ldots,l_k$ denote possible extensions for levels $1,\ldots,k$, respectively, then using (8):

$$
\begin{aligned}
E[|\mathcal{T}''_{S_1,\ldots,S_k}|] &= e^{-S_0} \sum_{\substack{\text{all } l_1\text{'s} \\ O_{l_1}=S_1}} \sum_{\substack{\text{all } l_2\text{'s} \\ O_{l_2}=S_2}} \cdots \sum_{\substack{\text{all } l_k\text{'s} \\ O_{l_k}=S_k}} \prod_{t=1}^{k} \prod_{E_j\in l_t} p_j \\
&= e^{-S_0} \sum_{\substack{\text{all } l_1\text{'s} \\ O_{l_1}=S_1}} \Big(\prod_{E_{j_1}\in l_1} p_{j_1} \sum_{\substack{\text{all } l_2\text{'s} \\ O_{l_2}=S_2}} \Big(\prod_{E_{j_2}\in l_2} p_{j_2} \cdots \sum_{\substack{\text{all } l_k\text{'s} \\ O_{l_k}=S_k}} \prod_{E_{j_k}\in l_k} p_{j_k} \Big)\Big) \ldots \Big). \qquad (10)
\end{aligned}
$$

Denote the set of all extensions R with $O_R = r$ of a set Q by $\mathrm{EXT}(Q,r)$. For a set Q of events let $X_{Q,r}$ be the number of extensions R such that $R \in \mathrm{EXT}(Q,r)$ and all events in R have heads tags. Therefore, by (6):

$$
E[X_{Q,r}] = \sum_{R:R\in\mathrm{EXT}(Q,r)} \Pr[Z'_R] = \sum_{R:R\in\mathrm{EXT}(Q,r)} \prod_{E_j\in R} p_j. \qquad (11)
$$

By this equation, the most internal summation in (10) is in fact $E[X_{l_{k-1},S_k}]$. In Section 6, Lemma 6.4, we show that $E[X_{Q,r}] \le e^{-r/8} e^{O_Q/16}$. Therefore, $E[X_{l_{k-1},S_k}] \le e^{-S_k/8} e^{S_{k-1}/16}$. Using this fact, Inequality (10) can be written as:

$$
E[|\mathcal{T}''_{S_1,\ldots,S_k}|] \le e^{-S_0} \prod_{i=0}^{k-1} e^{-S_{i+1}/8} e^{S_i/16}. \qquad (12)
$$

We need the following combinatorial lemma to prove Lemma 3.1.

Lemma 3.2 *Let N_x^+ denote the set of integers greater than or equal to x. For $1 \le k \le \delta\psi$, define EQ_k to be the equation $S_1 + S_2 + \ldots + S_k = \psi$, where the domain of each variable S_i ($1 \le i \le k$) is $N^+_{\frac{1}{\delta}}$. Then for sufficiently small $\delta > 0$, the sum of the number of the solutions of all equations EQ_k ($1 \le k \le \delta\psi$) is at most $e^{\psi/80}$.*

Proof: Let's call the equation $r_1 + r_2 + \ldots + r_{\delta\psi} = \psi$, in which the r_i's are the variables whose domain is N_0^+, the *reference* equation. To each solution of equation EQ_k, for $1 \le k \le \delta\psi$, we associate a unique solution of the reference equation: set $r_i = S_i$, for $1 \le i \le k$, and $r_j = 0$, for $k < j \le \delta\psi$. Therefore, the sum of the number of solutions of all EQ_k equations (for $1 \le k \le \delta\psi$) with domain N_k^+, is not more than the number of solutions of the reference equation with domain N_0^+. From elementary combinatorics we know that the number of non-negative integer solutions of the reference equation is $\binom{\psi+\delta\psi-1}{\delta\psi-1}$, which is less than $\binom{\psi+\delta\psi}{\delta\psi}$. Using Stirling's approximation for $n!$:

$$
\begin{aligned}
\binom{\psi+\delta\psi}{\delta\psi} &\le \frac{[\psi(1+\delta)]^{\psi+\delta\psi}}{(\delta\psi)^{\delta\psi}\psi^{\psi}} \\
&\le \frac{e^{\delta\psi}(1+\delta)^{\delta\psi}}{\delta^{\delta\psi}} \\
&\le e^{\delta\psi(1+\ln(1+\frac{1}{\delta}))} \\
&\le e^{\psi/80}
\end{aligned}
$$

