Coloring Non-uniform Hypergraphs: A New Algorithmic Approach to the General Lovász Local Lemma*

Artur Czumaj and Christian Scheideler
Department of Mathematics & Computer Science
and Heinz Nixdorf Institute
Paderborn University, 33095 Paderborn, Germany
Email: {artur,chrsch}@uni-paderborn.de

Abstract

The Lovász Local Lemma is a sieve method to prove the existence of certain structures with certain prescribed properties. In most of its applications the Lovász Local Lemma does not supply a polynomial-time algorithm for finding these structures. Beck was the first who gave a method of converting some of these existence proofs into efficient algorithmic procedures, at the cost of loosing a little in the estimates. He applied his technique to the symmetric form of the Lovász Local Lemma and, in particular, to the problem of 2-coloring uniform hypergraphs.

In this paper we investigate the general form of the Lovász Local Lemma. Our main result is a randomized algorithm for 2-coloring *non-uniform* hypergraphs that runs in expected *linear* time. Even for uniform hypergraphs no algorithm with such a runtime bound was previously known, and no polynomial-time algorithm was known at all for the class of non-uniform hypergraphs we will consider in this paper. Our algorithm and its analysis provide a novel approach to the general Lovász Local Lemma that may be of independent interest. We also show how to extend our result to the c-coloring problem.

*Research partially supported by DFG-Sonderforschungsbereich 376 "Massive Parallelität: Algorithmen, Entwurfsmethoden, Anwendungen."

1 Introduction

The probabilistic method is used to prove the existence of objects with desirable properties by showing that a randomly chosen object from an appropriate probability distribution has the desired properties with positive probability. In most applications this probability is not only positive, but is actually high and frequently tends to 1 as the parameters of the problem tend to infinity. In such cases, the proof usually supplies an efficient randomized algorithm for producing a structure of the desired type.

There are, however, certain examples, where one can prove the existence of the required combinatorial structure by probabilistic arguments that deal with rare events; events that hold with positive probability which is exponentially small in the size of the input. This happens often when using the Lovász Local Lemma. We state it in its general form.

Lemma 1.1 (Lovász Local Lemma [8]) *Let $A_1, \ldots, A_n$ be "bad" events in an arbitrary probability space. Let G be a dependency graph for the events $A_1, \ldots, A_n$. (That is, for every i, $1 \le i \le n$, the event A_i is mutually independent of all the events A_j with $(i,j) \notin G$.) Assume that there exist $x_j \in [0,1)$ for all $1 \le j \le n$ with*

$$\Pr[A_i] \le x_i \prod_{(i,j)\in G} (1 - x_j)$$

for all $1 \le i \le n$. Then

$$\Pr[\bar{A}_1 \cap \ldots \cap \bar{A}_n] \ge \prod_{i=1}^{n} (1 - x_i) \ ,$$

that is, with positive probability no bad event A_i holds.

Many applications of the Lovász Local Lemma can be found in the literature (see, e.g., [2, 4, 5, 7, 8, 9, 11, 12, 13, 14, 16, 15, 19, 20, 21, 22]). In all of these applications the Lovász Local Lemma only guarantees an extremely small (though positive) probability that no bad event holds. To turn proofs using the Lovász Local Lemma into efficient algorithms, even random ones, proved to be difficult for many of these applications. In a breakthrough paper [6], Beck presented a method of converting some applications of the Lovász Local Lemma into polynomial-time algorithms (with some sacrifices made with regards to the constants in the original application). Alon [1] provided a parallel variant of the algorithm and simplified the arguments used. His method was further generalized by Molloy and Reed [17] to yield efficient algorithms for a number of applications of the Lovász Local Lemma. However, all of these approaches can be applied only to the *symmetric* form of the Lovász Local Lemma, in which a very regular structure of the events under consideration is required.

Molloy and Reed [17] also found methods that could possibly be applied to problems that require the general Lovász Local Lemma, but as it was pointed out by the authors, they may require to prove some (possibly difficult) concentration-like properties for each problem under consideration.

1.1 New results

In this paper we present a novel method for turning some applications of the general LLL into efficient algorithms. We apply it to the problem of 2-coloring non-uniform hypergraphs. No polynomial-time algorithm has been found so far for this problem.

To start with, we first require some notation. Given a hypergraph $\mathcal{H} = (V, E)$, let the *(1-)neighborhood* $N(e)$ of an edge $e \in E$ be defined as $N(e) = \{e' \in E \setminus \{e\} \mid e \cap e' \neq \emptyset\}$. Similarly, for $\mathcal{E} \subseteq E$ we define $N(\mathcal{E}) = \bigcup_{e\in\mathcal{E}} N(e) \setminus \mathcal{E}$. For any $e \in E$, let $|e|$ denote the *size* of edge e (i.e., the number of nodes e contains).

A hypergraph $\mathcal{H} = (V, E)$ is *c-colorable* if c colors suffice to color its nodes in such a way that no hyperedge in E is monochromatic (i.e., has only nodes of a single color).

The following result due to Erdős and Lovász [8] follows easily from the Lovász Local Lemma.

Theorem 1.2 [8] *Consider any hypergraph $\mathcal{H} = (V, E)$ in which every edge has at least $k \geq 2$ nodes and no edge intersects more than d other edges. If $\mathrm{e}\,(d+1) \leq 2^{k-1}$, then $\mathcal{H}$ is 2-colorable.*

We note that Radhakrishnan and Srinivasan [19] recently improved this theorem. They showed that if no edge intersects more than $0.17 \cdot \sqrt{k/\ \ln k} \cdot 2^k$ edges, then for a sufficiently large k, hypergraph $\mathcal{H}$ is 2-colorable. However, their result is also non-contructive.

Efficient algorithms for finding such a 2-coloring were given by Beck [6] and later by Alon [1]. They show the following result.

Theorem 1.3 [1, 6] *There is a positive constant $c > 1$, such that if $\mathrm{e}\,(d+1) \leq 2^{k/c}$, then any hypergraph $\mathcal{H}$ in which every edge contains at least k nodes and no edge intersects more than d other edges can be 2-colored in polynomial time.*

Theorems 1.2 and 1.3 are interesting mostly for uniform hypergraphs. (Indeed, the way how one could color the nodes of a hypergraph $\mathcal{H}$ in which every edge contains at least k nodes is by reducing each edge to arbitrarily chosen k of its nodes and solving the problem for uniform hypergraphs.) However, they provide no reasonable bounds for non-uniform hypergraphs in which the edge sizes can vary arbitrarily (especially, when some edges might be of very small size and some other edges of very large size).

One can generalize Theorem 1.2 to the following result for non-uniform hypergraphs.

Theorem 1.4 *Let $\mathcal{H}$ be a hypergraph with edges $e_1, \dots, e_m$. For every i, $1 \leq i \leq m$, let A_i be the event that e_i is monochromatic. Furthermore, let $x_i = 2^{-\delta_i\,|e_i|}$ (for a suitably chosen $0 < \delta_i < 1$) for all $1 \leq i \leq m$. If it holds that*

$$\Pr[A_i] \leq x_i \prod_{e_j \in N(e_i)} (1 - x_j)$$

for all $1 \leq i \leq m$ for the case that each node is colored independently and uniformly at random (i.u.r.), then $\mathcal{H}$ is 2-colorable.

None of the techniques developed so far can be used to construct an efficient algorithm for finding a 2-coloring for this class of hypergraphs. We will present a randomized algorithm that finds a 2-coloring in an efficient way for the following hypergraphs.

Theorem 1.5 *There exist constants $K, \Delta, \mathcal{E} > 0$ such that for any $k \geq K$, $0 < \delta \leq \Delta$, and $0 < \epsilon \leq \mathcal{E}$ it holds:*

Consider any hypergraph $\mathcal{H}$ with edges $e_1, \dots, e_m$, in which every edge is of size at least k. Let A_i be the event that e_i is monochromatic, $1 \leq i \leq m$. Furthermore, let $x_i = 2^{-\delta\,|e_i|}$ for all $1 \leq i \leq m$. If it holds that

$$(\Pr[A_i])^{\epsilon} \leq x_i \prod_{e_j \in N(e_i)} (1 - x_j)$$

for all $1 \leq i \leq m$ for the case that the color of each node is chosen i.u.r., then there is a randomized algorithm that finds a 2-coloring of $\mathcal{H}$ in expected time linear in $\sum_i |e_i|$.

Remark 1.6 *In this paper we did not make any attempt to optimize the values of K, Δ and $\mathcal{E}$. In our proofs we require $\epsilon \leq 1/24$, $\delta \leq \epsilon^2/12$, and $k \geq 1/\delta$ to obtain a polynomial-time algorithm. For a linear-time algorithm we need $\epsilon \leq 1/24^2$ and $\delta \leq \epsilon^{3/2}/12$. If at least 4 colors are available then we show in Section 5 that the hyperedges are allowed to be of arbitrary size, i.e., $K = 1$.*

Observe that Theorem 1.5 contains as a special case Theorem 1.3. Thus it is its generalization to non-uniform hypergraphs. Most remarkable about this theorem is that we are able to construct an algorithm that runs in expected *linear time*. No such a fast algorithm was known before even for uniform hypergraphs.

