Coloring random graphs – an algorithmic perspective

Michael Krivelevich

1 Introduction

Algorithmic Graph Coloring and Random Graphs have long become one of the most prominent branches of Combinatorics and Combinatorial Optimization. It is thus very natural to expect that their mixture will produce quite many very attractive, diverse and challenging problems. And indeed, the last thirty or so years witnessed rapid growth of the field of Algorithmic Random Graph Coloring, with many researchers working in this area and bringing there their experience from different directions of Combinatorics, Probability and Computer Science. One of the most distinctive features of this field is indeed the diversity of tools and approaches used to tackle its central problems.

This survey is not intended to be a very detailed, monograph-like coverage of Algorithmic Random Graph Coloring. Instead, our aim is to acquaint the reader with several of the main problems in the field and to show several of the approaches that proved most fruitful in attacking those problems. We do not and we simply cannot provide all details of the proofs, referring the (hopefully) enthusiastic reader to the papers where those proofs are presented in full, or to his/her previous experience in Random Graphs, which should be sufficient to recover many sketched arguments. But of course, the best way to become fluent in this field is to learn from the best, most influential papers, and above all, to engage in independent research, which will undoubtedly bring new and exciting results.

2 Graph coloring is hard

Graph coloring ([21]) has long been one of the central notions in Graph Theory and Combinatorial Optimization. Great many diverse problems can be formulated in terms of finding a coloring of a given graph in a small number of colors or calculating, exactly or approximately, the chromatic number of the graph. Unfortunately, it turns out that these computational problems are very hard. Karp proved already in 1972 [25] that is it NP-complete to decide, for any fixed $k \geq 3$, whether a given graph G is k-colorable. Recent results show that one should not even hope to obtain an efficient algorithm which approximates the chromatic number within a non-trivial approximation ratio. Specifically, Feige and Kilian proved [12] that, unless $coRP = NP$, there is no approximation algorithm for the chromatic number whose approximation ratio over graphs on n vertices is less than $n^{1-\epsilon}$, for any fixed $\epsilon > 0$. Coloring 3-colorable graphs in four colors is NP-complete as well ([28], [18]). Altogether, the situation does not appear particularly encouraging, from both theoretical and practical points of view.

3 The chromatic number of random graphs

The picture changes dramatically when one switches from the somewhat pessimistic worst case scenario to possibly more applicable in practice average case analysis. Already the term "average case analysis" suggests that there should be some underlying probability distribution, whose ground set is composed of graphs, and whose purpose is to help to measure the typical or average performance of various coloring algorithms. Usually this underlying probability space is composed of graphs with the same number n of vertices.

It appears that the most natural and interesting definition of the probability measure on graphs on n vertices is the so called *binomial random graph* $G(n,p)$. This is the probability space whose elements are all labeled graphs $G=(V,E)$ with vertex set $V=\{1,\dots,n\}$, where each pair of vertices $(i,j): 1\le i<j\le n$ is chosen to be an edge of G independently and with probability p, in general p may be a function of the number of vertices n: $p=p(n)$. Thus $G(n,p)$ can be viewed as a product probability space, formed by $\binom{n}{2}$ i.i.d. Bernoulli random variables with parameter p. The probability of an individual graph G on n vertices in $G(n,p)$ is easily seen to be $Pr[G]=p^{|E(G)}(1-p)^{\binom{n}{2}-|E(G)|}$. The special case $p=0.5$ occupies a very prominent position in the study of random graphs, as for this case the probabilities of every pair (i,j) to be an edge or be a non-edge are equal, resulting in the uniform distribution on the set on all labeled graphs on n vertices: $Pr[G]=2^{-\binom{n}{2}}$. Therefore studying asymptotic properties of the random graph $G(n,0.5)$ is in a sense equivalent to counting graphs on n vertices with specified properties.

Usually asymptotic properties of random graphs $G(n,p)$ are of interest. For this reason we will assume that the number of vertices n tends to infinity. Also, for a graph property A (where "graph property" means just a family of graphs closed under isomorphism), we say that A holds *almost surely*, or a.s. for brevity, in $G(n,p)$, if the probability that a random graph G, drawn according to the distribution $G(n,p)$, possesses A tends to 1 as n tends to infinity. With some abuse of notation we will use $G(n,p)$ both for the probability distribution on graphs on n vertices and for a graph G drawn from this distribution.