if δ is sufficiently small. ■

Proof of Lemma 3.1: Using (12), definition of $\mathcal{T}''_{S_1,\ldots,S_k}$, and Lemma 3.2:

$$\begin{aligned} E[|\mathcal{T}''|] &\le \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta}\le S_1,\ldots,S_k\le\psi \\ \sum_{1\le i\le k} S_i=\psi}} E[|\mathcal{T}''_{S_1,\ldots,S_k}|] \\ &\le e^{-S_0} \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta}\le S_1,\ldots,S_k\le\psi \\ \sum_{1\le i\le k} S_i=\psi}} \prod_{j=0}^{k-1} e^{-S_{j+1}/8} e^{S_j/16} \\ &\le e^{-S_0} e^{-\psi/8} e^{\Psi/16} \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta}\le S_1,\ldots,S_k\le\psi \\ \sum_{1\le i\le k} S_i=\psi}} 1 \\ &\le e^{-\Psi/8} e^{\Psi/16} e^{\Psi/80} \\ &= e^{-\Psi/20} \end{aligned} \tag{13}$$

for sufficiently small δ. Therefore, using (9) and (13):

$$E[|\mathcal{T}'|] \le e^{-\Psi/20}. \tag{14}$$

Since we have at most m events that can be the initial event of a (1,2)-tree T with total order at least Ψ, using the bound in (14), after Step 1 of the algorithm:

$$\begin{aligned} E[|\{(1,2)\text{-trees } T \text{ of order at least } \Psi\}|] &\le m \sum_{k\ge\Psi} e^{-k/20} \\ &\le 21me^{-\Psi/20}. \end{aligned}$$

■

Proof of Lemma 2.2:

(i) Consider the bound in Lemma 3.1. The expected number of such (1,2)-trees is at most $m^{-\alpha}$ if $\Psi \ge 20((\alpha+1)\ln m + 4)$. For each such (1,2)-tree T, the number of trials of T is at most $\frac{\Psi}{\epsilon}$. Therefore, for any constant $\alpha > 0$, with probability at least $1 - \frac{1}{m^\alpha}$ there is no (1,2)-tree T for which the number of trials of the basic events corresponding to the vertices in $V_C(T)$ is larger than $\frac{20}{\epsilon}((\alpha+1)\ln m + 4) \in O(\frac{\alpha}{\epsilon}\ln m)$. Thus, with probability at least $1 - \frac{1}{m^\alpha}$ there is no 2-component of size greater than $O(\frac{\alpha}{\epsilon}\ln m)$.

(ii) Recall that in Step 3, we have a new set of basic events for each 2-component C that we consider: for each event E_i which is a core event or a participating event in CE_C, we consider $E'_i = E_i|_{F'_i}$ as a basic event, where $F'_i \subseteq F_i$ is the set of trials of E_i that are in CE_C. First we must show that if the assumptions of Remark 1 hold then the conditions of Theorem 1.5 hold for this new set of basic events.

Suppose that $|F'_i| = \alpha_i|F_i|$, for some $\alpha_i \le 1$. Note that since $|F'_i| \ge \epsilon|F_i|$: $\alpha_i \ge \epsilon$. For this new set of basic events, we have $p'_i = e^{-\epsilon|F'_i|} = e^{-\epsilon\alpha_i|F_i|}$, and for all $S \subseteq F'_i$:

- if $|S| > \epsilon|F'_i|$ then $\Pr[E'_i|_S] \le p'_i$, and
- if $|S| \le \epsilon|F'_i|$ then $\Pr[E'_i|_S] = 0$.

We need to show that with $x_i = e^{-\delta\epsilon^2|F'_i|}$: $x_i \le e^{-1}$ and $p'^{\epsilon}_i \le x_i \prod_{E'_j \in N(E'_i)}(1-x_j)$, or equivalently

$$\frac{p'^{\epsilon}_i}{x_i} \le \prod_{E'_j \in N(E'_i)} (1-x_j).$$

If the conditions of Remark 1 hold, i.e. for $x'_i = e^{-\delta\epsilon^3|F_i|}$: $x'_i \le e^{-1}$ and (2) holds then, since $x_i = e^{-\delta\epsilon^2\alpha_i|F_i|} \le e^{-\delta\epsilon^3|F_i|} = x'_i$, therefore: $x_i \le e^{-1}$ and

$$\frac{p_i^{\epsilon^2}}{x'_i} = e^{-(1-\delta)\epsilon^3|F_i|} \le \prod_{E'_j \in N(E'_i)} (1-x'_j) = \prod_{E'_j \in N(E'_i)} (1-e^{-\delta\epsilon^3|F_j|}). \tag{15}$$