Theorem 1.5 implies the following result, answering an open question of J. Beck.

Corollary 1.7 *Let $\mathcal{H} = (V, E)$ be a hypergraph. There exist positive constants c_1, c_2, c_3 so that if every edge in $\mathcal{H}$ is of size at least c_1 and for every integer k no edge $e \in E$ intersects more than $c_2 \cdot |e| \cdot 2^{c_3 \cdot k}$ edges of size at most k, then one can find in polynomial time a 2-coloring of $\mathcal{H}$, w.h.p.*

The underlying structure of our 2-coloring algorithm is similar to that of Beck [6] and Alon [1] for uniform hypergraphs:

Step 1: Color each node of $\mathcal{H}$ i.u.r.
Step 2: Select nodes that require a recoloring.
Step 3: Recolor the selected nodes.

The main novel idea of our algorithm is a *much more restrictive selection of the nodes that need a recoloring* and *allowing edges to be reduced to a fraction of their nodes* if necessary. These two features of our algorithm enable us to prove that each set of selected edges covers a small (at most logarithmic) number of nodes, with high probability. This will allow us to find in polynomial time a 2-coloring even for non-uniform hypergraphs.

As in the paper due to Alon [1], our approach to bound the runtime of the algorithm is to count certain structures that could "witness" a bad behavior of the algorithm. Our structures, however, are chosen in such a way that they allow us to obtain much more precise bounds than those obtained by Alon. Their investigation is the main new technique in the analysis of our algorithm.

Our techniques for the 2-coloring problem easily extend to the problem of c-coloring non-uniform hypergraphs. Many other applications also seem to be in reach:

Consider any discrete, finite application of the general Lovász Local Lemma. Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be its set of events and $\mathcal{T} = \{t_1, \ldots, t_m\}$ be the set of independent trials the events are based on. Each trial has a set of possible outcomes which may be viewed as its set of colors. Suppose that A_i only depends on a set of trials $T_i \subseteq \mathcal{T}$. Then the problem of making all events in $\mathcal{A}$ false can be interpreted as a generalized form of the hypergraph coloring problem, where the T_i represent the hyperedges and $\mathcal{T}$ represents the set of nodes.

In this paper we shall only present randomized, sequential algorithms. We remark, however, that our algorithms for the 2-coloring and c-coloring problem can be transformed into deterministic polynomial-time algorithms using the techniques in [3, 10, 18] (see also the discussion in [1, p.371] and in [19, Section 3]). Our method can also be implemented as a parallel $\mathcal{NC}$ algorithm. Since these two results can be obtained using standard methods, we leave their details to the reader.

1.2 Organization of the paper

In the next section we will present a polynomial-time algorithm for the 2-coloring problem. Section 3 contains the analysis of the algorithm. In Section 4 we show how to transform the polynomial-time algorithm into a linear-time algorithm. Afterwards, we present in Section 5 a generalization to the c-coloring problem and conclude with some open problems. Some technical parts of our calculations can be found in the appendix.

2 Description of the Algorithm

We first present a polynomial-time algorithm for the 2-coloring problem. Consider any hypergraph $\mathcal{H} = (V, E)$ that fulfills the conditions in Theorem 1.5, and let ϵ be chosen as in Remark 1.6. Our algorithm consists of three steps.

Step 1: Color all the nodes of $\mathcal{H}$ by choosing for each node one out of two possible colors i.u.r.

Before we describe Step 2, we first introduce some notation and provide the ideas behind that step. An edge e is called *bad* if more than $(1 - 2\,\epsilon)\,|e|$ of its nodes have the same color after Step 1. Otherwise the edge is called *good*. That is, a good edge e has at least $2\,\epsilon\,|e|$ nodes of each color.

Clearly, after Step 1 there might be many monochromatic edges left in $\mathcal{H}$. In this case we have to recolor certain nodes in these edges. Similarly to [1, 6], we shall not only recolor the nodes in the monochromatic edges, but also those in the bad edges. The aim of Step 2.1 is to find a partition of the bad edges into node-disjoint groups (i.e., two bad edges from different groups should be node-disjoint). The key feature of our partitioning procedure is that we do not consider all the nodes covered by the bad edges. Instead, in the course of the algorithm we shall frequently *reduce* the edges, that is, we shall modify the edges by removing some subset of their nodes (see Figure 1).

Figure 1: *Reduction* of an edge e to edge e^*. Observe that if e^* is non-monochromatic, then so is e.

Step 2: Perform the following two substeps Step 2.1 and Step 2.2.

Step 2.1:

set $R = E$ // R denotes the set of remaining edges
repeat
  choose any bad edge e in R
  call **Build_1-Component(e)**
until there are no bad edges left in R

Algorithm Build_1-Component(e_0):

set $i = 0$ and $E_0 = \{e_0\}$
repeat
  set $E_{i+1} = \emptyset$
  for all edges $e \in E_i$ do:
    for all edges $e' \in N(e) \cap R$ do: // e' is a neighbor of e that has not yet been analyzed
      set $e'' = e' \cap (\bigcup_{j=0}^{i+1} \bigcup_{e^* \in E_j} e^*)$ // e'' is the set of nodes in e' already covered
      if $|e''| \geq \epsilon\,|e'|$ **then**
        remove e' from R, reduce it to e'', and add it to E_{i+1}
      else
        if e' is bad **then** remove it from R and add it to E_{i+1} $(*)$
  set $i = i + 1$
until $E_i = \emptyset$

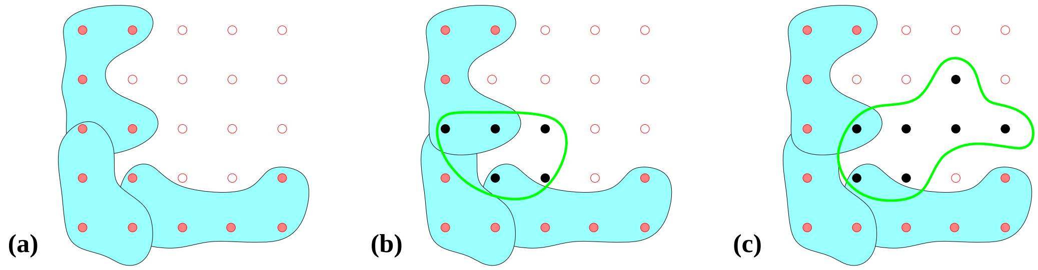

Figure 2: Illustration to Step 2.1. (a) An example of three edges defining set $\bigcup_{j=0}^{i+1} E_j$. (b) The case when a new edge e' intersects the edges in $\bigcup_{j=0}^{i+1} E_j$ in at least $\epsilon\,|e'|$ of its nodes. In this case e' is reduced to the three nodes in $\bigcup_{j=0}^{i+1} E_j$. In Step 3 of the algorithm we shall recolor these three nodes to ensure that e' will not be monochromatic. (c) The case when a new edge e' has a very small intersection with the edges in $\bigcup_{j=0}^{i+1} E_j$. In this case we do not reduce e'. If e' is good, then the nodes outside $\bigcup_{j=0}^{i+1} E_j$ will ensure that e' will remain monochromatic. However, if e' is bad, then we add it completely to the set E_{i+1}.

After performing Step 2.1, one could (even in polynomial time, as we will show) recolor the nodes within each group of bad edges independently of other groups such that all bad edges would become non-monochromatic. This, however, could make some of the good edges monochromatic. Therefore, in Step 2.2 we consider the good edges that are monochromatic after removing those of their nodes that belong to any group chosen in Step 2.1. The aim of Step 2.2 is to either *reduce* each such edge to the nodes covered by bad edges from one group chosen in Step 2.1, or to *combine* into one group all the groups covering the given edge. As we will see, the nodes covered by the groups constructed in Step 2.2 can be recolored independently so that now every edge is ensured to be non-monochromatic at the end.

Let the set $\bigcup_{i\geq 0} E_i$ built in the algorithm Build_1-Component be called a *1-component*. For each 1-component C, let $\bar{B}_C$ denote the set containing the edge e_0 in C and the edges that were added to C in $(*)$ because they were bad. These edges are called the *core edges* of C. They play an important role in the recoloring process in Step 3 because, as we shall show, our choice of ϵ ensures that it is sufficient to recolor only the nodes covered by the core edges in order to get a non-monochromatic coloring for all edges in C.

Define $\mathcal{C}$ to be the set of all 1-components. In the following, let $E^* = E \setminus (\bigcup_{C\in\mathcal{C}} C)$ denote the set of all edges not assigned to any of the 1-components. An edge e in E^* is called *dangerous* if at least $2\,\epsilon\,|e|$ of its nodes are covered by core edges[1]. The second substep works as follows.

Step 2.2:
set $R = \mathcal{C}$ // R denotes the set of remaining 1-components
repeat
  choose any 1-component C in R
  call **Build_2-Component(C)**
until R is empty

[1]This definition is oriented towards our analysis presented in the next section. For the purpose of the algorithm it would be enough to consider only those edges for which all nodes not covered by the core edges are of the same color.

