HUMBOLDT-UNIVERSITÄT ZU BERLIN

Mathematisch-Naturwissenschaftliche Fakultät II

Institut für Informatik
Lehrstuhl für Algorithmen und Komplexität

Coloring sparse random k-colorable graphs in polynomial expected time

Diplomarbeit

eingereicht von: Julia Böttcher

Gutachter: Dr. Amin Coja-Oghlan
Dr. Mihyun Kang

Berlin, den 5. Januar 2005

Erklärung

Hiermit erkläre ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet zu haben. Ich bin damit einverstanden, dass die vorliegende Arbeit in der Zentralbibliothek Naturwissenschaften der Humboldt-Universität zu Berlin öffentlich ausgelegt wird.

Berlin, den 5. Januar 2005

Julia Böttcher

Zusammenfassung

Nach wie vor stellt das Graphenfärbungsproblem eine der anspruchsvollsten algorithmischen Aufgaben innerhalb der Graphentheorie dar. So blieb bisher nicht nur die Suche nach effizienten Methoden, die dieses Problem exakt lösen, erfolglos. Gemäß eines Resultates von Feige und Kilian [22] ist auch die Existenz von signifikant besseren Approximationsalgorithmen als dem Trivialen, der jedem Knoten eine eigene Farbe zuordnet, sehr unwahrscheinlich.

Diese Tatsache motiviert die Suche nach Algorithmen, die das Problem entweder nicht in allen sondern nur den meisten Fällen entscheiden, dafür aber von polynomieller Laufzeitkomplexität sind, oder aber stets eine korrekte Lösung ausgeben, eine polynomielle Laufzeit jedoch nur im Durchschnitt garantieren.

Ein Algorithmus der ersterem Paradigma folgt, wurde von Alon und Kahale [2] für die folgende Klasse zufälliger k-färbbarer Graphen entwickelt: Konstruiere einen Graphen $\mathcal{G}_{n,p,k}$ mit einer Knotenmenge V der Kardinalität n durch Partitionierung von V in k Mengen gleicher Größe und anschließendes Hinzufügen der möglichen Kanten zwischen diesen Mengen mit Wahrscheinlichkeit jeweils p. Das Ergebnis von Alon und Kahale besagt nun, dass Graphen aus $\mathcal{G}_{n,p,k}$ mit hoher Wahrscheinlichkeit in polynomieller Zeit k-gefärbt werden können, falls $p \geq c/n$ für eine hinreichend große Konstante c ist. In dieser Arbeit wird für die gleichen Klasse von Graphen und den selben Bereich von p ein Algorithmus der zweiten Art für den Fall $k = 3$ vorgestellt. Genauer gesagt wird nachgewiesen, dass es möglich ist, $\mathcal{G}_{n,p,3}$ in polynomiell erwarteter Laufzeit mit drei Farben zu färben, falls $p \geq c/n$.

Zur Konstruktion dieses Algorithmuses werden die Ideen von Alon und Kahale mit Techniken aus dem Gebiet der semidefiniten Programmierung kombiniert und außerdem Mechanismen zur Behebung von Unzulänglichkeiten der von Alon und Kahale verwendeten Methode bei untypischen Graphen entwickelt. Die Berechnungen lassen sich auf beliebiges k verallgemeinern.

Abstract

The coloring problem on graphs remains one of the most demanding algorithmic tasks in graph theory. It is not only hard to efficiently find exact solutions to this problem, but it is also extremely unlikely, due to Feige and Kilian [22], that there exist approximation algorithms which substantially improve over the trivial one, i.e., color each vertex with a separate color.

This motivates the quest for algorithms that either solve the problem in most but not all cases, but are of polynomial time complexity, or that give a correct solution on all input graphs while guaranteeing a polynomial running time on average only.

An algorithm of the first kind was suggested by Alon and Kahale in [2] for the following type of random k-colorable graphs: A graph $\mathcal{G}_{n,p,k}$ on a vertex set V of cardinality n is constructed by first partitioning V into k equally sized sets and then adding each edge between these sets with probability p independently from each other. Now, the result of Alon and Kahale shows that graphs from $\mathcal{G}_{n,p,k}$ can be k-colored in polynomial time with high probability as long as $p \geq c/n$ for some sufficiently large constant c. In this thesis, we construct an algorithm of the second kind for $k = 3$ on the same type of graphs and for the same range of p. More precisely, we prove that it is possible to 3-color $\mathcal{G}_{n,p,3}$ in polynomial expected running time for $p \geq c/n$.

To obtain this result we combine the ideas developed by Alon and Kahale with techniques from semidefinite programming and develop error recovery mechanisms for atypical graphs. The calculations carry over to general k.

Contents

1 Introduction

“ There are only 3 colors, 10 digits, and 7 notes; it’s what we do with them that’s important. ”

Ruth Ross

The problem of computing the chromatic number of a graph has drawn much attention in graph theory. The objective of this problem is to find the minimal number of colors such that one color can be assigned to each vertex of a given graph without producing unicolored neighbors (for formal definitions see Chapter 2). While early interest in the subject was motivated by purely mathematical questions such as the famous 4-coloring problem, the development of the computer in the twentieth century, and with it the growing importance of algorithms, supplied practical applications of the problem. These applications include different kinds of scheduling problems [47, 68] as well as code optimization [9] and hardware design [60].

In compiler optimization for example, the goal is to schedule the usage of a minimal number of registers. The variables occuring in a given code fragment can be represented as the vertices of a graph. An edge is drawn between each pair of variables whose life span intersects. The problem of coloring the resulting graph corresponds to the original problem of assigning the variables to registers (cf. [61]).

In light of these applications, it would be desirable to have fast algorithms for solving the graph coloring problem, i.e. algorithms that exhibit a polynomial running time complexity. Unfortunately however, coloring is one of the classical $\mathcal{NP}$-hard problems (see [26]). As a consequence, it is rather unlikely that efficient algorithms for this problem exist.

If no exact answer to a problem can be found within a reasonable amount of time, one alternative is to search for fast algorithms that need not necessarily produce correct solutions but return at least sufficiently good approximations to the truth. However, for the coloring problem even this suboptimal approach fails. While the standard greedy heuristic, i.e., subsequently coloring each vertex with the least color that was not assigned to any of its neighbors, does not use more than twice of the minimum number of colors on “most” graphs (see Section 3.2 for details), guaranteeing a similar performance for all graphs proved infeasible under reasonable computational assumptions. In fact, Feige and Kilian [22] proved that for all $\epsilon > 0$ it is impossible to find an efficient algorithm that always uses at most $n^{1-\epsilon}$ times the optimum number of colors provided $\mathcal{ZPP} \neq \mathcal{NP}$, where n is the number of vertices of the input graph. Moreover, Khanna, Linial, and Safra [40] showed that coloring 3-colorable graphs (i.e., graphs that can be colored with 3 colors) with 4 colors is $\mathcal{NP}$-hard.

Accordingly, different approaches need to be pursued. One possibility is to ask for algorithms that manage to deal with the majority of graphs, failing only in a few exceptional cases. While finding algorithms of this type is not too dificult for dense k-colorable graphs (cf. [18, 44, 64]; for further explanations, see also Section 3.2), it turns out harder for sparse k-colorable graphs. For constructing such sparse graphs consider the following process: Partition the vertices into

k sets of equal size and allow only edges between these sets, taking each one independently with probability p. We denote graphs obtained in this way by $\mathcal{G}_{n,p,k}$. The number of neighbors for a vertex of $\mathcal{G}_{n,p,k}$ is $np \cdot (k-1)/k$ on average. Accordingly, we obtain graphs with relatively few edges, i.e., graphs that are sparse, by setting $p = c/n$ for constant c.

In 1997, Alon and Kahale [2] were able to prove that the k-coloring problem can efficiently be solved for "most" of these graphs. Here, by "most" we mean that the probability that $\mathcal{G}_{n,p,k}$ is colored correctly goes to 1 as n tends to infinity. Alternatively, we say that a valid coloring is constructed with high probability.

Theorem 1 (Alon & Kahale [2])
Let $p > c/n$ for some sufficiently large constant c. Then there is an efficient algorithm for k-coloring $\mathcal{G}_{n,p,k}$ with high probability.

But an algorithm that works with high probability has one drawback: For some inputs it does not provide any solution at all. Alternatively, we could require that the algorithm always gives a correct answer to the problem under study. By the remarks above, we need to compensate for this accuracy with a concession in running time. This motivates the search for algorithms that perform well on average (cf. [39]): An algorithm $\mathcal{A}$ with running time $t_{\mathcal{A}}(G)$ on input G has polynomial expected running time on $\mathcal{G}_{n,p,k}$ if $\sum_G t_{\mathcal{A}}(G) \cdot \mathbf{P}[\,\mathcal{G}_{n,p,k} = G\,]$ remains polynomial. Here, the sum ranges over all graphs on n vertices. Observe that this is a stronger condition than the requirement to work correctly with high probability: An algorithm that k-colors $\mathcal{G}_{n,p,k}$ in polynomial expected running time also solves the k-coloring problem with high probability in polynomial time.

For dense graphs, algorithms that color $\mathcal{G}_{n,p,k}$ in polynomial expected running time were given by Subramanian [62] and Coja-Oghlan [14]. In this thesis we modify the algorithm of Alon and Kahale for obtaining a corresponding result for sparse graphs.

Theorem 2
If $p > c/n$ for some sufficiently large constant c then $\mathcal{G}_{n,p,3}$ can be 3-colored in polynomial expected running time.

The calculations carry over to general k. This answers a question of Subramanian [62].

The main philosophy of the algorithm we will develop can be described as follows: In the beginning, a modified version $\mathcal{A}$ of the algorithm of Alon and Kahale is executed on the input graph G. Then, if no valid coloring is obtained in this way, the graph is split into two parts, one of them containing the vast majority of G and the other one all remaining vertices. Thereafter, $\mathcal{A}$ is rerun on the bigger part while the other part is treated with brute force coloring methods. If G is still not colored correctly then this process is repeated, more vertices handled by $\mathcal{A}$ are shifted to those taken care of by the brute force method and so on. This is continued until G is finally properly colored.

By assuring that on most graphs from $\mathcal{G}_{n,p,3}$ only a small number of vertices cannot be handled by $\mathcal{A}$, we verify that this procedure exhibits a running time that is polynomial in its expectation.

In addition, and in contrast to Alon and Kahale, we apply the concept of semidefinite programming for constructing a coloring of $\mathcal{G}_{n,p,3}$. Semidefinite programming is a branch of combinatorial optimization that was initiated in 1979 when Lovász [49] established a new bound, now called the Lovász number, on the independence number of graphs (i.e., the maximal cardinality of a set of vertices having no edges among each other). To this end, he characterized the

Lovász number in the form of a linear minimization problem over a set of positive semidefinite matrices with linear constraints. This seminal idea led to improved approximative algorithms for a number of graph problems and by now, semidefinite programming can be regarded as a standard tool for optimization problems on graphs.

In this work we use semidefinite programming techniques in order to obtain an initial coloring of $\mathcal{G}_{n,p,3}$ that already colors all but a small linear fraction of the input graph correctly (this method is based on ideas of Coja-Oghlan [14]). This approximative coloring is then refined in several iterations, leading to a valid coloring as desired.

The remainder of this thesis is structured as follows: In Chapter 2 we introduce some notation and definitions, and explain basic concepts which will be used later. Next, an outline of important results in the two fields forming the background of this work, namely algorithmic graph coloring and the theory of semidefinite programs, is presented in Chapters 3 and 4, respectively. Chapter 5 contains the main part of this thesis, i.e., the polynomial expected time coloring algorithm along with its analysis. Finally, in Chaper 6 we give some retrospective and prospective remarks.

2 Notation and Definitions

"A definition is the enclosing of a wilderness of idea within a wall of words."

Samuel Butler

In this chapter we provide most terminology used in this thesis and fix notational conventions. Further definitions are sporadically presented later, in special cases where more explanations are necessary to understand the ideas involved. Most of the terminology is standard for the field. What follows does therefore not contain a careful introduction to the underlying concepts. It may rather be regarded as a reference. We start with some general remarks, then turn to notions from graph theory and finally introduce semidefinite programming.

In the following we write $A := B$ ($A =: B$) for arbitrary terms A and B in order to express that the left (right) hand side of this equation is defined to equal the right (left) hand side.

Further, let S and $T = \{t_1, \ldots, t_k\}$ be two sets. We often also write $S - T$ or $S - t_1, \ldots, t_k$ for $S \setminus T$. Analogously $S + T$ and $S + t_1, \ldots, t_k$ are used for $S \cup T$. In addition, we designate the complement $S - S'$ of a subset S' of S also by $\overline{S'}$ if S is clear from the context.

In this text, the natural logarithm is denoted by ln, whereas log is used for the logarithm to basis 2.

2.1 Graphs

An *undirected finite graph* or simply *graph* $G = (V, E)$ consists of a finite set of *vertices*, or *nodes*, V and a symmetric binary relation $E \subseteq \binom{V}{2}$ called its *edge set*. Here, an edge between two vertices $u, v \in V$ is denoted by uv. From this definition it follows that we exclusively consider *simple* graphs, i.e., graphs where E does not contain multiple elements and is irreflexive. Where no confusion arises, we also refer to the vertex set V by just writing G and vice versa. In any case $|G|$ denotes the *order* $n := |V|$ of G and $m := |E|$ is also called its *size*. When dealing with different graphs, we set $V(G) := V$ and $E(G) := E$ to distinguish the vertex and edge sets involved.

An *empty graph* is a graph with an empty edge set and $V \neq \emptyset$. $G' = (V', E')$ is a *subgraph* of G, denoted by $G' \preceq G$, if $V' \subseteq V$ and $E' \subseteq E \cap \binom{V'}{2}$. If $E' = E \cap \binom{V'}{2}$, we say that G' is an *induced subgraph* of G or the *graph induced by* V'. An *independent set* $I \subseteq V$ in G is a set of vertices inducing an empty graph $G' = (I, \emptyset)$ in G.

For $W \subseteq V$, the set $\mathbf{N}_G(W) := \{u \in V : \exists v \in W \text{ with } uv \in E\}$ is called the *neighborhood* of W in G. Where the graph in question is clear from the context, we may omit the subscript G, and for simplicity, we also write $\mathbf{N}(v_1, \ldots, v_k)$ instead of $\mathbf{N}(\{v_1, \ldots, v_k\})$. The *degree* $deg_G(v)$ of a vertex v in G equals the size of its neighborhood in G. We denote the minimum value among the degrees of vertices in G (in the graph induced on a vertex set V) by $mindeg(G)$ ($mindeg(V)$) and call it the *minimal degree* of G (of the graph induced on V). The *average degree* of G

is the arithmetic mean of its vertex degrees. If W, W' are subsets of V, not neccessarily disjoint, then $E(W, W')$ contains all edges $ww' \in E$ such that $w \in W$ and $w' \in W'$ and $\mathbf{e}(W, W') = |E(W, W')|$.

A *trail* between two vertices u and w in G consists of a sequence $v_1, v_2, \ldots, v_k$ of vertices from V with $v_1 = u$, $v_k = w$, and $v_i v_{i+1} \in E$ for all $0 < i < k$. An inclusion-maximal set of vertices K is called a *component* of G if between each pair of vertices from K there exists some trail.

In general we use the term *coloring* for a mapping $\Upsilon : V \mapsto \Gamma$ from G to some set of *colors* $\Gamma \subset \mathbb{N}$. If $\Upsilon(u) \neq \Upsilon(v)$ for all $uv \in E$ we say that the coloring Υ is *valid* or *proper*. Here, $\Upsilon(v)$ is the *color* of v and the sets $\{v \in V : \Upsilon(v) = \gamma\}$ are called *color classes* of Υ or *partitions* of G. Moreover, if $|\Gamma| = k$, then Υ is a *k-coloring*.

The chromatic number $\chi(G)$ of a graph G is defined as $\chi(G) = min_\Upsilon(|\Gamma|)$, where the minimum is taken over all valid colorings $\Upsilon : V \mapsto \Gamma$ of $G = (V, E)$. The problem of finding a coloring of G that uses $\chi(G)$ colors is called the *coloring problem* of G whereas the term *k-coloring problem* is used if we search for a coloring with a predefined number k of colors.

In the following we will occasionally use the term *coloring* even if Υ is not defined on all elements of V. If we want to emphasize that this is the case we say that Υ is a *partial coloring*. If the domain of a partial coloring Υ is extended by some additional vertices to a set $V' \subseteq V$, we say that the coloring obtained in this way is an *extension* of Υ to V' (or to the graph $G(V')$), or alternatively that Υ is *extended* to V' (or to $G(V')$). Two valid partial colorings Υ and Υ' are *compatible* if they coincide on the vertices which their domains $\mathcal{D}(\Upsilon)$ and $\mathcal{D}(\Upsilon')$ share and if they can additionally be merged (in the obvious way) to form a valid coloring on $\mathcal{D}(\Upsilon) \cup \mathcal{D}(\Upsilon')$.

A k-cut of G is a partition of $V(G)$ into k disjoint sets $V_1, \ldots, V_k$. A 2-cut is also called *edge cut* or simply *cut*. Since the problem referred to is usually clear from the context, we again use the term *partitions* for the sets V_i. The *weight* of a k-cut is the total number of edges crossing the cut, i.e., edges that are connecting vertices in different partitions. MAXCUT is the problem of finding a cut of maximum weight in a given graph. Analogously, in MAX-k-CUT we ask for a maximum k-cut.

Formally, the (vertex) *expansion* of a connected graph $G = (V, E)$ is the minimal value of $|\mathbf{N}(U) \setminus U|/|U|$ for $U \subset V$ with $|U| \leq |V|/2$. In this text we use the term expansion only in qualitative statements. When we say that a graph has *good expansion* or *good expansion properties* we mean that the expansion of G is not "too small".

2.2 Random Graphs

A random graph $\mathcal{G}_{n,1/2}$ on n vertices is a graph that is taken uniformly at random from the set of all graphs on $V = [n] := \{1, 2, , \ldots, n\}$. Equivalently we can regard $\mathcal{G}_{n,1/2}$ as the result of the following random process: Choose each of the possible $\binom{n}{2}$ edges on $[n]$ independently with probability 1/2. Taking this approach, there is an obvious generalization of $\mathcal{G}_{n,1/2}$ to the so-called random graph model $\mathcal{G}_{n,p}$, where edges now are chosen with probability p. By linearity of expectation, the expected number of edges incident to a vertex of $\mathcal{G}_{n,p}$ equals $(n-1)p$. Therefore, the parameter p governs the average degree, or what we call the *density* of the graph. When used in a numerical context the term density directly refers to the expression np.

The main part of this thesis is concerned with a different random graph model (although the terms *random graph* and *random graph model* were introduced in connection with the standard model $\mathcal{G}_{n,p}$ we will use them in a more general way for various types of randomly generated graphs). Let $\mathcal{G}_{n,p,k}$ be a graph $G = (V, E)$ with $|V| = n$ that consists of k disjoint vertex sets or *partitions* C_i, $i \in \{1, \ldots, k\}$, of equal size, i.e., $V = \bigcup_{1 \le i \le k} C_i$ and $|C_i| = n/k$. Further, $\mathcal{G}_{n,p,k}$ has possible edges only between different partitions. Each edge occurs independently with probability p. Although p may take arbitrary values between 0 and 1, in the main part of this text we will generally assume that $p = d/n$ with $d = d(n) \ge c$ for some sufficiently large but constant c. At this, what is "sufficiently large" will be determined by the requirements of the analysis we will present.

When we refer to the partitions of $\mathcal{G}_{n,p,k}$ in the following with an index variable i we always assume that $i \in \{1, \ldots, k\}$ without mentioning this explicitely. For a coloring

$$\Upsilon(G) \,:\, V(G) \mapsto \{1, \ldots, k\}$$

we say that $v \in C_i$ is colored *incorrectly* or *wrongly* if $\Upsilon(v) \ne i$.

Now, let $\mathcal{G}$ be one of the graph models described above or a similar one. Then we say that $G \in \mathcal{G}$ if G is a random graph taken according to the distribution of $\mathcal{G}$. In times we also denote such a graph simply by $\mathcal{G}$.

We say that an event σ occurs *with high probability* (w.h.p.) in a random graph model $\mathcal{G}$, if $\mathbf{P}[\,\sigma\,] \to 1$ as n tends to infinity. A graph property holds *almost surely* if w.h.p. a graph from $\mathcal{G}$ has this property.

For analyzing the properties of random graphs, standard notions from probability theory such as the expectation $\mathbf{E}[\,X\,]$ or the variance $\mathrm{Var}\,[X]$ of a random variable X are used. Again, for a random variable X and a distribution Θ we write $X \in \Theta$ if X is distributed according to Θ. If a random variable X takes only nonnegative values, then we say that X is a *non-negative random variable*.

A *uniform distribution* is a distribution with constant density function. Essentially the only other probability distributions we will consider in the following are binomial distributions and standard normal distributions. Therefore, we finish this section by introducing some more terminology related to these two distributions. By $\mathrm{Bin}(n, p)$ we denote the *binomial distribution* with parameters n and p. For the *normal distribution* with expectation μ and variance ν we write $\Phi(\mu, \nu)$. $\Phi(0, 1)$ is also called the *standard normal distribution*. The *chi-squared distribution* $\mathrm{Chi}^2(n)$ *of dimension* n is the distribution derived by taking the sum of n squared standard normal distributed and independent variables. Therefore, $\mathbf{E}[\,X\,] = n$ for $X \in \mathrm{Chi}^2(n)$. The square root of an n-dimnesional chi-squared random variable follows a *chi distribution* $\mathrm{Chi}(n)$ *of dimension* n (see e.g., [66]). In standard literature $\mathrm{Chi}(n)$ and $\mathrm{Chi}^2(n)$ are often referred to as $\chi(n)$ and $\chi^2(n)$, respectively. Here, we avoid this notation since χ is already used for the chromatic number of a graph.

2.3 Algorithmic Aspects

We assume familiarity with basic notions from algorithmic theory such as the concepts of complexity and hardness or the classes $\mathcal{P}$ and $\mathcal{NP}$. In particular, recall that coloring, k-coloring for $k > 2$, MAXCUT and MAX-k-CUT for $k > 1$ are all $\mathcal{NP}$-hard problems (for an introduction to the field see [26]).

We say that an algorithm $\mathcal{A}$ *approximates* the solution to a *minimization* problem *within ratio* $f(n)$ (occassionally also simply *within* $f(n)$) if for each input of size n the optimal solution

OPT is guaranteed to be at most by a factor of $f(n)$ away from the solution $OUT(\mathcal{A})$ calculated by $\mathcal{A}$, i.e.,

$$\frac{OUT(\mathcal{A})}{OPT} \leq f(n).$$

For *maximization* problems this inequality is replaced by $OUT(\mathcal{A})/OPT \geq f(n)$. $f(n)$ is also called the *approximation ratio* of $\mathcal{A}$. Observe that this definition implies $f(n) \geq 1$ for minimization problems and $f(n) \leq 1$ for maximization problems.

An algorithm $\mathcal{A}$ runs in *polynomial expected running time* when applied to a distribution $\mathcal{G}$ of graphs if there is a constant l such that $\sum_{|G|=n} t_{\mathcal{A}}(G) \cdot \mathbf{P}[\,\mathcal{G} = G\,] \leq n^l$ for all sufficiently large n. Here, $t_{\mathcal{A}}(G)$ is the running time of $\mathcal{A}$ on input G.

For considerations on asymptotic behaviour we use standard notation: Let $f(n)$ and $g(n)$ be two real-valued functions depending on n. Then $f(n) = \mathcal{O}(g(n))$ if there is a constant c such that $|\,f(n)\,| \leq c \cdot g(n)$ for all sufficiently large n, $f(n) = \Omega\,(g(n))$ if $g(n) = \mathcal{O}(f(n))$, and $f(n) \sim g(n)$ if $|\,f(n)/g(n)\,|$ goes to 1 as n tends to infinity. Additionally, we write $f(n) = o\,(g(n))$ if for all $c > 0$ there is an n_c such that $|\,f(n)\,| < c \cdot g(n)$ for $n > n_c$, and $f(n) \ll g(n)$ if $f(n) \geq 0$ and $f(n) = o\,(g(n))$.

2.4 Convex Optimization

Semidefinite programming forms a special class of convex optimization problems, i.e., problems with a solution set that is convex. The origins of the techniques that evolved in this field can be found in the study of a simpler class of optimization problems, so-called linear programs (cf. [46]). For later reference, we will first describe the nature of these problems.

2.4.1 Linear Programming

Any optimization problem of the following form is called a *linear program*:

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} c_i x_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} a_{ij} x_i \geq b_j \quad j = 1, \ldots, n, \\ & \qquad x_i \geq 0 \quad\;\; i = 1, \ldots, m. \end{aligned}$$

The minimization refers to the x_i, which are also called *decision variables*, and “s.t.” is short for *subject to*. The decision variables may take values from the real numbers. The linear function $\sum_{i=1}^{m} c_i x_i$ is called the *objective function*. The remaining conditions provide the *constraints* to the minimization problem.

An assignment of specific values to the decision variables is called a *solution*. A solution is *feasible*, or a *feasible point*, if it satisfies all of the constraints and *optimal* if in addition the desired minimum is attained. The value the objective function attains for a given solution is also called the *objective value* for this solution.

To each linear minimization problem there is a corresponding maximization problem, the so-called *dual linear program*:

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} b_j y_j \\ \text{s.t.} \quad & \sum_{j=1}^{m} a_{ij} y_j \geq c_i \quad i = 1, \ldots, m, \\ & y_j \geq 0 \quad j = 1, \ldots, n. \end{aligned}$$

Of course the maximization in this problem is, again, taken over a set of decision variables ranging over the real numbers.

It is not difficult to show that the dual of the dual is again the original linear program. In addition, the most important result in the theory of linear programs states that if either of the two programs has a (finite) solution then the other program also has a (finite) solution with the same objective value (see [65]). This property is called *strong duality*. It plays an important role in the development of algorithms for solving linear programs since it provides a convenient tool for checking optimality (e.g., in the case that primal and dual solutions are constructed successively and with increasing precision; cf. [65]).

2.4.2 Semidefinite Programming

The ideas behind linear programming can be generalized and extended in several ways. One approach in the direction of a more general class of convex optimization problems that has attained more and more attention throughout the past years is the concept of semidefinite programs, also denoted by SDP in the following. This class of problems is somewhat more difficult to handle, but it still shares the nice properties of linear programming to a large extent. Moreover, the set of semidefinite programs contains all linear programs as well as many other well-known generalizations to them, such as convex quadratic programming (see [33]).

Before explaining the structure of semidefinite programs in detail, we need to fix the notation for a couple of basic concepts from linear algebra and recall some important properties of positive semidefinite matrices.

We denote by $\mathcal{M}_{m,n}$ the set of real $m \times n$ matrices $\mathbf{A} = (a_{ij})_{1 \leq i \leq m, 1 \leq j \leq n}$ and by $\mathcal{S}_n \subset \mathcal{M}_{n,n}$ those that are *symmetric*. The *rank* of a matrix $\mathbf{A} \in \mathcal{M}_{m,n}$ is the number of its linearly independent rows or columns. For a symmetric matrix $\mathbf{A}$ the *trace* $\operatorname{tr}(\mathbf{A})$ equals the sum of its diagonal elements. The *diagonal matrix* $\operatorname{diag}(\mathbf{v})$ of a vector $\mathbf{v}$ is the symmetric matrix having the elements of the vector on the diagonal and 0 in all other positions. The all-ones vector is denoted by $\mathbf{1}$.

Since $\mathcal{M}_{m,n}$ can be regarded as a vector space in $\mathbb{R}^{m \cdot n}$ the natural inner product between $\mathbf{A}, \mathbf{B} \in \mathcal{M}_{m,n}$ is given by

$$\langle \mathbf{A} \,|\, \mathbf{B} \rangle = \operatorname{tr}(\mathbf{B}^{\mathbf{T}} \mathbf{A}) = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} \cdot b_{ij}.$$

Consistent with this definition, the *scalar product* of two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is written as

$$\langle \mathbf{x} \,|\, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i \cdot y_i,$$

and $\sqrt{\langle \mathbf{x} \,|\, \mathbf{x} \rangle}$ is the (Euclidean) *norm* $\|\mathbf{x}\|$ of $\mathbf{x}$.

A matrix $\mathbf{A} \in \mathcal{S}_n$ is *positive semidefinite*, denoted by $\mathbf{A} \succeq 0$ or $\mathbf{A} \in \mathcal{S}_n^+$, if $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$. Equivalent definitions of positive semidefiniteness include that all eigenvalues of $\mathbf{A}$ are nonnegative or that $\mathbf{A} = \mathbf{C}^T\mathbf{C}$ for some matrix $\mathbf{C} \in \mathcal{M}_{m,n}$ having the same rank as $\mathbf{A}$.

We return to semidefinite programs shortly. But first we apply the preceeding definitions to formulate two graph theoretical concepts not introduced so far. Consider a graph $G = ([n], E)$. Then the *adjacency matrix* $\mathbf{A}(G) = (a_{vw})_{1 \leq v,w \leq n}$ of G is defined by

$$a_{vw} = \begin{cases} 1 & vw \in E, \\ 0 & otherwise. \end{cases}$$

An alternative algebraic representation of a graph is the so-called *Laplace matrix*, given by $\mathbf{L}(G) := \operatorname{diag}(\mathbf{A}(G) \cdot \mathbf{1}) - \mathbf{A}(G)$. Note that, by this definition, the entries on the diagonal of $\mathbf{L}$ equal the degrees of the corresponding vertices.