Thus:

$$\begin{aligned}
\frac{p'^{\epsilon}_i}{x_i} = e^{-(1-\delta)\epsilon^2\alpha_i|F_i|} &\le e^{-(1-\delta)\epsilon^3|F_i|} && \text{since } \alpha_i \ge \epsilon \\
&\le \prod_{E'_j \in N(E'_i)} (1-e^{-\delta\epsilon^3|F_j|}) && \text{by (15)} \\
&\le \prod_{E'_j \in N(E'_i)} (1-e^{-\delta\epsilon^2\alpha_i|F_j|}) && \text{since } \alpha_i \ge \epsilon \\
&= \prod_{E'_j \in N(E'_i)} (1-x_j)
\end{aligned}$$

as wanted. Therefore, we can run Step 1 on each 2-component and Lemma 2.1 holds.

Now we show that with high probability all the 2-components after this second run of Step 1 have size $O(\ln\ln m)$. Let C be a 2-component generated after the first run of Step 1 with order at most $\sigma \ln m$. We will prove in Section 6.2, Lemma 6.2, that there are at most $\sigma(\ln m)e^{\delta k}$ events of order at most k that are participating in C. Therefore, by Lemma 3.1, the expected number of (1,2)-trees of order at least Ψ is at most

$$\begin{aligned}
\sum_{k \ge \Psi} \sigma(\ln m) e^{\delta k} e^{-k/20} &\le \sigma(\ln m) \sum_{k \ge \Psi} e^{-k/25} \\
&\le 30\sigma(\ln m)e^{-\Psi/25}.
\end{aligned}$$

for sufficiently small δ, which in turn is at most $(\ln m)^{-\alpha}$ if $\Psi \ge 25((\alpha+1)\ln\ln m + \ln(30\sigma))$. Therefore, since the number of trials of a (1,2)-tree with order Ψ is at most $\frac{\Psi}{\epsilon}$, with probability at least $1 - \frac{1}{(\ln m)^{\alpha}}$, for any constant $\alpha > 0$, there is no 2-component of size greater than $\frac{25}{\epsilon}((\alpha+1)\ln\ln m + \ln(30\sigma)) \in O(\frac{\alpha}{\epsilon}\ln\ln m)$.

■

4 Expected Linear Time

In this section we show that under reasonably general assumptions (that are clarified below), the algorithm of Theorem 1.5 can be implemented in a way such that its expected running time is linear in $M = \sum_{i=1}^{m} |F_i|$.

The first assumption is that the requirements of Remark 1 hold, and therefore, we can run Steps 3 and 4 of the algorithm.

The second assumption is $t_1 = O(1)$, i.e. we can carry out each random trial in constant time. Let $\{\sigma_1, \sigma_2, \ldots, \sigma_x\}$ be the set of possible outcomes of a trial $f_i \in \mathcal{F}$, where x is a constant. For each (possibly reduced) event E_i we will keep x different integers $\eta_1, \ldots, \eta_x$, where η_j is the number of trials in F_i whose outcome is σ_j, $1 \le j \le x$. Let's call this set of integers Info(E_i).

The last assumption is that for each E_i, knowing Info(E_i), we can evaluate in $O(1)$ time whether E_i holds or not (and therefore $t_2 = O(1)$). Having these assumptions, we show that each step of the algorithm can be implemented in time $O(M)$.

Step 1: In this step, first we choose an outcome for each trial f_i uniformly at random. This can be done in time $O(n)$.

For the BFS procedure, the important point is that each event E_i will be evaluated at most once for each trial $f_a \in F_i$. Also, whenever $E_j|_{F'_j}$ is true (and so we will have a core event on F'_j), after performing $R = R - F'_j$, we update info(E_l) for all neighboring events E_l of E_j. Therefore, Info(E_l) will be updated at most $|F_l|$ times and each update takes constant time. So the overall running time of finding the 1-components will be $O(M)$.