Algorithm Build_2-Component(C):

set $i = 0$ and $E_0 = \{e_0\}$, where e_0 is any edge in B_C
repeat
  set $E_{i+1} = (N(E_i) \cap B_C) \setminus (\bigcup_{j=0}^{i} E_j)$ // initial 1,2-neighborhood of E_i
    for all edges $e' \in N(E_i) \cap E^*$ do: // e' has not yet been assigned to any 1-component or C
      set $e'' = e' \cap (\bigcup_{b \in B_C} b)$ // e'' is the set of nodes in e' already covered
      if $|e''| \geq \epsilon\, |e'|$
        then reduce e' to e'' and add it to C
        else
          if e' is dangerous **then** $(**)$
            for all 1-components C' in R that overlap with e' do:
              add to E_{i+1} all edges in $B_{C'}$ that intersect e'
              remove C' from R, add C' to C and $B_{C'}$ to B_C
            reduce e' to the nodes covered by $\bigcup_{b \in B_C} b$ and add it to C
      remove e' from E^*
  set $i = i + 1$
until $E_i = \emptyset$

Let the set C built in the algorithm Build_2-Component be called a *2-component*. For each 2-component C, let B_C denote the union of the sets $B_{C'}$ of the 1-components C' it consists of and let V_C be the set of nodes covered by the edges in B_C.

Remark 2.1 *Let us notice five important properties that we shall frequently use in our analysis and which follow immediately from our construction:*

(1) For every 2-component C, all edges in B_C are bad.
(2) For every 2-component C, it holds that $\bigcup_{e \in C} e = \bigcup_{e \in B_C} e$.
(3) For every edge e in the hypergraph H, if it is contained as a reduced edge e' in a 2-component C, then $|e'| \geq \epsilon\, |e|$.
(4) For every 2-component C, if $B_C = \{e_1, \ldots, e_\ell\}$, then there exists a partition of V_C into $W_1, \ldots, W_\ell$ such that for every i, $1 \leq i \leq \ell$, it holds that $W_i \subseteq e_i$ and $|W_i| \geq (1 - \epsilon)\, |e_i|$.
(5) If an edge e is bad (after completing Step 1), then there exists a 2-component which contains either e or the reduced edge of e.

Step 3: Find a coloring for each 2-component so that all of its edges are non-monochromatic.

In order to perform Step 3 efficiently, we use the following properties.

Lemma 2.2

(1) For every 2-component C, there is a coloring of the nodes in V_C such that all edges in C are non-monochromatic.
(2) All edges that are not assigned to any 2-component cannot become monochromatic by the recoloring process in Step 3.

Proof : We first prove (1). By Remark 2.1 (3), every edge in C has at least $\epsilon\, |e|$ of its nodes in V_C. Hence, the probability bound in Theorem 1.5 together with the LLL imply that there is a 2-coloring for the nodes in V_C so that every edge in C is non-monochromatic, no matter what the color of its nodes outside of V_C is.

In order to prove (2), let us consider any edge e that was not assigned to any 2-component. This means that it was neither bad nor dangerous. That is, it has at least $2\,\epsilon\,|e|$ nodes of each color and less than $2\,\epsilon\,|e|$ of its nodes are covered by core edges. Hence, the nodes of e not covered by the core edges are still non-monochromatic, and therefore e cannot become monochromatic by the recoloring process in Step 3. ∎

Lemma 2.2 and the fact that the node sets covered by distinct 2-components are disjoint imply the following vital property.

Corollary 2.3 *The 2-components can be recolored independently of each other in order to obtain a correct 2-coloring.*

We will prove in the next section (Theorem 3.1) that the number of nodes to be recolored in each 2-component is $O(\log m)$, w.h.p. Together with property (1) of Lemma 2.2, this allows us to find a proper coloring for each 2-component via exhaustive search in polynomial time. Hence, Corollary 2.3 implies that Step 3 can be performed in polynomial time. This results in a polynomial-time algorithm for the 2-coloring problem. To obtain a linear-time algorithm, we run in two phases. The first phase consists of Steps 1 and 2 above and the second phase is based on Steps 1 to 3 above, applied independently to each of the 2-components resulting from Phase 1. We shall describe this procedure in detail in Section 4. This will establish Theorem 1.5.

3 Analysis of the Algorithm

In this section we prove the following result.

Theorem 3.1 *If ϵ, δ, and k in Theorem 1.5 are set as $\epsilon = 1/24$, $\delta = \epsilon^2/12$, and $k = 1/\delta$, then with probability at least $1 - \frac{1}{m}$ the size of every 2-component is $O(\log m)$.*

In order to prove this theorem, we consider all possible structures of a certain kind that could witness a large 2-component. These structures are defined in Definition 3.3. For this we need the following definition.

Definition 3.2 *Given a hypergraph $\mathcal{H} = (V, E)$ and a set of edges $\mathcal{E} \subseteq E$, let*

- *the k-*neighborhood *of $\mathcal{E}$ be defined as*

$$N_k(\mathcal{E}) = \begin{cases} \mathcal{E} & : \quad k = 0 \\ N(\mathcal{E}) & : \quad k = 1 \\ N(N_{k-1}(\mathcal{E})) \setminus N_{k-2}(\mathcal{E}) & : \quad k > 1 \end{cases}$$

- *and the k,ℓ-neighborhood of $\mathcal{E}$ be defined as $N_{k,\ell}(\mathcal{E}) = \cup_{i=k}^{\ell} N_i(\mathcal{E})$.*

Definition 3.3 *Consider any hypergraph $\mathcal{H} = (V, E)$.*
A sequence $\mathcal{W} = \langle B_0, B_1, \ldots, B_d \rangle$ of edge sets $B_i \subseteq E$ is called a witness structure *of depth d if*

- *B_0 contains a single edge and*
- *all edges in B_{i+1} are in the 1,2-neighborhood of B_i.*

A witness structure is called a 2-component witness *if*

(1) for every edge $e \in \bigcup_{i=0}^{d} B_i$ a subset e' of its nodes of size at least $(1-\epsilon)\,|e|$ can be chosen such that all e' are disjoint and

(2) *for every $i \in \{0, \ldots, d-1\}$ the edges in B_{i+1} that are in the 2-neighborhood of B_i can be partitioned into sets $S_1, \ldots, S_r$ for some r, such that for every S_j there is a distinct edge $e_j \in N(B_i) \setminus B_{i+1}$ with $|e_j \cap (\bigcup_{e \in S_j} e')| \geq \epsilon\, |e_j|$ (note that e' is the part of e defined in (1)).*

Furthermore, a witness structure is called valid *if all edges in $\bigcup_{i=0}^{d} B_i$ are bad.*

Let $V_{\mathcal{W},i} = \bigcup_{e \in B_i} e$ denote the set of nodes covered by B_i and let $V_{\mathcal{W}} = \bigcup_{i=0}^{d} V_{\mathcal{W},i}$ denote the set of all nodes covered by $\mathcal{W}$.

We introduce these structures, because our aim will be to bound the expected number of valid 2-component witnesses of a certain size, rather than trying to bound the expected number of 2-components of a certain size constructed by the algorithm. This is important, since the latter method invokes dependencies that seem to be extremely difficult to handle, whereas the former method is purely combinatorial in that it is not based on a certain selection process. The next lemma enables us to switch to these witness structures.

Lemma 3.4 *For any 2-component C constructed by the algorithm, there is a valid 2-component witness that contains all edges in B_C.*

Proof : In order to construct a suitable witness structure for C, let us simply choose $B_i = E_i$ in the Build_2-Component algorithm for all $i \geq 0$. It is easy to check that this construction fulfills all requirements of a witness structure. Furthermore, by Remark 2.1, our algorithm ensures that disjoint node sets can be assigned to the edges in B_C so that every edge $e \in B_C$ is assigned to at least $(1-\epsilon)\,|e|$ of its nodes. Moreover, line $(**)$ in Step 2.2 ensures that Property (2) of Definition 3.3 is fulfilled. Hence, the witness structure constructed above is a 2-component witness. It is also valid since our algorithm requires all edges in B_C to be bad. ∎

Lemma 3.4 implies that if there is no valid 2-component witness over an edge set B then B cannot form a 2-component. The proof of Theorem 3.1 therefore follows directly from the following lemma.

Lemma 3.5 *There exists a positive constant γ such that*

$$\mathbf{E}\left[\left|\{\text{valid 2-component witnesses } \mathcal{W} : |V_{\mathcal{W}}| \geq \gamma \cdot \log m\}\right|\right] \leq \frac{1}{m} .$$

Before we prove the lemma, let us first introduce some notation which will be frequently used in the subsequent sections.

Definition 3.6 *Given any edge set $\mathcal{E}$, let $V_{\mathcal{E}}$ denote the set of all nodes covered by edges in $\mathcal{E}$. $\mathcal{E}$ is called a* core witness *if for each edge $e \in \mathcal{E}$ a subset e' of its nodes of size at least $(1-\epsilon)\,|e|$ can be chosen such that all e' are disjoint. Furthermore, $\mathcal{E}$ is called* valid *if all edges in $\mathcal{E}$ are bad. An edge set $\mathcal{F}$ is called a* 1,2-core witness *of $\mathcal{E}$ if $\mathcal{F}$ is a core witness and $\mathcal{F}$ is related to $\mathcal{E}$ as B_{i+1} is related to B_i in Definition 3.3 (2). Finally, let $\hat{v}_{\mathcal{E}}$ denote the maximum number of nodes a valid 1,2-core witness of $\mathcal{E}$ can cover.*

Lemma 3.5 now follows from the following three propositions.