The following definitions and remarks on semidefinite programming are taken from [33]. Details and further explanations can be found there.

Semidefinite programming is linear programming over positive semidefinite matrix variables, i.e., the objective function and the constraints still are given by linear functions. But now the variables of these functions may take values from $\mathcal{S}_n^+$. Accordingly, linear programming is a special case of semidefinite programming where the matrices involved are restricted to diagonal matrices.

The standard formulation of a semidefinite program, also called *primal semidefinite program* and abbreviated PSDP is of the following structure:

$$\begin{array}{ll} \min & \langle \mathbf{C} \,|\, \mathbf{X} \rangle \\ \text{s.t.} & \mathcal{A}\mathbf{X} = \mathbf{b}, \\ & \mathbf{X} \succeq 0. \end{array}$$

As indicated, the minimum is taken over all positive semidefinite matrices $\mathbf{X}$; $\mathbf{C}$, $\mathcal{A}$ and $\mathbf{b}$ are given as parameters of the semidefinite program. Here, the linear operator $\mathcal{A} : \mathcal{S}_n \mapsto \mathbb{R}^m$ combines the linear *constraints* $\mathbf{A}_i \in \mathcal{S}_n$ $(1 \leq i \leq m)$ in the form

$$\mathcal{A}\mathbf{X} := \begin{pmatrix} \langle \mathbf{A}_1 \,|\, \mathbf{X} \rangle \\ \vdots \\ \langle \mathbf{A}_m \,|\, \mathbf{X} \rangle \end{pmatrix}.$$

The notions of contraints, the objective function, a feasible point, a solution and the objective value carry over from linear programs. The matrix $\mathbf{C} \in \mathcal{S}_n$ is also called the *cost matrix*.

Analogous to the concepts developed for linear programs one can define a *dual semidefinite program* DSDP of PSDP. Given the primal program, the dual may be constructed by using Lagrange multipliers. The resulting maximization problem then reads as follows:

$$\begin{array}{ll} \max & \langle \mathbf{b} \,|\, \mathbf{y} \rangle \\ \text{s.t.} & \mathcal{A}^T\mathbf{y} + \mathbf{Z} = \mathbf{C}, \\ & \mathbf{y} \in \mathbb{R}^m, \, \mathbf{Z} \succeq 0. \end{array}$$

Here, the maximization runs over both the vector $\mathbf{y}$ and the so-called *slack matrix* $\mathbf{Z} \in \mathcal{S}_n^+$.

Although not immediately obvious from the representation given here, one can show that DSDP is a semidefinite program as well, in the sense that it may be reformulated in standard notation.

One important property of the dual pair PSDP and DSDP of optimization problems is what is refered to as *weak duality.*

Theorem 3 (weak duality; see e.g., [33])
The objective of any primal feasible solution is greater than or equal to the objective value of any dual feasible solution.

In contrast to linear programming, the objective values of optimal solutions for DSDP and PSDP do not necessarily coincide. However, if the pair of optimization problems fulfills some further reasonable requirements, strong duality can also be established for semidefinite programs.

We will not comment on the issues of duality again after this section. But it is not difficult to check that for all semidefinite programs occuring in this text strong duals do exist.

3 Coloring Graphs

"What good are the colors if I don't know how to paint."

MICHEL EYQUEM DE MONTAIGNE

As discussed in the introduction, both the coloring problem and the k-coloring problem are $\mathcal{NP}$-hard and therefore no feasible methods for solving them exactly are known. Accordingly, it is necessary to search for alternative approaches to obtain efficient algorithms.

One possibility is to introduce structural restrictions to the graphs allowed as input. This method has been considered in great depth and has been developed into many different directions (see, e.g. [19, 41, 57]). Its main motivation rests upon the fact that if graphs are used in applications, the structural properties of the problem under investigation lead to special structural properties in the resulting graphs.

Just to name an example, one class of graphs that is relevant to a number of graph coloring applications (see e.g., [4]) is the class of so-called interval graphs. These graphs may be represented by intersecting intervals over the real line. Given a representation of this form it is possible to efficiently order the vertices in such a way that the simple greedy heuristic produces an optimal coloring [54]. Thus the additional structural information on the input gives rise to fast coloring methods. Another important result of different flavor is provided by the celebrated strong perfect graph theorem (see e.g., [11]). From this and the possibility of efficiently calculating the Lovász number mentioned in the introduction it follows that for graphs containing neither odd cycles nor their complements as induced subgraphs the coloring problem can be solved in polynomial time. Conversely, for many other important and interesting graph classes negative results have been established over the years. For example, coloring planar graphs remains $\mathcal{NP}$-hard (although their chromatic number can be approximated very well). For a detailed treatment of structural results in graph coloring we refer to [35].

If restrictions to the type of graphs accepted as input are not desired, other parameters of the problem or the requirements on the algorithm need to be altered. In general the theory of computation knows different approaches for finding good, but possibly suboptimal, solutions to $\mathcal{NP}$-hard problems within a reasonable running time. We will briefly describe these approaches here, but solely refer to their application on graph problems.

First of all, there are *approximation algorithms*. These algorithms guarantee some non-trivial approximation ratio on any graph by, at least implicitly, constructing both, upper and lower bounds to the parameter investigated.

If due to non-approximability results it is unlikely that good approximative solutions can be obtained, then so-called *heuristics* may be considered as an alternative. Algorithms belonging to this class share the following properties: They always run in polynomial time and almost always output a correct or at least good solution to the problem studied.

What is problematic with heuristics is their insufficiency in coping with atypical cases. Algorithms with *polynomial expected running time* may be seen as a compromise solution. The running time of such algorithms is polynomial only on average (with respect to some underlying graph model) but the output is always a correct or at least reasonably good solution. These requirements of an algorithm are certainly stronger than those of pure heuristics.

Following this classification we will next comment on the state of the art as far as approximation algorithms for graph coloring are concerned. Turning to heuristics we will thereafter briefly sketch what is known about the chromatic number on standard random graph models, which allows us then to put in perspective the various results on finding exact graph colorings with high probability. Finally, algorithms with polynomial expected running time will be considered. The main contribution of this text is an algorithm of the last type. Therefore, we will provide an outline of this algorithm at the end of this chapter.

3.1 Approximating the Chromatic Number

One of the most discouraging facts in connection with algorithmic graph coloring is that this problem is not only intractable for itself, but also guaranteeing any modestly good solution proves to be conceivably hard.

Indeed, as was shown by Feige and Kilian [22], no polynomial time algorithm can approximate $\chi(G)$ within $n^{1-\epsilon}$ for any constant $\epsilon > 0$ unless $\mathcal{ZPP} = \mathcal{NP}$. Here, $\mathcal{ZPP}$ is the class of problems that can polynomially be decided with zero error probability, i.e., a corresponding algorithm may at times (for an ϵ fraction of all inputs) output the answer “unknown” but it never produces a wrong result and always halts in polynomial time. Accordingly, it is unlikely that a polynomial time approximation algorithm can be designed that considerably improves over the trivial one: Assign a seperate color to each vertex. Recently, Engebretsen and Holmerin [20] proved that the situation is even worse. According to their result the non-approximability bound from Feige and Kilian remains valid for non-constant ϵ, more precisely for $\epsilon = \mathcal{O}\big(1/\sqrt{\log\log n}\big)$. This gets rather close to the currently best known positive result in the field, which is due to Halldórsson [30] who approximated $\chi(G)$ within $n^{1-\mathcal{O}(\log\log n/\log n)}$. In comparison, a relatively simple algorithm with approximation ratio $\mathcal{O}(n/\log n)$ was provided by Johnson as early as 1974 [36]. Johnson's Algorithm repeatedly constructs maximal independent sets I by starting at an arbitrary vertex v, putting v into I, then deleting v and its neighborhood and continuing. Each independent set obtained in this way is assigned a new color and subsequently removed from the graph.

What is possible for an approximation algorithm significantly improves if we turn to graphs with small chromatic number. For example, Wigderson [67] developed a simple algorithm to color 3-colorable graphs with $\sqrt{n}$ colors. He exploited the fact that in a 3-colorable graph the neighborhood of each vertex is bipartite and can therefore easily be colored optimally.

The currently best approximation algorithm for the 3-coloring problem is due to Blum and Karger [5] and uses $n^{3/14+o(1)}$ colors. The principles building the basis of this algorithm were developed by Karger, Motwani, and Sudan [37]. Using semidefinite programming techniques, they proposed a coloring algorithm which achieves an approximation ratio of $\mathcal{O}\big(n^{1/4}\big)$. For this, they introduced the notion of a vector coloring. Up to date, all those coloring algorithms for graphs with small chromatic number that achieve the best approximation ratios make use of vector colorings as well. Therefore, we will review this concept. More details of the approach of Karger, Motwani, and Sudan are postponed until after the discussion of preliminary results in Section 4.4.

A *vector k-coloring* of a graph $G = ([n], E)$ consists of an assignment $\overrightarrow{\sigma} : V \mapsto \mathbb{R}^n$ of unit vectors $\mathbf{v}_i \in \mathbb{R}^n$ to the vertices i of the graph such that for any vertices i and j with $ij \in E$

$$\langle \mathbf{v}_i \,|\, \mathbf{v}_j \rangle \leq -\frac{1}{k-1}.$$

With this definition the vectors corresponding to two adjacent vertices are required to be "different", mimicking the fundamental property of proper colorings in a relaxed way. The *vector chromatic number* $\overrightarrow{\chi}(\mathrm{G})$ of G then is the minimal k such that G is vector k-colorable.

As we will see later, every k-colorable graph has a vector k-coloring and so vector colorings can be used to construct lower bounds for the chromatic number. In addition, a vector k-coloring (when it exists) can be found in polynomial time (see [37]).

Despite its success in approximative k-coloring, the vector chromatic number does unfortunately not provide a good approximation ratio for non-constant k. In fact, Feige, Langberg, and Schechtman [23] showed that there are graphs of vector chromatic number $\mathcal{O}(\log n/\log\log n)$ whose chromatic number gets as big as $n/(\log n)^c$ for some constant c.

Turning to negative results on k-coloring for small k, Charikar [10] constructed an infinite family of vector $2+\epsilon$-colorable graphs G for any $\epsilon > 0$ such that $\chi(G) \geq n^{\Omega(1)}$. It follows that vector coloring alone does not give rise to a $n^{o(1)}$ approximation for the 3-coloring problem.

The most important non-approximability result for coloring with a constant number of colors is due to Khanna, Linial, and Safra, who showed that it is $\mathcal{NP}$-hard to 4-color 3-colorable graphs unless $\mathcal{P} = \mathcal{NP}$ [40].

3.2 Coloring Random Graphs in Polynomial Time

The properties of the chromatic number have been extensively studied on $\mathcal{G}_{n,p}$. Early results on the independence number of a random graph led to a number of consequences for $\chi(\mathcal{G}_{n,p})$. Then, in 1987, Shamir and Spencer [59] obtained a sharp concentration result for the chromatic number in this graph model, using martingales. Surprisingly, they showed that for each p the chromatic number $\chi(\mathcal{G}_{n,p})$ almost surely lies in an interval $I(n,p)$ of roughly length $\sqrt{n}$. Moreover, for sparse random graphs $\mathcal{G}_{n,p}$ with $p < 1/n^{\beta}$ and $1/2 < \beta < 1$ the chromatic number is concentrated in an interval of constant length. As was observed by Łuczak [51], the concentration is even in width one in the case that $5/6 < \beta < 1$. Accordingly, for this range of p the chromatic number of $\mathcal{G}_{n,p}$ is (asymptotically) uniquely determined. This result was extended by Alon and Krivelevich [3], who showed that $\chi(\mathcal{G}_{n,p})$ is concentrated on two points if $1/2 < \beta < 1$.

For general p, Bollobás [7] and Łuczak [50] were able to calculate the probable value of $\chi(\mathcal{G}_{n,p})$:

$$\chi(\mathcal{G}_{n,\frac{1}{2}}) \sim \frac{n}{2\log n} \qquad \text{and} \qquad \chi(\mathcal{G}_{n,p}) \sim \frac{np}{2\log np} \quad \text{for} \quad \frac{c}{n} \leq p = o\,(1)\,.$$

Here, c is large but constant (see [34] for a detailed treatment of the methods and ideas used for obtaining these and related results). Further on, Achlioptas and Friedgut [1] proved that the property of being k-colorable also exhibits a sharp threshold. More precisely, for any constant k there exists a sequence $d_k = d_k(n)$ such that, for any $\epsilon > 0$ the random graph $\mathcal{G}_{n,p}$ is w.h.p. k-colorable if $p < (1-\epsilon)d_k(n)/n$, and w.h.p. non-k-colorable if $p > (1+\epsilon)d_k(n)/n$. It is conjectured but not known that for each k the sequence $d_k(n)$ converges to a constant as n goes to infinity (see [34]).

As far as algorithms are concerned, for $p = 1/2$ already the linear-time greedy heuristic achieves a factor 2-approximation, because w.h.p. it does not use more than $(1 + o(1))n/\log n$ colors on $\mathcal{G}_{n,1/2}$ (see [34]). Nonetheless, this heuristic fails to guarantee a non-trivial approximation ratio because it does not provide any lower bound on the chromatic number of the input graph. As motivated by the remarks in the preceeding section, such algorithms with non-trivial approximation ratio can only be obtained if a non-polynomial worst case running time complexity is taken into account. We will turn to results following this approach in the next section.

Algorithms that color random k-colorable graphs correctly with high probability were developed by Kučera [44], Turner [64], and Dyer and Frieze [18]. However, as most k-colorable graphs are quite dense (for a vertex, on average about half of the possible edges to all other color classes are present) the number of common neighbors of vertices having the same color considerably exceeds the number of common neighbors of vertices with distinct colors. As a consequence a coloring algorithm exploiting only this fact already finds the desired coloring almost surely.

The problem of coloring sparse random graphs turns out to be more difficult. First of all, the graph model introduced above needs to be modified in order to produce k-colorable graphs of a prescribed density on average. For this, vertices are assigned to color classes uniformly at random and the probability for the occurence of valid edges is determined by a parameter p.

For a graph G constructed in this way the partition obtained is balanced w.h.p., that is, all color classes are roughly of the same size. This follows from elementary considerations in probability theory and motivates why we restrict ourselves to graphs with equal sized color classes in the following. This leads to the random graph model $\mathcal{G}_{n,p,k}$ defined earlier. Moreover, G naturally comes with an associated k-coloring. We will often refer to this coloring as the coloring of the random k-colorable graph G.

First attempts to investigate the k-coloring problem on $\mathcal{G}_{n,p,k}$ were performed by Petford and Welsh [55]. They used a randomized heuristic for 3-coloring random 3-colorable graphs and provided experimental evidence that this heuristic works for most edge probabilities. Subsequently, Blum and Spencer [6] presented a polynomial time algorithm that colors 3-colorable graphs optimally w.h.p. provided $p \geq n^{\epsilon}/n$. As usual, ϵ may be chosen arbitrarily small but is fixed. The solution of Blum and Spencer is based on a path counting technique, which can be viewed as a generalization of the method of counting common neighbors described above.

In 1997 Alon and Kahale [2] improved on these early results by analyzing the spectral properties of $\mathcal{G}_{n,p,k}$. More specifically, they showed that it is possible to read off a rather accurate approximation to the color classes from the eigenvectors corresponding to the smallest $k-1$ eigenvalues of the adjacency matrix of a large subgraph of $\mathcal{G}_{n,p,k}$. Several iteratively performed improvements then produce a proper k-coloring as desired.

The polynomial expected time algorithm presented in this work is inspired by the approach of Alon and Kahale. Therefore, we will describe their ideas in greater detail in Section 5.1. But now we turn our attention to previous results on coloring algorithms with non-polynomial worst case complexity.

3.3 Coloring Fast on Average

For solving the coloring problem on random graphs $\mathcal{G}_{n,p}$, successively different algorithms with non-trivial approximation ratio and polynomial expected running time were developed. For the case $p > 1/\sqrt{n}$ Krivelevich and Vu [43] were able to guarantee a $\mathcal{O}\big(\sqrt{np}/\log(np)\big)$ approximation by applying spectral techniques. Coja-Oghlan and Taraz [16] then constructed lower-bounds via the Lovász number θ in order to approximate the chromatic number within $\mathcal{O}\big(\sqrt{np}\big)$ in polynomial expected running time for $p > \log^7 n/n$. They additionally developed an exact algorithm for $p \leq 1.01/n$. A connection between the MAX-k-CUT problem and graph coloring led to the complementation of these results: In [15] Coja-Oghlan, Moore, and Sanwalani provided an efficiently computable lower bound on $\chi(\mathcal{G}_{n,p})$ by using a semidefinite programming relaxation for MAX-k-CUT. With this they extend the $\mathcal{O}\big(\sqrt{np}\big)$ approximation result to $1/n \leq p$, i.e., essentially all values of p. The approximation ratio was subsequently improved by Kuhtz [45] to $\mathcal{O}\big(\sqrt{np}/\ln np\big)$.

Additional results could be obtained for deciding in polynomial expected time whether a graph from $\mathcal{G}_{n,p}$ is k-colorable. For $p \geq \exp(\Omega\,(k))\,/n$ this problem was addressed by Krivelevich [42]. Later, Coja-Oghlan [13] solved the case $k = o\,(\sqrt{n})$ and $p \geq ck^2/n$ for some $c > 0$.

Turning to random graph models other than $\mathcal{G}_{n,p}$, Dyer and Frieze [18] provided an algorithm for k-coloring randomly chosen k-colorable graphs in polynomial expected running time. The investigation of $\mathcal{G}_{n,p,k}$ in this context was started by Subramanian [62] who proved that, for constant k and $p \geq n^{\epsilon}/n$, graphs from $\mathcal{G}_{n,p,k}$ may be colored optimally in polynomial expected time. Non-constant values of k were considered by Coja-Oghlan [14], who established a k-coloring algorithm that terminates in polynomial expected time on $\mathcal{G}_{n,p,k}$ if $np \geq c \cdot \max\big(k \ln n, k^2\big)$ where $k = k(n)$ and c is constant.

In this work we consider random graphs that are sparser than those studied by Subramanian and Coja-Oghlan. We provide an algorithm COLOR$\mathcal{G}_{n,p,3}$ that runs in polynomial time on average and outputs a valid 3-coloring for graphs from $\mathcal{G}_{n,p,3}$ provided that $p \geq c/n$ for some sufficiently large constant c.

The main steps of COLOR$\mathcal{G}_{n,p,3}$ resemble those that Alon and Kahale designed in [2] for coloring $\mathcal{G}_{n,p,3}$. The operation of their algorithm can be summarized as follows: First, the algorithm constructs an initial coloring that already colors most vertices correctly. This coloring is then refined according to a local majority vote, i.e., each vertex is assigned the color that is least favorite among its neighbors. After repeating this step a certain number of times, the algorithm then tries to eliminate peculiarities of the coloring obtained so far. To this end, the algorithm repeatedly uncolors those vertices which have suspiciously few neighbors of some color other than their own. The resulting set of uncolored vertices is finally recolored by using a brute force approach.

For the result of Alon and Kahale it is sufficient to produce a valid coloring on the vast majority of input graphs. Therefore, the algorithm just described extensively relies on exploiting probable structures of $\mathcal{G}_{n,p,3}$. However, we also have to take care of exceptional cases. Thus algorithm COLOR$\mathcal{G}_{n,p,3}$ needs to introduce suitable modifications in the main steps of the algorithm presented above. Here, the basic idea is to combine these main steps with an adequate though computationally expensive repair mechanism. This mechanism comes into play only under the unlikely circumstance that no proper coloring can be found via the main steps.

Another central difference between the algorithm of Alon and Kahale and COLOR$\mathcal{G}_{n,p,3}$ is the initial phase. In both algorithms, the purpose of this phase is to construct a coloring of the input graph which might fail on a small constant fraction of the vertices but which is sufficiently good to serve as a starting point for further improvements. In contrast to the approach of Alon and Kahale which solely relies on the spectrum of the graph to achieve this goal, our methods are based on semidefinite programming. Similar methods were used by Coja-Oghlan in [14].

4 Semidefinite Programming

"Narrative is linear, but action has breadth and depth as well as height and is solid."

Thomas Carlyle

As discussed earlier semidefinite programming has its roots in the theory of linear optimization. This subject in turn goes back to the study of systems of linear inequalities, which already caught the interest of Fourier in 1826 (this and all following remarks on linear programming are taken from [12] and [65]). Its popularity in the first half of the twentieth century was in some sense initiated by the interest in military advancement at the time: In 1947 Danzig invented the simplex method for solving linear planning problems of the U.S. Air Force. Independently the Russian mathematician Kantorovich had already developed a rudimentary algorithm for solving a certain class of linear programs eight years earlier. His work was interrupted by World War II, however, and his results remained hidden from the western world by an iron curtain for many years.

The early applications of linear programming also included a huge variety of problems in production management as well as calculations in connection with classical economic theories. Later the algebraic methods that were developed along with the theory of linear programs also lead to mathematical concepts of independent interest. It could be shown, for instance, that von Neumann's famous Minimax Theorem may be obtained as a direct consequence of linear programming duality.

While practically convincing an increasingly automated world of the usefulness of algorithmic solutions to optimization problems, from a theoretical point of view the simplex method remained unsatisfactory. It shows a good performance for essentially all real world problems, its worst case complexity, however, is exponential. It was not before 1979 that this insufficiency could be eliminated. In that year Khachian introduced the ellipsoid method for solving linear programs in polynomial time. Even so, in applications none of the polynomial time methods suggested since then can keep up with the simplex algorithm from 1947.

In comparison to linear methods, mastering nonlinear optimization problems is usually considered much more difficult. Convex problems that can be formulated as semidefinite programs however, can be regarded as an exception to this general rule of thumb. The systematic study of this kind of problems over the past decades has led to many new results in the area and a much better understanding of the topic. Its most important applications can be found in control theory [8, 21], signal processing [53], eigenvalue optimization [48], and combinatorial optimization (cf. [33]). Recently, semidefinte programs also proved useful in 3-D object recognition [17].

While linear programming is restricted to optimization over the reals a semidefinite program may additionally use positive semidefinite matrix variables. The introduction of such matrix variables allows for convex feasible sets other than polyhedral ones. In spite of this difference the optimization is still subject to linear constraints (for formal definitions see Section 2.4).

It follows, as argued earlier, that linear programming is just a special case of semidefinite programming.

Moreover, when applied to integer programs, the space of feasible solutions often becomes smaller when a suitable SDP is used as a relaxation instead of a linear program. Thus the application of semidefinite programming techniques can also be helpful in the case that solving a corresponding linear program would consume more time than desirable.

Turning to algorithms, in 1981 Grötschel, Lovász, and Schrijver [28] showed how to apply the ellipsoid method for solving an optimization problem that can be expressed as a semidefinite program. Indeed, the ellipsoid method can be used for calculating the solution to a semidefinite program within any desired precision (cf. [29]). However, as in the case of linear programs the ellipsoid method is computationally expensive. Therefore, the result of Grötschel, Lovász, and Schrijver was mainly of theoretical interest. It made it possible to show the feasibility of certain computational problems by reducing their solution to the calculation of a semidefinite program.

SDPs were not considered of practical relevance until faster algorithms for solving them became available. The development of such algorithms got its start in 1984 with Karmarkar's work [38] on so-called interior point methods. The algorithms that arose from these and resulting investigations (cf. [33]) were faster than those based on the ellipsoid method but in general harder to implement. Fortunately, today a number of standard tools are available for dealing with semidefinite programs in practice. Among the fastest are the DSDP program by Benson, Ye, and Zhang, the CSDP package by Borchers, and the SBmethod software by Helmberg (see [32] for an overview on SDP solver implementations).

In the following sections we explain how semidefinite programming is used in combinatorial optimization. We follow the presentation of Helmberg [33].

4.1 Semidefinite Programs and Combinatorial Optimization

Problems in combinatorial optimization can be phrased as finding an optimal subset in a given family of subsets over some finite set. Here, optimality is defined with respect to a cost function. In general, solving such problems requires complete enumeration of all possible subsets. But fortunately, often structural properties of the constraints involved may be exploited for reducing the number of items to be enumerated.

Another approach to efficiently solving combinatorial optimization problems is that of *convex relaxation*. The basic idea is to reformulate the problem as an optimization problem of a linear cost function over a finite set of *integral points*. In a next step this set of integral points is relaxed to a convex set containing all these points. Although this method can not be used for all problems in the field, it has been successfully applied to many important ones.

The tightest convex relaxation to a set of integral points is their convex hull, a polyhedral set that can be described by linear constraints. Therefore, one possibility for obtaining a relaxation to the set of integral points is to use their convex hull or a linear approximation to it. Unfortunately, the result might be an exponential number of linear inequalities. Therefore, it is not surprising that often no reasonable linear approximation of the convex hull is available. In this case, considering other than linear relaxations of the problem's feasible set may prove useful. As far as such methods are concerned, the application of semidefinite programs has recently received considerable attention. As mentioned earlier the properties of these optimization problems are now well understood and efficient methods for their solution are not only known but also implemented and used.

With respect to graph theory the application of semidefinite programming in approximation algorithms was initiated by Lovász [49] and received increasing attention after the work of Goemans and Williamson [27], who served as a starting point of a series of papers building on their results.

The basic principle underlying a variety of the semidefinite programs designed for graph problems is simple and elegant: Each vertex is labeled with a vector and relations between vertices, e.g., the edges present in the graph, are expressed in terms of the inner product of the associated vectors. A semidefinite program is then obtained by using this vector labeling for phrasing the problem under investigation in an algebraic way and formulating an appropriate relaxation to it.

One of the most important applications of this technique is the concept of vector coloring introduced by Karger, Motwani, and Sudan [37] which was already mentioned in Section 3.1. We will sketch their ideas in further detail shortly since they play a key role in approximative graph coloring. Before turning to vector coloring though, we will illustrate the semidefinite programming approach to combinatorial optimization on a simpler and by now classical example, the MAXCUT problem. We then explain how to generalize the obtained approximative solution for this problem to an approximation algorithm for MAX-k-CUT. The methods involved in this process and some consequences will be used later as part of the coloring algorithm presented in the main part of this text.

4.2 Maximum Cuts

One problem originating from graph theory that has been extensively studied in combinatorial optimization is the problem of finding an edge cut of maximum cardinality in a graph.

As noted earlier, this problem is $\mathcal{NP}$-hard, which justifies the search for approximative solutions. Indeed, MAXCUT can be approximated with approximation ratio 0.5 by a simple greedy algorithm, as was first observed by Sahni and Gonzales [58].

A number of minor improvements over their result were suggested over the years but a real breakthrough could not be obtained before 1995 when Goemans and Williamson [27] showed how to use semidefinite programming for constructing an algorithm that approximates MAXCUT within 0.87856.

The importance of this result becomes evident when compared to a corresponding non-approximability bound: The results of Håstadt [31] and Trevisan, Sorkin, Sudan, and Williamson [63] show that no polynomial time algorithm can approximate MAXCUT with a ratio greater than $16/17 \approx 0.94118$ unless $\mathcal{P} = \mathcal{NP}$.

Because of its importance to the developments in the field of semidefinite programming we will now have a closer look at the approach of Goemans and Williamson. Several concepts reused in the following for different optimization problems will be introduced along the way.

As suggested by the remarks on vector labelings, we start by constructing an algebraic version of MAXCUT. The basic idea is to use vectors $\mathbf{x} \in \{-1, 1\}^n$ for representing cuts in a graph $G = ([n], E)$: we set $x_v = 1$ for vertices v in one of the partitions of the cut and -1 for vertices in the other one. The problem of finding a maximum cut then becomes the following maximization problem:

$$\max_{\mathbf{x}\in\{-1,1\}^n} \left(\sum_{vw\in E(G)} \frac{1-x_v x_w}{2} \right). \tag{4.1}$$

The objective function of this problem can be reformulated by using the adjacency matrix $\mathbf{A} = (a_{vw})$ and the according Laplace matrix $\mathbf{L} = (l_{vw})$ of G:

$$\sum_{vw\in E(G)} \frac{1-x_v x_w}{2} = \frac{1}{4}\sum_{v,w} a_{vw}(1-x_v x_w) = \frac{1}{4}\mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{4}\left\langle \mathbf{L} \,\middle|\, \mathbf{x}\cdot\mathbf{x}^T \right\rangle.$$

With this we can now naturally relax the maximization problem to a semidefinite program:

$$\begin{array}{ll} \max & \frac{1}{4}\langle \mathbf{L} \,|\, \mathbf{X}\rangle \\ \text{s.t.} & \operatorname{diag}(\mathbf{X}) = \mathbf{1} \\ & \qquad \mathbf{X} \succeq 0. \end{array}$$

In the following we will refer to this semidefinite program as $\mathcal{SDP}$.