To compute the 2-components, it is obvious that each event is checked at most once to see if it is dangerous or not. Thus, the total running time of finding the 2-components is $O(M)$. We repeat Step 1 until all the 2-components produced have "*order*" at most $\mu \ln m$, for some constant μ (where we mean *order* in the sense that we defined it in Section 3). By the proof of Lemma 2.2, the expected number of times we have to repeat Step 1 would be $O(1)$. So, in time $O(M)$ we find 2-components all of which have "*order*" at most $\mu \ln m$.

Since we want to obtain expected linear time, we do not perform exhaustive search (Step 2). Instead we perform Step 3.

Step 3: Suppose that C^2 is some 2-component with "*order*" Ψ. Therefore, the sum of the number of trials of all (possibly reduced) events of C^2 is at most $\frac{\Psi}{\epsilon}$. We apply Step 1 to this 2-component, independently from the other 2-components. We repeat this until the produced 2-components of C^2 all have "*order*" at most $\mu \ln \frac{\Psi}{\epsilon}$. Again, using the proof of Lemma 2.2, the expected number of times that we have to repeat this is at most $O(1)$. So the total time spent to find small 2-components in C^2 would be $O(\frac{\Psi}{\epsilon})$. By summing this up over all 2-components, the total running time of Step 3 will be at most $O(M)$.

Step 4: Assume that the 2-components produced for C^2 in Step 3 are $C_1^2, C_2^2, \ldots, C_r^2$. Finding a good set of outcomes for the trials of C_i^2 using exhaustive search takes at most $O(x^{|C_i^2|} \sum_{E_j \in C_i^2} |F_j|)$. Therefore, using the fact that $\sum_{E_j \in C_i^2} |F_j| \le \frac{\Psi}{\epsilon}$, the total running time of Step 4 on C^2 would be at most:

$$\sum_{i=1}^{r} O\left(x^{\frac{\mu}{\epsilon} \ln \frac{\Psi}{\epsilon}} (\Psi/\epsilon)\right) = O((\Psi/\epsilon)^{\frac{\mu}{\epsilon} \ln x + 2}).$$

By Lemma 3.1, the expected number of 2-components C^2 having "order" at least Ψ is at most

$21me^{-\Psi/20}$. Since

$$\sum_{\Psi=1}^{\infty} O((\Psi/\epsilon)^{\frac{\mu}{\epsilon}\ln x+2})e^{-\Psi/20}$$

is a convergent sum, the total running time for the exhaustive search on all 2-components is at most:

$$\sum_{\Psi=1}^{\frac{\mu}{\epsilon}\ln m} O((\Psi/\epsilon)^{\frac{\mu}{\epsilon}\ln x+2})me^{-\Psi/20} = O(m).$$

5 Proof of Theorem 1.6

Assume that we color each vertex uniformly at random with colors $\{1, 2, \ldots, C\}$. For each vertex v_i of the hypergraph we define a trial f_i that has C outcomes, one for each possible color. For each edge e_j we define C sets of trials, one for each color $1 \leq c \leq C$. For each color c, let $F_{c,j}$ be the set of trials whose corresponding vertices are in e_j. Note that the set of trials for a fixed j and different values of c is the same, $|F_{c,j}| = |e_j|$, and $F_{c,j}$ is intersecting at most $C(\beta|e_j|e^{\gamma k} + 1)$ other sets of size at most k. For each set S of trials and each color c, define $d_c(S)$ to be the number of trials in S whose outcome is c, and define the bad event $E_{c,j}$ to be the event that $|d_c(F_{c,j}) - \frac{|e_j|}{C}| > \frac{\alpha|e_j|}{C}$. Using Chernoff's bound for the tails of binomial distribution we have:

$$\begin{aligned}
\Pr[E_{c,j}] &= \Pr[|BIN(|e_j|, \frac{1}{C}) - \frac{|e_j|}{C}| > \frac{\alpha|e_j|}{C}] \\
&\leq 2e^{-\alpha^2|e_j|/3C} \\
&\leq e^{-\alpha^2|e_j|/4C}.
\end{aligned}$$