Proposition 3.7 *There is a constant $c > 0$ so that for every edge e of size larger than $c\,\log m$ it holds*

$$\Pr[e \text{ is bad}] \leq \frac{1}{2m^2} .$$

In particular,

$$\Pr[\text{there is a valid 2-component witness } \mathcal{W} \text{ with } |V_{\mathcal{W},0}| \geq c\,\log m] \leq \frac{1}{2m} .$$

The proof of this proposition is obvious. Recall that, for each edge of a 2-component witness, we can select at least a $(1-\epsilon)$-fraction of its nodes that is disjoint from the node sets assigned to other edges. Therefore, we can assume that the probability of an edge to be bad is "independent" of other edges by considering only the subset of nodes chosen for it and assuming the worst possible case for the remaining nodes (cf. Claim 3.14). Furthermore, Property (2) of Definition 3.3 allows us to assume that also the probability of an edge to be dangerous is "independent" (in the above sense) of other edges. These are key properties which will enable us to obtain the following result.

Proposition 3.8 *There are constants $0 < \alpha < 1$ and $0 < \beta \leq 1/2$ so that for every core witness $\mathcal{E}$ we have $\mathbf{E}[\hat{\nu}_{\mathcal{E}}] \leq \beta |V_{\mathcal{E}}|$ and for every $\nu \geq |V_{\mathcal{E}}|/2$ it holds that $\Pr[\hat{\nu}_{\mathcal{E}} \geq \nu] \leq \alpha^{\nu}$.*

Proposition 3.8 will be proven in Section 3.4 after some preparations in Sections 3.1–3.3. The proof of the next proposition is given in Appendix A.

Proposition 3.9 *Let α and β be positive constants with $0 < \alpha < 1$ and $0 < \beta \leq \frac{1}{2}$. Let $X_0, X_1, \ldots,$ be any sequence of non-negative integer random variables satisfying the following four conditions:*

- $X_0 = \lambda$,
- $\mathbf{E}[X_{i+1} \mid X_0, X_1, \ldots, X_i] \leq \beta \cdot X_i$ *for every* $i \geq 0$,
- *if* $X_i = 0$ *then* $X_{i+1} = 0$, *and*
- $\Pr[X_{i+1} \geq t \mid X_0, X_1, \ldots, X_i] \leq \alpha^t$ *for every* $i \geq 0$ *and every* $t \geq \frac{1}{2} \cdot X_i$.

Then $\mathbf{E}\left[\sum_{j \geq 0} X_j\right] \leq 2 \cdot \lambda$, *and there is a positive constant c such that for every $s > 0$*

$$\Pr\left[\sum_{j \geq 0} X_j \geq c \cdot (\lambda + \log s)\right] \leq \frac{1}{s} .$$

Proof of Lemma 3.5 : From Proposition 3.7 it follows that

$$\mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W},0}| \geq c_1 \log m\}|] \leq \frac{1}{2m} ,$$

and from Propositions 3.8 and 3.9 we get that

$$\mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W}}| \geq c_2(|V_{\mathcal{W},0}| + \log m)\}|] \leq \frac{1}{2m} ,$$

for some positive constants c_1 and c_2. Hence, for $\gamma \geq c_2(c_1 + 1)$,

$$\begin{aligned}
&\mathbf{E}\left[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W}}| \geq \gamma \cdot \log m\}|\right] \\
&\quad = \mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W},0}| \geq c_1 \log m \text{ and } |V_{\mathcal{W}}| \geq \gamma \cdot \log m\}|] + \\
&\qquad\ \mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W},0}| < c_1 \log m \text{ and } |V_{\mathcal{W}}| \geq \gamma \cdot \log m\}|] \\
&\quad \leq \mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W},0}| \geq c_1 \log m\}|] + \\
&\qquad\ \mathbf{E}[|\{\text{valid 2-component witnesses } \mathcal{W}\text{: } |V_{\mathcal{W}}| \geq c_2(|V_{\mathcal{W},0}| + \log m)\}|] \\
&\quad \leq \frac{1}{2m} + \frac{1}{2m} = \frac{1}{m} .
\end{aligned}$$

This completes the proof of Lemma 3.5. ∎

3.1 A technical lemma

In this section we present a lemma that will be essential for estimating the expected number of valid 1,2-core witnesses that cover a given number of nodes.

Suppose that for every edge $e \in E$ we have an integer random variable Λ_e which we call its *contribution*. For any set of edges $\mathcal{F}$, let the *contribution* $\Lambda_{\mathcal{F}}$ of $\mathcal{F}$ be defined as the sum of the contributions of all edges in $\mathcal{F}$. Let

$$S_\lambda^{\mathcal{E}} = \{\mathcal{F} \subseteq N(\mathcal{E}) \ : \ \Lambda_{\mathcal{F}} = \lambda\} \ .$$

The next lemma provides an estimation for $\mathbf{E}[|S_\lambda^{\mathcal{E}}|]$.

Lemma 3.10 *Let c and γ be arbitrary positive constants with $\gamma \leq 1/288$. Furthermore, let $\mathcal{E}$ be any set of edges and $(\Lambda_e)_{e \in N(\mathcal{E})}$ be any sequence of contributions with the property that*

(1) the contribution of any edge in $N(\mathcal{E})$ is either 0 or at least $1/\gamma$,

(2) $|\{e \in N(\mathcal{E}) : \Pr[1 \leq \Lambda_e \leq k] > 0\}| \leq c \cdot 2^{\gamma \cdot k}$, and

(3) there is a $\phi \geq 1/6$ so that for every edge $e \in N(\mathcal{E})$, $\Pr[\Lambda_e = k] \leq 2^{-\phi \cdot k}$ independently of other events of positive contribution.

Then

$$\mathbf{E}[|S_\lambda^{\mathcal{E}}|] \leq 2^{-\phi\lambda/2} \cdot 2^{c/48} \ .$$

Proof : Let $m_\lambda^{\mathcal{E}}$ denote the number of all possible candidates for sets in $S_\lambda^{\mathcal{E}}$. We first bound $m_\lambda^{\mathcal{E}}$. For every set of edges $\{e_1, \ldots, e_r\} \in N(\mathcal{E})$, a sequence $\langle s_1, \ldots, s_\lambda\rangle$ is the *contribution-characterization* of $\{e_1, \ldots, e_r\}$ if $s_k = |\{1 \leq j \leq r : \Lambda_{e_j} = k\}|$. Since there are at most $c \cdot 2^{\gamma \cdot k}$ edges of contribution k in $N(\mathcal{E})$, there can be at most

$$\prod_{k=1}^{\lambda} \binom{c \cdot 2^{\gamma \cdot k}}{s_k}$$

sets $\mathcal{F} \subseteq N(\mathcal{E})$ that fulfill a prescribed contribution-characterization $\langle s_1, \ldots, s_\lambda\rangle$.

Let $\mathcal{P}(\lambda) = \{\langle s_1, \ldots, s_\lambda\rangle : s_j$ is a non-negative integer and $\sum_{j=1}^{\lambda} j \cdot s_j = \lambda\}$. Furthermore, let $\mathcal{P}(\lambda; \kappa) = \{\langle s_1, \ldots, s_\lambda\rangle \in \mathcal{P}(\lambda) : s_1 = \cdots s_{\kappa-1} = 0\}$. By our discussion above we obtain

$$\begin{aligned} m_\lambda^{\mathcal{E}} &\leq \sum_{\langle s_1,\ldots,s_\lambda\rangle \in \mathcal{P}(\lambda;1/\gamma)} \prod_{k=1}^{\lambda} \binom{c \cdot 2^{\gamma \cdot k}}{s_k} \\ &\leq \sum_{\langle s_1,\ldots,s_\lambda\rangle \in \mathcal{P}(\lambda;1/\gamma)} \prod_{k \geq 1/\gamma,\, s_k > 0} \left(\frac{\mathrm{e} \cdot 2^{\gamma \cdot k} \cdot c}{s_k}\right)^{s_k} \ . \end{aligned}$$

The bound given in the lemma for the probability that an edge has a contribution of k yields

$$\begin{aligned} \mathbf{E}[|S_\lambda^{\mathcal{E}}|] &\leq \sum_{\langle s_1,\ldots,s_\lambda\rangle \in \mathcal{P}(\lambda;1/\gamma)} \prod_{k \geq 1/\gamma,\, s_k > 0} \left(\frac{\mathrm{e} \cdot 2^{\gamma \cdot k} \cdot c}{s_k} \cdot 2^{-\phi \cdot k}\right)^{s_k} \\ &= 2^{-2\phi \cdot \lambda/3} \sum_{\langle s_1,\ldots,s_\lambda\rangle \in \mathcal{P}(\lambda;1/\gamma)} \prod_{k \geq 1/\gamma,\, s_k > 0} \left(\frac{\mathrm{e} \cdot 2^{\gamma \cdot k} \cdot c}{s_k \cdot 2^{\phi \cdot k/3}}\right)^{s_k} \ . \end{aligned}$$