Goemans and Williamson [27] showed how to interpret the feasible set of $\mathcal{SDP}$ geometrically by rephrasing it in vector notation. For this, they use the fact that each feasible matrix $\mathbf{X}$ is semidefinite and may therefore be factorized into $\mathbf{X} = \mathbf{Y}^T\mathbf{Y}$ for a $\mathbf{Y} \in \mathcal{S}_n$. If we denote the vectors formed by the columns of $\mathbf{Y}$ by $\mathbf{y}_v$ we obtain the following equivalent version of $\mathcal{SDP}$:

$$\begin{array}{lll} \max & \sum_{v,w} \frac{1}{4} l_{vw}\mathbf{y}_v^T\mathbf{y}_w & \\ \text{s.t.} & \mathbf{y}_v^T\mathbf{y}_v = 1 & \forall v \in V \\ & \mathbf{y}_v \in \mathbb{R}^n & \forall v \in V. \end{array}$$

Here, the first set of constraints arises from the constraint $\operatorname{diag}(\mathbf{X}) = \mathbf{1}$ in the original formulation. If each vector $\mathbf{y}_v$ is associated with vertex v of G then we can directly interpret $\mathcal{SDP}$ as a relaxed version of (4.1): The vectors $\mathbf{y}_v$ are relaxations of the $x_v \in \{-1,1\}$ to the n-dimensional unit sphere and the products $x_v x_w \in \{-1,1\}$ are relaxed to $\mathbf{y}_v^T\mathbf{y}_v \in [-1,1]$.

The intuitive interpretation of a solution to this semidefinite program is as follows: If the angle between two vectors of the solution is large then the corresponding vertices should be assigned to different partitions of the cut. Note, however, that conflicts may arise in this process. In order to deal with the resulting complications Goemans and Williamson proposed the use of rounding techniques. To this end they chose a random hyperplane in $\mathbb{R}^n$. The two resulting halfspaces then determine the two partitions of the cut.

4.3 Maximum k-Cuts

Since MAX-k-CUT is a generalization of MAXCUT it is natural to ask whether the approach of Goemans and Williamson to the later problem can be generalized to arbitrary k. And in fact, a corresponding semidefinite programming approximation could be obtained by Frieze and Jerrum [25].

For the MAX-k-CUT problem partitionings of $V(G)$ into k sets are considered. Therefore, the algebraic formulation (4.1) of MAXCUT has to be extended in such a way that variables taking one of k different values instead of just two can be handled. One approach to this problem is to use k unit vectors in $\mathbb{R}^{k-1}$ pointing as "far apart" as possible, i.e., all enclosing the same angle. As can easily be verified, the solution to this is a system of vectors with the pairwise scalar product $-1/(k-1)$ (see e.g., [33]). This corresponds to an enclosed angle of $\arccos(-1/k-1)$ radians, and so geometrically these vectors form an equilateral $(k-1)$-dimensional simplex.

Now, let $\{\mathbf{s}_1, \dots, \mathbf{s}_k\}$ be the vectors of the equilateral $(k-1)$-dimensional simplex. Then MAX-k-CUT can be phrased as the following maximization problem:

$$\begin{aligned} \max \quad & \sum_{vw \in E(G)} \frac{k-1}{k} \left(1 - \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle\right) \\ \text{s.t.} \quad & \mathbf{x}_v \in \{\mathbf{s}_1, \dots, \mathbf{s}_k\} \quad \forall v \in V. \end{aligned}$$

The semidefinite relaxation resulting from this formulation gives an upper bound on MAX-k-CUT. We will denote it by $\mathcal{SDP}_k$:

$$\begin{aligned} \max \quad & \sum_{vw \in E(G)} \frac{k-1}{k} \left(1 - \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle\right) \\ \text{s.t.} \quad & \|\mathbf{x}_v\| = 1 && \forall v \in V, \\ & \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle \geq -\frac{1}{k-1} && \forall v, w \in V. \end{aligned}$$

In order to construct an approximation algorithm from this SDP, analogous to the MAXCUT algorithm of Goemans and Williamson, it remains to replace the random hyperplane in the rounding strategy. For this, Frieze and Jerrum used k random vectors. The partitions of the k-cut are then formed by assigning each vector of a solution of $\mathcal{SDP}_k$ to the random vector with which it has minimal scalar product.

For $k = 3$, Goemans and Williamson [27] proved that the algorithm resulting from this approach has an approximation ratio of 0.836008. For comparison, observe that a greedy algorithm achieves a $(1 - 1/k)$-approximation for MAX-k-CUT. This also resembles the size of a random k-cut.

4.4 Coloring

As we will see by the end of this section, the MAX-k-CUT semidefinite program defined above also plays an important role for the k-coloring problem on graphs. But before turning to this aspect, we will describe an independent semidefinite programming approach to approximative coloring developed by Karger, Motwani, and Sudan [37].

The vector chromatic number was introduced in Section 3.1. By the definition given there, determining this parameter for a graph $G = ([n], E)$ corresponds to the following minimization problem:

$$\min \quad k \quad \text{s.t.} \quad \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle \leq -\frac{1}{k-1} \quad \forall vw \in E, \quad \|\mathbf{x}_v\| = 1 \quad \forall v \in [n]. \tag{4.2}$$

This is the vector formulation of a semidefinite program and a solution can therefore be efficiently calculated. In fact, Karger, Motwani, and Sudan showed that computing this minimum

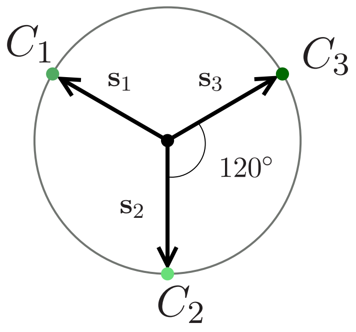

Figure 4.1: *An optimal solution for $\mathcal{SDP}_3$ on 3-colorable graphs with color classes C_1, C_2, C_3 using no more than 2 dimensions*

is equivalent to solving the matrix optimization problem

$$\begin{aligned} \min \quad & \kappa \\ \text{s.t.} \quad & x_{vw} \le \kappa \quad \forall vw \in E, \\ & \operatorname{diag}(\mathbf{X}) = \mathbf{1}, \\ & \mathbf{X} \succeq 0 \end{aligned}$$

where $k = (\kappa - 1)/\kappa$ and $\mathbf{X} := (x_{vw})_{1 \le v,w \le n}$.

Moreover, if a graph G is k-colorable, then the equilateral $k-1$ dimensional simplex encountered in the last section also provides a feasible solution to (4.2): Map each color class to one of the k vectors generating the simplex and assign to all vertices the vector corresponding to their color class. As a consequence, a graph is vector k-colorable if it is k-colorable.

Inspired by this relation, Karger, Motwani, and Sudan iteratively applied the semidefinite program above to obtain an approximative graph coloring. The main idea is, again, a rounding technique, namely to represent the k color classes by k random vectors. But contrary to the approach of Frieze and Jerrum to MAX-k-CUT, vertices v from G are now only mapped to a color c if the scalar product between the vector assigned to v and the random vector corresponding to c does not fall below a certain value and if no conflicting color assignments are produced in this way. The algorithm then proceeds with the graph induced by the remaining uncolored vertices and uses new colors.

As indicated above, the equilateral $k-1$ dimensional simplex does not only give rise to a feasible solution of (4.2), but also of the MAX-k-CUT relaxation $\mathcal{SDP}_k$. Indeed, one way to realize a feasible solution of $\mathcal{SDP}_k$ corresponding to a k-cut $C_1, C_2, \ldots, C_k$ of G is to assign the same vector $\mathbf{s}_i$ to each vertex in C_i in such a way that $\langle \mathbf{s}_i \,|\, \mathbf{s}_j \rangle = -1/(k-1)$ for $i \neq j$. For $k = 3$ this solution is depicted in Figure 4.1.

Observe further that each edge of G contributes some value between 0 and 1 to $\mathcal{SDP}_k$. So $\mathcal{SDP}_k(G_1) \le \mathcal{SDP}_k(G_2)$ whenever $G_1 \preceq G_2$. Moreover, there is a rather obvious connection between maximum k-cuts and k-colorings: In the case of a k-colorable graph G a maximum k-cut simply contains all edges. Then, we know that each edge contributes exactly 1 to the value of $\mathcal{SDP}_k$. In this case, the special feasible solution to $\mathcal{SDP}_k$ discussed above is optimal.

Conversely, if the optimal solution of $\mathcal{SDP}_k(G)$ possesses this structure, then it is clearly easy to read off a proper k-coloring of G from this solution. Unfortunately, the position of the vectors $\mathbf{x}_v$ can get "far away" from this ideal picture in general. Next, we will give an example to illustrate this fact for $k = 3$.

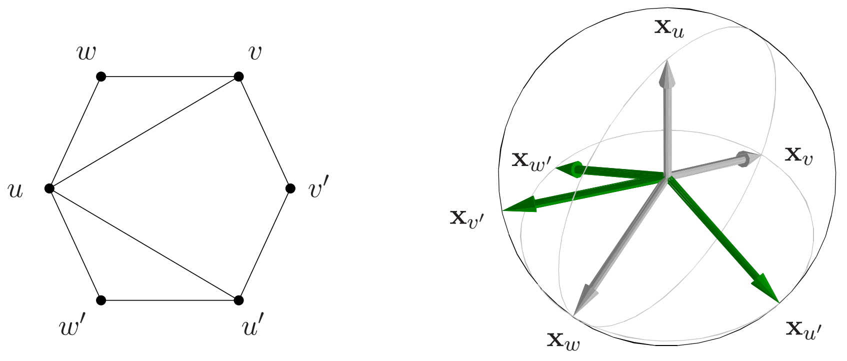

Figure 4.2: *A 3-dimensional optimal solution for $\mathcal{SDP}_3$ on a 3-colorable graph*

Example:

Figure 4.2 shows a 3-colorable graph $G = (V, E)$ on the 6 vertices u,v,w,u',v', and w' and an optimal solution $\mathcal{X} = (\mathbf{x}_i)_{i \in V}$ for $\mathcal{SDP}_3(G)$, that uses 3 dimensions. Since the vertices u, v, and w form a triangle in G, the vectors corresponding to this triple have to lie on a disc as indicated by the grey circle through $\mathbf{x}_u$, $\mathbf{x}_v$, and $\mathbf{x}_w$ in Figure 4.2. In addition all other pairs of vectors corresponding to the vertices of an edge of G also enclose an angle of $120°$. One way to obtain an optimal solution for $\mathcal{SDP}_3$ is to set $\mathbf{x}_{u'} = \mathbf{x}_w$, $\mathbf{x}_{v'} = \mathbf{x}_u$, and $\mathbf{x}_{w'} = \mathbf{x}_v$. However, there are other optimal solutions. In Figure 4.2 we construct one where the 6 vectors are distributed over the unit sphere in such a way that determining 3 color classes from this solution by just evaluating pairwise scalar products might produce errors: First, $\mathbf{x}_v$, $\mathbf{x}_u$, and $\mathbf{x}_w$ are fixed. Then $\mathbf{x}_{u'}$ is chosen to be the vector that encloses an angle of $120°$ with $\mathbf{x}_u$, but has maximal difference to $\mathbf{x}_v$ and $\mathbf{x}_w$, i.e., $\mathbf{x}_{u'}$ encloses an angle of $\arccos(5/8)$ with each of them. Now, $\mathbf{x}_{v'}$ is forced to be either identical with $\mathbf{x}_u$ or it is the vector obtained by reflecting $\mathbf{x}_u$ at the plane spanned by $\mathbf{x}_v$ and $\mathbf{x}_{u'}$ (since v' also is adjacent to both, v and u'). We choose the second possibility. Finally, $\mathbf{x}_{w'}$ can be constructed by reflecting $\mathbf{x}_{u'}$ at the plane through $\mathbf{x}_u$, $\mathbf{x}_v$, and $\mathbf{x}_w$.

It is now easy to verify that $\mathcal{X}$ obeys the constraints of $\mathcal{SDP}_3$ and is optimal. In short, while the relative position of the vectors $\mathbf{x}_u$, $\mathbf{x}_v$, and $\mathbf{x}_w$ is fixed up to rotation, the edges outside the triangle formed by the corresponding vertices do not force the remaining vectors to coincide with these three color class representatives (here, we say that $\mathbf{x}_u$, $\mathbf{x}_v$, and $\mathbf{x}_w$ represent the color classes, since in any valid coloring, u, v, and w will receive pairwise distinct colors). $\diamond$

In Section 5.3 we show however that with high probability such a scenario does not occur in the case of random 3-colorable graphs from $\mathcal{G}_{n,p,3}$. Although for such graphs the vectors corresponding to vertices of one color class do not necessarily need to be equal, most of them will be comparably close. As was pointed out by Coja-Oghlan [14] in a different context the main reason for this behaviour is that $\mathcal{G}_{n,p,3}$ is likely to have good expansion properties.

This fact will be of particular importance in the initial phase of the algorithm COLOR$\mathcal{G}_{n,p,3}$ that is presented in the oncoming main part of this text.

5 An Algorithm for Coloring $\mathcal{G}_{n,p,3}$ in Polynomial Expected Time

"I can't work without a model. I won't say I turn my back on nature ruthlessly in order to turn a study into a picture, arranging the colors, enlarging and simplifying; but in the matter of form I am too afraid of departing from the possible and the true."

Vincent Van Gogh

5.1 The Algorithm COLOR$\mathcal{G}_{n,p,3}$

In this section we introduce the algorithm COLOR$\mathcal{G}_{n,p,3}$ that colors a graph from $\mathcal{G}_{n,p,3}$ with 3 colors in polynomial expected running time if $p := d/n$ and d is bounded from below by some large but fixed constant c. More specifically, the algorithm aims at determining the original partitions C_1, C_2, and C_3 of $\mathcal{G}_{n,p,3}$, but might occasionally come up with an alternative solution. As we noted earlier, this algorithm is partly based on methods and ideas developed by Alon and Kahale [2].

We will start by giving an informal description of the algorithm. An exact formulation is then presented in the form of pseudo-code (see page 27). In the informal description, we refer to the lines of this code where appropriate. A detailed discussion is, however, delayed until the analysis of COLOR$\mathcal{G}_{n,p,3}$'s running time complexity.

When we explain the mechanisms of COLOR$\mathcal{G}_{n,p,3}$, we will often state that a particular step results in a (partial) coloring of G with certain properties with *sufficiently high probability*. By this, we mean that these probabilities are small enough to allow for the use of exponential time methods in case the mentioned properties do not hold, while leaving the expected running time polynomial. Moreover, if a coloring with such properties is obtained on input G, we generally say that the corresponding step was *successful*, and otherwise that it *failed*, or made a *mistake*.

Roughly speaking, there are two basic principles underlying the mechanisms of COLOR$\mathcal{G}_{n,p,3}$. On the one hand a number of steps (also called the *main steps*) aim at constructing a valid 3-coloring of the input graph with sufficiently high probability. However, since we are interested in returning a valid coloring for each graph, COLOR$\mathcal{G}_{n,p,3}$ also has to take care of exceptional cases; possibly by using computationally expensive methods as a trade off. Accordingly, the purpose of the remaining operations is to fix the mistakes of the main steps. This constitutes the second principle, the so-called *recovery procedure* of COLOR$\mathcal{G}_{n,p,3}$.

In order to simplify explanations, the remainder of this section does not provide a line by line account of the algorithm, but is structured according to the two principles we just described. We start by commenting on the main steps and will then turn to the recovery procedure.

The *initial phase* (Steps 1, 5, and 6; see page 27) of COLOR$\mathcal{G}_{n,p,3}$ is concerned with finding an initial coloring Υ_0 of the input graph $G = (V, E)$ such that Υ_0 fails on at most ϵn vertices of G. Here, ϵ is a small constant that will be determined in Section 5.7. To obtain Υ_0 we apply the SDP relaxation $\mathcal{SDP}_3$ of MAX-3-CUT introduced in Section 4.2 on page 20. To be more precise, we first determine an optimal solution of this semidefinite program, i.e., an optimal assignment of vectors from $\mathbb{R}^n$ to the vertices of G. As mentioned in Chapter 4 this solution can efficiently be computed within any numerical precision. This solution then gives rise to the coloring Υ_0 by grouping vertices whose corresponding vectors have small scalar product into the same color class. This grouping process can be achieved in different ways. In the algorithm COLOR$\mathcal{G}_{n,p,3}$ we use a randomized method, that chooses three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ uniformly at random from an optimal solution of $\mathcal{SDP}_3(G)$. Each vertex v of G is then allocated to the vector $\mathbf{x}_i$ having minimal scalar product with the vector corresponding to v. This attempt is repeated a linear number of times in order to increase the probability of success.

A motivation and the details of this process are given in Section 5.3. There, we will also discuss alternative solutions to the grouping problem, e.g., a vector rounding technique, and show that Υ_0 successfully colors a large fraction of the vertices with sufficiently high probability.

The next steps of COLOR$\mathcal{G}_{n,p,3}$ are used to refine the initial coloring Υ_0. We can summarize this refinement as follows: An *iterative recoloring procedure* (Step 8) is combined with an *uncoloring procedure* (Step 9) in order to obtain a valid partial coloring on all but $\alpha_0 n$ vertices. We describe the strategy behind these two steps in the following paragraph. The constant α_0 will be specified in Section 5.4. It is much smaller than ϵ and depends on the (constant) lower bound on $d = np$.

As explained, Υ_0 correctly colors a constant proportion of the vertices w.h.p. In the iterative recoloring procedure the hope is that this initial coloring can be used to find a valid coloring of a much larger vertex set by applying the concept of majority vote. In detail, the iterative recoloring procedure repeats the following step at most a logarithmic number of times: Assign to each vertex in G the color that is the least favorite among its neighbors. In Section 5.4 we will show that this approach is indeed successful, in the sense that with sufficiently high probability at most $\alpha_0 n$ vertices are still colored incorrectly after the execution of this iterative recoloring procedure. Thus, for the aim formulated above, it remains to detect those vertices of G that could not be colored correctly up to this point. Here, the uncoloring procedure comes into play. This phase of COLOR$\mathcal{G}_{n,p,3}$ proceeds iteratively as well. In each step, those vertices of G get uncolored for which the coloring behaves "locally obscure": The uncoloring procedure uncolors all vertices that have less than $d/6$ neighbors of some color other than their own. Observe that in a "typical" graph from $\mathcal{G}_{n,p,3}$ a "typical" vertex and its neighbors will not have this property if they are colored correctly, since the average degree in $\mathcal{G}_{n,p,3}$ is $2d/3$.

If both the iterative recoloring procedure and the uncoloring procedure were carried out successfully, then all vertices that remain colored received the correct color and relatively few vertices are uncolored. COLOR$\mathcal{G}_{n,p,3}$ proceeds with Step 10. Here, an exact coloring method is used to extend the partial coloring obtained to the whole graph $\mathcal{G}_{n,p,3}$. In this process the components induced on the uncolored vertices are treated seperately. On each such component K, the algorithm tries all possible colorings until it finds one that is compatible to the coloring of the rest of G. Recall that colorings need not be valid in our terminology and notice that a coloring of K that is valid for K and compatible with the colored vertices will certainly be compatible with valid compatible colorings of other components in the uncolored vertices. In what follows, Step 10 will be referred to as the *extension step*, its analysis is provided in Section 5.6.

As indicated, the steps of the algorithm discussed so far are all we need for 3-coloring $\mathcal{G}_{n,p,3}$ with high probability (this will be formally proven in Sections 5.4 to 5.6). If no valid coloring of G could be obtained via the semidefinite program and by the recoloring, the uncoloring, and the extension step, we need to find ways of fixing the insufficiencies of these steps (as noted we call them the main steps of COLOR$\mathcal{G}_{n,p,3}$). In our algorithm, this problem is handled by the the recovery procedure, i.e., the loops in Steps 2 to 7. The concept is as follows. Assume that, for an input graph G, the main steps of COLOR$\mathcal{G}_{n,p,3}$ fail on vertex set Y. Then an easy way of "repairing" the coloring obtained is to exhaustively test all valid colorings of Y. Of course, we neither know this set Y nor its size $|Y|$. To deal with these two problems, COLOR$\mathcal{G}_{n,p,3}$ proceeds rather naively by trying all possible subsets Y of V with $|Y| = y$. Here, we start with $y = 0$ and then gradually increase the value of y until a valid coloring of G is determined. This is performed in Steps 2 and 3 of the recovery procedure. In the following we also call Step 3, where all colorings of all vertex sets of size y are constructed, the *brute force coloring method* or *repair mechanism* for these sets.

For establishing results on likely properties of the outcome of some of the main steps, it will often be necessary to make certain assumptions about the input to that step, i.e., the outcome of earlier stages of the algorithm. Therefore, the successes or failures of different steps of COLOR$\mathcal{G}_{n,p,3}$ are dependent on each other and so a mistake made in an earlier step may influence the outcome of a later step in an unpredictable way. For this reason we do not attempt to fix the coloring after executing the main steps but we already use the coloring of Y, produced by the brute force coloring method, while running the main steps. That is why these steps are nested inside the loops of the recovery procedure.

At times, the recovery procedure must make even more involved interventions into the mechanisms of the main steps. This can be observed in the pseudo-code of COLOR$\mathcal{G}_{n,p,3}$ in the form of Algorithm 5.1 given below (notice for example the additional loop in Step 7). Again, the details of these corrective actions are postponed to the analysis of the algorithm; their interplay and influence on each other is explained in Section 5.7.

Throughout Algorithm 5.1, exact coloring procedures are used (namely in Steps 3 and 10). These procedures have exponential running time in the size of their input of course and are therefore expensive as far as computational complexity is concerned. In order to leave the expected running time polynomial, this has to be compensated for by using the exact coloring procedures on rare occasions only. As we will see this is the case for COLOR$\mathcal{G}_{n,p,3}$ when the input graphs are distributed according to $\mathcal{G}_{n,p,3}$.

Since the size of Y in the recovery procedure (Step 3) is increased until a valid coloring is obtained, the correctness of Algorithm 5.1 is inherent (ultimately, a proper coloring will be found in the last iteration, when Y contains all vertices of G) . An analysis of the complexity of Algorithm 5.1 will be presented in the next few sections. An outline of the different parts of this analysis was already provided with the discussion above. For convenience we give a brief overview once more.

We start in Section 5.3 by analyzing the initial coloring Υ_0 that is produced by $\mathcal{SDP}_3$. Section 5.4 then investigates the probability that the recoloring procedure performs well on a large subgraph of G and Section 5.5 studies possible mistakes of the uncoloring procedure. In Section 5.6 we will show that Step 10, i.e., the extension step, runs in polynomial expected time on average (cf. Lemma 5.6.2). Section 5.7 finally combines the different parts of the analysis and explaines how they lead to an overall running time of the recovery procedure that is polynomial on average (see Lemma 5.7.1). Some technical preliminaries are provided

```
5.1: COLOR$\mathcal{G}_{n,p,3}$(G)
Input: a graph $\mathcal{G}_{n,p,3} = G = (V, E)$
Output: a valid coloring of $\mathcal{G}_{n,p,3}$
begin
1   compute an optimal solution $(\mathbf{x}_v)_{v \in V(G)}$ to $\mathcal{SDP}_3(G)$ with sufficient numerical precision;
2   for $0 \le y \le n$ do
3       foreach vertex set $Y$ of size $y$ and each valid 3-coloring $\Upsilon_Y$ of $Y$ do
4           for $\mathcal{O}(n)$ times do
                /**   The initial phase   **/
5               Randomly choose three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \{\mathbf{x}_v \mid v \in V(G)\}$ ;
6               Extend $\Upsilon_Y$ to a coloring $\Upsilon_0$ of $G$ by setting $\Upsilon_0(v) := i$ for all $v \in G - Y$ where
                $i$ is such that $\langle \mathbf{x}_v \,|\, \mathbf{x}_{v_i} \rangle$ is maximal ;
7               for  $t = \log n$ downto $t = 0$ do
                    /**   The iterative recoloring procedure   **/
8                   for  $0 \le s < t$ do
                        Construct a coloring $\Upsilon_{s+1}$ of $G$ with $\Upsilon_{s+1}(v) := \Upsilon_Y(v)$ for $v \in Y$ and
                        $\Upsilon_{s+1}(v) := i$ for $v \notin Y$ where $i$ minimizes $|\mathbf{N}(v) \cap \Upsilon_s^{-1}(i)|$ ;
                    Set $\Upsilon' := \Upsilon_t$ ;
                    /**   The uncoloring procedure   **/
9                   while  $\exists v \in G - Y$ with $\Upsilon'(v) = i$ and $|\{w \mid w \in \mathbf{N}(v) \wedge \Upsilon'(w) = j\}| < d/6$
                    for some $j \neq i$ do
                        uncolor $v$ in $\Upsilon'$ ;
                    /**   The extension step   **/
10                  if each component of uncolored vertices is of size at most $\min(y, \alpha_0 n)$ then
                        Extend the partial coloring $\Upsilon'$ to a coloring $\Upsilon$ of $G$ by exhaustively
                        trying each coloring of each component in the set of uncolored vertices ;
11                      if $\Upsilon$ is a valid coloring of $G$ then
                            return $\Upsilon(G)$ and stop;
end
```

in Section 5.2.

Apart from the recovery procedure and the extension step, all other steps of Algorithm 5.1, i.e., Steps 1, 6, 8, 9, can obviously be executed in polynomial time. Together with the analysis of the expected running time of the recovery procedure and the extension step (cf. Sections 5.7 and 5.6, respectively) this implies the following theorem.

Theorem 4
Algorithm 5.1 finds a 3-coloring of a graph from $\mathcal{G}_{n,p,3}$ in polynomial expected running time if $d = pn > c$ for some sufficiently large constant c.

As mentioned, the details of Algorithm 5.1 that were not explained so far, such as the significance of t and the constant α_0, will become clear in the subsequent sections. In the following we usually let $y =: \alpha n$ for simplifying calculations. Here, y is the variable from Step 3 in Algorithm 5.1 and α might be a function in n.

5.2 Preliminaries

The analysis presented in the following sections is technical in nature, mainly relying on methods from probability theory and elementary analysis. As preparation, we will introduce some basic tools for bounding probabilities and provide a number of simple observations in this section.

The binary entropy function

The *binary entropy function* $\mathrm{H}(x)$ is defined for $x \in (0,1)$ by the relation

$$\mathrm{H}(x) := -x \log x - (1-x)\log(1-x).$$

This function is useful for establishing bounds on binomial coefficients as we will see shortly. But first we explain how to bound $\mathrm{H}(x)$ itself. For this, note that $-x\log x$ has a unique local maximum at $x = 1/2$ and that $-(1/4)\log(1/4) > -(1-1/4)\log(1-1/4)$. By symmetry it follows that

$$\mathrm{H}(x) \le -2 \cdot x \log x \qquad \text{for} \quad x \le \frac{1}{2}. \tag{5.1}$$

Bounds on binomial coefficients

When estimating the probabilities of events for combinatorial objects one often encounters binomial coefficients. These coefficients can either be bounded with the help of

$$\left(\frac{a}{b}\right)^b \le \binom{a}{b} \le \left(e \cdot \frac{a}{b}\right)^b \tag{5.2}$$

or by applying the entropy function defined above

$$\binom{n}{xn} \le \exp(\mathrm{H}(x)\cdot n) \le \exp(-2x\log x \cdot n)\,. \tag{5.3}$$

Here, we generally assume that xn is integral, although this inequality also remains valid for a generalized form of binomial coefficients.

In the case that standard probability distributions, such as the binomial distribution, need to be evaluated, more sophisticated tools are at hand.

Markov and Chernoff bounds

Let X be a non-negative random variable with expectation $\mathbf{E}[\,X\,]$. Then *Markov's inequality* asserts that the following holds true for all $t > 0$:

$$\mathbf{P}[\,X \ge t\,] \le \frac{\mathbf{E}[\,X\,]}{t}. \tag{5.4}$$

Upper and lower tails of binomially distributed variables may be estimated with the help of *Chernoff bounds* (see e.g., [34], Chapter 2). In our analysis we will use the following example. For $X \in \mathrm{Bin}(n,p)$ the expectation $\mathbf{E}[\,X\,]$ is given by $\lambda = np$. Now, if $t \ge 0$, then

$$\mathbf{P}[\,X \ge \mathbf{E}[\,X\,] + t\,] \le \exp\left(-\frac{t^2}{2(\lambda + t/3)}\right) \tag{5.5}$$

$$\mathbf{P}[\,X \le \mathbf{E}[\,X\,] - t\,] \le \exp\left(-\frac{t^2}{2\lambda}\right). \tag{5.6}$$

Multi dimensional normal distributions

As indicated in Chapter 4 randomized rounding is a standard technique in connection with semidefinite programming. In Section 5.3 we will apply a strategy of this type to the semidefinite program used in the algorithm COLOR$\mathcal{G}_{n,p,3}$. For preparing the arguments involved in this rounding process, we need to recall some facts about n-dimensional normal distributions.

We say that a vector $\mathbf{r}$ is distributed according to the *n-dimensional standard normal distribution* if, independently of each other, the components of $\mathbf{r}$ are standard normally distributed. Then, $\mathbf{r}$ is also called a *random vector.*

The main reason for considering n-dimensional normal distributions in connection with n-dimensional vector spaces is their high spherical symmetry (see, e.g. [24]). This allows for simple conversions if the problem is restricted to a lower dimensional subspace. One well-known result along these lines is the fact that random vectors are in a certain sense invariant under orthogonal transformations.