The last inequality holds if λ is sufficiently small. For each set $S \subseteq F_{c,j}$, we define $E_{c,j}|_S$, the event $E_{c,j}$ restricted to S, to be the event that $|d_c(S) - \frac{|S|}{C}| > \frac{\alpha|e_j|}{3C}$. Let $\epsilon = \frac{\alpha^2}{60C}$, $p_{c,j} = e^{-\epsilon|e_j|}$. Since we want to obtain expected linear time, we must show that the stronger assumptions of Remark 1 hold. So let $x_{c,j} = e^{-\delta\epsilon^3|e_j|}$, for sufficiently small δ. By the Chernoff bound, $\Pr[E_{c,j}] \leq p_{c,j}$ and if $|S| > \epsilon|e_j|$ then $\Pr[E_{c,j}|_S] \leq p_{c,j}$ and if $|S| \leq \epsilon|e_j|$ then $\Pr[E_{c,j}|_S] = 0$. Using the fact that $1 - x \geq e^{-2x}$, for $0 \leq x \leq \frac{1}{2}$, we have:

$$\begin{aligned}
x_{c,j} \prod_{F_j \cap F_t \neq \emptyset} (1 - x_{s,t}) &\geq e^{-\delta\epsilon^3|e_j|} \prod_{k \geq 1/\lambda} \left(1 - e^{-\delta\epsilon^3 k}\right)^{C(\beta|e_j|e^{\gamma k}+1)} \\
&\geq e^{-\delta\epsilon^3|e_j|} \exp\left\{-2\beta C'|e_j| \sum_{k\geq 1/\lambda} e^{-\delta\epsilon^3 k} e^{\gamma k}\right\} \\
&\qquad \text{(For } C' = C+1 \text{ and sufficiently small } \lambda) \\
&\geq \exp\left\{\left[-\delta\epsilon^3 - \left(\frac{2\beta C' a^{1/\lambda}}{1-a}\right)\right]|e_j|\right\} \qquad \text{(where } a = e^{\gamma - \delta\epsilon^3}) \\
&\geq e^{-\epsilon^3|e_j|} \qquad \text{(if } \beta \text{ and } \gamma \text{ are sufficiently small)} \\
&= p_{c,j}^{\epsilon^2}.
\end{aligned}$$

So we satisfy all the requirements of Theorem 1.5 and Remark 1, and therefore, there is a randomized algorithm that runs in expected linear time that finds the outcomes of the trials, such that each set

$F_{c,j}$ is partitioned into at most 3 subsets $S_{c,j,1}, S_{c,j,2}, S_{c,j,3}$ such that $E_{c,j}$ restricted to each of them does not hold, i.e. for $1 \leq k \leq 3$, $|d_c(S_{c,j,k}) - \frac{|S|}{C}| \leq \frac{\alpha|e_j|}{3C}$. Thus, for each set $F_{c,j}$:

$$\begin{aligned} |d_c(F_{c,j}) - \frac{|e_j|}{C}| &\leq \sum_{1\leq k\leq 3} |d_c(S_{c,j,k}) - \frac{|S|}{C}| \\ &\leq \frac{\alpha|e_j|}{C}. \end{aligned}$$

This gives the required coloring of the vertices of H.

6 Details of the Proof of Theorem 1.5

Proof of Inequality (4): Suppose that $T_1^1, T_2^1, \ldots, T_r^1$ are the 1-trees of T and let $Z_{T_i^1}$ be the event that T_i^1 is built during Step 1 of the algorithm. Then

$$\Pr[Z_T] \leq \Pr[\bigwedge_{i=1}^{r} Z_{T_i^1}].$$

For each i, let Q_i be the sequence of 1-trees (not necessarily in T) corresponding to the 1-components that are built before the first 1-component containing any reduced event of a basic event corresponding to a vertex of T_i^1. Note that by this definition, given Q_i and T_i^1 we can uniquely determine the set of core events that must be formed in any 1-component whose corresponding 1-tree is T_i^1. Letting this set of (possible) core events be P and by using (3):

$$\Pr[Z_{T_i^1}|Q_i] = \Pr[Z_P] \leq \prod_{E_j:v_j\in V_C(T_i^1)} p_j. \tag{16}$$

We also have:

$$\Pr[Z_T] \leq \prod_{i=1}^{r} \Pr[Z_{T_i^1}| \bigwedge_{j=1}^{i-1} Z_{T_j^1}]. \tag{17}$$

If we define $\mathcal{Q}_i$ to be the set of all sequences Q_i each of which contains $T_1^1, \ldots, T_{i-1}^1$, then:

$$\Pr[Z_{T_i^1}| \bigwedge_{j=1}^{i-1} Z_{T_j^1}] = \sum_{Q_i\in\mathcal{Q}_i} \Pr[Z_{T_i^1}|Q_i] \times \Pr[Q_i| \bigwedge_{j=1}^{i-1} Z_{T_j^1}]. \tag{18}$$