It is easy to check that, for any $s, n > 0$, $(\frac{n}{s})^s$ is maximal for $s = n/2$. Hence, for any $\gamma \leq 1/72$,

$$\left(\frac{\mathrm{e}\, 2^{\gamma\cdot k}\cdot c}{s_k\cdot 2^{\phi\cdot k/3}}\right)^{s_k} \leq 2^{\frac{\mathrm{e}\, c}{2}\cdot\frac{2^{\gamma\cdot k}}{2^{\phi\cdot k/3}}} \leq 2^{\frac{\mathrm{e}\, c}{2}\cdot 2^{-\phi\cdot k/4}} .$$

Furthermore, for $\gamma \leq 1/288$ and $\phi \geq 1/6$ we have

$$\sum_{k\geq 1/\gamma} 2^{-\phi\cdot k/4} \leq 2^{-1/24\gamma}\sum_{k\geq 0} 2^{-k/24} \leq 2^{-12}\cdot 36 .$$

Hence, we get

$$\prod_{\substack{k\geq 1/\gamma:\\ s_k>0}} \left(\frac{\mathrm{e}\cdot 2^{\gamma\cdot k}\cdot c}{s_k}\right)^{s_k} \leq 2^{\frac{\mathrm{e}\, c}{2}\sum_{k\geq 1/\gamma} 2^{-\phi\cdot k/4}} \leq 2^{c/48} .$$

Thus, altogether we obtain

$$\mathbf{E}[S_\lambda^{\mathcal{E}}] \leq 2^{-2\phi\lambda/3} \sum_{\langle s_1,\dots,s_\lambda\rangle\in\mathcal{P}(\lambda;1/\gamma)} 2^{c/48} \leq 2^{-2\phi\lambda/3}\cdot|\mathcal{P}(\lambda;1/\gamma)|\cdot 2^{c/48} . \tag{1}$$

Obviously, for all $\lambda, \kappa \in \mathbb{N}$ it holds that

$$|\mathcal{P}(\lambda;\kappa)| \leq \begin{cases} 0 & : \quad \lambda < \kappa \\ 1 + \sum_{\lambda'=\kappa}^{\lambda-\kappa} |\mathcal{P}(\lambda';\kappa)| & : \quad \lambda \geq \kappa \end{cases} .$$

With this formula we can show the following claim.

Claim 3.11 *For all integers $\lambda, \kappa \geq 234$ it holds that $|\mathcal{P}(\lambda;\kappa)| \leq 2^{\lambda/36}$.*

Proof : We will prove the claim by induction on λ. Clearly, according to the formula for $|\mathcal{P}(\lambda;\kappa)|$ above, for all $\lambda \leq \kappa$ we have $|\mathcal{P}(\lambda;\kappa)| \leq 2^{\lambda/36}$. Now suppose that for some $\lambda > \kappa$ it has already been shown for all $\lambda' < \lambda$ that $|\mathcal{P}(\lambda';\kappa)| \leq 2^{\lambda'/36}$. Then it holds

$$\begin{aligned} |\mathcal{P}(\lambda;\kappa)| &\leq 1 + \sum_{\lambda'=\kappa}^{\lambda-\kappa} |\mathcal{P}(\lambda';\kappa)| \\ &\leq 1 + \sum_{\lambda'=\kappa}^{\lambda-\kappa} 2^{\lambda'/36} \\ &\leq 1 + 36\cdot 2^{1+(\lambda-\kappa)/36} = 1 + 36\cdot 2^{1-\kappa/36}\cdot 2^{\lambda/36} \\ &\leq 1 + 0.8\cdot 2^{\lambda/36} \leq 2^{\lambda/36} . \end{aligned}$$

∎

Thus, for our choice of γ and ϕ we have $|\mathcal{P}(\lambda;1/\gamma)| \leq 2^{\phi\lambda/6}$. Using this in inequality (1) yields the lemma. ∎

3.2 Counting valid 1-neighborhood sets

In this section we show how to apply Lemma 3.10 to bound the expected number of edge sets of certain size that form the 1-neighborhood of a valid 1,2-core witness. For this we first need some simple claims.

Claim 3.12 *For every edge $e \in E$ it holds*

$$\sum_{e' \in N(e)} 2^{-\delta \cdot |e'|} \leq \epsilon \cdot \ln 2 \cdot |e| \ .$$

Proof : Observe that from the assumption of Theorem 1.5 we obtain for every $e \in E$ that

$$\left(2^{1-|e|}\right)^{\epsilon} \leq 2^{-\delta \cdot |e|} \cdot \prod_{e' \in N(e)} \left(1 - 2^{-\delta \cdot |e'|}\right) \ .$$

This clearly implies that

$$2^{-\epsilon \cdot |e|} \leq \prod_{e' \in N(e)} \left(1 - 2^{-\delta \cdot |e'|}\right) \leq \prod_{e' \in N(e)} \exp\left(-2^{-\delta \cdot |e'|}\right) = \exp\left(-\sum_{e' \in N(e)} 2^{-\delta \cdot |e'|}\right) \ ,$$

which immediately yields the claim. ∎

Claim 3.13 *For every edge $e \in E$ and $k > 0$ it holds that*

$$|\{e' \in N(e) : |e'| \leq k\}| \leq 2^{\delta \cdot k} \cdot \epsilon \cdot \ln 2 \cdot |e| \ .$$

Proof : Follows directly from Claim 3.12 above. ∎

Claim 3.14 *Consider any $e \in E$. If we fix arbitrarily the colors of up to $\epsilon \cdot |e|$ nodes of e and then randomly color the other nodes, then it holds that*

$$\Pr[e \textit{ is bad}] \leq 2^{-|e|/2} \ .$$

Proof : Suppose that the colors of $s \leq \epsilon \cdot |e|$ nodes of e are determined. Then, clearly, the probability that e is bad is at most the probability that among the remaining $|e| - s$ nodes there are less than $2 \cdot \epsilon \cdot |e|$ nodes of one color. Therefore

$$\begin{aligned} \Pr[e \text{ is bad}] &\leq 2 \cdot \sum_{k=0}^{\lfloor 2\epsilon|e| \rfloor} \binom{|e|-s}{k} 2^{-(|e|-s)} \leq 4 \cdot 2^{-(|e|-s)} \cdot \binom{|e|-s}{2\epsilon|e|} \leq 4 \cdot 2^{-(|e|-s)} \cdot \left(\frac{\mathrm{e}}{2\epsilon}\right)^{2\epsilon|e|} \\ &\leq 4 \cdot 2^{-|e|(1-\epsilon)} \cdot \left(\frac{\mathrm{e}}{2\epsilon}\right)^{2\epsilon|e|} \ . \end{aligned}$$

For $\epsilon \leq 1/24$ this is at most $2^{-|e|/2}$ for any e with $|e| \geq 12/\epsilon^2$. ∎

Now we are ready to apply Lemma 3.10. Consider any core witness $\mathcal{E}$. Let the contribution Λ_e of an edge e be defined either as the size of e if e is bad, or 0 otherwise. Clearly, requirement (1) of Lemma 3.10 is fulfilled, since any edge in E must have a size of at least $1/\delta \geq 288$. Our assumption on $\mathcal{E}$ and Claim 3.13 together imply that

$$|\{e' \in N(\mathcal{E}) : |e'| \leq k\}| \leq 2^{\delta \cdot k} \cdot \epsilon \cdot \ln 2 \cdot \frac{|V_{\mathcal{E}}|}{1-\epsilon} \leq \epsilon |V_{\mathcal{E}}| \cdot 2^{\delta \cdot k} \ .$$

Thus, requirement (2) holds with $c = \epsilon|V_{\mathcal{E}}|$. Requirement (3) follows from Claim 3.14 and the definition of a core witness.

Since all requirements of Lemma 3.10 are fulfilled, the expected number of possibilities $C^{(1)}_{\mathcal{E},\lambda}$ of choosing edges of contribution λ for a valid 1,2-core witness of $\mathcal{E}$ that belong to $N(\mathcal{E})$ satisfies

$$C^{(1)}_{\mathcal{E},\lambda} \leq 2^{-\lambda/4} \cdot 2^{\epsilon|V_{\mathcal{E}}|/48} \ . \qquad (2)$$

3.3 Counting valid 2-neighborhood sets

In this section we show how to apply Lemma 3.10 to bound the expected number of edge sets of certain size that form the 2-neighborhood of a valid 1,2-core witness.

Let us fix some core witness $\mathcal{E}$. We say an edge $e \in N(\mathcal{E})$ is of *type* k if there is a valid core witness $\mathcal{F} \in N(e) \setminus \mathcal{E}$ with $\sum_{e' \in \mathcal{F}} |e'| = k$. (Note that e can be of different types at the same time if there are different $\mathcal{F}$ of different size.) Recall from Definition 3.3 that in order e to be of type $k > 0$ the number of nodes in e covered by $\mathcal{F}$ must be at least $\epsilon \cdot |e|$. Therefore, we must have $|e| \leq k/\epsilon$. With this we can show the following claims.