Theorem 5 (Rényi [56])
The projections of an n-dimensional random vector $\mathbf{r}$ onto two lines l_1 and l_2 are independent (and normally distributed) iff l_1 and l_2 are orthogonal.

The following corollary (cf. [37]) will be usefull for rounding the solution of a semidefinite program via vector projections in section 5.3.

Corollary 5.2.1
Let $\mathbf{r} \in \mathbb{R}^n$ be a random vector and $\mathbf{u}$ be an arbitrary unit vector. Then the projection $\langle \mathbf{r} \,|\, \mathbf{u} \rangle$ of $\mathbf{r}$ along $\mathbf{u}$ is distributed according to the standard normal distribution.

Another consequence of Theorem 5 and the spherical symmetry of the n-dimensional normal distribution is the independence of projections of random vectors onto orthogonal subspaces of $\mathbb{R}^n$.

Corollary 5.2.2
Let $\mathbf{r} \in \mathbb{R}^n$ be a random vector, R an n'-dimensional subspace of $\mathbb{R}^n$, and $R^\perp$ the orthogonal complement of R in $\mathbb{R}^n$. Then the projection of $\mathbf{r}$ onto R is distributed according to an n'-dimensional standard normal distribution. Similarly, the projection of $\mathbf{r}$ onto $R^\perp$ follows an $n - n'$-dimensional standard normal distribution. Moreover, these two distributions are independent.

In addition, the n-dimensional standard normal distribution naturally gives rise to a uniform distribution on a sphere centered at the origin. More precisely, for a random vector $\mathbf{r}$ the vector $\mathbf{r}/\left\|\mathbf{r}\right\|$ is distributed uniformly over the unit sphere.

Coloring vertex sets

We denote by **3-COL** (n) the time an algorithm needs to find all valid colorings of a graph G of order n. This task can certainly be performed in 3^x steps and so we define

$$\mathbf{3\text{-}COL}\,(x) := 3^x.$$

Note that coloring all subgraphs of G on x vertices in all possible ways then takes time

$$\mathbf{3\text{-}COL}\,(x)\binom{n}{x}.$$

Some simple observations

In the observations given below we will refer to a graph $G = (V, E)$ that may be generated by $\mathcal{G}_{n,p,k}$ as well as by any other random graph model where edges are chosen independently from each other.

Observation 5.2.3
Let $a, b, c \in \mathbb{R}$ be non-negative, not necessarily constant, and V_1 and V_2 be two disjoint subsets of V such that $b \leq |V_2|$. Then

$$\mathbf{P}[\, \exists A \subset V_1, B \subset V_2 : \, |A| \geq a \wedge |B| \leq b \wedge \mathbf{e}(A, B) \geq c \cdot |A| \,] \\ = \mathbf{P}[\, \exists A \subset V_1, B \subset V_2 : \, |A| = a \wedge |B| = b \wedge \mathbf{e}(A, B) \geq c \cdot |A| \,] \,.$$

Clearly the first probability exceeds the second one. The other direction holds since the avarage size of $\mathbf{N}_{B+v}(v)$ for vertices $v \in A$ is not decreased if a vertex w with $|\mathbf{N}_{B+w}(w)\,|$ smaller than this average is removed from A. Trivially such a vertex does always exist. (Note that we need the fact that $V_1 \cap V_2 = \emptyset$ at this point.)

The next two observations are very elementary but provide useful tools that are applied throughout the calculations in the following sections. So they are stated here in order to avoid commenting on these issues much later.

Observation 5.2.4
Let $c_1, c_2, c_3 > 0$ be constant, $c'_1 > c_1$, and d be arbitrary. Then

$$c_1 \cdot d^{c_2} + c_3 \leq c'_1 \cdot d^{c_2} \tag{5.7}$$

as long as d is sufficiently large. Equivalent forms of this statement include $c_3 - c'_1 \cdot d^{c_2} \leq -c_1 \cdot d^{c_2}$, $1/(c'_1 \cdot d^{c_2} - c_3) \leq 1/(c_1 \cdot d^{c_2})$ and $c_4 \cdot \exp(-c'_1 \cdot d^{c_2}) \leq \exp(-c_1 \cdot d^{c_2})$ for $c_4 \geq 1$.

For satisfying Equation (5.7) it suffices to guarantee that $d \geq (c_3/(c'_1 - c_1))^{1/c_2}$. For the other statements this lower bound on d needs to be adjusted accordingly.

Consistent with the definition of $\mathcal{G}_{n,p,k}$, the value of $pn \geq c$ will usually be used for d when Observation 5.2.4 is applied. Here we always asume that c is chosen suitably large to justify these applications and we mention this fact only occasionally.

The following observation concerns random graphs in our context, but it could be formulated in a much more general random setting.

Observation 5.2.5
Let A be a given subset of V and $\sigma_P(A)$ be the event that some property P holds for A. Then

$$\mathbf{P}[\, \sigma_P(A) \,] \leq \mathbf{P}[\, \exists X \subset V : \, \sigma_P(X) \,] \,.$$

Analogously we could require A and X to be subsets of E.

Of course this statement can be generalized to events involving more than one set.

5.3 Finding an Initial Coloring

In this section we will analyze the first phase of COLOR$\mathcal{G}_{n,p,3}$, consisting of Steps 1 and 6 of Algorithm 5.1. In these steps an initial coloring $\Upsilon_0(G)$ is constructed for the input graph G

by using the SDP relaxation $\mathcal{SDP}_3$ of MAX-3-CUT due to Frieze and Jerrum [25] which was introduced in Section 4.3. We will restate the corresponding semidefinite program here for convenience:

$$\begin{aligned} \max \quad & \sum_{vw \in E(G)} \frac{2}{3}\left(1 - \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle\right) & \\ \text{s.t.} \quad & \|\mathbf{x}_v\| = 1 & \forall v \in V, \\ & \langle \mathbf{x}_v \,|\, \mathbf{x}_w \rangle \geq -\frac{1}{2} & \forall v, w \in V. \end{aligned}$$

Recall that the maximum runs over all vector assignments $(\mathbf{x}_v)_{v \in V(G)}$ obeying $\mathbf{x}_v \in \mathbb{R}^{|V|}$ and that for graphs G_1 and G_2 with $G_1 \preceq G_2$ we have the relation $\mathcal{SDP}_3(G_1) \leq \mathcal{SDP}_3(G_2)$.

The goal of the following analysis is to estimate the probability that the initial coloring obtained by COLOR$\mathcal{G}_{n,p,3}$ colors at least $(1-\epsilon)n$ vertices of G correctly.

In [15] Coja-Oghlan, Moore, and Sanwalani studied the behaviour of $\mathcal{SDP}_3$ on $\mathcal{G}_{n,p}$. They obtained the following result, which will be the key ingredient to our analysis of $\mathcal{SDP}_3$ on graphs from $\mathcal{G}_{n,p,3}$.

Theorem 6 (Coja-Oghlan, Moore & Sanwalani [15])
If $p \geq c/n$ for sufficiently large c then

$$\mathcal{SDP}_3(\mathcal{G}_{n,p}) \leq \frac{2}{3}\binom{n}{2}p + \mathcal{O}\left(\sqrt{n^3 p(1-p)}\right) \tag{5.8}$$

with probability at least $1 - \exp(-3n)$.

Note that $2\binom{n}{2}p/3$ is also the size of a random 3-cut in $\mathcal{G}_{n,p}$. So Theorem 6 estimates the difference of the sizes of a maximum 3-cut and a random 3-cut in $\mathcal{G}_{n,p}$.

Let $G = (V, E) \in \mathcal{G}_{n,p,3}$. Then we can construct a random graph $G^* \in \mathcal{G}_{n,p}$ from G by inserting additional edges with probability p within each color class. The following lemma investigates the effect of this process on the value of $\mathcal{SDP}_3$. Similar techniques were applied by Coja-Oghlan in [14].

Lemma 5.3.1
Consider a graph $G^* = (V, E^*)$ from $\mathcal{G}_{n,p}$ with $V = [n]$ and let $G = (V, E) \preceq G^*$ be the graph on V having edges $E = E^* \cap \{vw \,|\, \lceil 3v/n \rceil \neq \lceil 3w/n \rceil\}$. Then for some constant c' not depending on d

$$\mathcal{SDP}_3(G^*) - \mathcal{SDP}_3(G) \leq c' n\sqrt{d} \tag{5.9}$$

with probability at least $1 - \exp(-5n/2)$.

Proof:
In order to establish this result we prove that

$$\frac{2}{3}\binom{n}{2}p - \frac{c'}{2}\sqrt{n^3p} \leq \mathcal{SDP}_3(G) \leq \mathcal{SDP}_3(G^*) \leq \frac{2}{3}\binom{n}{2}p + \frac{c'}{2}\sqrt{n^3p}$$

holds with the same probability.

In fact, the second inequality holds by construction since $G \preceq G^*$ and the third inequality is asserted by Theorem 6 if we choose c' accordingly. Thus it remains to show the first inequality. This is obtained by a straightforward application of the Chernoff bound (5.6) and the fact that $\mathcal{SDP}_3(G) = |E|$ as mentioned earlier: The number of edges $|E|$ is a binomially distributed random variable with expectation $2\binom{n}{2}p/3$ and thus

$$\begin{aligned}\mathbf{P}\left[\frac{2}{3}\binom{n}{2}p - \frac{c'}{2}\sqrt{n^3p} \le |E|\right] &= \mathbf{P}\left[\mathbf{E}[\,|E|\,] - \frac{c'}{2}\sqrt{n^3p} \le |E|\right]\\ &= 1 - \mathbf{P}\left[\,|E| < \mathbf{E}[\,|E|\,] - \frac{c'}{2}\sqrt{n^3p}\right]\\ &\overset{(5.6)}{\ge} 1 - \exp\left(-\frac{{c'}^2n^3p}{2\cdot\frac{2}{3}\binom{n}{2}p}\right) > 1 - \exp\left(-{c'}^2\frac{3}{2}n\right).\end{aligned}$$

This settles the proof since we can certainly choose c' in such a way that $c' > 2$ in Theorem 6. □

Equation (5.9) asserts that the values of $\mathcal{SDP}_3$ for G and G^* are not likely to differ much, if the additional edges within the color classes C_i of G are chosen at random. It follows that in an optimal solution to $\mathcal{SDP}_3(G)$, most of the vectors corresponding to vertices of C_i for a particular i can not be far apart. This is shown in the next lemma.

If not stated otherwise, we consider $\mathcal{SDP}_3$ on input G from now on. Let $\mathcal{X} = \{\mathbf{x}_v \mid v \in V\} \subset \mathbb{R}^{|V|}$ be an optimal solution to $\mathcal{SDP}_3(G)$. Then we call

$$\mathbf{N}^{\mu}(v) := \left\{v' \in V \;\middle|\; \langle \mathbf{x}_v \,|\, \mathbf{x}_{v'}\rangle > 1 - \mu\right\} \tag{5.10}$$

the *μ-neighborhood* of v.

Lemma 5.3.2
For fixed ϵ with $0 < \epsilon < 1/2$ there is a constant $0 < \mu < 1/2$ such that for any μ' with $\mu \le \mu' < 1/2$ the following holds with probability greater than $1 - \exp(-7n/3)$: For each $i \in \{1,2,3\}$ there is a vertex $v_i \in C_i$ such that the set $\mathbf{N}^{\mu}(v_i)$ contains at least $(1-\epsilon)n/3$ vertices of C_i and the set $\mathbf{N}^{\mu'}(v_i)$ contains at most $\epsilon n/3$ vertices from other color classes.

Proof:
We first show that the first statement of Lemma 5.3.2 holds with probability at least $1 - 2\exp(-5n/2)$ for a fixed i. Then, we proceed by proving that for any pair $i \ne j$ and appropriate choices of v_i and μ' the probability that $\mathbf{N}^{\mu'}(v_i)$ contains more than $\epsilon n/6$ vertices is at most $\exp(-\Omega\,(dn))$. Since

$$3\cdot 2\cdot \exp\left(-\frac{5}{2}n\right) + 6\cdot\exp(-\Omega\,(dn)) < 7\cdot\exp\left(-\frac{5}{2}n\right) \le \exp\left(-\frac{7}{3}n\right) \tag{5.11}$$

for d and n sufficiently large this establishes the lemma (the probabilities are multiplied by the number of choices for i and j).

For proving the first part of Lemma 5.3.2 let $G_i^{\mu} = (C_i, E_i^{\mu})$ be the graph on one color class C_i of G with edge set $E_i^{\mu} := \{vw \mid \langle \mathbf{x}_v \,|\, \mathbf{x}_w\rangle \le 1 - \mu\}$ (recall that $\mathcal{X} = \{\mathbf{x}_v \mid v \in V\} \subset \mathbb{R}^{|V|}$ is an optimal solution to $\mathcal{SDP}_3(G)$). Now, consider the edges $E_i^{\mu} \cap E^*$ that this graph shares with the random graph G^* defined above. Since E_i^{μ} only depends on the optimal solution to $\mathcal{SDP}_3(G)$ and edges in $E^* \setminus E$ do not influence this solution, $E_i^{\mu} \cap E^*$ spans a random subgraph of G^*. Therefore, this edge set contains $p\,|E_i^\mu|$ edges in expectation. Moreover,

$$\begin{aligned}\mathbf{P}\Big[\,|E_i^\mu \cap E^*| \le \mathbf{E}[\,|E_i^\mu \cap E^*|\,] - 3n\sqrt{d}\,\Big] &= \mathbf{P}\Big[\,|E_i^\mu \cap E^*| \le p\cdot|E_i^\mu| - 3n\sqrt{d}\,\Big]\\ &\le \exp\Big(-\frac{3^2n^2d}{2p\cdot|E_i^\mu|}\Big)\\ &\le \exp\Big(-\frac{9n}{2}\Big)\\ &\le \exp(-5n/2)\end{aligned} \tag{5.12}$$

by the Chernoff bound (5.6).

Observe that, $\mathcal{X}$ also is a feasible (but not necessarily optimal) solution to $\mathcal{SDP}_3(G^*)$. Consequently, we can apply Lemma 5.3.1 for concluding that

$$\begin{aligned}\frac{2}{3}\mu\cdot\Big(p\,|E_i^\mu| - 3n\sqrt{d}\Big) \overset{(5.12)}{\le} \frac{2}{3}\mu\cdot|E_i^\mu\cap E^*| \;&\le\; \frac{2}{3}\sum_{vw\in E_i^\mu\cap E^*}(1-\langle \mathbf{x}_v\,|\,\mathbf{x}_w\rangle)\\ &\le\; \frac{2}{3}\sum_{vw\in E^*\setminus E}(1-\langle \mathbf{x}_v\,|\,\mathbf{x}_w\rangle) \overset{(5.9)}{\le} c'n\sqrt{d}\end{aligned}$$

holds with probability at least $1-\exp(-5n/2)-\exp(-5n/2)$ where the second inequality follows from the definition of G_i^μ. By rearranging terms and dividing by n we arrive at a conclusion about the average degree $|E_i^\mu|/n$ of G_i^μ:

$$\frac{|E_i^\mu|}{n} \le \Big(\frac{3c'}{2\mu}+3\Big)\frac{n}{\sqrt{d}} =: \frac{c''}{\sqrt{d}}\cdot\frac{n}{3}.$$

This implies that there is some vertex v_i in C_i with

$$\frac{c''}{\sqrt{d}}\cdot\frac{n}{3} \ge deg_{G_i^\mu}(v_i) = \frac{n}{3} - |\mathbf{N}^\mu(v_i)\cap C_i|\,,$$

and thus we establish the first part of Lemma 5.3.2 by choosing μ in such a way that

$$\epsilon \ge \frac{c''}{\sqrt{d}}. \tag{5.13}$$

It remains to show that at most $\epsilon n/3$ vertices of other color classes are contained in $\mathbf{N}^{\mu'}(v_i)$ given that $|\mathbf{N}^\mu(v_i)\cap C_i| \ge (1-\epsilon)n/3 \ge n/6$. Assume, for contradiction, that for some color class C_j with $i\ne j$ we have $|\mathbf{N}^\mu(v_i)\cap C_j| \ge \epsilon/2\cdot n/3$. Since each edge vw of G contributes exactly one to the optimal value of $\mathcal{SDP}_3(G)$ we know that $\langle \mathbf{x}_v\,|\,\mathbf{x}_w\rangle = -1/2$ and so $\mathbf{x}_v$ and $\mathbf{x}_w$ enclose an angle of 120°. Therefore, the set $\mathbf{N}^{\mu'}(v)\supset\mathbf{N}^\mu(v)$ induces an empty graph in G (because $\mu\le\mu'<1/2$ and $\arccos(1/2)=60°$).

Accordingly, we can bound the desired probability by calculating the probability that some vertex sets $Y_i\subset C_i$ and $Y_j\subset C_j$ exist in G with $|Y_i|\ge n/6$, $|Y_j|\ge\epsilon n/6$, and $\mathbf{e}(Y_i,Y_j)=0$:

$$\begin{aligned}&\mathbf{P}\Big[\,\exists Y_i\subset C_i, Y_j\subset C_j : |Y_i|\ge\frac{n}{6}\wedge|Y_j|\ge\epsilon\frac{n}{6}\wedge\mathbf{e}(Y_i,Y_j)=0\Big]\\ &\le \binom{\frac{n}{3}}{\frac{n}{6}}\binom{\frac{n}{3}}{\epsilon\frac{n}{6}}\Big(1-\frac{d}{n}\Big)^{\epsilon\frac{n^2}{36}} \overset{(5.3)}{\le} \exp\Big(\Big(\mathrm{H}\Big(\frac{1}{2}\Big)+\mathrm{H}\Big(\frac{\epsilon}{2}\Big)\Big)\frac{n}{3}\Big)\cdot\exp\Big(-d\cdot\epsilon\frac{n}{36}\Big)\\ &= \exp(-\Omega\,(dn))\end{aligned} \tag{5.14}$$

where the second inequality follows from $(1 - d/n)^n \leq \exp(-d)$ for $d < n$. □

In the following we will call a vertex $v \in C_i$ obeying the properties asserted by Lemma 5.3.2 a *strong* (ϵ, μ, μ')*-representative* or briefly *strong representative* for color class C_i. In the case $\mu' = \mu$ we omit the parameter μ' and say that v is a *weak* (ϵ, μ)*-representative* or just *weak representative*. Both, weak and strong representatives are *representatives*. A set or triple of *representatives for* G contains one representative for each color class.

Observe that the choice of μ in Lemma 5.3.2 depends on ϵ. But since we can get the right hand side of Equation (5.13) arbitrarily small by raising the lower bound c on d accordingly we can virtually choose any fixed μ between 0 and 0.5 for a given ϵ.

Moreover, let v_i and v_j be two strong (ϵ, μ, μ')-representatives for color classes C_i and C_j respectively with $i \neq j$. Then, Calculation (5.14) also gives a lower bound on the probability that there is no edge between $\mathbf{N}^\mu(v_i) \cap C_i$ and $\mathbf{N}^\mu(v_j) \cap C_j$. Adding this probability to the left hand side of (5.11) implies the following remark that will be useful later.

Remark 5.3.3
With probability at least $1 - \exp(7n/3)$ *there is a triple* v_1, v_2, v_3 *of strong* (ϵ, μ, μ')*-representatives such that* $\mathbf{e}(\mathbf{N}^\mu(v_i), \mathbf{N}^\mu(v_j)) \neq \emptyset$ *for all pairs* $i \neq j$.

Lemma 5.3.2 guarantees that, given an optimal solution of $\mathcal{SDP}_3(G)$, we can construct a reasonably good initial coloring Υ_0 via the sets $\mathbf{N}^\mu(v_i)$ by choosing an appropriate representative v_i for each color class C_i and setting $\Upsilon_0(v) := i$ for all $v \in N^\mu(v_i)$. Here, ties are broken arbitrarily and vertices not appearing in any of the sets $N^\mu(v_i)$ get assigned an arbitrary color.

The main difficulty with this approach is that we neither know the color classes C_i of G nor the graph G_i^μ constructed in the proof of Lemma 5.3.2 and so there is no way to determine appropriate representatives for the color classes of G. The easiest way to deal with this problem is to simply try all different triples of vertices from G as representatives. This, however, introduces an extra factor of n^3 in the running time.

In order to reduce this factor to a linear one, Algorithm 5.1 proceeds differently. Let v_1, v_2 and v_3 be strong (ϵ, μ, μ')-representatives of the color classes C_1, C_2, and C_3, respectively, as constructed in the proof of Lemma 5.3.2. We now make use of the following observation which is a direct consequence of a more general result provided in the next section (cf. Observation 5.3.8).

Observation 5.3.4
If $\mu' > 4\mu + \sqrt{2\mu}$*, then each vertex* $v \in \mathbf{N}^\mu(v_i)$ *is a weak* $(\epsilon, 4\mu)$*-representative of color class* C_i.

By choosing $\mu' > 4\mu + \sqrt{2\mu}$ (and μ sufficiently small such that $\mu' < 1/2$) we therefore get at least $\epsilon \cdot n/3$ weak representatives per color class. But then the probability of obtaining a set of weak representatives for G by picking three vertices r_1, r_2, r_3 from V at random is at least $(1 - 3 \cdot \epsilon)/9$. Repeating this process raises the probability of success. More specifically, the probability that in $c'n$ trials none yields a triple of weak $(\epsilon, 4\mu)$-representatives is smaller than $(8/9 + \epsilon/3)^{c'n}$. Here, $c' > 0$ is an arbitrary constant. Observe additionally that if r_1, r_2, r_3 indeed form a triple of weak representatives for G, then $\langle \mathbf{x}_v \,|\, \mathbf{x}_{r_i} \rangle > \langle \mathbf{x}_v \,|\, \mathbf{x}_{r_j} \rangle$ for $v \in N^{4\mu}(v_i)$ and $i \neq j$. Therefore, we then get a coloring Υ_0 of G that colors at least $\epsilon \cdot n$ vertices of G correctly by assigning each vertex v the color i such that $\langle \mathbf{x}_v \,|\, \mathbf{x}_{r_i} \rangle$ is maximal.

The strategy just described is applied in Step 4 of Algorithm 5.1. We conclude that also this randomized approach gives rise to a valid coloring of an $(1-\epsilon)$-fraction of the graph with probability at least

$$1 - \exp\left(-\frac{7}{3}n\right) - (8/9+\epsilon/3)^{c'n} \geq 1 - 2\exp\left(-\frac{7}{3}n\right) \geq 1 - 10^{-n} \tag{5.15}$$

for c' and n sufficiently large and ϵ small enough, e.g. $c' = 30$ and $\epsilon < 1/10$. Here, the probability is taken with respect to the input graphs $\mathcal{G}_{n,p,k}$ and to the random choices of representatives.

In the following subsection we will present even a different way of obtaining this initial coloring by using rounding via vector projections. We investigate this technique since it is standard to the field of semidefinite programming. As we will show, this method achieves a similar success probability.

First however, we state this section's main result which is an immediate corollary of the foregoing remarks on a randomized selection of the representatives of G in combination with Equation (5.15) and Lemma 5.3.2. For this pupose, consider the coloring Υ_0 constructed in Step 6 of Algorithm 5.1 and let F_{SDP} be a vertex set of minimal cardinality such that Υ_0 colors at most $\epsilon n/3$ vertices incorrectly in each set $C_i \setminus F_{SDP}$.

Corollary 5.3.5
For all $y > 0$ the following relation holds true:

$$\mathbf{P}[\,|F_{SDP}| \geq 1\,] \leq 10^{-n}.$$

On the rare occasion that $F_{SDP} \neq \emptyset$ the recovery procedure of Algorithm 5.1 is responsible for coloring the vertices in F_{SDP} correctly (cf. Section 5.7). This implies the following statement.

Remark 5.3.6
In the following analysis, we can assume that less than ϵn vertices of G are colored incorrectly after the initial phase of Algorithm 5.1.

5.3.1 Rounding the SDP Solution

In this section we describe a rounding technique for an optimal solution to the semidefinite program $\mathcal{SDP}_3(G)$. This technique then allows for a randomized construction of an initial coloring that is valid on at least $(1-\epsilon)n$ vertices of G. One principle that is common to all the rounding strategies for semidefinite programs mentioned in the introduction is the usage of random vectors in order to subdivide the solution space into different regions. This paradigm will be applied here as well.

The main idea is simple but effective. Let $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ be the vectors corresponding to strong representatives of the color classes C_1, C_2, and C_3 respectively, as constructed in the proof of Lemma 5.3.2. In the rounding process, we will choose three random unit vectors $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$. With constant probability these vectors will be "close" to $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ respectively (see Lemma 5.3.11). Here, by "close" we mean that the following process yields a correct classification for at least $(1-\epsilon)n$ vertices of G: We construct Υ_0 by assigning color i to those $n/3$ vertices having maximal scalar product to $\mathbf{r}_i$ (again, ties are broken arbitrarily).

As desired, it follows that Υ_0 fails on at most ϵn vertices with probability $1 - 10^{-n}$ in at least one trial if this process is repeated a sufficiently big but linear number of times.

The analysis of this rounding method is split into two steps. First, we will study geometric properties of $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$. This allows us then to compute the probability that the rounding method is indeed successful.

Geometrical Considerations

As indicated, for arguing how to use the properties of $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ in the rounding process we first need to investigate their relative positions in $\mathbb{R}^n$. Here, our goal is to construct a set of alternative representatives $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, $\hat{\mathbf{x}}_3$ from $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ that obey a geometrical structure that is completely symmetric: each pair $\hat{\mathbf{x}}_i$, $\hat{\mathbf{x}}_j$ of these vectors is required to include exactly an angle of 120°. Note that this implies that $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$ all lie on one plane $[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3]$.

For obtaining the desired vectors we proceed as follows: Since already $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ almost obey this structure, we can choose vectors in the neighborhood of these original representatives as $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$. It then remains to show that the new vectors actually serve as representatives for the color classes C_1, C_2, and C_3, in the sense that $\langle \hat{\mathbf{x}}_i \,|\, \mathbf{x}_v \rangle$ is comparably big for most vertices v in C_i and small for most other vertices.

This will be formulated more accurately in Lemma 5.3.9 below. But before we can state this lemma we need to complement existing conventions. We extend the definition (5.10) of the *μ-neighborhood* $\mathbf{N}^\mu(v)$ to arbitrary vectors $\mathbf{x} \in \mathbb{R}^n$ in the obvious way

$$\mathbf{N}^\mu(\mathbf{x}) := \{ v \in V \mid \langle \mathbf{x} \,|\, \mathbf{x}_v \rangle > 1 - \mu \} \,.$$

Similarly a vector $\mathbf{x}$ is called a *strong* (ϵ, μ, μ')*-representative* of color class C_i if at least $(1-\epsilon)n/3$ vertices of C_i are contained in $\mathbf{N}^\mu(\mathbf{x})$ and at most $\epsilon n/6$ vertices of any other color class are contained in $\mathbf{N}^{\mu'}(\mathbf{x})$. If $\mu = \mu'$, then we also say that $\mathbf{x}$ is a *weak* (ϵ, μ)*-representative.*

The following observation will be useful for determining the difference between a unit vector $\mathbf{x}$ and another unit vector $\mathbf{x}'$ from its μ-neighborhood.

Observation 5.3.7
Let $\mathbf{x}$, $\mathbf{x}'$, $\mathbf{x}''$, $\mathbf{y}$, and $\mathbf{y}'$ be n-dimensional unit vectors such that $\mathbf{x}', \mathbf{x}'' \in \mathbf{N}^{\mu_1}(\mathbf{x})$ and $\mathbf{y}' \in \mathbf{N}^{\mu_2}(\mathbf{y})$ with $0 < \mu_1, \mu_2 < 1$. If moreover $\mathbf{x}' =: \mathbf{x} + \mathbf{x}^{\mu_1}$ and $\mathbf{y}' =: \mathbf{y} + \mathbf{y}^{\mu_2}$, then

1. $\|\mathbf{x}^{\mu_1}\| \le \sqrt{2\mu_1}$,
2. $\mathbf{x}'' \in \mathbf{N}^{4\mu_1}(\mathbf{x}')$, *and*
3. $|\, \langle \mathbf{x}' \,|\, \mathbf{y}' \rangle - \langle \mathbf{x} \,|\, \mathbf{y} \rangle \,| \le \sqrt{2\mu_1} + \sqrt{2\mu_2} + 2\sqrt{\mu_1\mu_2}$.