Using (16) and (18):

$$\begin{aligned} \Pr[Z_{T_i^1}| \bigwedge_{j=1}^{i-1} Z_{T_j^1}] &\leq \prod_{E_k:v_k\in V_C(T_i^1)} p_k \sum_{Q_i\in\mathcal{Q}_i} \Pr[Q_i| \bigwedge_{j=1}^{i-1} Z_{T_j^1}] \\ &\leq \prod_{E_k:v_k\in V_C(T_i^1)} p_k. \end{aligned}$$

Now combining this with (17) yields:

$$\Pr[Z_T] \leq \prod_{i=1}^{r} \prod_{E_j:v_j\in V_C(T_i^1)} p_j = \prod_{E_j:v_j\in V_C(T)} p_j.$$

■

In the rest of this section, we prove the upper bound of $E[X_{Q,r}]$ used to prove Inequality (12). To do that, the following technical lemma, which is proved in [6], is very helpful.

6.1 A Technical Lemma

Consider some fixed set Q of basic events. Assume that for every event $A \in Q$ we have a non-negative random variable Λ_A, called *contribution*. Let us define the contribution Λ_Q of Q to be the sum of the contributions of all events of Q. Also,

$$S_\psi^Q = \{R \subseteq N(Q) : \Lambda_R = \psi\}.$$

Lemma 6.1 *[6] Let $\gamma > \frac{1}{288}$ be an arbitrary constant. Furthermore, let Q be any set of basic events and $(\Lambda_B)_{B \in N(Q)}$ be any sequence of contributions with the property that:*

(i) the contribution of any event in $N(R)$ is either 0 or at least $1/\gamma$.

(ii) $|\{(A, B) \in Q \times N(Q) : \Pr[1 \leq \Lambda_B \leq k] > 0\}| \leq ce^{\gamma k}$.

(iii) there is a $\phi \geq \frac{1}{6}$ so that for every event $B \in N(Q)$, $\Pr[\Lambda_B = k] \leq e^{-\phi k}$ independently of other events of positive contributions.

Then

$$E[|S_\psi^Q|] \leq e^{-\phi\psi/2} e^{c/48}.$$

6.2 Bounding the Expected Number of Extensions of a Specific Order

The main purpose of this section is to bound $E[X_{Q,r}]$. First we focus on those extensions of Q that only use 1-tree edges.

Lemma 6.2 *For every event E_i:*

$$|\{E_j \in N(E_i) : |F_j| \leq k\}| \leq \epsilon^2 e^{\delta\epsilon^2 k} |F_i|$$

Proof: By Inequality (1) of Theorem 1.5:

$$\begin{aligned}
e^{-\epsilon^2|F_i|} &\leq e^{-\delta\epsilon^2|F_i|} \prod_{E_j \in N(E_i)} (1 - e^{-\delta\epsilon^2|F_j|}) \\
&\leq \prod_{E_j \in N(E_i)} \exp\left(-e^{-\delta\epsilon^2|F_j|}\right) \\
&= \exp\left(- \sum_{E_j \in N(E_i)} e^{-\delta\epsilon^2|F_j|}\right) \\
\Longrightarrow \epsilon^2|F_i| &\geq \sum_{E_j \in N(E_i)} e^{-\delta\epsilon^2|F_j|}.
\end{aligned}$$

The lemma then follows easily from the last inequality. ■

Lemma 6.3 *Every event E_i has order at least $\frac{1}{\epsilon\delta}$.*

Proof: By one of the requirements of Theorem 1.5: $x_i \leq \frac{1}{e}$. Therefore:

$$e^{-\delta\epsilon^2|F_i|} \leq e^{-1} \Longrightarrow O_{E_i} = \epsilon|F_i| \geq \frac{1}{\epsilon\delta}.$$

∎

Now we are going to apply Lemma 6.1. For every event $E_j \in N(Q)$, let the contribution Λ_{E_j} of E_j be defined as O_{E_j}, if E_j gets heads tag, after performing the helper experiment, and 0 otherwise. So condition (i) of Lemma 6.1 holds by Lemma 6.3.