Claim 3.15 *For every edge $e \in E$ and positive k it holds*

$$|\{e' \in N(e) : e' \text{ can be of type in } \{1, \ldots, k\}| \leq 2^{k \cdot \delta/\epsilon} \cdot \epsilon \cdot \ln 2 \cdot |e| \ .$$

Proof : The proof of the claim follows directly from Claim 3.13. ∎

Claim 3.16 *For every $e \in E$ it holds that*

$$\Pr[e \text{ is of type } k] \leq 2^{-k/6} \ .$$

Proof : If an edge e is of type k there must be a core witness $\mathcal{F} \in N(e)$ with $\sum_{e' \in \mathcal{F}} |e'| = k$. Since the size of e can be at most k/ϵ in order to be of type k, we get from formula (1) in Section 3.2 that

$$\Pr[e \text{ is of type } k] \quad \leq \quad 2^{-k/4} \cdot 2^{\epsilon\,|e|/48} \leq 2^{-k/6} \ .$$

∎

For every edge $e_i \in N(\mathcal{E})$ let us introduce a set $\{e_{i,\epsilon\,|e_i|}, \ldots, e_{i,\lambda}\}$ of copies of e_i (λ will be specified below). We define the contribution $\Lambda_{e_{i,j}}$ of $e_{i,j}$ to be j if e_i is of type j and 0 otherwise. Now we apply Lemma 3.10 to the copies of all edges in $N(\mathcal{E})$. Clearly, requirement (1) of Lemma 3.10 holds, since every edge must have a type of either 0 or at least ϵ/δ. Requirement (2) follows from Claim 3.15 and requirement (3) follows from Claim 3.16 and the definition of a core witness. Thus, it follows from Lemma 3.10 that the expected number of possibilities $C^{(2)}_{\mathcal{E},\lambda}$ of choosing edges of contribution λ for a valid 1,2-core witness of $\mathcal{E}$ that belong to $N_2(\mathcal{E})$ satisfies

$$C^{(2)}_{\mathcal{E},\lambda} \leq 2^{-\lambda/12} \cdot 2^{\epsilon|V_{\mathcal{E}}|/48} \ . \tag{3}$$

3.4 Proof of Proposition 3.8

We are now ready to prove Proposition 3.8. Consider any fixed 1,2-core witness $\mathcal{E}$. Let the contribution of $\mathcal{E}$ be the sum of the edge sizes of its core edges. If $\mathcal{E}$ has contribution λ, then it must cover ν nodes, where $(1-\epsilon)\lambda \leq \nu \leq \lambda$ by Remark 2.1. Thus, we can apply the formulas (2) and (3) from Sections 3.2 and 3.3 to obtain the following bound for all $\nu \geq 6\epsilon\,|V_{\mathcal{E}}|$.

$$\begin{aligned}
\Pr[\hat{\nu}_{\mathcal{E}} \geq \nu] &\leq \mathbf{E}\left[|\{\mathcal{F} \;:\; \mathcal{F} \text{ is a 1,2-core witness of } \mathcal{E} \text{ and } |V_{\mathcal{F}}| \geq \nu\}|\right] \\
&\leq \sum_{\lambda \geq (1-\epsilon)\nu} \sum_{\kappa=0}^{\lambda} C^{(1)}_{\mathcal{E},\kappa} \cdot C^{(2)}_{\mathcal{E},\lambda-\kappa} \\
&\leq \sum_{\lambda \geq (1-\epsilon)\nu} \sum_{\kappa=0}^{\lambda} 2^{-\kappa/4} \cdot 2^{\epsilon|V_{\mathcal{E}}|/48} \cdot 2^{-(\lambda-\kappa)/12} \cdot 2^{\epsilon|V_{\mathcal{E}}|/48} \\
&\leq 2^{\epsilon|V_{\mathcal{E}}|/24} \sum_{\lambda \geq (1-\epsilon)\nu} 2^{-\lambda/12} \sum_{\kappa=0}^{\lambda} 2^{-\kappa/6} \\
&= 2^{\epsilon|V_{\mathcal{E}}|/24} \sum_{\lambda \geq (1-\epsilon)\nu} 2^{-\lambda/12} \cdot \frac{1}{1-2^{-1/6}} \\
&= 2^{\epsilon|V_{\mathcal{E}}|/24} \cdot 2^{-(1-\epsilon)\nu/12} \cdot \frac{1}{(1-2^{-1/12}) \cdot (1-2^{-1/6})} \\
&\leq 164 \cdot 2^{-(1-\epsilon)\nu/12} \cdot 2^{\epsilon|V_{\mathcal{E}}|/24} \leq 164 \cdot 2^{-(1-\epsilon)\nu/12} \cdot 2^{\nu/144} \leq 2^{-\nu/24} \;.
\end{aligned}$$

From this it follows that

$$\mathbf{E}[\hat{\nu}_{\mathcal{E}}] \leq 6\epsilon \cdot |V_{\mathcal{E}}| + \sum_{\nu > 6\epsilon \cdot |V_{\mathcal{E}}|} 2^{-\nu/24} \leq |V_{\mathcal{E}}|/2 \;.$$

This completes the proof of Proposition 3.8. ∎

4 A randomized linear-time algorithm for 2-coloring

In this section we show how to modify the algorithm presented in Section 2 to obtain a randomized algorithm that returns a 2-coloring of a hypergraph in expected linear time. In the following, let m denote the number of edges and let M denote the sum of the sizes of all edges in the input hypergraph. Our algorithm runs in two phases.

Phase 1

In the first phase we run Step 1 and Step 2 of the algorithm presented in Section 2 (however, the ϵ there has to be replaced by $\sqrt{\epsilon}$).

Clearly, the time required for Step 1 is $O(M)$. Step 2 requires a careful implementation to obtain a runtime of $O(M)$. This can be achieved for Step 2.1 by ensuring the properties that

- for every hyperedge it has to be checked at most once via its nodes whether it is bad (afterwards, this property can be retrieved from some variable assigned to that hyperedge), and
- for every node covered by a 1-component, the set of edges adjacent to it is evaluated at most once (the hyperedges can store the number of nodes they have in common with a 1-component by some counter; this counter suffices to check whether an ϵ-fraction of a hyperedge has already been covered by the current 1-component).

In Step 2.2 it is easy to see that every hyperedge has to be considered at most once. Whatever decision is done for the hyperedge, it is removed afterwards from the set of edges that still have to be checked. Hence, the overall runtime of Phase 1 is $O(M)$.

In Section 3 we investigated the structure of the 2-components obtained. It follows from Theorem 3.1 that with high probability there is no 2-component of size larger than $c \log m$ for a sufficiently large c. As we will show, it follows also from the analysis presented in Section 3 that the expected number of 2-components of size k is at most $\frac{m}{2^{\xi k}}$ for a suitably chosen constant ξ. Indeed, since each 2-component of size k has to start from a bad edge e of size less than or equal to k, the expected number of 2-components of size k is bounded by the following sum:

$$\sum_{e:\ |e| \leq k/(2c)} \Pr[\text{2-component } \mathcal{W} \text{ having } V_{\mathcal{W},0} = \{e\} \text{ is of size } k] + \sum_{e:\ |e| \in \{k/(2c)+1,\dots,k\}} \Pr[e \text{ is bad}] \ ,$$

where c is chosen as in Proposition 3.9. We can apply Propositions 3.8 and 3.9 to bound the first term from above by $|\{e \ : \ |e| \leq k/(2c)\}| \cdot 2^{-k/(2c)}$. By Claim 3.14, the second term is bounded by $|\{e \ : \ |e| > k/(2c)\}| \cdot 2^{-k/(4c)}$. Hence, the expected number of 2-components of size k is at most

$$\sum_{e:\ |e| \leq k/(2c)} 2^{-k/(2c)} + \sum_{e:\ |e| > k/(2c)} 2^{-k/(4c)} \leq m\, 2^{-k/(4c)} \ .$$

Phase 2

In the second phase we consider each 2-component $\mathcal{C}$ obtained after Phase 1 independently. For every $\mathcal{C}$ we run Steps 1 and 2 (again, with ϵ replaced by $\sqrt{\epsilon}$). Let $\mathcal{E}_{\mathcal{C}}$ be the event that all of the 2-components obtained out of $\mathcal{C}$ after performing Steps 1 and 2 are of size at most $\varsigma \log |V_{\mathcal{C}}|$, where ς is a suitably chosen constant and $V_{\mathcal{C}}$ is the set of the nodes covered by the edges in $\mathcal{C}$. We repeat independently Steps 1 and 2 for $\mathcal{C}$ until the event $\mathcal{E}_{\mathcal{C}}$ holds. Then, we perform Step 3 as described in Section 2.