Indeed, by evaluating the norm of $\mathbf{x}'$ we find

$$1 = \left\|\mathbf{x}'\right\|^2 = \|\mathbf{x} + \mathbf{x}^{\mu_1}\|^2 = \langle \mathbf{x} + \mathbf{x}^{\mu_1} \,|\, \mathbf{x} + \mathbf{x}^{\mu_1} \rangle = 1 + \langle \mathbf{x}^{\mu_1} \,|\, \mathbf{x}^{\mu_1} \rangle + 2 \langle \mathbf{x} \,|\, \mathbf{x}^{\mu_1} \rangle$$

and so

$$\langle \mathbf{x}^{\mu_1} \,|\, \mathbf{x}^{\mu_1} \rangle = -2 \langle \mathbf{x} \,|\, \mathbf{x}^{\mu_1} \rangle = -2 \left\langle \mathbf{x} \,\middle|\, \mathbf{x}' - \mathbf{x} \right\rangle = -2(\left\langle \mathbf{x} \,\middle|\, \mathbf{x}' \right\rangle - 1) < -2(1 - \mu_1 - 1) = 2\mu_1.$$

Similarly, $\langle \mathbf{y}^{\mu_2} \,|\, \mathbf{y}^{\mu_2} \rangle < 2\mu_2$. In addition, the angle enclosed by $\mathbf{x}'$ and $\mathbf{x}''$ is certainly less than $2 \cdot \arccos(1 - \mu_1)$ since these vectors are both in $\mathbf{N}^{\mu_1}(\mathbf{x})$. It follows that

$$\left\langle \mathbf{x}' \,\middle|\, \mathbf{x}'' \right\rangle \ge \cos(2 \cdot \arccos(1 - \mu_1)) = 2 \cdot \cos^2(\arccos(1 - \mu_1)) - 1 = 1 - 4\mu_1 + 2\mu_1^2 \ge 1 - 4\mu_1$$

implying $\mathbf{x}'' \in \mathbf{N}^{4\mu_1}(\mathbf{x}')$. The third relation is a consequence of the Cauchy-Schwarz inequality which asserts $|\langle \mathbf{x}^{\mu_1} \,|\, \mathbf{y}\rangle| \leq \|\mathbf{x}^{\mu_1}\| \cdot \|\mathbf{y}\| \leq \sqrt{2\mu_1}$, $|\langle \mathbf{x} \,|\, \mathbf{y}^{\mu_2}\rangle| \leq \sqrt{2\mu_2}$ and

$$|\langle \mathbf{x}^{\mu_1} \,|\, \mathbf{y}^{\mu_2}\rangle| \leq \|\mathbf{x}^{\mu_1}\| \cdot \|\mathbf{y}^{\mu_2}\| \leq 2\sqrt{\mu_1\mu_2}.$$

Thus,

$$\begin{aligned} \big|\big\langle \mathbf{x}' \,\big|\, \mathbf{y}'\big\rangle - \langle \mathbf{x} \,|\, \mathbf{y}\rangle \big| &= |\langle \mathbf{x} + \mathbf{x}^{\mu_1} \,|\, \mathbf{y} + \mathbf{y}^{\mu_2}\rangle - \langle \mathbf{x} \,|\, \mathbf{y}\rangle| \\ &= |\langle \mathbf{x}^{\mu_1} \,|\, \mathbf{y}\rangle + \langle \mathbf{x} \,|\, \mathbf{y}^{\mu_2}\rangle + \langle \mathbf{x}^{\mu_1} \,|\, \mathbf{y}^{\mu_2}\rangle| \leq \sqrt{2\mu_1} + \sqrt{2\mu_2} + 2\sqrt{\mu_1\mu_2}. \end{aligned}$$

For $\mathbf{y}' = \mathbf{y}$ and $\mathbf{y} \notin \mathbf{N}^{\mu_1'}(\mathbf{x})$ for some μ_1', this asserts

$$\big\langle \mathbf{x}' \,\big|\, \mathbf{y}\big\rangle \leq \langle \mathbf{x} \,|\, \mathbf{y}\rangle + \sqrt{2\mu_1} \leq 1 - \mu_1' + \sqrt{2\mu_1}$$

and so $\mathbf{x}' \notin \mathbf{N}^{\mu_1' - \sqrt{2\mu_1}}(\mathbf{x})$. In conjunction with the second statement of Observation 5.3.7, this immediately implies the following result.

Observation 5.3.8
Let $\mathbf{x}$ be a strong (ϵ, μ, μ')-representative of G with $\mu' > 4\mu + \sqrt{2\mu}$. If $\hat{\mathbf{x}} \in \mathbf{N}^{\mu}(\mathbf{x})$ then $\hat{\mathbf{x}}$ is a strong $(4\mu, \mu' - \sqrt{2\mu}, \epsilon)$-representative of G.

Note that this observation contains Observation 5.3.4 as a special case.

We are now ready to construct the vectors $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$.

Lemma 5.3.9
For μ sufficiently small the constant μ' can be chosen in such a way that for each triple $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ of strong (ϵ, μ, μ')-representatives there are three vectors $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3 \in \mathbb{R}^n$ with $\|\hat{\mathbf{x}}_1\| = \|\hat{\mathbf{x}}_2\| = \|\hat{\mathbf{x}}_3\| = 1$ such that $\langle \hat{\mathbf{x}}_i \,|\, \hat{\mathbf{x}}_j\rangle = -1/2$ for $i \neq j$ and each $\hat{\mathbf{x}}_i$ is a weak (ϵ, μ')-representative for C_i.

Proof:
By Remark 5.3.3, we can assume that for each pair of indices $i \neq j$ there are vectors $\mathbf{x}_{i \to j} \in \mathbf{N}^{\mu}(\mathbf{x}_i)$ and $\mathbf{x}_{j \to i} \in \mathbf{N}^{\mu}(\mathbf{x}_j)$ such that $\langle \mathbf{x}_{i \to j} \,|\, \mathbf{x}_{j \to i}\rangle = -1/2$. We will use these vectors for constructing a triple $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3$ of representatives pairwise enclosing an angle of 120°. For this, choose $\hat{\mathbf{x}}_1 := \mathbf{x}_{1 \to 2}$, $\hat{\mathbf{x}}_2 := \mathbf{x}_{2 \to 1}$ and $\hat{\mathbf{x}}_3 := -\hat{\mathbf{x}}_1 - \hat{\mathbf{x}}_2$. By construction, $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3$ are unit vectors and enclose the desired angles. It remains to demonstrate that they are weak (ϵ, μ')-representatives for C_1, C_2, and C_3, respectively.

Here, the basic idea is to use the fact that $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ are "close" to each other for each $i \in \{1, 2, 3\}$. While this property is asserted for $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ by $\hat{\mathbf{x}}_1 \in \mathbf{N}^{\mu}(\mathbf{x}_1)$ and $\hat{\mathbf{x}}_2 \in \mathbf{N}^{\mu}(\mathbf{x}_2)$ it is not immediately obvious for $\hat{\mathbf{x}}_3$. Below however, we will prove that a similar relation holds for this vector, namely that $\hat{\mathbf{x}}_3 \in \mathbf{N}^{10\sqrt{\mu}}(\mathbf{x}_3)$. For establishing the lemma it therefore suffices to show that for a strong (ϵ, μ, μ')-representative $\mathbf{x}$ and appropriate μ' each vector $\hat{\mathbf{x}} \in \mathbf{N}^{10\sqrt{\mu}}(\mathbf{x}) \supset \mathbf{N}^{\mu}(\mathbf{x})$ is a weak (ϵ, μ')-representative.

This is obtained as a direct consequence of Observation 5.3.8. In fact, if $\mu' > 10\sqrt{\mu}$ then $\mathbf{x}$ certainly is a strong $(\epsilon, 10\sqrt{\mu}, \mu')$-representative. But then $\hat{\mathbf{x}}$ is a strong $(\epsilon, 40\sqrt{\mu}, \mu'')$-representative with

$$\mu'' := \mu' - \sqrt{20\sqrt{\mu}}$$

by Observation 5.3.8. Accordingly, we can assert that $\hat{\mathbf{x}}$ is a weak (ϵ, μ')-representative by choosing $\mu' := 2 \cdot \sqrt{20\sqrt{\mu}}$, since then

$$\mu'' = \mu' - \sqrt{20\sqrt{\mu}} = \sqrt{20\sqrt{\mu}} > 40\sqrt{\mu}$$

(and $\mu' > 10\sqrt{\mu}$) if μ is sufficiently small (i.e., $\mu < 1/6400$).

For completing the proof we need to show that $\hat{\mathbf{x}}_3 \in \mathbf{N}^{10\sqrt{\mu}}(\mathbf{x}_3)$. For this, observe that $\mathbf{x}_{1\to3} \in \mathbf{N}^{4\mu}(\hat{\mathbf{x}}_1)$ by Observation 5.3.7 since $\hat{\mathbf{x}}_1, \mathbf{x}_{1\to3} \in \mathbf{N}^{\mu}(\mathbf{x}_1)$. Similarly $\mathbf{x}_{2\to3} \in \mathbf{N}^{4\mu}(\hat{\mathbf{x}}_2)$. In addition, recall that $\mathbf{x}_{3\to1}, \mathbf{x}_{3\to2} \in \mathbf{N}^{\mu}(\mathbf{x}_3)$. The third statement of Observation 5.3.7 applied once to $\mathbf{x}_{1\to3} =: \hat{\mathbf{x}}_1 + \mathbf{x}_3^{4\mu}$ and $\mathbf{x}_{3\to1} =: \mathbf{x}_3 + \mathbf{x}_3^{\mu}$, and once to $\mathbf{x}_{1\to2} =: \hat{\mathbf{x}}_1 + \mathbf{x}_2^{4\mu}$ and $\mathbf{x}_{2\to1} =: \mathbf{x}_2 + \mathbf{x}_2^{\mu}$ then entails

$$\begin{aligned}
\langle \hat{\mathbf{x}}_3 \,|\, \mathbf{x}_3 \rangle &= -\langle \hat{\mathbf{x}}_1 \,|\, \mathbf{x}_3 \rangle - \langle \hat{\mathbf{x}}_2 \,|\, \mathbf{x}_3 \rangle \\
&\geq (\frac{1}{2} - \sqrt{2 \cdot 4\mu} - \sqrt{2 \cdot \mu} - 2 \cdot \sqrt{4\mu \cdot \mu}) + (\frac{1}{2} - \sqrt{2 \cdot 4\mu} - \sqrt{2 \cdot \mu} - 2 \cdot \sqrt{4\mu \cdot \mu}) \\
&= 1 - 3\sqrt{2\mu} - 8\mu \geq 1 - 10\sqrt{\mu},
\end{aligned}$$

since $\langle \mathbf{x}_{1\to3} \,|\, \mathbf{x}_{3\to1} \rangle = \langle \mathbf{x}_{2\to3} \,|\, \mathbf{x}_{3\to2} \rangle = -1/2$. □

By Lemma 5.3.9, we now know that the vectors $(x_v)_{v\in V}$ roughly form three well seperated clusters. The next step is to figure out how these clusters can be located in relatively few steps. As indicated earlier, our approach here is a randomized one.

Random Representatives

Let $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$ be the weak representatives for C_1, C_2, and C_3 from Lemma 5.3.9. Since we are only concerned with weak representatives in this section, we replace μ' by μ for simplifying the notation. Accordingly, $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are weak (ϵ, μ)-representatives.

In the following we consider random vectors $\mathbf{r}$, i.e., vectors where each component is independently distributed according to the standard normal distribution with mean 0 and variance 1. For such a vector, denote by $\hat{\mathbf{N}}(\mathbf{r})$ those $n/3$ vectors of $(\mathbf{x}_v)_{v\in V}$ with maximal scalar product to $\mathbf{r}$.

As explained earlier, the strategy now is to choose three random vectors $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$ and construct the initial coloring Υ_0 by assigning color i to each vertex v with $\mathbf{x}_v \in \hat{\mathbf{N}}(\mathbf{r}_i)$. This motivates the following definition. We say that a random vector $\mathbf{r} \in \mathbb{R}^n$ is a *random* (ϵ, μ)-*representative* for color class C if $\mathbf{N}^{\mu}(\hat{\mathbf{x}}) \cap \{\mathbf{x}_v \mid v \in V\} \subset \hat{\mathbf{N}}(\mathbf{r})$ where $\hat{\mathbf{x}}$ is some weak (ϵ, μ)-representative for C. Note that this implies that a triple of random representatives for G produces a valid coloring on at least $(1-\epsilon)n$ vertices.

For understanding the intuition behind this approach, consider the idealized situation that $\mathbf{x}_v = \hat{\mathbf{x}}_i$ for each $v \in C_i$. Since the (ϵ, μ)-representatives $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$ are maximally "far apart" in $\mathbb{R}^n$, a random vector $\mathbf{r}_i$ would then be a random representative for C_i iff $\hat{\mathbf{x}}_i$ and $\mathbf{r}_i$ enclose an angle of less than 60°. The probability of this event is exactly 1/3. Therefore, independently picking three random vectors and repeating this process sufficiently often w.h.p. would yield a triple of random representatives for G in at least one of these trials. As we will show only minor changes to this idea are necessary in the case that the vectors $\mathbf{x}_v$ with $v \in C_i$ do not coincide with $\hat{\mathbf{x}}_i$ exactly but are in $\mathbf{N}^{\mu}(\hat{\mathbf{x}}_i)$. In Lemma 5.3.11, we will prove that the probability for a random vector $\mathbf{r}_i$ to be a random (ϵ, μ)-representative for C_i is greater than some positive constant as desired.

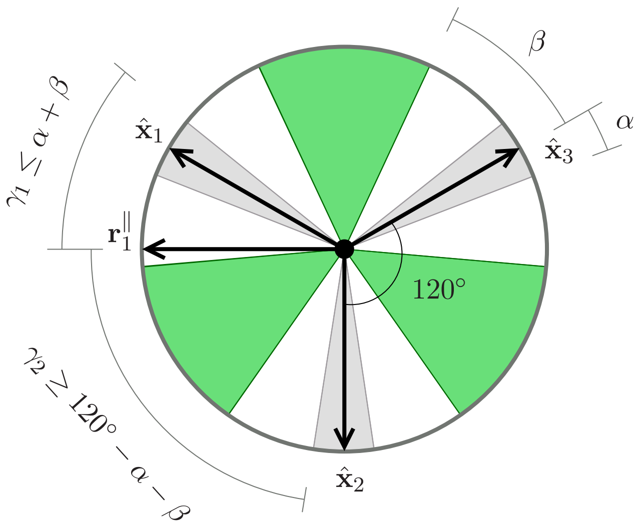

Figure 5.1: *The weak (ϵ, μ) representatives $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$ and $\hat{\mathbf{x}}_3$ and a random representative $\mathbf{r}_1$ for C_1: For all vertices $v_i \in C_i$ the angle enclosed by $\mathbf{x}_{v_i}^{\|}$ and $\hat{\mathbf{x}}_i$ is at most $\alpha \leq \arccos(1-\mu)$. The event σ_B forces the angle β enclosed by $\mathbf{r_1}^{\|}$ and $\hat{\mathbf{x}}_1$ to be such that $\beta \leq \arccos(0.9) \leq 26°$. It follows that $\gamma_1 < \gamma_2$ for μ sufficiently small.*

Recall that $[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3]$ denotes the plane spanned by the three representatives $\hat{\mathbf{x}}_1$, $\hat{\mathbf{x}}_2$, and $\hat{\mathbf{x}}_3$. For analyzing the effect of small perturbations on $\hat{\mathbf{x}}_i$, it will be useful to decompose the vectors under investigation into their projection onto this plane and a proportion perpendicular to it. Thus, for an arbitrary vector $\mathbf{y}$ let $\mathbf{y}^{\|}$ be the projection of $\mathbf{y}$ onto $[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3]$ and $\mathbf{y}^{\perp} := \mathbf{y} - \mathbf{y}^{\|}$ be the projection of $\mathbf{y}$ onto the orthogonal complement of $[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{\mathbf{x}}_3]$. Moreover, denote by $\hat{\mathbf{x}}_v$ the representative of the color class of v and let $\mathbf{x}_v =: \hat{\mathbf{x}}_v + \mathbf{x}_v^{\mu}$ and

$$V^{\mu} := \{v \mid v \in \mathbf{N}^{\mu}(\hat{\mathbf{x}}_v)\}\,.$$

The vector $\mathbf{x}_v^{\mu}$ is also called the *perturbation* of vector $\mathbf{x}_v$.

We next provide a sufficient condition for a random vector to be a random representative of G.

Observation 5.3.10
Let $C_i \in \{C_1, C_2, C_3\}$ be a color class of G and $\hat{\mathbf{x}}_i$ be the corresponding weak (ϵ, μ)-representative with $\epsilon < 1/4$ and μ sufficiently small. A random vector $\mathbf{r}_i$ is a random $(2\epsilon, \mu)$-representative for color class C_i if the following three conditions are satisfied:

1. **event $\sigma_A(\mathbf{r}_i)$:** $\|\mathbf{r}_i^{\|}\| \geq 1$,
2. **event $\sigma_B(\mathbf{r}_i)$:** $\left\langle \frac{\mathbf{r}_i^{\|}}{\|\mathbf{r}_i^{\|}\|} \,\middle|\, \hat{\mathbf{x}}_i \right\rangle \geq 0.9$,
3. **event $\sigma_C(\mathbf{r}_i)$:** $|\langle \mathbf{r}_i^{\perp} \,|\mathbf{x}_v^{\mu}\rangle| \geq 0.1$ *for at most $\epsilon \cdot n/3$ vertices $v \in V^{\mu}$.*

For this, denote by $V^{\|}(\mathbf{r}_i)$ the set of vertices v with "small" othogonal perturbation along $\mathbf{r}_i$, i.e., $|\langle \mathbf{r}_i^{\perp} \,|\, \mathbf{x}_v^{\mu} \rangle| < 0.1$. Event σ_C then asserts that there are at least $(1-2\epsilon)\cdot n/3$ vertices $v_i \in V^{\|}(\mathbf{r}_i) \cap \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i)$ with $\mathbf{x}_{v_i} \in C_i$ and at most $2\epsilon \cdot n/3$ vertices $v_j \in V^{\|}(\mathbf{r}_i) \cap \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i)$ with $\mathbf{x}_{v_j} \notin C_i$. Consequently, it suffices to show that $V^{\|}(\mathbf{r}_i) \cap \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i) \subset V^{\|}(\mathbf{r}_i) \cap \hat{\mathbf{N}}(\mathbf{r}_i)$.

Now, let $u \in V$ be an arbitrary vertex. Then the projection of $\mathbf{r}_i$ onto $\mathbf{x}_u$ is given by

$$\begin{aligned}\langle \mathbf{r}_i \,|\, \mathbf{x}_u \rangle &= \langle \mathbf{r}_i \,|\, \hat{\mathbf{x}}_u + \mathbf{x}_u^{\mu} \rangle = \langle \mathbf{r}_i \,|\, \hat{\mathbf{x}}_u \rangle + \langle \mathbf{r}_i \,|\, \mathbf{x}_u^{\mu} \rangle \\ &= \langle \mathbf{r}_i^{\|} \,|\, \hat{\mathbf{x}}_u \rangle + \langle \mathbf{r}_i^{\perp} \,|\, \hat{\mathbf{x}}_u \rangle + \langle \mathbf{r}_i^{\|} \,|\, \mathbf{x}_u^{\mu} \rangle + \langle \mathbf{r}_i^{\perp} \,|\, \mathbf{x}_u^{\mu} \rangle = \langle \mathbf{r}_i^{\|} \,|\, \hat{\mathbf{x}}_u \rangle + \langle \mathbf{r}_i^{\|} \,|\, \mathbf{x}_u^{\mu} \rangle + \langle \mathbf{r}_i^{\perp} \,|\, \mathbf{x}_u^{\mu} \rangle .\end{aligned}$$

By definition, if $u \in V^{\|}(\mathbf{r}_i)$ then $-0.1 < \langle \mathbf{r}_i^{\perp} \,|\, \mathbf{x}_u^{\mu} \rangle < 0.1$. Moreover, the first statement of Observation 5.3.7 asserts that $\|\mathbf{x}_u^{\mu}\| \leq \sqrt{\mu}$. By the Cauchy-Schwarz inequality we conclude $\langle \mathbf{r}_i^{\|} \,|\, \mathbf{x}_u^{\mu} \rangle \leq \|\mathbf{r}_i^{\|}\| \cdot \sqrt{2\mu}$. It remains to evaluate the term $\langle \mathbf{r}_i^{\|} \,|\, \hat{\mathbf{x}}_u \rangle$ for different vertices $u \in V^{\|}(\mathbf{r}_i)$.

Let $\hat{\mathbf{x}}_j \neq \hat{\mathbf{x}}_i$ be the representative of another color class of G and choose $v_i \in \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i) \cap V^{\|}(\mathbf{r}_i)$ and $v_j \in \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i) \cap V^{\|}(\mathbf{r}_i)$. Event σ_B implies that $\langle \mathbf{r}_i^{\|} \,|\, \hat{\mathbf{x}}_i \rangle \geq 0.9 \cdot \|\mathbf{r}_i^{\|}\|$. Therefore, the angle enclosed by $\mathbf{r}_i^{\|}$ and $\hat{\mathbf{x}}_i$ is at most $\arccos(0.9) < 26°$ and so $\mathbf{r}_i^{\|}$ and $\hat{\mathbf{x}}_i$ enclose an angle of at least $120° - 26° > 90°$. It follows that $\langle \mathbf{r}_i^{\|} \,|\, \hat{\mathbf{x}}_j \rangle \leq 0$ (see also Figure 5.1).

Accordingly, $\langle \mathbf{r}_i \,|\, \mathbf{x}_{v_i} \rangle \geq 0.9 \cdot \|\mathbf{r}_i^{\|}\| - 0.1 - \sqrt{2\mu} \cdot \|\mathbf{r}_i^{\|}\|$ and $\langle \mathbf{r}_i \,|\, \mathbf{x}_{v_j} \rangle \leq 0.1 + \sqrt{2\mu} \cdot \|\mathbf{r}_i^{\|}\|$. Thus, by event σ_A we have

$$\langle \mathbf{r}_i \,|\, \mathbf{x}_{v_i} \rangle - \langle \mathbf{r}_i \,|\, \mathbf{x}_{v_j} \rangle \geq (0.9 - 2\sqrt{2\mu}) \cdot \|\mathbf{r}_i^{\|}\| - 0.2 > 0$$

for μ sufficiently small and so $V^{\|}(\mathbf{r}_i) \cap \mathbf{N}^{\mu}(\hat{\mathbf{x}}_i) \subset V^{\|}(\mathbf{r}_i) \cap \hat{\mathbf{N}}(\mathbf{r}_i)$ as desired.

Note that the constants 0.9 and 0.1 in the second and the third statement of Observation 5.3.10 merely ensure that the angle between $\mathbf{r}_i^{\|}$ and $\hat{\mathbf{x}}_i$ and the perturbations along $\mathbf{r}_i^{\perp}$, are not too big. These constants may therefore be replaced by any other suitable constants.

With the help of Observation 5.3.10, we can now calculate the probability that three random vectors form a set of random representatives for G.

Lemma 5.3.11
Let $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ be n-dimensional random vectors. Then

$$\mathbf{P}[\,\mathbf{r}_1,\, \mathbf{r}_2,\, \textit{and } \mathbf{r}_3 \textit{ are random } (\epsilon,\mu)\textit{-representatives for } C_1,\, C_2, \textit{ and } C_3 \textit{ respectively}\,] \tag{5.16}$$

is greater than some positive constant for sufficiently small μ.

Proof:
Since $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ are independent random vectors, Observation 5.3.10 implies that the probability we are interested in is at least $(\mathbf{P}[\,\sigma_A(\mathbf{r}) \wedge \sigma_B(\mathbf{r}) \wedge \sigma_C(\mathbf{r})\,])^3$ where σ_A, σ_B, and σ_C are the events referred to in Observation 5.3.10 and $\mathbf{r} \in \mathbb{R}^n$ is a random vector.

The event σ_A is only concerned with the length of $\mathbf{r}^{\|}$ while σ_B only considers the direction of this vector and σ_C only considers the proportion of $\mathbf{r}$ perpendicular to $\mathbf{r}^{\|}$. By Corollary 5.2.2 the events σ_A, σ_B, and σ_C are therefore independent. Thus, it suffices to bound the probabilities $\mathbf{P}[\,\sigma_A(\mathbf{r})\,]$, $\mathbf{P}[\,\sigma_B(\mathbf{r})\,]$, and $\mathbf{P}[\,\sigma_C(\mathbf{r})\,]$ from below by some positive constants.

First we investigate σ_B. Note that since $\|\hat{\mathbf{x}}\| = 1$ this condition is equivalent to requiring an angle of at most $\arccos(0.9) \geq 18°$ between $\hat{\mathbf{x}}$ and $\mathbf{r}^{\|}$. This immediatly implies the following result

$$\mathbf{P}[\,\sigma_B\,] \geq 0.1.$$

For calculating the probability of σ_A, observe that $\|\mathbf{r}^{\|}\|^2$ is the sum of two squared standard normal distributions. It follows that $\|\mathbf{r}^{\|}\|$ is distributed according to the 2-dimensional chi distribution and so numerical evaluations[1] yield

$$\mathbf{P}[\,\sigma_A\,] \geq 0.393.$$

The probability of σ_C finally is estimated by applying a Markov bound. For this, consider $\sum_{v\in V}\langle \mathbf{r}\,|\,\mathbf{x}_v\rangle^2$. In the following we will show that this quantity usually remains small. It then follows that for most vertices v the term $\langle \mathbf{r}^{\perp}\,|\,\mathbf{x}_v\rangle^2$ cannot be large either.

Recall that $V^\mu = \{v \mid v \in \mathbf{N}^\mu(\hat{\mathbf{x}}_v)\}$. By the first statement of Observation 5.3.7 we have $\langle \mathbf{x}_v^\mu\,|\,\mathbf{x}_v^\mu\rangle \leq 2\mu$ for all vertices $v \in V^\mu$. Now, assume that $\langle \mathbf{r}\,|\,\mathbf{x}_v^\mu\rangle^2 \geq \sqrt{\mu}$ for more than ϵn vertices in V^μ. Then certainly $\sum_{v\in V^\mu}\langle \mathbf{r}\,|\,\mathbf{x}_v^\mu\rangle^2 \geq \sqrt{\mu}\cdot\epsilon\,|V^\mu|$. But

$$\sum_{v\in V^\mu}\langle \mathbf{r}\,|\,\mathbf{x}_v^\mu\rangle^2 = \sum_{v\in V^\mu}\|\mathbf{x}_v^\mu\|^2\left\langle \mathbf{r}\,\middle|\,\frac{\mathbf{x}_v^\mu}{\|\mathbf{x}_v^\mu\|}\right\rangle^2 \leq 2\mu\sum_{v\in V^\mu}\left\langle \mathbf{r}\,\middle|\,\frac{\mathbf{x}_v^\mu}{\|\mathbf{x}_v^\mu\|}\right\rangle^2$$

and $\sum_{v\in V^\mu}\langle\,\mathbf{r}\mid\mathbf{x}_v^\mu/\,\|\mathbf{x}_v^\mu\|\,\rangle^2$ is distributed according to a $|V^\mu|$-dimensional chi-squared distribution by Lemma 5.2.1. Therefore, $\mathbf{E}[\,2\mu\sum_{v\in V^\mu}\langle\,\mathbf{r}\mid\mathbf{x}_v^\mu/\,\|\mathbf{x}_v^\mu\|\,\rangle^2\,] = 2\mu\cdot|V^\mu|$ and so by Markov's inequality (5.4) we have

$$\mathbf{P}\left[\sum_{v\in V^\mu}\langle \mathbf{r}\,|\,\mathbf{x}_v^\mu\rangle^2 \geq \sqrt{\mu}\cdot\epsilon\,|V^\mu|\right] \leq \mathbf{P}\left[2\mu\sum_{v\in V^\mu}\left\langle \mathbf{r}\,\middle|\,\frac{\mathbf{x}_v^\mu}{\|\mathbf{x}_v^\mu\|}\right\rangle^2 \geq \sqrt{\mu}\cdot\epsilon\,|V^\mu|\right]$$
$$\leq \frac{2\mu\cdot|V^\mu|}{\sqrt{\mu}\cdot\epsilon\,|V^\mu|} = \frac{2\sqrt{\mu}}{\epsilon}.$$

Since we can assume $\epsilon \geq 10\sqrt{\mu}$ it follows that

$$\mathbf{P}[\,\sigma_C\,] > 1 - 0.2.$$

□

Observe that three random vectors $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are likely to have similar lengths. Therefore the rule for assigning vertices to color classes with the help of random vectors (cf. page 5.3) could easily be simplified in the following way: Assign to each vertex v the color i that minimizes $\langle \mathbf{x}_v\,|\,\mathbf{r}_i\rangle$.

5.4 Extending the Initial Coloring to a Large Subgraph H of G

After the initial coloring $\Upsilon_0(G)$ is constructed, Step 8 of Algorithm 5.1 aims at improving this coloring iteratively. In this section we show that this attempt is indeed successful on a subgraph H of G, in the sense that all vertices of H will be colored correctly after Step 8 of Algorithm 5.1 with high probability. This result will be used later for showing that the recoloring procedure can afford to color the part of G where this iterative recoloring process fails by using exact but expensive methods.

The approach outlined above can only be effective, however, if we guarantee additionally that only a very small number of vertices of G are not contained in H for most instances G.

[1]Numerical evaluations have been performed using the Mathematica software.

So, before turning to the performance of the recoloring procedure on H, we will tackle the problem of determining the expected size of H. First though, we provide explicit rules for the construction of the subgraph we will be studying.