Recall that the edges between Q and R can have either of two possible directions. That is, there are up to two possibilities for connecting $E_i \in Q$ to $E_j \in R$. To consider both of these possibilities, we assume that there are (virtually) two copies of each $E_j \in R$, one for each possible directed edge. Therefore, using Lemma 6.2, the above assumption, and the fact that $O_{E_j} \leq k$ implies $|F_j| \leq \frac{k}{\epsilon}$:

$$\begin{aligned}|\{(E_l, E_j) \in Q \times N(Q) : \Pr[1 \leq \Lambda_{E_j} \leq k] > 0\} \cup \{(E_j, E_l) \in N(Q) \times Q : \Pr[1 \leq \Lambda_{E_j} \leq k] > 0\}| \\ \leq \textstyle\sum_{E_l \in Q} 2\epsilon^2 e^{\delta\epsilon k}|F_l| \\ \leq 2e^{\delta k} O_Q.\end{aligned}$$

So condition (ii) is true with $c = 2O_Q$. For condition (iii), we are going to use the same trick as used by Czumaj and Scheideler [7]. That is, we are going to use only part of the probability for an event to get heads tag and save the other part for when we want to consider 2-tree edges. Since each event E_j gets heads tag with probability p_j, therefore $\Pr[\Lambda_{E_j} = O_{E_j}] = p_j$. For the reason mentioned above, and by setting $\phi = \frac{1}{2}$, we get $\Pr[\Lambda_{E_j} = O_{E_j}] \leq e^{-\phi O_{E_j}}$ and so condition (iii) of Lemma 6.1 holds, too. Thus, if we denote by $X^1_{Q,r}$ the number of extensions R with contribution r where all the events in R have heads tags and are connected to the events in Q using only 1-tree edges after performing the helper experiment, then:

$$E[X^1_{Q,r}] \leq e^{-r/4} e^{O_Q/24}. \tag{19}$$

Now consider the case that there is a 2-tree edge from an event in Q having heads tags to a pseudo dangerous event E_j. This case is significantly different from the previous one since E_j itself does not have a probability. Instead, using the same idea as in [7], we "borrow" probabilities from the basic events that are intersecting E_j and have a core event (this is the point we use the other part of the probability of a core event that we reserved in the previous case). We define the *virtual* order of the pseudo-dangerous event E_j, denoted by w_{E_j}, to be the sum of the orders of the set S of core events that are intersecting E_j and whose corresponding pseudo 1-components were merged into a pseudo 2-component because of E_j. That is: $w_{E_j} = \sum_{E_s \in S} O_{E_s}$. Note that the events in S must cover at least $\epsilon|F_j|$ trials of E_j. Therefore:

$$w_{E_j} = \sum_{E_s \in S} O_{E_s} = \sum_{E_s \in S} \epsilon|F_s| \geq \epsilon^2|F_j| = \epsilon O_{E_j}. \tag{20}$$

Since for every core event $E_s \in S$ we still have a probability $e^{-O_{E_s}/2}$ available, we can assign a probability of $e^{-w_{E_j}/2}$ to E_j. Now we are ready to apply Lemma 6.1 again. Define the contribution

Λ_{E_j} of E_j as w_{E_j} if the events in S got heads tags after the helper experiment, and 0 otherwise. Condition (i) follows with $\gamma = \delta$ from Lemma 6.3 and (20). By Lemma 6.2 and (20):

$$|\{E_j \in N(E_i) : w_{E_j} \leq k\}| \leq |\{E_j \in N(E_i) : \epsilon^2|F_j| \leq k\}| \leq \epsilon^2 e^{\delta k}|F_i| = e^{\delta k} w_{E_i}.$$

So we satisfy condition (ii). Finally, condition (iii) holds with $\phi = 1/2$. Thus, it follows from Lemma 6.1 that if we denote the expected number of extensions R with contribution r of events having heads tags that are connected to Q by a path of length two of 2-tree edges, by $X^2_{Q,r}$, then:

$$E[X^2_{Q,r}] \leq e^{-r/4} e^{O_Q/48}. \tag{21}$$

Lemma 6.4 *For sufficiently small δ:*

$$E[X_{Q,r}] \leq e^{-r/8} e^{O_Q/16}.$$

Proof: Using (19) and (21), we bound the expected number of extensions R of a set Q with $O_R = r$. Let $U_{\delta,r} = \{0, 1/\delta, 1/\delta + 1, \ldots, r\}$.