In the following we estimate the running time of Phase 2. Since we are using the same algorithm as for the whole hypergraph, we can apply our analysis presented in Section 3 to study the 2-components obtained in Phase 2 out of the 2-component $\mathcal{C}$. We start with the first part in which, for every 2-component $\mathcal{C}$ obtained after Phase 1, we repeat Steps 1 and 2 until the event $\mathcal{E}_{\mathcal{C}}$ holds. Then, by Claim 3.13, there are at most $|V_{\mathcal{C}}| \cdot 2^{\delta' k}$ edges of (possibly reduced) size at most k in $\mathcal{C}$, where $\delta' = \delta/\sqrt{\epsilon}$. Since $\Pr[e \text{ is bad}] \leq 2^{-|e|/2}$, the probability that there is a bad edge e in $\mathcal{C}$ that has a size of larger than or equal to $c_1 \log |V_{\mathcal{C}}|$ is bounded from above by

$$\begin{aligned}\sum_{\substack{e \in \mathcal{C}:\\ |e| \geq c_1 \log |V_{\mathcal{C}}|}} 2^{-|e|/2} \quad &\leq \quad \sum_{k \geq c_1 \log |V_{\mathcal{C}}|} |V_{\mathcal{C}}| \cdot 2^{\delta' k} \cdot 2^{-k/2} \leq \sum_{k \geq c_1 \log |V_{\mathcal{C}}|} |V_{\mathcal{C}}| \cdot 2^{-k/3} \\ &\leq \quad 5 \cdot |V_{\mathcal{C}}| \cdot 2^{-c_1 \log |V_{\mathcal{C}}|/3} \ .\end{aligned}$$

Therefore, if we choose c_1 sufficiently large, then with probability at least $1 - 1/|V_{\mathcal{C}}|$ each $V_{\mathcal{C},0}$ obtained in Step 2 is of size not larger than $c_1 \log |V_{\mathcal{C}}|$. Similar arguments (by applying Propositions 3.8 and 3.9) imply that with probability at least $1 - 1/|V_{\mathcal{C}}|$ each 2-component within $\mathcal{C}$ will cover at most $\varsigma \log |V_{\mathcal{C}}|$ nodes, for a sufficiently large constant ς. Therefore, the expected running time in Steps 1 and 2 until the event $\mathcal{E}_{\mathcal{C}}$ holds is of the order of the sum of the sizes of the (possibly reduced) edges in $\mathcal{C}$. Summing this over all 2-components, the expected running time is at most $O(M)$.

Once the event $\mathcal{E}_{\mathcal{C}}$ holds, the running time of Step 3 for all the 2-components $\mathcal{C}_1, \dots, \mathcal{C}_r$ obtained from $\mathcal{C}$ in

Phase 2 is bounded by

$$\sum_{i=1}^{r} O\left(2^{|V_{\mathcal{C}_i}|} \cdot \sum_{e \in \mathcal{C}_i} |e|\right) = \sum_{i=1}^{r} O\left(2^{|V_{\mathcal{C}_i}|} \cdot \left(2^{|V_{\mathcal{C}_i}|} \cdot |V_{\mathcal{C}_i}|\right)\right)$$
$$= O\left(r \cdot \left(2^{\varsigma \log |V_{\mathcal{C}}|}\right)^2 \cdot |V_{\mathcal{C}}|\right) = O(|V_{\mathcal{C}}|^{2\varsigma+2}) ,$$

where $V_{\mathcal{C}_i}$ is the set of nodes in $\mathcal{C}_i$ at the beginning of Step 3 in Phase 2 and we have used the trivial upper bound $|\mathcal{C}_i| \le 2^{|V_{\mathcal{C}_i}|}$.

Since after Phase 1 the expected number of 2-components with k nodes is bounded by $\frac{m}{2^{\xi k}}$, the expected time required to perform Step 3 for all 2-components is bounded by

$$\sum_{\text{2-components } \mathcal{C} \text{ of Phase 1}} O(|V_{\mathcal{C}}|^{2\varsigma+2}) = O\left(\sum_{k=1}^{c \log M} k^{2\varsigma+2} \cdot \frac{m}{2^{\xi k}}\right) = O(m) .$$

Hence, the total expected running time of our algorithm is $O(M)$.

5 Extensions to c-coloring

In this section we sketch how to extend our results to c-coloring. First, we observe that the algorithms presented in the previous sections can be easily extended to deal with any c-coloring. A simple modification of the proof of Theorem 1.5 yields the following theorem.

Theorem 5.1 *Let $c \ge 2$ be an arbitrary constant. Then there exist constants K, Δ, $\mathcal{E} > 0$ such that for any $k \ge K$, $0 < \delta \le \Delta$, and $0 < \epsilon \le \mathcal{E}$ it holds:*

Consider any hypergraph $\mathcal{H}$ with edges $e_1, \ldots, e_m$, in which every edge is of size at least k. Let A_i be the event that e_i is monochromatic, $1 \le i \le m$. Further, let $x_i = c^{-\delta |e_i|}$ for all i, $1 \le i \le m$. If it holds that

$$(\Pr[A_i])^{\epsilon} \le x_i \prod_{e_j \in N(e_i)} (1 - x_j)$$

for all i, $1 \le i \le m$, in the case that the color of each node is chosen i.u.r., then there is a randomized algorithm that finds a c-coloring of $\mathcal{H}$ in expected time linear in $\sum_i |e_i|$.

One of the restrictions in Theorems 1.5 and 5.1 is that the size of the hyperedges must exceed some constant. In the following theorem we demonstrate that this condition can already be avoided if 4 colors are available.

Theorem 5.2 *There exist constants $\Delta', \mathcal{E}' > 0$ so that for any $0 < \delta \le \Delta'$ and $0 < \epsilon \le \mathcal{E}'$ it holds:*

Consider any hypergraph $\mathcal{H}$ with edges $e_1, \ldots, e_m$ of any size. Let A_i be the event that e_i is monochromatic. Further, let $x_i = 4^{-\delta |e_i|}$. If it holds that

$$(\Pr[A_i])^{\epsilon} \le x_i \prod_{e_j \in N(e_i)} (1 - x_j) \tag{4}$$

for all $i \in \{1, \ldots, m\}$, in the case that the color of each node is chosen i.u.r., then there is a randomized algorithm that finds a 4-coloring of $\mathcal{H}$ in polynomial time.

Proof : Let us choose $\mathcal{E}' = \mathcal{E}/6$ and $\Delta' = \Delta/2$, where $\mathcal{E} = 1/24$ and $\Delta = \mathcal{E}^2/12$. From Theorem 1.5 and Remark 1.6 we know that in this case there is a 2-coloring for all hyperedges in $\mathcal{H}$ of size at least $1/\Delta$. Thus, there are 2 colors left we can use to color the hyperedges of size below $1/\Delta$. All possible combinations of the 2 sets of 2 colors require only 4 colors, and hence a 4-coloring of $\mathcal{H}$ could be found if the small hyperedges could be 2-colored.

For any edge e_j of the small edges it clearly holds that $x_j = 4^{-\delta\,|e_j|} \geq 2^{-\Delta\,|e_j|} \geq 1/2$. Hence, $1 - x_j \leq 1/2$ and therefore $\Pr[A_i]^\epsilon \leq x_i \prod_{e_j \in N(e_i)}(1 - x_j)$ for $|N(e_i)| > 0$ only if $4^{-\epsilon|e_i|} \leq 1/2$. Thus, any e_i with $|N(e_i)| > 0$ has to have a size of at least $3/\mathcal{E}$. Hence, all e_i with $|e_i| < 3/\mathcal{E}$ are isolated from all other small edges and thus can be colored without any problem. It therefore remains to consider only hyperedges of size $k \in \{3/\mathcal{E}, \ldots, 1/\Delta\}$. For each of these edges the probability to become monochromatic is less than $2 \cdot 2^{-k}$, and any of these edges can intersect at most $(\mathcal{E}/3) \cdot k$ other edges without violating inequality (4). This suffices to use one of the algorithms for 2-coloring uniform hypergraphs to find a 2-coloring for the small hyperedges.■

We remark also that the higher the value of c in Theorem 5.2 is, the better are the values that can be found for Δ' and $\mathcal{E}'$ (for instance, the property that small hyperedges are isolated from other small hyperedges can be avoided if c is sufficiently large).

6 Conclusions

We presented a powerful method which, as we believe, will allow to provide polynomial-time algorithms for many applications of the general Lovász Local Lemma.

There are still many open problems left. For instance, what kind of properties do applications of the general Lovász Local Lemma have to fulfill to be able to construct polynomial-time algorithms for them? What class of applications can be covered by (generalizations of) our technique? What is the largest ϵ for which our algorithm still runs in polynomial time? $\epsilon = 1/6$ might already work to obtain a polynomial-time algorithm for the 2-coloring problem. Is there a polynomial-time algorithm that finds a 2-coloring if there is no restriction on the minimum size of the hyperedges?

References

[1] N. Alon. A parallel algorithmic version of the Local Lemma. *Random Structures and Algorithms*, 2(4):367–378, 1991. A preliminary version appeared in *Proceedings of the 32nd IEEE Symposium on Foundations of Computer Science*, pages 586–593, San Juan, Puerto Rico, October 1–4, 1991. IEEE Computer Society Press, Los Alamitos, CA.

[2] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg. On the complexity of radio communication. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 274-285, Seattle, WA, May 15–17, 1989. ACM Press, New York, NY.

[3] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992. An addendum appeared in *Random Structures and Algorithms*, 4, 1993. A preliminary version appeared in *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pages 544–553, St. Louis, MO, October 22–24, 1990. IEEE Computer Society Press, Los Alamitos, CA.

[4] N. Alon, C. McDiarmid, and B. Reed. Acyclic coloring of graphs. *Random Structures and Algorithms*, 2(3):277–288, 1991.