Definition: *Let H be the subgraph of G obtained by the following process:*

1. *Delete all vertices in* $\overline{H}^{(+)} := \left\{ v \in V \,\middle|\, v \in C_i \wedge \exists j \neq i : \, deg_{C_j}(v) > (1+\delta)\frac{d}{3} \right\}$
2. *Delete all vertices in* $\overline{H}^{(-)} := \left\{ v \in V \,\middle|\, v \in C_i \wedge \exists j \neq i : \, deg_{C_j}(v) < (1-\delta)\frac{d}{3} \right\}$
3. *Iteratively delete all vertices having more than $\delta d/3$ neighbors that were deleted earlier in C_i for some i, i.e., delete all vertices in $\bigcup\limits_{0<l} \overline{H}^{(l)}$, where $\overline{H}^{(0)} := \overline{H}^{(+)} \cup \overline{H}^{(-)}$ and*

$$\overline{H}^{(l)} := \left\{ v \in V \,\middle|\, \exists i : \mathbf{N}(v) \cap C_i \cap \bigcup_{l'<l} \overline{H}^{(l')} > \delta \frac{d}{3} \right\}$$

   for $l > 0$.

We also denote $G - H$ by $\overline{H}$.

In the following, we call steps 1 and 2 of this proecess the *initial steps* and step 3 the *iterative deletion procedure*. For step l of this construction, i.e., the step when $\overline{H}^{(l)}$ is deleted, let $\overline{H}^{it} := \bigcup_{0<l'\leq l} \overline{H}^{(l')}$. In addition, $\overline{H}^{it}$ may also be used without reference to a particular step l. It then denotes $\bigcup_{0<l'} \overline{H}^{(l')}$.

Observe that H is close to being regular, more precisely

$$\{deg_H(v) \mid v \in V(H)\} \subset \left[(1-\delta)\frac{2d}{3}\,,\,(1+\delta)\frac{2d}{3} \right].$$

This implies that subsets of H are likely to have good expansion. In addition, vertices in H do not have too many neighbors outside of H by construction. These two properties will play a key role in the proofs presented below.

5.4.1 The Size of H

The choice of the parameter δ in the procedure for constructing H certainly influences the size of this graph. This parameter needs to be small enough to ensure that the iterative coloring procedure colors H correctly w.h.p. In Section 5.4.2 we will show that it suffices to choose δ small but constant for this purpose. This motivates why we assume that δ is constant in this section.

We now determine the probability that the size of H falls below a certain value. More precisely, we will not investigate the size of H directly. Instead we turn to the complement $\overline{H} = G - H$ and bound its size from above. Here, we are not interested in the absolute value of $|\overline{H}|$, but we want to know which fraction of the whole graph is taken up by $\overline{H}$. Therefore, we parametrize $|\overline{H}|$ by setting $\alpha := |\overline{H}|/n$. The lemma below gives an estimate on the probability that $|\overline{H}|$ reaches size αn.

Lemma 5.4.1
Let $0 < \alpha < \frac{1}{2}$ and $0 < \delta < \frac{1}{2}$ be constant in the definition of H. Then

$$\mathbf{P}\left[\, |\overline{H}| \geq \alpha n \,\right] \leq \exp(-\left(\log\alpha + \Omega\left(d\right)\right) \cdot \alpha n) + \exp(\Omega\left(d \cdot \log\alpha \cdot \alpha n\right)) \,. \qquad (5.17)$$

Proof:
For estimating this probability we will distinguish two cases. Either the vertices $\overline{H}^{(0)}$ deleted from H initially already contain the majority of vertices from $\overline{H}$, i.e., $|\overline{H}^{(0)}| \geq \alpha n/2$. Or $\overline{H}^{it}$ grows beyond the size of $\overline{H}^{(0)}$. For both cases, it is relatively unlikely that the number of vertices deleted in total exceeds a small constant fraction of $|V|$. Therefore, we get a low overall probability for $|\overline{H}| \geq \alpha n$ for α not too small, as desired.
Following this strategy, we obtain

$$\begin{aligned}\mathbf{P}\left[\, |\overline{H}| \geq \alpha n \,\right] &\leq \mathbf{P}\left[\, |\overline{H}| \geq \alpha n \;\middle|\; |\overline{H}^{(0)}| \geq \frac{\alpha}{2} n \right] \cdot \mathbf{P}\left[\, |\overline{H}^{(0)}| \geq \frac{\alpha}{2} n \right] \\ &\quad + \mathbf{P}\left[\, |\overline{H}| \geq \alpha n \;\middle|\; |\overline{H}^{(0)}| < \frac{\alpha}{2} n \right] \cdot \mathbf{P}\left[\, |\overline{H}^{(0)}| < \frac{\alpha}{2} n \right] \\ &\leq \mathbf{P}\left[\, |\overline{H}^{(0)}| \geq \frac{\alpha}{2} n \right] + \mathbf{P}\left[\, |\overline{H}| \geq \alpha n \;\middle|\; |\overline{H}^{(0)}| < \frac{\alpha}{2} n \right],\end{aligned}$$

and so the lemma stated above follows from

$$\mathbf{P}\left[\, |\overline{H}^{(0)}| \geq \frac{\alpha}{2} n \right] \leq \exp(-\left(\log\alpha + \Omega\left(d\right)\right) \cdot \alpha n)$$

and

$$\mathbf{P}\left[\, |\overline{H}^{it}| \geq \frac{\alpha}{2} n \;\middle|\; |\overline{H}^{(0)}| < \frac{\alpha}{2} n \right] \leq \exp(\Omega\left(d \cdot \log\alpha \cdot \alpha n\right)) \,.$$

These relations will be established subsequently in Lemma 5.4.3 and Lemma 5.4.4, respectively.
□

Before turning to the results the proof of Lemma 5.4.1 is based on, we will take a closer look at Equation (5.17). Let $c_{(0)}$ be a constant such that the expression $\log\alpha + \Omega\left(d\right)$ in this equation can be written as $\log\alpha + c_{(0)} d$. As we will see $c_{(0)}$ depends on the constant δ in the definition of H. If δ is fixed for the construction of H then we will choose $c_{(0)}$ in such a way that Lemma 5.4.1 remains valid if we reduce δ slightly to δ'. This will be needed in Section 5.6, where we employ a modification of the construction process for H. The use of δ' in the definition of $c_{(0)}$ is justified by the fact that we are interested in bounding the size of H from below in this section. This quantity increases with larger δ and so using δ' instead of δ gives a stronger condition for the choice of $c_{(0)}$.

Note further that $\log\alpha + c_{(0)} d$ gets negative for $\alpha < 2^{-c_{(0)} d}$ and consequently the bound (5.17) on $\mathbf{P}[\, |\overline{H}^{(0)}| \geq \alpha n/2 \,]$ gets trivial in this case. As we will see in Section 5.7, however, this bound is small enough for our purposes if

$$\alpha > \alpha_0 := 2^{-c_{(0)} d/10},$$

i.e., the recovery procedure can extend a coloring of H to the whole graph G in polynomial expected time if $\overline{H} > \alpha_0 n$ (cf. Lemma 5.7.4). This motivates why we use the following assumption on $\overline{H}$ in subsequent stages of Algorithm 5.1.

Remark 5.4.2
We can assume that $\overline{H} < \alpha_0 n$ whenever discussing the actions of Algorithm 5.1 on $\overline{H}$ in the following analysis.

We now return to the investigation of $|\overline{H}|$ by first considering the size of $\overline{H}^{(0)}$ and then studying the iterative deletion procedure.

The Number of Vertices Deleted in the Initial Steps

Recall that $\overline{H}^{(0)} = \overline{H}^{(+)} \cup \overline{H}^{(-)}$ is formed by those vertices having too many ($\overline{H}^{(+)}$) or too few ($\overline{H}^{(+)}$) neighbors in some color class of G other than their own. For proving the next lemma we determine the sizes of these two vertex sets separately.

Lemma 5.4.3
For $0 < \alpha < 1/2$ and constant $0 < \delta < 1/2$,

$$\mathbf{P}\left[\,|\overline{H}^{(0)}| \geq \frac{\alpha}{2}n\,\right] \exp(-\left(\log\alpha + \Omega\left(d\right)\right)\cdot \alpha n)\,.$$

Proof:
We start by considering the vertices in $\overline{H}^{(+)}$ only and give an estimate on $\mathbf{P}[\,|\overline{H}^{(+)}| \geq \alpha n/4\,]$. The corresponding probability for $\overline{H}^{(-)}$ will then be deduced from this result by an argument that uses the symmetry of these two sets with regard to $\mathbf{E}\left[\,deg_{C_i}(v)\,\right] = d/3$ for vertices $v \in C_j$, $i \neq j$.

Idea: We first calculate the probability $\overline{p}$ that a vertex in C_i has degree greater than $(1+\delta)d/3$ in some partition C_j with $i \neq j$. Then the probability distribution of $\overline{H}_i^{(+)} := |\overline{H}^{(+)} \cap C_i|$ is given by the binomial distribution $\mathrm{Bin}(\overline{p}, n/3)$.

Calculating $\overline{p}$: By symmetry we may consider a vertex $v \in C_1$ without loss of generality. For computing the desired probability we make use of $\mathbf{E}[\,deg_{C_2}(v)\,] = \mathbf{E}[\,deg_{C_3}(v)\,] = d/3$, the fact that δ is constant, and the Chernoff bound given in (5.5):

$$\begin{aligned}\overline{p} &\leq \mathbf{P}\left[\,deg_{C_2}(v) \geq (1+\delta)\frac{d}{3}\,\right] + \mathbf{P}\left[\,deg_{C_3}(v) \geq (1+\delta)\frac{d}{3}\,\right]\\ &\overset{(5.5)}{\leq} 2\exp\left(-\frac{\delta^2 d^2}{6(d+\frac{d\delta}{3})}\right) \leq 2\exp\left(-\frac{\delta^2 d}{7}\right) \overset{(5.7)}{\leq} \exp\left(-\frac{\delta^2 d}{8}\right).\end{aligned}$$

Here, we apply Equation 5.7 from Observation 5.2.4 since we can assume that $d = pn$ is sufficiently large.

Calculating $\mathbf{P}[\,|\overline{H}^{(+)}| \geq \alpha n/4\,]$: Without loss of generality, let C_1 be the partition that maximizes $\overline{H}_i^{(+)}$, i.e., if $|\overline{H}^{(+)}| \geq \alpha n/4$ then $|\overline{H}_1^{(+)}| \geq \alpha n/12$. Thus

$$\begin{aligned}\mathbf{P}\left[\,|\overline{H}^{(+)}| \geq \frac{\alpha}{4}n\,\right] &\leq 3\cdot\mathbf{P}\left[\,\overline{H}_1^{(+)} \geq \frac{\alpha n}{12}\,\right]\\ &\leq 3\binom{\frac{n}{3}}{\frac{\alpha}{4}\cdot\frac{n}{3}}\overline{p}^{\alpha n/12} \overset{(5.3)}{\leq} 3\exp\left(-\frac{\alpha}{2}\log\frac{\alpha}{4}\cdot\frac{n}{3}\right)\cdot\exp\left(-\frac{\delta^2 d}{8}\cdot\frac{\alpha n}{12}\right)\\ &\leq \frac{1}{2}\exp\left(\left(\ln 6 - \frac{1}{6}\log\frac{\alpha}{4} - \frac{\delta^2 d}{96}\right)\alpha n\right)\\ &\overset{(5.7)}{\leq} \frac{1}{2}\exp\left(-\left(\log\alpha + \frac{\delta^2 d}{97}\right)\alpha n\right) = \frac{1}{2}\exp(-\left(\log\alpha + \Omega\left(d\right)\right)\alpha n)\end{aligned} \tag{5.18}$$

for constant δ and $\alpha < 0.5$.

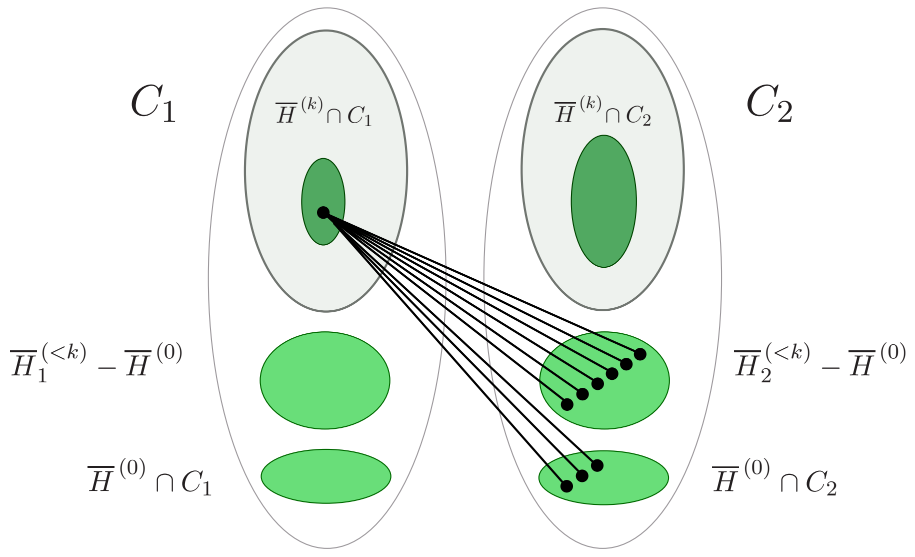
C_1
$\overline{H}^{(k)} \cap C_1$
$\overline{H}^{(k)} \cap C_2$
C_2
$\overline{H}_1^{(<k)} - \overline{H}^{(0)}$
$\overline{H}_2^{(<k)} - \overline{H}^{(0)}$
$\overline{H}^{(0)} \cap C_1$
$\overline{H}^{(0)} \cap C_2$

Figure 5.2: *Vertices in $\overline{H}^{(k)}$ are deleted from H because they have too many neighbors outside H in C_i for some i.*

Since $\mathbf{P}[\,|\overline{H}^{(0)}| \geq \alpha n\,] \leq \mathbf{P}[\,|\overline{H}^{(+)}| \geq \alpha n\,] + \mathbf{P}[\,|\overline{H}^{(-)}| \geq \alpha n\,]$ it remains to argue that an analogue of (5.18) holds for $|\overline{H}^{(-)}|$ in order to establish Lemma 5.4.3. For this purpose, notice that the Chernoff bound (5.5) exceeds the corresponding lower tail bound (5.6). Thus, by the definition of $\overline{H}^{(-)}$ and $\overline{H}^{(+)}$, the probability that $\overline{H}^{(-)}$ reaches a certain size can be estimated using the same methods as above for $\overline{H}^{(+)}$ and so

$$\mathbf{P}\left[\,|\overline{H}^{(-)}| \geq \alpha n\,\right] \leq \frac{1}{2}\exp(-\left(\log\alpha + \Omega\left(d\right)\right)\alpha n)\,.$$

$\square$

The Number of Vertices Deleted in the Iterative Deletion Procedure

The iterative deletion procedure removes vertices having too many neighbors in the set of formerly deleted vertices. The aim of the following calculations is to estimate the probability that the number of these vertices gets considerably bigger than the set of vertices deleted in the initial steps.

For this purpose we condition on the event $|\overline{H}^{(0)}| \leq 3 \cdot |\overline{H}^{it}|$. The constant 3 in this bound is not required for analyzing the performance of the recovery procedure on H. We introduce it for obtaining calculations robust enough to allow certain changes to the construction of H. This will be needed in Section 5.6.

Lemma 5.4.4
Assume that $0 < \alpha < 1/2$ and that $0 < \delta < 1/2$ is constant in the definition of H. Then

$$\mathbf{P}\left[\,|\overline{H}^{it}| \geq \frac{\alpha}{2}n \;\middle|\; |\overline{H}^{(0)}| \leq \frac{\alpha}{2}n\right] \leq \mathbf{P}\left[\,|\overline{H}^{it}| \geq \frac{\alpha}{2}n \;\middle|\; |\overline{H}^{(0)}| \leq \frac{3\alpha}{2}n\right] \leq \exp(\Omega\left(d\cdot\log\alpha\cdot\alpha n\right))\,.$$

Proof:
The first inequality holds since edges are chosen independently in G and by the construction of H.

For the second inequality, consider the first step in the deletion process, say step k, when more than $\alpha n/2$ vertices have been deleted in addition to $\overline{H}^{(0)}$ and denote the vertices deleted prior to step k by $\overline{H}^{(<k)}$. Then

$$\overline{H}^{(<k)} \leq 2\alpha n$$

since $|\overline{H}^{(0)}| \leq 3\alpha n/2$ by assumption. Observe that each vertex in $\overline{H}^{(<k)} \setminus \overline{H}^{(0)}$ was deleted because it had more than $\delta d/3$ neighbours in $\overline{H}^{(<k)}$ (since $\overline{H}^{(<k)} \setminus \overline{H}^{(0)} \subset \overline{H}^{it}$). Accordingly the graph induced on $\overline{H}^{(<k)}$ contains more than $\delta d/3 \cdot \alpha n/2$ edges. At this point it is important to notice that $\overline{H}^{(<k)}$ is not a random subset of V, but this sets is selected deterministically. Therefore, we can not treat the distribution of edges in $\overline{H}^{(<k)}$ as randomly generated. To deal with this difficulty we use Observation 5.2.5 and bound the desired probability from above by calculating the probability that there exists any set $Y \subset V$ of size at most $2\alpha n$ in G with $\mathbf{e}(Y,Y) \geq \delta d/3 \cdot \alpha n/2$.

In the following calculation we show that with high probability a subgraph of this density does not even exist in $\mathcal{G}_{n,p}$ and therefore the same result follows for $\mathcal{G}_{n,p,3}$:

$$\begin{aligned}
&\mathbf{P}\left[|\overline{H}^{it}| \geq \frac{\alpha}{2}n \;\middle|\; |\overline{H}^{(0)}| \leq \frac{3\alpha}{2}n\right] \leq \mathbf{P}\left[\exists Y \subset V : |Y| \leq 2\alpha n \wedge \mathbf{e}(Y,Y) \geq \delta\frac{d}{3}\cdot\frac{\alpha}{2}n\right] \\
&\leq \mathbf{P}\left[\exists Y \subset V : |Y| = 2\alpha n \wedge \mathbf{e}(Y,Y) \geq \delta\frac{d}{3}\cdot\frac{\alpha}{2}n\right] \leq \binom{n}{2\alpha n}\binom{\binom{2\alpha n}{2}}{\delta\frac{d}{3}\cdot\frac{\alpha}{2}n} p^{\delta\frac{d}{3}\cdot\frac{\alpha}{2}n} \\
&\overset{(5.2)}{\leq} \left(\frac{e}{2\alpha}\right)^{2\alpha n}\left(\frac{3e\cdot 2\alpha n(2\alpha n-1)}{\delta d\cdot\alpha n}\right)^{\delta\frac{d}{3}\cdot\frac{\alpha}{2}n}\left(\frac{d}{n}\right)^{\delta\frac{d}{3}\cdot\frac{\alpha}{2}n} \leq \left(\left(\frac{e}{2\alpha}\right)^{12}\left(\frac{12e\cdot\alpha}{\delta}\right)^{\delta d}\right)^{\frac{\alpha}{6}n} \\
&= \left(\left(\frac{e}{2\alpha}\right)^{12}\left(\frac{12e\cdot\alpha}{\delta}\right)^{12}\left(\frac{12e\cdot\alpha}{\delta}\right)^{\delta d-12}\right)^{\frac{\alpha}{6}n} \overset{(5.7)}{=} \exp(\Omega\left(d\cdot\log\alpha\cdot\alpha n\right)).
\end{aligned}$$

□

With this the proof of Lemma 5.4.1 is complete. We next investigate the effect of COLOR$\mathcal{G}_{n,p,3}$ on H.

5.4.2 Coloring H

After assuring that H is a large subgraph of G in most cases, the aim of this section is to show that this subgraph gets colored correctly with high probability by the iterative recoloring procedure of Algorithm 5.1. To begin with, recall that in each step this procedure assigns to each vertex the color that is least favorite among its neighbors.

For the following considerations it is important to notice that the coloring algorithm has no means of determining the graph H defined by the deletion process in the last section. Nevertheless, we can use the structural properties of H to show that the algorithm succeeds on H with high probability. Note that no conclusions are drawn on properties of the coloring obtained for vertices outside of H by the following analysis. So, being pessimistic, in what follows we need to assume that all of them are colored incorrectly by the end of the recoloring procedure.

Idea: Since H has good expansion properties, it is likely that the number of vertices in H that are colored incorrectly decreases by more than a factor of 2 in each of the recoloring steps. If this performance is actually achieved, we call the corresponding step *successful*, otherwise we say that it *fails*. Algorithm 5.1 performs at most $\log n$ of these steps. Afterwards, either the entire graph H is colored correctly or one of the steps failed. As a consequence of the results derived below, we will show in Section 5.7 that the later appears so seldom that the brute force techniques provided by the recovery procedure can be used to repair the coloring of H in this case. More specifically, the idea is that Algorithm 5.1 runs the iterative recoloring procedure until just before the step, say step t, when it fails for the first time. The information that all previous steps were successful provides an upper bound y on how many vertices of H are colored incorrectly at that time. The algorithm then proceeds by exhaustively trying all colorings on all subsets of G of size y and thus fixes the coloring of H in this way. However, since Algorithm 5.1 can not discover whether a particular step of the recoloring procedure succeeds or fails another iteration is necessary at this point. Algorithm 5.1 applies the strategy of simply trying to repair each of the steps of the recoloring procedure subsequently, starting with the last one and proceeding until it reaches step t. This explains the innermost loop of the recovery procedure (Step 7 of Algorithm 5.1). Once the recovery procedure repaired step t, all vertices of H are colored correctly.

For later reference this is summarized in the following remark.

Remark 5.4.5
After executing the iterative coloring procedure of Algorithm 5.1 and possibly the brute force coloring method on appropriate vertex sets, the graph H is colored correctly.

Notice that the recovery procedure simultaneously corrects the mistakes of other steps of the underlying algorithm and so the effects on the recoloring step are not made explicit in Algorithm 5.1. A more detailed account on the overall picture is given in Section 5.7.

Based on the ideas described above we now calculate the probability that the iterative recoloring procedure fails on αn vertices of H.

Lemma 5.4.6
Consider the first step of the recoloring procedure that fails and let $F_H \subseteq H$ denote the set of vertices of H that were colored incorrectly in H before this step. Then

$$\mathbf{P}[\,|F_H| = \alpha n\,] \leq \exp(\Omega\,(d \cdot \log \alpha \cdot \alpha n))$$

for δ sufficiently small but constant in the definition of H.

PROOF:
Let $F^{(s)} \subseteq H$ be the set of vertices in H colored incorrectly after step s of the iterative recoloring procedure and denote by $F_i^{(s)}$ the vertices of $F^{(s)}$ that belong to C_i.

Now, consider step t of the recoloring procedure and assume that all previous steps were successful, but step t fails, i.e., $F^{(t-1)} = F_H$ and $|F^{(t)}| \geq |F^{(t-1)}|/2$. Let $i \neq j$ and $v \in F^{(t)}$ be a vertex in color class C_i that received color j in step t. Observe that, as v remains colored incorrectly after step t, v can have at most $deg_G(v)\,/3$ neighbors that were colored j in the previous step. But by construction, H contains only vertices which have at most $(1+\delta)d/3$ neighbors in each color class other than their own and so $deg_G(v) \leq 2 \cdot (1+\delta)d/3$ and $deg_{C_j \cap H}(v) \geq (1-\delta)d/3$. This implies that v has at most $(2 \cdot (1+\delta)d/9)$ neighbors in

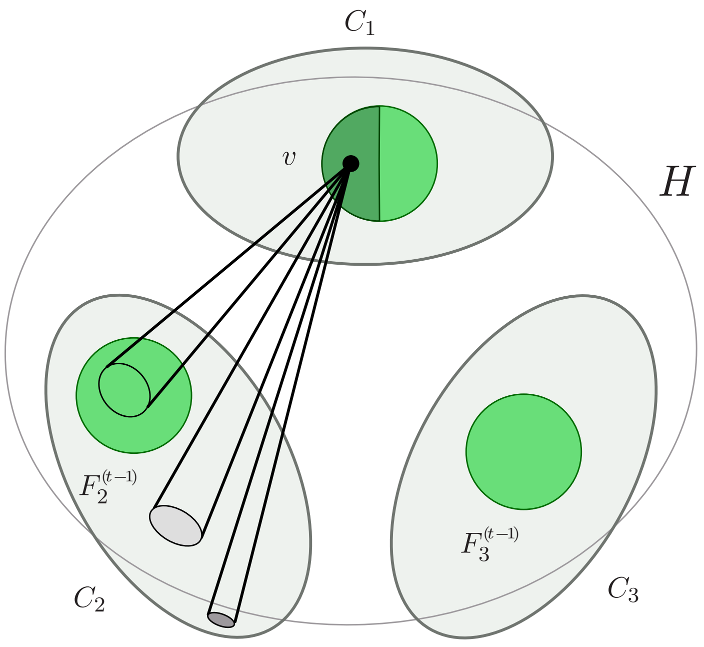

Figure 5.3: *Vertices v in C_1 that received color 2 in step t of the recoloring procedure need to have many neighbors in a small vertex set $F_2^{(t-1)}$ of C_2.*

$C_j \cap H - F^{(t-1)}$ and so all other neighbors of v in $C_j \cap H$, namely, at least

$$(1-\delta)\frac{d}{3} - 2\cdot(1+\delta)\frac{d}{9} = (1-5\delta)\,\frac{d}{9}$$

are contained in $F^{(t-1)}$. Figure 5.3 illustrates the different types of neighbors of v in C_j. We conclude that the graph induced on $F^{(t-1)}$ contains at least $(1-5\delta)\,d/9\cdot|F^{(t)}|$ edges. Using Observation 5.2.5, we can therefore bound $\mathbf{P}[\,|F_H| = \alpha n\,]$ by calculating the probability that some vertex set Y exists in V with $|Y| = \alpha n$ such that $\mathbf{e}(Y,Y) \geq (1-5\delta)d/9\cdot\alpha n/2$. Notice that the estimation of this probability is similar to the one carried out for evaluating the size of $\overline{H}^{it}$:

$$\begin{aligned}
&\mathbf{P}\left[\exists Y \subset V : |Y| = \alpha n \wedge \mathbf{e}(Y,Y) \geq (1-5\delta)\,\frac{d}{9}\cdot\frac{\alpha}{2}n\right]\\
&\leq \binom{n}{\alpha n}\binom{\binom{\alpha n}{2}}{(1-5\delta)\frac{d}{9}\cdot\frac{\alpha}{2}n}\cdot p^{(1-5\delta)\frac{d}{9}\cdot\frac{\alpha}{2}n}\\
&\overset{(5.2)}{\leq} \left(\frac{e}{\alpha}\right)^{\alpha n}\left(\frac{9e\cdot\alpha n(\alpha n-1)}{(1-5\delta)d\cdot\alpha n}\right)^{(1-5\delta)\frac{d}{9}\cdot\frac{\alpha}{2}n}\left(\frac{d}{n}\right)^{(1-5\delta)\frac{d}{9}\cdot\frac{\alpha}{2}n} \leq \left(\left(\frac{e}{\alpha}\right)^{18}\left(\frac{9e\cdot\alpha}{1-5\delta}\right)^{(1-5\delta)d}\right)^{\frac{\alpha}{18}n}\\
&\leq \left(\left(\frac{e}{\alpha}\right)^{18}\left(\frac{9e\cdot\alpha}{1-5\delta}\right)^{18}\left(\frac{9e\cdot\alpha}{1-5\delta}\right)^{(1-5\delta)d-18}\right)^{\frac{\alpha}{18}n} \overset{(5.7)}{=} \exp(\Omega\,(d\cdot\log\alpha\cdot\alpha n))\,,
\end{aligned}$$

for δ sufficiently small. As argued above, this implies $\mathbf{P}[\,|F_H| = \alpha n\,] \leq \exp(\Omega\,(d\cdot\log\alpha\cdot\alpha n))$ as desired. $\square$

Note that the definition of F_H above and Remark 5.3.6 imply an upper bound on the size of F_H.

Remark 5.4.7
Let F_H be the vertex set defined in Lemma 5.4.6. Then $|F_H| < \epsilon n$.

5.5 Uncoloring the Remaining Incorrectly Colored Vertices

As we saw in the last section, the iterative recoloring procedure refines the initial coloring of G. At the end of this procedure, COLOR$\mathcal{G}_{n,p,3}$ tries to determine those vertices that might still be colored incorrectly and uncolor them. This task is performed in the uncoloring procedure (Step 9 of Algorithm 5.1). The uncoloring procedure proceeds by repeatedly uncoloring all those vertices of G that have less than $d/6$ neighbors of some color other than their own.

By the definition of H, this procedure does not effect the coloring of H if we assume that H is colored correctly before this procedure is called as justified by Remark 5.4.5.

Observation 5.5.1
If H is colored correctly before the application of the uncoloring procedure, no vertex of H gets uncolored by this procedure.

Indeed, since each vertex v in H has at least $(1-\delta)d/3$ neighbors in $C_i \cap H$ for each i such that $v \notin C_i$ and all these neighbors are colored with color i, v does not get uncolored as long as $\delta < 1/2$.

Accordingly, in what follows, we will turn our attention to vertices $v \notin H$.

5.5.1 Uncoloring the Vertices Outside of H

Many vertices outside of H do not share the strong regularity properties of vertices from H. Some vertices of $\overline{H}$ might therefore neither be colored correctly by previous steps of Algorithm 5.1, nor will they get uncolored by the uncoloring procedure. In the following we show, however, that this is comparably unlikely.