The theory of random graphs is one of the most rapidly developing areas of Combinatorics, with already thousands papers devoted to the subject. We certainly do not intend to cover it here, instead referring the reader to recent monographs [20] and [9]. We will however represent the state of the art of one aspect of random graphs, relevant to the subject of this survey – the asymptotic behavior of the chromatic number of random graphs.

To begin with, consider the most important case $p=0.5$. For an integer k, let

$$f(k)=\binom{n}{k}\left(\frac{1}{2}\right)^{\binom{k}{2}} \ . \tag{1}$$

Obviously, $f(k)$ is just the expectation of the number of independent sets of size k in $G(n,0.5)$. When $f(k)=o(1)$, this expectation tends to zero as n grows, and applying Markov's inequality we get immediately that a.s. $G(n,0.5)$ does not contain an independent set of size k. We thus set

$$k_0=\max\{k: f(k)\ge 1\} \ . \tag{2}$$

Applying standard asymptotic estimates for the binomial coefficients, one can easily solve asymptotically the above equation for k_0, getting $k_0 = (1-o(1))2\log_2 n$. Hence a.s. $\alpha(G(n,0.5)) \le (1-o(1))2\log_2 n$. Providing a matching lower bound on the independence number of $G(n,0.5)$ requires more effort, but this can be done as follows [41]. Let X_k be a random variable, counting the number of independent sets of size k in $G(n,0.5)$. Clearly, $E[X_k] = f(k)$. If k is chosen around k_0, using direct calculations one can show that $VAR[X_k] = o(E[X_k]^2)$. Thus Chebyshev's inequality applies, and we get that X_k is concentrated around its mean. In particular, when $f(k) \to \infty$, we derive that a.s. $X_k \ge 1$, which means exactly that $\alpha(G) \ge k$. To satisfy the former condition it is enough to choose $k = k_0$ or $k = k_0 + 1$. Altogether we get that a.s. $\alpha(G(n,0.5)) = (1-o(1))2\log_2 n$.

As for every graph G, $\chi(G) \ge |V(G)|/\alpha(G)$, the above asymptotic (upper) bound on $\alpha(G(n,0.5))$ supplies an easy asymptotic bound for the chromatic number – a.s. $\chi(G(n,0.5)) \ge (1+o(1))n/(2\log_2 n)$. Providing a matching upper bound for the chromatic number was one of the major open questions in the theory of random graphs for about quarter of a century until Béla Bollobás [8] discovered a very inspiring proof of the following theorem.

Theorem 3.1 *Almost surely in the probability space* $G(n,0.5)$, $\chi(G) \le (1+o(1))\frac{n}{2\log_2 n}$.

Proof. Set $m = n/\log^2 n$ and define

$$k_1 = \max\left\{k : \binom{m}{k}\left(\frac{1}{2}\right)^{\binom{k}{2}} \ge n^3\right\}.$$

The parameter k_1 is chosen so as to guarantee that the expected number of independent subsets of size k_1 in a *fixed* subset $V_0 \subseteq V$ of m vertices is at least n^3. An asymptotic computation very similar to the one mentioned above for k_0 shows that still $k_1 = (1-o(1))2\log_2 n$.

The theorem will easily follow from the following lemma.

Lemma 3.2 *Almost surely in* $G(n,0.5)$, *every subset* V_0 *of* m *vertices contains an independent set of size* k_1.

We will return to the proof of the lemma shortly, but let us see first how it implies Theorem 3.1. Assume that a graph G on n vertices satisfies the conclusion of the lemma. We will prove then that $\chi(G) < n/k_1 + m = (1+o(1))n/(2\log_2 n)$. To show this, we will act in a rather typical for existential coloring arguments way, coloring the graph G by excavation. As long as G contains at least m uncolored vertices, there exists an independent set I of size k_1 in G, all of whose vertices are still uncolored. We then color I by a fresh color and discard all of its vertices from the graph. Clearly this procedure is repeated at most n/k_1 times. Once less than m vertices are left uncolored, we can color each one of them in a new and separate color, resulting in less than m additional colors. The total number of colors used by the above argument is less than $n/k_1 + m$, as promised.