[5] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, New York, 1992.

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

[7] A. Z. Broder, A. M. Frieze, and E. Upfal. Static and dynamic path selection on expander graphs: A random walk approach. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 531–539, El Paso, TX, May 4–6, 1997. ACM Press, New York, NY.

[8] P. Erdős and L. Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In A. Hajnal, R. Rado, and V. T. Sós, editors, *Infinite and Finite Sets (to Paul Erdős on his 60th birthday)*, volume II, pages 609–627. North-Holland, Amsterdam, 1975. Colloquia Mathematica Societatis János Bolyai, 10. Infinite and Finite Sets, Keszthely, Hungary, 1973.

[9] P. Erdős and J. Spencer. Lopsided Lovász local lemma and latin transversals. *Discrete Applied Mathematics*, 30:151–154, 1991.

[10] G. Even, O. Goldreich, M. Luby, N. Nisan, and B. Veličković. Efficient approximations of product distributions. *Random Structures and Algorithms*, 13(1):1–16, 1998. A preliminary version appeared in *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 10–16, Victoria, British Columbia, Canada, May 4–6, 1992. ACM Press, New York, NY.

[11] U. Feige and C. Scheideler. Improved bounds for acyclic job shop scheduling. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 624–633, Dallas, TX, May 23–26, 1998. ACM Press, New York, NY.

[12] H. Hind, M. Molloy, and B. Reed. Colouring a graph frugally. *Combinatorica*, 17(4):469–482, 1997.

[13] F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet routing and job-shop scheduling in O(Congestion + Dilation) steps. *Combinatorica*, 14(2):167–180, 1994. A preliminary version entitled "Universal packet routing algorithms" appeared in *Proceedings of the 29th IEEE Symposium on Foundations of Computer Science*, pages 256–271, White Plains, NY, October 24-26, 1988. IEEE Computer Society Press, Los Alamitos, CA.

[14] F. T. Leighton, B. M. Maggs, and A. W. Richa. Fast algorithms for finding O(Congestion + Dilation) packet routing schedules. *Combinatorica*, to appear, 1998. A preliminary version appeared as Technical Report CMU–CS–96–152, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, July 1996.

[15] T. Leighton, S. Rao, and A. Srinivasan. New algorithmic aspects of the Local Lemma with applications to routing and partitioning. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 643–652, Baltimore, MD, January 17–19, 1999. SIAM, Philadelphia, PA.

[16] M. Molloy. The probabilistic method. In M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 1–35. Springer-Verlag, Berlin, 1998.

[17] M. Molloy and B. Reed. Further algorithmic aspects of the Local Lemma. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 524–529, Dallas, TX, May 23–26, 1998. ACM Press, New York, NY.

[18] J. Naor and J. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993. A preliminary version appeared in *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 213–223, Baltimore, MD, May 14–16, 1990. ACM Press, New York, NY.

[19] J. Radhakrishnan and A. Srinivasan. Improved bounds and algorithms for hypergraph two-coloring. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 684–693, Palo Alto, CA, November 8–11, 1998. IEEE Computer Society Press, Los Alamitos, CA.

[20] B. Reed. ω, Δ, and χ. *Journal of Graph Theory*, 27(4):177–212, 1998.

[21] J. Spencer. *Ten Lectures on the Probabilistic Method*. 2nd Edition. SIAM, Philadelphia, 1994.

[22] A. Srinivasan. An extension of the Lovász Local Lemma, and its applications to integer programming. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 6–15, Atlanta, GA, January 28–30, 1996. SIAM, Philadelphia, PA.

Appendix

A Proof of Proposition 3.9

Proof of Proposition 3.9 : To show the first claim of Proposition 3.9, let us observe that since $\mathbf{E}[X_{i+1} \mid X_0, X_1, \ldots, X_i] \le \beta \cdot X_i$, we immediately obtain $\mathbf{E}[X_t] \le \beta^t \cdot \lambda$. This implies that

$$\mathbf{E}[\sum_{j \ge 0} X_j] \le \sum_{j \ge 0} \beta^j \cdot \lambda = \tfrac{1}{1-\beta} \cdot \lambda \le 2 \cdot \lambda$$

since $\beta \le 1/2$. Now we prove the second claim. Since $\mathbf{E}[X_t] \le \beta^t \cdot \lambda$, it follows from the Markov Inequality that

$$\Pr[X_\tau > 0] \le \beta^\tau \cdot \lambda \le \frac{1}{2\,s} \tag{5}$$

for $\tau \ge \log_{1/\beta}(2\,s\,\lambda)$. Let us now fix $\tau = \max[\log_{1/\beta}(2\,s\,\lambda), \log(2s)]$ and condition on the event $X_\tau = 0$.

For a given sequence $X_0, \ldots, X_\tau$, we say X_i is *ascending* if $i \ge 1$ and $X_i > \frac{1}{2} \cdot X_{i-1}$. Observe that if $X_j, X_{j+1}, \ldots, X_s > 0$ are not ascending, then they decrease at least geometrically. Therefore, if $X_{i_1}, X_{i_2}, \ldots, X_{i_r}$ are all ascending random variables in $X_0, \ldots, X_\tau$, then we have

$$\sum_{j \ge 0} X_j \le 2 \cdot \lambda + \sum_{j=1}^{r} 2 \cdot X_{i_j} \ . \tag{6}$$

Thus from now on our aim is to bound $\sum_{j=1}^{r} X_{i_j}$.

Let $Y_1, \ldots, Y_\tau$ be independent, geometrically distributed random variables with $\Pr[Y_j = t] = \alpha^{t-1}(1-\alpha)$ for every $t \in \mathbb{N}$. It is easy to see that $\Pr[Y_j \ge t] = \alpha^{t-1}$ for all $t \in \mathbb{N}$. Since, by assumption, $\Pr[X_j \ge t] \le \alpha^t$ for all $t \ge \frac{1}{2} X_{j-1}$, $(Y_j - 1)$ stochastically dominates any ascending X_j. Thus, $\sum_{j \ge 0} X_j$ is stochastically dominated [2] by $2 \cdot (\lambda + \sum_{j=1}^{\tau} (Y_j - 1))$. Hence, it remains to bound $\sum_{j=1}^{\tau} Y_j$. Since the Y_j are geometrically distributed, we have $\mathbf{E}[Y_j] = 1/(1-\alpha)$ for all j and therefore $\mathbf{E}[Y] = \tau/(1-\alpha)$. It remains to prove a probability bound that shows that w.h.p. Y is not too far away from $\mathbf{E}[Y]$. For this we will use factorial moments. For any $r \in \mathbb{N}$, we call

$$\mathbf{E}[Y^{[r]}] = \mathbf{E}[Y \cdot (Y+1) \cdot (Y+r-1)]$$

the rth ascending factorial moment of Y.

Lemma A.1 *Let Y be the sum of n identical geometric random variables with parameter p (that is, $\Pr[Y_i = k] = (1-p)^{k-1} p$ for any $k \in \mathbb{N}$ and $i \in \{1, \ldots, n\}$). Then it holds for any $k \in \mathbb{N}$ that*

$$\Pr[Y \ge k] \le \frac{n^{[n]}}{p^n \cdot k^{[n]}} \ .$$

Proof : It follows easily from the Markov Inequality that for any $k \in \mathbb{N}$ we have

$$\Pr[Y \ge k] = \Pr[Y^{[n]} \ge k^{[n]}] \le \frac{\mathbf{E}[Y^{[n]}]}{k^{[n]}} \ . \tag{7}$$

[2] Recall that a real valued random variable A is *stochastically dominated* by a real valued random variable B if for every $x \in \mathbb{R}$ holds $\Pr[A \ge x] \le \Pr[B \ge x]$.

Since Y can be interpreted as a negative binomial random variable with parameters n and p, we obtained with the help of Maple that

$$\mathbf{E}[Y^{[r]}] = \frac{n^{[r]}}{p^r} \tag{8}$$

for any $r \in \mathbb{N}$. Combining (7) and (8) yields the lemma. ∎

Thus, it follows from Lemma A.1 that for any $\delta \geq 3$,

$$\begin{aligned} \Pr[Y \geq (1+\delta)\mathbf{E}[Y]] &\leq \frac{\tau^{[\tau]}}{(1-\alpha)^\tau \cdot [(1+\delta)\tau/(1-\alpha)]^{[\tau]}} \leq \frac{(2\tau)^\tau}{(1-\alpha)^\tau \cdot [(1+\delta)\tau/(1-\alpha)]^\tau} \\ &\leq 2^{-\tau} \; . \end{aligned}$$

Thus,

$$\Pr[Y \geq 4\tau/(1-\alpha)] \leq 2^{-\tau} \leq \frac{1}{2\,s}$$

for τ chosen as above. Therefore, we have

$$\Pr\left[\sum_{j\geq 0} X_j \geq 2(\lambda + (4/(1-\alpha)-1)\tau) \mid X_\tau = 0\right] \leq \frac{1}{2\,s} \; .$$

Together with (5) this implies

$$\Pr\left[\sum_{j\geq 0} X_j \geq 2\left(\lambda + (\tfrac{4}{1-\alpha}-1)\tau\right)\right] \leq \frac{1}{s} \; .$$

∎