Lemma 5.5.2
Let $F_\Upsilon \subset G - H$ be the set of vertices that are colored incorrectly and remain colored after the execution of the uncoloring procedure. Then

$$\mathbf{P}[\,|F_\Upsilon| = \alpha n\,] \leq \exp(\Omega\,(d \cdot \log \alpha \cdot \alpha n))$$

for $0 < \alpha < \frac{1}{2}$.

Proof:
More generally we will bound the probability that $|F_\Upsilon| \geq \alpha n$ from above.

If a vertex v in C_i is colored incorrectly, say with color j, and remains colored after the uncoloring procedure, v must have at least $d/6$ neighbors of color i. Since v is not adjacent to any vertex in its own color class C_i, all these neighbors are elements of F_Υ as well. Hence it suffices to determine the probability that there is some set $Y \subset V(G)$ with $|Y| = \alpha n$ and minimum degree at least $d/6$:

$$\begin{aligned}
\mathbf{P}[\,|F_\Upsilon| \geq \alpha n\,] &\leq \mathbf{P}[\,\exists Y \subset V(G) : |Y| = \alpha n \wedge \mathit{mindeg}(Y) \geq d/6\,]\\
&\leq \mathbf{P}\left[\exists Y \subset V(G) : |Y| = \alpha n \wedge \mathbf{e}(Y,Y) \geq \frac{d}{12}\alpha n\right]\\
&\leq \binom{n}{\alpha n}\binom{\binom{\alpha n}{2}}{\frac{d}{12}\alpha n} p^{d\cdot\alpha n/12} \overset{(5.2)}{\leq} \left(\frac{e}{\alpha}\right)^{\alpha n}\left(\frac{6e(\alpha n-1)}{d}\right)^{d\cdot\alpha n/12}\left(\frac{d}{n}\right)^{d\cdot\alpha n/12}\\
&\leq \left(\frac{e}{\alpha}\,(6e\alpha)^{d/12}\right)^{\alpha n} \leq \left(6e^2\,(6e\alpha)^{d/12-1}\right)^{\alpha n} \overset{(5.7)}{=} \exp(\Omega\,(d\cdot\log\alpha\cdot\alpha n))\,.
\end{aligned}$$

□

5.6 Recoloring the Uncolored Vertices

Knowing that the uncoloring procedure succeeds in uncoloring all vertices of wrong color with high probability, we are now left with the task of assigning a new color to these uncolored vertices. In Algorithm 5.1 this is taken care of by the extension step in Step 10. Here, an exact coloring routine makes sure that the vertices are forced to be colored correctly this time. In spite of having exponential worst time complexity, this step performs polynomial on average as we will see in this section. For this purpose, we use techniques developed by Alon and Kahale in [2], to show that all components induced on the set of uncolored vertices are rather small. The main goal is to prove the following lemma.

Lemma 5.6.1
For $\alpha < \alpha_0$,

$$\mathbf{P}\left[\,\textit{there is a component of order } \alpha n \textit{ in } \overline{H}\;\right] \leq \left(\frac{d}{\exp(\Omega\,(d))}\right)^{\alpha n}.$$

Recall that $\alpha_0 = 2^{-c_{(0)}d/10}$ for a constant $c_{(0)}$ (see Section 5.4.1, page 43) and notice that it is sufficient to consider only α smaller than α_0 by Remark 5.4.2.

From Lemma 5.6.1 it follows that with high probability a valid coloring of H can indeed be extended to the whole graph G by Step 10 of Algorithm 5.1 as long as $\overline{H}$ does not get too large. This will be proven in the next lemma.

Lemma 5.6.2
The extension step (Step 10) of Algorithm 5.1 has polynomial expected running time.

PROOF:
As explained, in Step 10 of Algorithm 5.1 an exact coloring method is used to extend the partial coloring obtained in earlier steps to the whole graph G. In this process the components induced on the vertices uncolored by the uncoloring procedure are considered independently. On each such component the algorithm tries all possible colorings until it finds one that is compatible to the coloring of the rest of G.

By definition the expected running time of this procedure is determined by the complexity of the exact coloring method used and the probability that it has to be executed on vertex sets of a certain size. Trivially there are at most n components in the set of uncolored vertices and so it suffices to show that the probability that a component of $G - H$ has αn vertices multiplied by **3-COL** (αn) remains small for all $\alpha < \alpha_0$:

$$\begin{aligned}&\mathbf{P}[\,\text{there is a component of order } \alpha n \text{ in } G - H\;\,] \cdot \textbf{3-COL}\,(\alpha n)\\ &\leq \left(\frac{d}{\exp(\Omega\,(d))}\right)^{\alpha n} \cdot 3^{\alpha n} \leq \left(\frac{3d}{\exp(\Omega\,(d))}\right)^{\alpha n} = \mathcal{O}(1)\,.\end{aligned}$$

□

Some more preparation is needed before we can turn to the proof of Lemma 5.6.1. Let αn be the size of the largest component induced on the uncolored vertices. Clearly

$$\begin{aligned}&\mathbf{P}[\,\text{there is a component of order } \alpha n \text{ in } G - H\;\,] \leq \mathbf{P}[\,\exists \text{ a tree } T \preceq G - H \text{ of order } \alpha n\,]\\ &\quad = \mathbf{P}[\,\exists \text{ a tree } T \text{ with } |T| = \alpha n \;:\; E(T) \subset E(G) \wedge T \cap H = \emptyset\,]\,.\end{aligned} \tag{5.19}$$

Unfortunately the two events occuring on the last line of Equation (5.19) are not independent of each other since H is not a random subgraph of G. For this reason, we modify the construction of H depending on the tree T under study, resulting in a new subgraph H' of G. For this subgraph the corresponding events can be seperated, giving an upper bound on the probability we are interested in.

So, consider a fixed tree T with $V(T) \subset V(G)$ and all edges between different color classes of G, but not necessarily satisfying $E(T) \subset E(G)$. Denote by $T_{<4}$ all vertices v of T obeying $deg_T(v) < 4$, set $T_{\geq 4} := T - T_{<4}$, and let H' be the graph obtained from G by the following process:

1. Construct a graph $G' = (V, E')$ from G by discarding all edges in $E(T)$ and then reconsidering their occurence by throwing a new die : $E' = (E \setminus E(T)) \cup E'(T)$ where $E'(T) \subset E(T)$ contains each edge of T with probability p.

2. Delete all vertices in $T_{\geq 4}$ from G'.

3. Apply the procedure for constructing H to G' with δ slightly reduced to δ' (see page 42).

Again, we refer to Steps 1 and 2 of the construction procedure for H within this process as the two *initial steps*. Step 3 of the procedure for H is called the *iterative deletion process*, and the corresponding sets are $\overline{H}^{(0)'}$ and $\overline{H}^{it'}$.

Note that the graph G' constructed in Step 1 above still is a 3-colorable graph with partitions C_1, C_2, and C_3 since all edges of T run between different partitions of G by definition. When refering to the degree of a vertex $v \in V$ into one of these partitions C_i in G' we write $deg_{C_i'}(v)$.

The next lemma shows that this revised construction does indeed lead to independent events. A similar version first appeared in [2] although the construction for H' used here is slightly changed and so our analysis involves arguments different from those in [2].

Lemma 5.6.3
For any fixed tree T,

$$\mathbf{P}[\,E(T) \subset E(G) \wedge T \cap H = \emptyset\,] \leq \mathbf{P}[\,E(T) \subset E(G)\,] \cdot \mathbf{P}\big[\,T \cap H' = \emptyset\,\big]\,.$$

Proof:
By

$$\mathbf{P}\big[\,E(T) \subset E(G) \wedge T \cap H' = \emptyset\,\big] = \mathbf{P}[\,E(T) \subset E(G)\,] \cdot \mathbf{P}\big[\,T \cap H' = \emptyset \bigm| E(T) \subset E(G)\,\big]$$

this lemma is a direct consequence of $T \cap H' \subset T \cap H$ since the events $T \cap H' = \emptyset$ and $E(T) \subset E(G)$ are clearly independent due to Step 1 in the construction of H'. Accordingly, it remains to show that $v \notin H \cap T$ implies $v \notin H' \cap T$. We assert this by comparing the different deletion steps in the construction of H versus H'.

First, consider vertices $v \in \overline{H}^{(0)}$ deleted in one of the first two steps while constructing H (i.e., vertices v with $deg_{C_i}(v) \lesseqgtr (1 \mp \delta)d/3$ for some i). If $v \notin T$ then v is clearly deleted in the initial steps of the construction of H'. All vertices $v \in T$ with $v \notin T_{<4}$ are deleted from H' as well. For $v \in T_{<4}$ finally we have either $deg_{C_i'}(v) \geq (1+\delta)d - 3 \geq (1+\delta')d$ or $deg_{C_i'}(v) \leq (1-\delta)d + 3 \leq (1-\delta')d$ by the definition of $T_{<4}$ and so $v \in \overline{H}^{(0)\prime}$ in this case too.

Now, let us turn to vertices $v \in \overline{H}^{it}$. By the preceeding arguments we know that $\overline{H}^{(0)} \subset \overline{H}^{(0)\prime}$. We proceed by induction on the steps in the recursive deletion process. In each of these steps in the construction of H, those vertices v are deleted from H that have more than $2\delta d/3$ formerly deleted neighbors. Again, in the case $v \notin T$ such a deletion carries over to a deletion from H' since $\overline{H}^{(0)} \subset \overline{H}^{(0)\prime}$ and by applying the induction hypothesis. The case $v \in T_{\geq 4}$ may be omitted since these vertices were deleted from H' already. Thus, it remains to consider $v \in T_{<4}$ (observe that we can not use the same argument here as in the case $v \notin T$ since the edges on vertices from T are different in H and H'). By induction hypothesis and the definition of $T_{<4}$ we know that v has at least $2\delta d/3 - 3 \leq 2\delta' d/3$ neighbors that were deleted from H' in earlier steps. It follows that v is deleted from H as well.

This concludes the proof of $H' \cap T \subset H \cap T$ and shows more generally that $H' \subset H$. Therefore, the validity of Lemma 5.6.3 is verified. □

With this we are ready to prove Lemma 5.6.1. There, we are interested in trees T with $|T| = \alpha n$. In this case $|T_{<4}| \geq \alpha n/2$ because $|T_{<4}| \geq |T|/2$ follows from

$$|T| - 1 = |E(T)| = \frac{1}{2}\left(\sum_{v \in T_{<4}} deg_T(v) + \sum_{v \in T_{\geq 4}} deg_T(v)\right) \geq \frac{1}{2} \cdot 4\,|T_{\geq 4}|\,.$$

The following discussion will partly refer to the analysis provided in Section 5.4. Keep in mind though that the symbol α is used in a different context there.

Proof of Lemma 5.6.1:

By Equation (5.19) and Lemma 5.6.3, we can rewrite the probability we are interested in as follows:

$$\mathbf{P}[\,\exists T \text{ with } |T| = \alpha n \,:\, E(T) \subset E(G) \wedge T \cap H = \emptyset\,] \leq \sum_T \mathbf{P}[\,E(T) \subset E(G)\,] \cdot \mathbf{P}\big[\,T \cap H' = \emptyset\,\big]\,.$$

Since the edges of G (and G') are chosen independently from each other, the probability that $E(T) \subset E(G)$ for a fixed tree T is simply $p^{\alpha n}$. So in the main part of this proof we will investigate the probability $\mathbf{P}[\,T \cap H' = \emptyset\,]$ and show that

$$\mathbf{P}\big[\,T \cap H' = \emptyset\,\big] < \exp(-\alpha \cdot \Omega\,(dn))\,.$$

This suffices for establishing the desired bound: It is well known that the number of labeled trees on y vertices is y^{y-2} (see e.g., [52]) and consequently

$$\begin{aligned}
&\sum_T \mathbf{P}[\, E(T) \subset E(G)\,] \cdot \mathbf{P}\big[\, T \cap H' = \emptyset\,\big] \\
&\leq \binom{n}{\alpha n} (\alpha n)^{\alpha n - 2} \cdot p^{\alpha n - 1} \cdot \exp(-\alpha \cdot \Omega\,(dn)) \\
&\overset{(5.2)}{\leq} \left(\frac{e}{\alpha} \cdot \alpha n \cdot \frac{d}{n} \cdot \exp(-\Omega\,(d)) \right)^{\alpha n} \leq \left(\frac{d}{\exp(\Omega\,(d))} \right)^{\alpha n}.
\end{aligned}$$

Calculating $\mathbf{P}[\, T \cap H' = \emptyset\,]$: H' is obviously not a random subgraph of G. However, the event $T \cap H' = \emptyset$ does not depend on the structure of H', but only on the vertices contained in this subgraph. Moreover, the construction of H' is independent of the labeling of the vertices involved. Now, consider a fixed graph F. Then, by the foregoing remarks, among all graphs from $\mathcal{G}_{n,p,3}$ the event that F is induced on a particular set of vertices and forms the graph H' has the same probability as the event that this happens for any other set of vertices. Consequently, we get an invariance under permutation of the labeling of G (while leaving the labeling of T fixed) and so the only information about H' that is necessary for determining the desired probability is its size. For this reason, we will next try to estimate $|H'|$. Since the construction of H' resembles that of H apart from minor changes, it will not be difficult to argue why the calculations concerning the size of H in Section 5.4 can be applied to H' as well. But before providing these arguments, it is a good idea to recall some observations made when investigating the size of H and use them to develop a reasoning for the strategy we will follow from here on.

In Section 5.4, the probability that $|G - H|$ exceeds αn was bounded by

$$\exp(-\,(\log \alpha + \Omega\,(d)) \cdot \alpha n) + \exp(\Omega\,(d \cdot \log \alpha \cdot \alpha n))$$

(see Lemma 5.4.1). In this connection we also remarked that the given estimation does not allow for a nontrivial upper bound when $\alpha \ll \alpha_0 = 2^{-c_{(0)}d/10}$, where $c_{(0)}$ is the constant defined on page 43. As mentioned, we will show that we can use the same bound for a corresponding result on $|H'|$ which will, again, only be useful in the case that $|G - H'|$ is at least of size $\alpha_0 n$. This gives a motivation for rewriting $\mathbf{P}[\, T \cap H' = \emptyset\,]$ by conditioning on the event $|H'| \geq (1 - \alpha_0)\, n$:

$$\begin{aligned}
\mathbf{P}\big[\, T \cap H' = \emptyset\,\big] &= \mathbf{P}\big[\, T \cap H' = \emptyset \;\big|\; \big|H'\big| \geq (1 - \alpha_0)\, n\,\big] \cdot \mathbf{P}\big[\, \big|H'\big| \geq (1 - \alpha_0)\, n\,\big] \\
&\quad + \mathbf{P}\big[\, T \cap H' = \emptyset \;\big|\; \big|H'\big| < (1 - \alpha_0)\, n\,\big] \cdot \mathbf{P}\big[\, \big|H'\big| < (1 - \alpha_0)\, n\,\big] \\
&\leq \mathbf{P}\big[\, T \cap H' = \emptyset \;\big|\; \big|H'\big| \geq (1 - \alpha_0)\, n\,\big] + \mathbf{P}\big[\, \big|H'\big| < (1 - \alpha_0)\, n\,\big].
\end{aligned} \tag{5.20}$$

Since, as discussed above, $\mathbf{P}[\, T \cap H'\,]$ does only depend on the size of H' we can bound the first term in Equation (5.20) in the following way:

$$\begin{aligned}
\mathbf{P}\big[\, T \cap H' = \emptyset |\, \big|H'\big| \geq (1 - \alpha_0)\, n\,\big] &\leq \frac{\binom{\alpha_0 n}{|T|}}{\binom{n}{|T|}} \leq \left(\frac{e \cdot \alpha_0 n}{|T|} \cdot \frac{|T|}{n} \right)^{|T|} \\
&\leq \left(e \cdot 2^{-\frac{1}{10} c_{(0)} \cdot d} \right)^{\alpha n} = \exp(-\alpha \cdot \Omega\,(dn))\,.
\end{aligned}$$

For evaluating the second term we return to the discussion on the size of H'.

The initial steps in the construction of H and H' (i.e., removing vertices of high and low degree in G and G', respectively) are identical apart from the value of δ and δ', respectively. The computations concerning the size of the set $\overline{H}^{(0)} \subset G - H$ of initially deleted vertices have been performed for δ' in Section 5.4 (see Lemma 5.4.3). Since G and G' are both graphs from $\mathcal{G}_{n,p,3}$, we consequently can adopt the results derived there for concluding that $|\overline{H}^{(0)'}| < \alpha_0 n/4$ holds with probability at least

$$1 - \exp\Big(-\Big(\log\frac{\alpha_0}{2} + c_{(0)}\cdot d\Big)\cdot\frac{\alpha_0}{2}n\Big) = 1 - \exp\Big(-\Big(-\frac{1}{10}c_{(0)}\cdot d - 1 + c_{(0)}\cdot d\Big)\, 2^{-\frac{1}{10}c_{(0)}\cdot d-1}n\Big)$$
$$= 1 - \exp\Big(-\frac{d}{2^{\mathcal{O}(d)}}\,\Omega\,(n)\Big).$$

$T_{\geq 4}$ is of order at most $\alpha n/2$ and therefore this set contains at most $\alpha_0 n/2$ vertices. So the number $|\overline{H}^{(0)'} \cup T_{\geq 4}|$ of vertices removed from H' before the iterative deletion procedure starts is certainly less than $3\alpha_0 n/4$ with the same probability.

Later, in Lemma 5.4.4 we calculated $\mathbf{P}\Big[\,|\overline{H}^{it}| \geq y \;\Big|\; |\overline{H}^{(0)}| \leq 3y\,\Big]$ for the number $|\overline{H}^{it}|$ of vertices removed in the iterative deletion procedure. Recall that the corresponding analysis solely relied on the probability that a set $\overline{H}^{it}_{i\to j}$ of vertices deleted in this procedure has got many neighbors in a set $\overline{H}^{it}_{j}$ of formerly deleted vertices. Here, this probability does not depend on the structure induced on the set $\overline{H}^{it}_{j}$. In particular, the structure of the graph deleted before the iterative deletion procedure takes effect is irrelevant to the results established in this connection. We therefore conclude in accordance to Lemma 5.4.4 that given

$$|\overline{H}^{(0)'} \cup T_{\geq 4}| \leq 3\alpha_0 n/4,$$

the event $|\overline{H}^{it'}| < \alpha_0 n/4$ holds with probability at least

$$\begin{aligned} & 1 - \exp(\Omega\,(d\cdot\log\alpha_0\cdot\alpha_0 n)) \\ = \;& 1 - \exp\Big(\Omega\,\Big(d\cdot\log 2^{-c_{(0)}d/10}\cdot 2^{-c_{(0)}d/10}n\Big)\Big) \\ \overset{(5.7)}{\geq} & 1 - \exp\Big(\Omega\,\Big(-d\cdot d\cdot 2^{-c_{(0)}d/10}\cdot n\Big)\Big) = 1 - \exp\Big(-\frac{d}{2^{\mathcal{O}(d)}}\,\Omega\,(n)\Big). \end{aligned}$$

It follows that

$$\mathbf{P}\big[\,|H'| \geq (1-\alpha_0)\,n\,\big] \geq 1 - \exp\Big(-\frac{d}{2^{\mathcal{O}(d)}}\,\Omega\,(n)\Big)$$

and so, using the bound on $\mathbf{P}[\,T\cap H' = \emptyset \mid |H'| \geq (1-6\cdot\alpha_0)\,n\,]$ calculated earlier, we finally arrive at

$$\begin{aligned} \mathbf{P}\big[\,T\cap H' = \emptyset\,\big] &\leq \mathbf{P}\big[\,T\cap H' = \emptyset \;\big|\; |H'| \geq (1-6\cdot\alpha_0)\,n\,\big] + \mathbf{P}\big[\,|H'| < (1-6\cdot\alpha_0)\,n\,\big] \\ &\leq \exp(-\alpha\cdot\Omega\,(dn)) + \exp\Big(-\frac{d}{2^{\mathcal{O}(d)}}\,\Omega\,(n)\Big) \overset{\alpha<\alpha_0}{\leq} \exp(-\alpha\cdot\Omega\,(dn)). \end{aligned}$$

□

5.7 The Performance of the Recovery Procedure

As mentioned earlier all main steps of Algorithm 5.1, i.e., the construction of the initial coloring, the recoloring procedure, and the uncoloring procedure are executed in polynomial time. Moreover, Lemma 5.6.1 guarantees that the extension step of COLOR$\mathcal{G}_{n,p,3}$ has polynomial expected running time. It therefore remains to investigate the recovery procedure consisting of the loops in Steps 2, 3, 4 and 7 of Algorithm 5.1.

The results derived in the last few sections estimate the probabilities that one of the main steps of Algorithm 5.1 fails on a vertex set Y of size y. As explained in Section 5.1, these vertex sets are taken care of by the recovery procedure. In this section we study the overall running time of the recovery procedure by bringing together these previous results. We establish the following lemma and thus complete the proof of Theorem 4.

Lemma 5.7.1
The recovery procedure (i.e., Steps 2 to 7) of Algorithm 5.1 has polynomial expected running time.

For motivating the strategy we use, Figure 5.4 provides an overview on the interactions between the main steps of Algorithm 5.1 and the recovery procedure. In this figure the main steps are given as nodes. The "+"-signs denote success, the "−"-signs failure of the corresponding steps. In the later case the action that needs to be performed in order to fix the failure is given in *italic* on the edge to the next node. All these actions are implicitly executed by the recovery procedure of Algorithm 5.1. They are not mentioned separately since the algorithm cannot compute the various subgraphs of G used in different parts of the analysis.

The notation used in Figure 5.4 is inherited from the preceeding sections. In particular, recall that H is the maximal subgraph of G with $(1+\delta)d/3 \leq deg_{C_i}(v) \leq (1-\delta)d/3$ for all $v \in C_j$, $i \neq j \in \{1,2,3\}$, and some appropriate δ. F_H are those vertices in H which are colored incorrectly after the recoloring procedure of Algorithm 5.1. F_{SDP} is the set of vertices that need to be assigned a different color for obtaining a valid coloring on an $(1-\epsilon)$-fraction of G after the inital phase. The set of vertices colored incorrectly after the uncoloring procedure finally is denoted by F_Υ and

$$\alpha_0 = 2^{-c_{(0)}d/10}$$

for some constant $c_{(0)}$ (cf. page 43).

Note that we do not rely on the uncoloring procedure and the extension step if the graph H gets too small. In this case, it is sufficient to assume that the recovery procedure takes care of all vertices in $\overline{H}$. A justification of this strategy is provided with the following calculations.

Proof of Lemma 5.7.1:
Consider the vertex sets Y from Algorithm 5.1 that are colored correctly in Step 3 of the recovery procedure and let $t(y)$ be the time the algorithm needs to execute this step in the case $|Y| = y$. Further, denote by F the set Y used in the iteration when the algorithm finally obtains a valid coloring and recall that **3-COL** (x) is the time needed to find all colorings for a graph of order x.

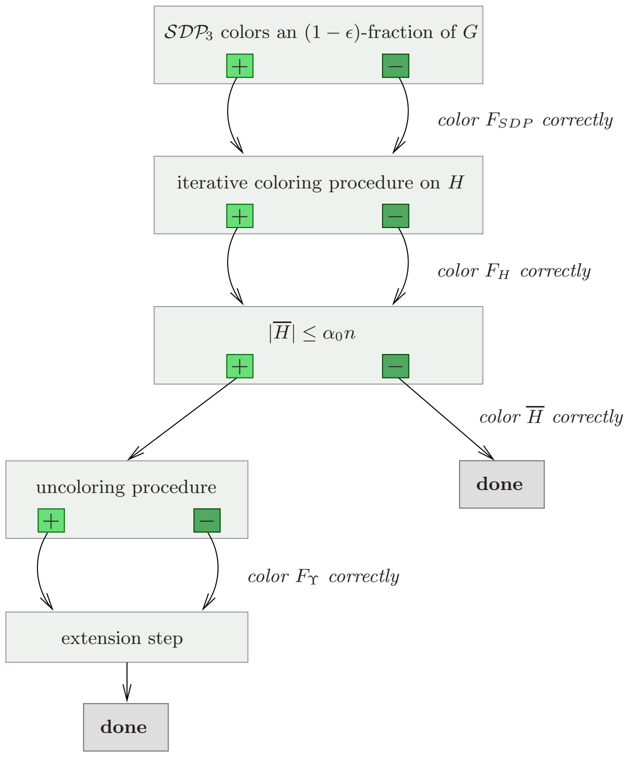
$\mathcal{SDP}_3$ colors an $(1-\epsilon)$-fraction of G
+
−
color F_{SDP} correctly
iterative coloring procedure on H
+
−
color F_H correctly
$|\overline{H}| \leq \alpha_0 n$
+
−
color $\overline{H}$ correctly
uncoloring procedure
+
−
done
color F_Υ correctly
extension step
done

Figure 5.4: *A graph illustrating the calculations implicit to Algorithm 5.1.*

The expected running time $\mathbf{E}[\,t\,]$ of the repair mechanism can then be written as

$$\begin{aligned}\mathbf{E}[\,t\,] &= \sum_{y\leq n} \mathbf{P}[\,|F| = y\,]\cdot t(y)\\ &= \sum_{y\leq n} \mathbf{P}[\,|F| = y\,]\cdot \sum_{y'\leq y}\binom{n}{y'}\textbf{3-COL}\,(y')\\ &\leq \mathcal{O}(n)\sum_{y\leq n}\mathbf{P}[\,|F| = y\,]\cdot\binom{n}{y}\textbf{3-COL}\,(y)\,.\end{aligned} \tag{5.21}$$

We proceed by splitting the summands $\mathbf{P}[\,|F| = y\,]\cdot\binom{n}{y}\textbf{3-COL}\,(y)$ into different components and show that each of them evaluates to a polynomial. To begin with, we bound $\mathbf{P}[\,|F| = y\,]$ by rewriting F as sum of F_{SDP}, F_H, F_Υ and possibly $\overline{H}$. For this, observe that

$$F = F_{SDP}\cup F_H\cup F_\Upsilon.$$

Consistent with the strategy outlined in Figure 5.4, however, the following analysis only applies this partitioning of F in the case that $\left|\overline{H}\right| \leq \alpha_0 n$. If $\left|\overline{H}\right| \leq \alpha_0 n$ we use

$$F \subset F_{SDP}\cup F_H\cup \overline{H}.$$

Since all probabilities involved are monotone decreasing, we get

$$\begin{aligned}\mathbf{P}[\,|F| = y\,] \le &\sum_{\substack{y_1+y_2+y_3=y\\ y_3\le\alpha_0 n}} \mathbf{P}[\,|F_{SDP}| = y_1 \wedge |F_H| = y_2 \wedge |F_\Upsilon| = y_3\,]\\ &+ \sum_{\substack{y_1+y_2+y_3=y\\ y_3>\alpha_0 n}} \mathbf{P}\left[\,|F_{SDP}| = y_1 \wedge |F_H| = y_2 \wedge \left|\overline{H}\right| = y_3\,\right].\end{aligned}$$

One of the vertex sets involved in the conjunctions of this sum certainly contains more than $y/3$ vertices and so

$$\mathbf{P}[\,|F| = y\,] \le n^3 \begin{cases} \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_H| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_\Upsilon| = \frac{y}{3}\,\right]\right) & \text{if } \frac{y}{3} \le \alpha_0 n,\\ \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_H| = \frac{y}{3}\,\right], \mathbf{P}\left[\,\left|\overline{H}\right| = \frac{y}{3}\,\right]\right) & \text{otherwise.}\end{cases}$$

By Remark 5.4.7, the size of F_H does not exceed ϵn. Therefore, we can refine this equation in the following way:

$$\mathbf{P}[\,|F| = y\,] \le n^3 \begin{cases} \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_H| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_\Upsilon| = \frac{y}{3}\,\right]\right) & \text{if } \frac{y}{3} \le \alpha_0 n,\\ \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,\left|\overline{H}\right| = \frac{y}{3}\,\right]\right) & \text{if } \frac{y}{3} \ge \epsilon n,\\ \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,|F_H| = \frac{y}{3}\,\right], \mathbf{P}\left[\,\left|\overline{H}\right| = \frac{y}{3}\,\right]\right) & \text{otherwise.}\end{cases} \tag{5.22}$$

Now, let $y = \alpha n$ and recall that

$$\mathbf{P}[\,|F_H| = \alpha n\,] \le (c_H\alpha)^{c'_H d\cdot\alpha n} =: f_H(\alpha),$$

$$\mathbf{P}[\,|F_\Upsilon| = \alpha n\,] \le (c_\Upsilon\alpha)^{c'_\Upsilon d\cdot\alpha n} =: f_\Upsilon(\alpha),$$

and

$$\mathbf{P}\left[\,\left|\overline{H}\right| = \alpha n\,\right] \le \mathbf{P}\left[\,\left|\overline{H}\right| \ge \alpha n\,\right] \le \underbrace{\exp\left(-\alpha n\left(\log\alpha + c_{(0)}\cdot d\right)\right)}_{=:f_{(0)}(\alpha)} + \underbrace{(c_{it}\alpha)^{c'_{it}d\cdot\alpha n}}_{=:f_{it}(\alpha)} =: f_{\overline{H}}(\alpha)$$

by Lemmas 5.4.6, 5.5.2, and 5.4.1, respectively, for appropriate constants c_H, c'_H, c_Υ, c'_Υ, $c_{(0)}$, c_{it}, and c'_{it}. With this we can rewrite (5.22) as

$$\mathbf{P}[\,|F| = y\,] \le n^3 \begin{cases} \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], f_H(\frac{y}{3n}), f_\Upsilon(\frac{y}{3n})\right) & \text{if } \frac{y}{3} \le \alpha_0 n,\\ \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], \mathbf{P}\left[\,\left|\overline{H}\right| = \frac{y}{3}\,\right]\right) & \text{if } \frac{y}{3} \ge \epsilon n,\\ \max\left(\mathbf{P}\left[\,|F_{SDP}| = \frac{y}{3}\,\right], f_H(\frac{y}{3n}), \mathbf{P}\left[\,\left|\overline{H}\right| = \frac{y}{3}\,\right]\right) & \text{otherwise.}\end{cases} \tag{5.23}$$

The reason why we did not bound $\mathbf{P}\left[\,\left|\overline{H}\right| = y/3\,\right]$ by $f_{\overline{H}}$ will become clear later. We next exploit the fact that $f_\Upsilon(\alpha)$, $f_H(\alpha)$, and $f_{it}(\alpha)$ are of similar structure.