$$\begin{aligned} E[X_{Q,r}] &\leq \sum_{k \in U_{\delta,r}} E[X^1_{Q,k}] E[X^2_{Q,(r-k)}] \\ &\leq \sum_{k \in U_{\delta,r}} e^{-k/4} e^{O_Q/24} e^{-(r-k)/4} e^{O_Q/48} \\ &\leq (r - \frac{1}{\delta} + 1) e^{-r/4} e^{O_Q/16} \\ &\leq e^{-r/8} e^{O_Q/16}, \end{aligned}$$

if δ is sufficiently small. ∎

Acknowledgment: Most of this work was done when I was a graduate student in the Department of Computer Science at the University of Toronto. I would like to thank Mike Molloy, for introducing me to the problem and for many useful discussions and suggestions. I extend my appreciation to two anonymous referees for their very helpful suggestions and comments.

References

[1] N. Alon, "A parallel algorithmic version of the Local Lemma", *Random Structures and Algorithms* 2 (1991), 367-378. A preliminary version appeared in Proc. of 32nd IEEE Symp. Foundations of Computer Science, (FOCS) 1991.

[2] N. Alon, C. McDiarmid, and B. Reed, "Acyclic coloring of graphs", *Random structures and Algorithms* 2 (1991), 277-288.

[3] N. Alon and J.H. Spencer, "The probabilistic Method", WileyInterscience Ser. Discrete Math and Optimization, Wiley, New York, 2000.

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

[5] J. Beck and S. Lodha, “Efficient Proper 2-coloring of Almost Disjoint Hypergraphs”, in *Proc. of 13th annual ACM-SIAM Symp. on Discrete Algorithms*, (SODA) 2002.

[6] A. Czumaj and C. Scheideler, “Coloring non-uniform hypergraphs: A new algorithmic approach to the general Lovász local lemma”, *Random Structures and Algorithms* 17 (2000). Earlier version in *Proc. of 11th annual ACM-SIAM Symp. on Discrete Algorithms*, (SODA) 2000, 30-39.

[7] A. Czumaj and C. Scheideler, “An algorithmic approach to the general Lovász local lemma with applications to scheduling and satisfiability problems”, *Proc. of 32nd ACM Symp. on Theory of Computing*, (STOC) 2000, 38-47.

[8] P. Erdös and L. Lovász, “Problems and results on 3-chromatic hypergraphs and some related questions”, Infinite and Finite Sets (A. Hajnal et. al. eds), Colloq. Math. Soc. J. Bolyai, 11 (1975), pp. 609-627.

[9] P. Erdös and J. Selfridge, “On a combinatoriall game”, *J. Comb. Theory Ser. A* 14 (1973), 298-301.

[10] A.M. Frieze, “Disjoint paths in expander graphs via random walks: a short survey”, *Proc. Of Random '98 LNCS 1518* (1998), 1-14.

[11] A.M. Frieze, “Edge disjoint paths in expander graphs”, *SIAM J. on Computing* 30(6) 2001, 1790-1801.

[12] A.M. Frieze and M. Molloy, “Splitting an expander graph”, *J. of Algorithms* 33 (1999), 166-172.

[13] J. Kahn, “Asymptotically good list-colorings”, *J. Comb. Theory Ser. A* 73 (1996), 1-59.

[14] T. Leighton, C. Lu, S. Rao, and A. Srinivasan, “New algorithmic aspects of the Local Lemma with applications to routing and partitioning”, *SIAM J. on Computing* 31(2), 2001, 626-641. Earlier version appeared as two separate papers, both in *Proc. of the 10th annual ACM-SIAM Symp. on Discrete Algorithms*, (SODA) 1999.

[15] T. Leighton, B.M. Mags, and S.B. Rao, “Packet routing and job-shop scheduling in $O(Congestion + Dilation)$ steps”, *Combinatorica* 14 (1994), 167-180.

[16] T. Leighton and S. Rao, “Circuit switching: a multicommodity flow based approach”, *Proc. of a Workshop on Randomized parallel computing*, 1996.

[17] M. Molloy and B. Reed, “Graph Colouring via the Probabilistic Method”, Springer, 2002.

[18] M. Molloy and B. Reed, “Further algorithmic aspects of the Local Lemma”, *In Proc. of the 30th ACM Symp. on Theory of Computing*, (STOC) 1998, 524-529.

[19] M. Molloy and B. Reed, “A bound on the total chromatic number”, *Combinatorica* 18 (1998), 241-280.

[20] B. Reed, “ω, Δ, and χ”, *J. Graph Theory* 27 (1998), 177-212.