Observation 5.7.2
Let $c_{max} := \max(c_H, c_\Upsilon, c_{it})$. Then there is some constant c'_{max} such that each of the functions $f_\Upsilon(\alpha)$, $f_H(\alpha)$, and $f_{it}(\alpha)$ is less than or equal to

$$f_{max}(\alpha) := (c_{max}\alpha)^{c'_{max}d\cdot\alpha n}$$

in the whole interval $0 < \alpha < 1$.

Indeed, our estimates on the three probabilities in question are all of the form $(c\alpha)^{c'd\cdot\alpha n}$ where $c > 1$ and $c' > 0$. Note that for comparing two functions of this form, we can omit the exponent $d \cdot \alpha n$. So, consider two functions $f_1(y) := (c_1 y)^{c_1'}$ and $f_2(y) := (c_2 y)^{c_2'}$ with $c_1, c_2 > 1$, $c_1', c_2' > 0$, and $c_1 \geq c_2$. Then $f_1 \geq f_2$ for $y < 1$ provided that $c_1' \leq c_2'$ and ${c_1}^{c_1'} \geq {c_2}^{c_2'}$. These inequalities may be asserted by choosing $c_2' := c_1' + \nu$ with ν sufficiently small since

$$c_1^{c_1'} \geq c_2^{c_1'+\nu} \quad \Leftrightarrow \quad \nu \leq c_1' \frac{\ln c_1 - \ln c_2}{\ln c_2}.$$

For $c_1 \geq c_2$ this choice is always possible in such a way that $\nu \geq 0$. If follows that for each function $f \in \{f_\Upsilon, f_H, f_{it}\}$, we can find a constant c_f' such that $(c_{max}\alpha)^{c_f'\alpha n d} \geq f(\alpha)$ for $\alpha < 1$. Then, letting c_{max}' be the maximum of these constants c_f gives a function f_{max} with the required properties.

Accordingly, (5.23) can be bounded by

$$\mathbf{P}[\,|F| = y\,] \leq n^3 \begin{cases} \max\big(\mathbf{P}\big[\,|F_{SDP}| = \frac{y}{3}\,\big]\,,\, f_{max}(\frac{y}{3n})\big) & \text{if} \quad \frac{y}{3} \leq \alpha_0 n, \\ \max\big(\mathbf{P}\big[\,|F_{SDP}| = \frac{y}{3}\,\big]\,, \mathbf{P}\big[\,|\overline{H}| = \frac{y}{3}\,\big]\big) & \text{if} \quad \frac{y}{3} > \epsilon n, \\ \max\big(\mathbf{P}\big[\,|F_{SDP}| = \frac{y}{3}\,\big]\,, f_{max}(\frac{y}{3n}),\ \mathbf{P}\big[\,|\overline{H}| = \frac{y}{3}\,\big]\big) & \text{otherwise.} \end{cases}$$

Returning to the evaluation of $\mathbf{P}[\,|F| = y\,] \cdot \binom{n}{y} \textbf{3-COL}\,(y)$ in Equation (5.21), we therefore can certainly guarantee the required polynomial bound on $\mathbf{E}[\,t\,]$ by asserting the following three relations

$$\mathbf{P}\Big[\,|F_{SDP}| = \frac{y}{3}\,\Big] \cdot \binom{n}{y} \textbf{3-COL}\,(y) = \mathcal{O}(1) \quad \text{for all} \quad y \leq n, \tag{5.24a}$$

$$\mathbf{P}\Big[\,|\overline{H}| = \frac{y}{3}\,\Big] \cdot \binom{n}{y} \textbf{3-COL}\,(y) = \mathcal{O}(1) \quad \text{for all} \quad \frac{y}{3} > \alpha_0 n, \tag{5.24b}$$

$$f_{max}\left(\frac{y}{3n}\right) \cdot \binom{n}{y} \textbf{3-COL}\,(y) = \mathcal{O}(1) \quad \text{for all} \quad \frac{y}{3} < \epsilon n. \tag{5.24c}$$

The first of these equations can be derived with the help of Corollary 5.3.5: Let $y = \alpha n$. Then

$$\begin{aligned} \mathbf{P}\Big[\,|F_{SDP}| = \frac{y}{3}\,\Big] \cdot \binom{n}{y} \textbf{3-COL}\,(y) \;&=\; \mathbf{P}\Big[\,|F_{SDP}| = \frac{\alpha n}{3}\,\Big] \cdot \binom{n}{\alpha n} \textbf{3-COL}\,(\alpha n) \\ &\overset{(5.2)}{\leq}\; 10^{-n} \cdot \left(\frac{e}{\alpha}\right)^{\alpha n} 3^{\alpha n} \leq 10^{-n} \cdot 10^{n} \end{aligned}$$

since $(e \cdot 3/\alpha)^\alpha < 10$ for all $0 < \alpha \leq 1$.

Moreover, (5.24b) and (5.24c) are proven below in Lemma 5.7.4 and Lemma 5.7.3 respectively. This concludes the proof of Lemma 5.7.1. □

So the remainder of this section is dedicated to establishing the second and the third assertion of (5.24). We start by showing (5.24c) in Lemma 5.7.3. Here we make use of the terminology introduced in the proof of Lemma 5.7.1.

Lemma 5.7.3
For $\alpha/3 < \epsilon$,

$$f_{max}\left(\frac{\alpha}{3}\right) \cdot \binom{n}{\alpha n} \textbf{3-COL}\,(\alpha n)$$

tends to a constant as n goes to infinity.

Proof:
A straight forward calculation shows

$$\begin{aligned} f_{max}\left(\frac{\alpha}{3}\right)\cdot\binom{n}{\alpha n}\textbf{3-COL}\,(\alpha n) &\le (c_{max}\alpha/3)^{c'_{max}d\cdot\alpha n/3}\cdot\binom{n}{\alpha n}\cdot\exp(\ln 3\cdot\alpha n)\\ &\overset{(5.3)}{\le} \exp\left(\left((\ln c_{max}+\ln\alpha-\ln 3)\,c'_{max}\frac{d}{3}-2\log\alpha+\ln 3\right)\alpha n\right)\\ &\le \exp\left(\left(\ln c_{max}\cdot\frac{c'_{max}}{3}d+\ln\alpha\left(c'_{max}\frac{d}{3}-3\right)\right)\alpha n\right), \end{aligned}$$

which is of order $\mathcal{O}(1)$ for

$$\ln\alpha < \frac{\ln c_{max}\cdot c'_{max}\frac{d}{3}}{3-c'_{max}\frac{d}{3}}.$$

By Observation 5.2.4 this is satisfied for

$$\ln\alpha < 2\frac{\ln c_{max}\cdot c'_{max}\frac{d}{3}}{-c'_{max}\frac{d}{3}} = \ln\frac{1}{c^2_{max}}$$

since $c_{max} > 1$. Thus, Lemma 5.7.3 follows if we choose

$$\epsilon \le \frac{1}{3c^2_{max}}. \tag{5.25}$$

□

For completing the proof of Lemma 5.7.1 it remains to complement the preceeding result by an argument for (5.24b). This is provided by the following lemma.

Lemma 5.7.4
For $\alpha/3 > \alpha_0$,

$$\mathbf{P}\left[\,|\overline{H}| = \frac{\alpha}{3}n\,\right]\cdot\binom{n}{\alpha n}\textbf{3-COL}\,(\alpha n)$$

is of order $\mathcal{O}(1)$.

Proof:
Set $\alpha' := \alpha/3$ and recall that

$$\mathbf{P}\left[\,|\overline{H}| = \alpha' n\,\right] \le f_{(0)}(\alpha') + f_{it}(\alpha') \tag{5.26}$$

with $f_{(0)}(\alpha') = \exp\bigl(-\alpha' n\left(\log\alpha' + c_{(0)}\cdot d\right)\bigr)$ and $f_{it}(\alpha') = (c_{it}\alpha')^{c'_{it}d\cdot\alpha' n}$. Notice that $f_{it}(\alpha') \ge 1$ for $\alpha' \ge 1/c_{it} \ge 1/c_{max} > \epsilon$ and so this bound on $\mathbf{P}\left[\,|\overline{H}| = \alpha' n\,\right]$ gets trivial for such α'. Therefore, we proceed by discussing the cases $\alpha_0 \le \alpha' < \epsilon$ and $\alpha' \ge \epsilon$ separately. In both cases we will derive a constant bound on

$$\mathbf{P}\left[\,|\overline{H}| = \alpha' n\,\right]\cdot\binom{n}{\alpha n}\textbf{3-COL}\,(\alpha n) = \mathbf{P}\left[\,|\overline{H}| = \frac{\alpha}{3}n\,\right]\cdot\binom{n}{\alpha n}\textbf{3-COL}\,(\alpha n)\,. \tag{5.27}$$

This establishes the lemma.

First, assume $\alpha' < \epsilon$. We evaluate the two terms $f_{(0)}(\alpha')$ and $f_{it}(\alpha')$ contributing to the bound in (5.26) independently. By Observation 5.7.2 and Lemma 5.7.3, we know that

$$f_{it}\left(\frac{\alpha}{3}\right) \cdot \binom{n}{\alpha n} \mathbf{3\text{-}COL}\,(3\alpha n) = \mathcal{O}(1)\,.$$

Using (5.3) we obtain a corresponding result for $f_{(0)}(\alpha')$:

$$\begin{aligned} f_{(0)}\left(\frac{\alpha}{3}\right) \cdot \binom{n}{\alpha n} \mathbf{3\text{-}COL}\,(\alpha n) &\le \exp\Bigl(-\frac{\alpha}{3} n \left(\log\frac{\alpha}{3} + c_{(0)} \cdot d\right)\Bigr) \cdot \binom{n}{\alpha n} \cdot \exp(\ln 3 \cdot \alpha n) \\ &\overset{(5.3)}{\le} \exp\Bigl(-\frac{\alpha}{3} n \left(\log\alpha - \log 3 + c_{(0)} \cdot d + 3 \cdot 2\log(\alpha) - 3\ln 3\right)\Bigr) \\ &\overset{(5.7)}{\le} \exp\Bigl(-\frac{\alpha}{3} n \left(6\log\alpha + \frac{6}{10} c_{(0)} \cdot d\right)\Bigr)\,. \end{aligned}$$

This term tends to a constant for

$$-\log\alpha < \frac{1}{10} c_{(0)} \cdot d,$$

which holds for $\alpha/3 = \alpha' > \alpha_0$ since we chose $\alpha_0 = 2^{-c_{(0)}d/10}$ in Section 5.4.1 (see page 43). This settles the case $\alpha_0 < \alpha' < \epsilon$.

For $\alpha' \ge \epsilon$ we need to follow a different strategy. By monotonicity we have

$$\mathbf{P}\bigl[\,|\overline{H}| = \alpha' n\,\bigr] \le \mathbf{P}\bigl[\,|\overline{H}| \ge \alpha' n\,\bigr] \le \mathbf{P}\bigl[\,|\overline{H}| \ge \alpha'' n\,\bigr]$$

for $\alpha'' \le \alpha'$. Moreover,

$$\binom{n}{\alpha n} \mathbf{3\text{-}COL}\,(\alpha n) \overset{(5.2)}{\le} \left(\frac{e}{\alpha}\right)^{\alpha n} 3^{\alpha n} \le 10^n$$

since $(e \cdot 3/\alpha)^{\alpha} < 10$ for all $0 < \alpha \le 1$. Accordingly, (5.27) can be bounded from above by

$$\mathbf{P}\bigl[\,|\overline{H}| \ge \epsilon n\,\bigr] \cdot 10^n \le f_{(0)}(\epsilon) \cdot 10^n + f_{it}(\epsilon) \cdot 10^n.$$

For these two summands we can now easily provide constant bounds:

$$\begin{aligned} f_{(0)}(\epsilon) \cdot 10^n &= \exp(-\epsilon n\,(\log\epsilon + \Omega\,(d)))\,\exp(\ln 10 \cdot n) = \exp(n(\mathcal{O}(1) - \Omega\,(d))) \\ f_{it}(\epsilon) \cdot 10^n &= (c_{it} \cdot \epsilon)^{c'_{it} d \cdot \epsilon n}\,\exp(\ln 10 \cdot n) = \exp(n(\mathcal{O}(1) - \Omega\,(d)))\,. \end{aligned}$$

□

Observe that the second part of this proof demonstrates in particular that Algorithm 5.1 can even afford to use the recovery procedure on the whole graph G if $|\overline{H}|$ grows beyond ϵn.

6 Conclusions

"I conclude that there are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies and the other way is to make it so complicated that there are no obvious deficiencies."

Tony Hoare

We proved that random 3-colorable graphs taken from $\mathcal{G}_{n,p,3}$ can be 3-colored in polynomial time on average if $p \geq c/n$, where c is some sufficiently large constant. For obtaining this result we applied techniques from the theory of semidefinite programming developed for problems in combinatorial optimization. In combination with the methods designed by Alon and Kahale in [2] for coloring $\mathcal{G}_{n,p,3}$ with high probability, these techniques led to an algorithm that achieves the desired running time complexity.

The methods developed here can also be used for obtaining a similar result for $\mathcal{G}_{n,p,k}$ with values of k other than 3. More precisely, the calculations presented in this thesis carry over to arbitrary k for $pn \geq c_k$ where c_k is a constant depending on k.

One obvious question is whether it is possible to design an algorithm for coloring $\mathcal{G}_{n,p,k}$ in polynomial expected running time for all values of p. In particular, it is not clear how to deal with graphs of density $pn = c'$ for constant $c' \ll c_k$. Additionally, one could require the coloring that is constructed not only to be valid but to be optimal. Asserting this for graphs G from $\mathcal{G}_{n,p,3}$ is trivial: If $\chi(G) < 3$ then G is bipartite and can easily be colored optimally. It is not obvious how optimality can be ensured if $k > 3$.

While good standard solutions for solving semidefinite programs are available by now, calculating these kind of problems still consumes an undesirable amount of time. This often makes methods based on SDPs impracticable for graphs on more than 10^4 vertices. Spectral properties of graphs can be calculated more easily. It is therefore natural to ask whether the semidefinite programming techniques used in the algorithm COLOR$\mathcal{G}_{n,p,3}$(G) may be replaced by calculations merely involving the spectrum of G, similar to those suggested by Alon and Kahale [2].

Since the study of random graphs revealed that these objects obey very special structural properties, several authors have suggested the usage of *semirandom models* for inspecting the behaviour of algorithms in a somewhat more realistic setting. In such models the random generation of a graph is usually combined with an *adversary* that is allowed to insert edges in a restricted manner.

Among such semirandom models, $\mathcal{G}^*_{n,p,k}$ is of particular interest to this work since it is an extension of the random graph $\mathcal{G}_{n,p,k}$ as suggested by its name. For obtaining a graph from $\mathcal{G}^*_{n,p,k}$ the edge set of a graph taken from $\mathcal{G}_{n,p,k}$ is complemented by an adversary who may insert an arbitrary number of valid edges, i.e., edges between different color classes. This model was studied in [14] where the following result could be established.

Theorem 7 (Coja-Oghlan [14])
For $k = k(n)$ and $p = p(n)$ with $np \geq c\max\bigl(k \cdot \ln(n), k^2\bigr)$ where c is a certain constant, $\mathcal{G}^*_{n,p,k}$ can be k-colored in polynomial time on average.

In the same article, a negative result shows that this is near to best possible. However, the techniques used for proving this result rely on the fact that the adversary may embed an arbitrary k-colorable graph into $\mathcal{G}_{n,p,k}$ by adding many new edges. If the adversary, on the other hand, was only allowed to add a small constant number of new edges to the graph an algorithm for k-coloring $\mathcal{G}_{n,p,k}$ would certainly carry over to $\mathcal{G}^*_{n,p,k}$. Therefore, one could ask how the adversary's power can be restricted in such a way that the range of p for which polynomial expected time coloring algorithms exist can be increased.

Acknowledgements

“It would be quite unfair to expect a machine straight from the factory to compete on equal terms with a university graduate. The graduate has had contact with human beeings for twenty years or more. This contact has throughout that period been modifying his behaviour pattern. His teachers have been intentionally trying to modify it. At the end of the period a large number of standard routines will have been superimposed on the original pattern of his brain.”

Alan Turing

I would like to thank my supervisor Dr. Amin Coja-Oghlan for suggesting the topic of this thesis to me and for his excellent guidance, for many helpful explanations and fruitful discussions. Without his support and assistance this thesis would not have been possible.

In addition I am indepted to Dr. Mihyun Kang. Her careful examination of this text and her valuable comments added considerably to the quality of this thesis.

I am also grateful to the research group for algorithms and complexity at the department of computer science at Humboldt-Universität Berlin for providing a working atmosphere I truly enjoyed during the past two years. In particular I would like to thank Eva Sandig not only for frequently updating the stock of pencils but also for many encouraging words, and to Ralf Oelschlägel for helping me out of any trouble with computers that I had got myself into.

Late-night hours and other unexpected incidents gave rise to various mistakes in my writing and thinking. I am convinced that not many of these mistakes survived the careful reading of Dorotha Drake, Matthias Füssel, Stefan Kirchner, Jan Kretzschmar, and Dirk Schlatter. So, thanks for their corrections and for their comments on earlier versions of this thesis.

Thanks to Lars Kuhtz for helpful discussions on technical details and for bringing a huge roll of paper to our office, to Valentin Ziegler for sharing experience, tea and chocolate, and to Stefan Kirchner, Dirk Schlatter, and Taral Guldahl Seierstad for understanding that work on a diploma thesis should regularly be interrupted by playing frisbee.

Most important of all, I would like to thank my family and friends for supporting me emotionally and financially during the past six years. It is due to them that life in Berlin and Toronto was as interesting and versatile as I had hoped it would be.

Bibliography

“ How far my efforts agree with those of other philosophers I will not decide. Indeed what I have here written makes no claim to novelty in points of detail; and therefore I give no references, because it is indifferent to me whether what I have thought has already been thought before me by another.”

LUDWIG WITTGENSTEIN

[1] D. Achlioptas and E. Friedgut. A sharp threshold for k-colorability. *Random Structures and Algorithms*, 14(1):63–70, 1999. 3.2

[2] N. Alon and N. Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM Journal on Computing*, 26(6):1733–1748, 1997. (document), 1, 1, 3.2, 3.3, 5.1, 5.6, 5.6, 6

[3] N. Alon and M. Krivelevich. The concentration of the chromatic number of random graphs. *Combinatorica*, 17:303–313, 1997. 3.2

[4] M. Biró, M. Hujter, and Z. Tuza. Cross fertilisation of graph theory and aircraft maintenance scheduling. In *Proceedings of the Thirty-Second Annual Symposium of the Airline Group of the International Federation of Operational Research Societies*, pages 307–318, 1992. 3

[5] A. Blum and D. Karger. An $\tilde{\mathcal{O}}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Information Processing Letters*, 61(1):49–53, 1997. 3.1

[6] A. Blum and J. Spencer. Coloring random and semi-random k-colorable graphs. *Journal of Algorithms*, 19(2):204–234, 1995. 3.2

[7] B. Bollobás. The chromatic number of random graphs. *Combinatorica*, 8:49–55, 1988. 3.2

[8] S. Boyd and L. Vandenberghe. Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization. In *Communications, Computation, Control and Signal Processing: a Tribute to Thomas Kailath*, chapter 15, pages 279–288. Kluwer, 1997. 4

[9] G. J. Chaitin. Register allocation & spilling via graph coloring. In *Proceedings of the ACM SIGPLAN 82 Symposium on Compiler Construction*, pages 98–105, 1982. 1

[10] M. Charikar. On semidefinite programming relaxations for graph coloring and vertex cover. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 616–620, 2002. 3.1

[11] V. Chvátal. The strong perfect graph theorem. http://www.cs.rutgers.edu/~chvatal/perfect/spgt.html. 3

[12] V. Chvátal. *Linear Programming*. Freeman, New York, 1983. 4

[13] A. Coja-Oghlan. The Lovász number of random graphs. Accepted for publication, Preprint available from http://www.informatik.hu-berlin.de/~coja/, preliminary version in Proceedings of the 7th International Workshop on Randomization and Approximation Techniques in Computer Science, 228–239, 2003. 3.3

[14] A. Coja-Oghlan. Coloring semirandom graphs optimally. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, pages 383–395, 2004. 1, 1, 3.3, 4.4, 5.3, 6, 7

[15] A. Coja-Oghlan, C. Moore, and V. Sanwalani. MAX k-CUT and approximating the chromatic number of random graphs. In *Proceedings of the 30th International Colloquium on Automata, Languages and Programming*, pages 200–211, 2003. 3.3, 5.3, 6

[16] A. Coja-Oghlan and A. Taraz. Exact and approximative algorithms for coloring G(n,p). *Random Structures and Algorithms*, 24(3):259–278, 2004. 3.3

[17] T. Denton, M. Demirci, J. Abrahamson, A. Shokoufandeh, and S. Dickinson. Selecting canonical views for view-based 3-d object recognition. In *Proceedings of the 17th International Conference on Pattern Recognition*, 2004. 4

[18] M. E. Dyer and A. M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *Journal of Algorithms*, 10:451–489, 1989. 1, 3.2, 3.3

[19] K. Edwards. The complexity of coloring problems on dense graphs. *Theoretical Computer Science*, 43(2-3):337–334, 1986. 3

[20] L. Engebretsen and J. Holmerin. Towards optimal lower bounds for clique and chromatic number. *Theoretical Computer Science*, 299(1-3):537–584, 2003. 3.1

[21] B. Fares, D. Noll, and P. Apkarian. Robust control via sequential semidefinite programming. *SIAM Journal on Control and Optimization*, 40(6):1791–1820, 2002. 4

[22] U. Feige and J. Kilian. Zero knowledge and the chromatic number. *Journal of Computer and System Sciences*, 57(2):187–199, 1998. (document), 1, 3.1

[23] U. Feige, M. Langberg, and G. Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. *SIAM Journal on Computing*, 33(6):1338–1368, 2004. 3.1

[24] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. John Wiley & Sons, New York, 1971. 5.2

[25] A. M. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, 18:61–77, 1997. 4.3, 5.3

[26] M. R. Garey and D. S. Johnson. *Computers and Intractability*. W.H. Freeman and Company, 1979. 1, 2.3

[27] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995. 4.1, 4.2, 4.2, 4.3

[28] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981. 4

[29] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, Berlin, 1993. 4

[30] M. M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Information Processing Letters*, 45(1):19–23, 1993. 3.1

[31] J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001. 4.2

[32] C. Helmberg. SDP software.
http://www-user.tu-chemnitz.de/~helmberg/sdp_software.html. 4

[33] C. Helmberg. Semidefinite programming for combinatorial optimization. Habilitationsschrift. Report ZR-00-34, Zuse Institut Berlin, 2000. 2.4.2, 3, 4, 4.3

[34] S. Janson, T. Łuczak, and A. Ruciński. *Random Graphs*. John Wiley & Sons, New York, 2000. 3.2, 5.2

[35] T. R. Jensen and B. Toft. *Graph Coloring Problems*. John Wiley & Sons, New York, 1995. 3

[36] D. Johnson. Worst case behaviour of graph coloring algorithms. In *Proceedings of the 5th South Eastern Conference on Combinatorics, Graph Theory and Computing. Congressus Numerantium*, pages 513–527, 1974. 3.1

[37] D. R. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45:246–265, 1998. 3.1, 4.1, 4.4, 5.2

[38] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984. 4

[39] R. Karp. The probabilistic analysis of combinatorial optimization algorithms. In *Proceedings of the International Congress of Mathematicians*, pages 1601–1609, 1984. 1

[40] S. Khanna, N. Linial, and S. Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000. 1, 3.1

[41] D. Král, J. Kratochvíl, Z. Tuza, and J. Woeginger. Complexity of coloring graphs without forbidden induced subgraphs. In *Proceedings of the 27th International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 254–262, 2001. 3

[42] M. Krivelevich. Deciding k-colorability in expected polynomial time. *Information Processing Letters*, 81:1–6, 2002. 3.3

[43] M. Krivelevich and V. H. Vu. Approximating the independence number and the chromatic number in expected polynomial time. *Journal of Combinatorial Optimization*, 6:143–155, 2002. 3.3

[44] L. Kučera. Expected behavior of graph colouring algorithms. In *Proceedings of the 1977 International Conference on Fundamentals of Computation Theory*, pages 447–451, 1977. 1, 3.2

[45] L. Kuhtz. Colouring $\mathcal{G}_{n,p}$ and spectral techniqes. Diplomarbeit. Humboldt-Universität zu Berlin, 2004. 3.3

[46] M. Laurent and F. Rendl. Semidefinite programming and integer programming. Report PNA-R0210, CWI, Amsterdam, 2002. 2.4

[47] F. T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Reasearch of the National Bureau of Standards*, 84:489–506, 1979. 1

[48] A. Lewis and M. Overton. Eigenvalue optimization. *Acta Numerica*, 5:149–190, 1996. 4

[49] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, IT-25:1–7, 1979. 1, 4.1

[50] T. Łuczak. The chromatic number of random graphs. *Combinatorica*, 11:45–54, 1991. 3.2

[51] T. Łuczak. A note on the sharp concentration of the chromatic number of random graphs. *Combinatorica*, 11:295–297, 1991. 3.2

[52] J. W. Moon. Various proofs of Cayley's formula for counting trees. In *A Seminar on Graph Theory*, pages 70–78. Holt, Rinehart and Winston, Inc., 1976. 5.6

[53] R. Niemistö, B. Dumitrescu, and I. Tabus. SDP design procedures for near-optimum IIR compaction filters. *Signal Processing*, 82(6):911–924, 2002. 4

[54] S. Olariu. Optimal greedy heuristic to color interval graphs. *Information Processing Letters*, 37(1):21–25, 1991. 3

[55] A. D. Petford and D. J. A. Welsh. A randomised 3-colouring algorithm. *Discrete Mathematics*, 74:253–261, 1989. 3.2

[56] A. Rényi. *Probability Theory.* Elsevier, New York, 1970. 5

[57] N. Robertson, D. P. Sanders, P. Seymour, and R. Thomas. Efficiently four-coloring planar graphs. In *Proceedings of the 28th ACM Symposium on the Theory of Computing*, pages 571–575, 1996. 3

[58] S. Sahni and T. Gonzales. $\mathcal{P}$-complete approximation problems. *Journal of the ACM*, 23(3):555–565, 1976. 4.2

[59] E. Shamir and J. H. Spencer. Sharp concentration of the chromatic number on random graphs $\mathcal{G}_{n,p}$. *Combinatorica*, 7(1):121–129, 1987. 3.2

[60] A. Singh and M. Marek-Sadowska. Circuit clustering using graph coloring. In *Proceedings of the 1999 International Symposium on Physical Design*, pages 164–169, 1999. 1

[61] S. S. Skiena. *The Algorithm Design Manual*. Springer-Verlag, New York, 1998. 1

[62] C. R. Subramanian. Algorithms for coloring random k-colorable graphs. *Combinatorics, Probability and Computing*, 9:45–77, 2000. 1, 1, 3.3

[63] L. Trevisan, G. Sorkin, M. Sudan, and D. Williamson. Gadgets, approximation, and linear programming. *SIAM Journal on Computing*, 29:2074–2097, 2000. 4.2

[64] J. S. Turner. Almost all k-colorable graphs are easy to color. *Journal of Algorithms*, 9:253–261, 1988. 1, 3.2

[65] R. J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer, 1997. 2.4.1, 4

[66] E. W. Weisstein. Chi distribution.
http://mathworld.wolfram.com/ChiDistribution.html.
From MathWorld – A Wolfram Web Resource. 2.2

[67] A. Wigderson. Improving the performance guarantee for approximate graph coloring. *Journal of the ACM*, 30(4):729–735, 1983. 3.1

[68] D. C. Wood. A technique for coloring a graph applicable to large scale optimization problems. *Computer Journal*, 12:317, 1969. 1