Of course, the above derivation was pretty easy, so the crux of the proof of Theorem 3.1 lies in proving Lemma 3.2. In order to prove Lemma 3.2, fix a subset $V_0 \subset V$ of cardinality $|V_0| = m$. The subgraph of $G(n,0.5)$, induced by V_0, behaves like a random graph $G(m, 0.5)$. Let Y be the number of independent sets of size k_1 inside V_0. Then

$$E[Y] = \binom{m}{k} \left(\frac{1}{2}\right)^{\binom{k}{2}} \geq n^3 ,$$

due to the definition of k_1. To prove the lemma it is enough to prove that

$$Pr[Y = 0] \ll \frac{1}{\binom{n}{m}} , \tag{3}$$

as then by the simple Union bound almost surely every subset of size m in $G(n, 0.5)$ contains an independent set of the required size.

Notice that (3) is a typical large deviation statement – one needs to show that the probability that a random variable (Y) deviates from its expectation ($E[Y] \geq n^3$) by a large quantity (n^3) is exponentially small. However, this task, rather standard and accessible by now, was very challenging fifteen years ago! The main contribution of Bollobás was first to switch from Y to another random variable Z, easier to tackle and such that the positivity of Z implies the positivity of Y, and then to provide an exponential bound for $Pr[Z = 0]$, using martingales – a very novel by then tool for combinatorialists. Currently, there are at least three alternative proofs of Lemma 3.2, based on three different large deviation techniques – the one of Bollobás through martingales, a proof through the so called generalized Janson Inequality, and a proof using the Talagrand concentration of measure inequality. Since all three of them require some technical calculations, we prefer not to present any of them here, instead suggesting the reader to consult [5], where all three tools are discussed in great details.

Bollobas' argument works also for smaller values of $p(n)$ down to $p(n) = n^{-a}$ for a small positive constant a. Later, Luczak [37] was able to establish the asymptotic value of the chromatic number of $G(n, p)$ for all values of $p(n)$ down to $p(n) \geq C/n$ for a large enough constant $C > 0$:

Theorem 3.3 *There exists C_0 such that for every $p = p(n)$ satisfying $C_0/n \leq p \leq \log^{-7} n$ a.s. in $G(n, p)$*

$$\frac{np}{2\log(np) - 2\log\log(np) + 1} \leq \chi(G) \leq \frac{np}{2\log(np) - 40\log\log(np)} .$$

Luczak's argument is quite challenging technically and relies heavily on the so called *expose-and-merge approach* invented by Matula [42]. We will not discuss it here. For future reference we summarize the above discussion by noting that the chromatic number of $G(n, p)$ is almost surely $(1+o(1))n\log_2(1(1-p))/(2\log_2 n)$ for a constant edge probability p, and $(1 + o(1))np/(2\ln(np))$ for $C/n \leq p(n) \leq o(1))$, where the $o(1)$ term tends to 0 as np tends to infinity.

The above described results of Bollobás and Luczak have settled the most important problem in random graph coloring – the asymptotic value of the chromatic number of a random graph. Still many quite significant and attractive problems in this area remain unsolved, for example, the concentration of the chromatic number of random graphs ([45], [38], [2]), list coloring ([3], [29], [33]), thresholds for non-k-colorability for a fixed value of $k \geq 3$ (see a recent survey of Molloy [43]), to mention just a few. And of course, there are many algorithmic problems related to random graph coloring, some of them to be addressed later in this survey.

4 The greedy algorithm for coloring random graphs

The *greedy algorithm*, sometimes also called the first fit algorithm for reasons to become evident immediately, is probably the simplest imaginable algorithm for graph coloring. The greedy algorithm proceeds as follows: given a graph $G = (V, E)$ on n vertices, one first fixes some order $(v_1, \ldots, v_n)$ of the vertices of G, and then scans the vertices of G according to the chosen order, each time coloring a current vertex v_i in the first available color, not used by any already colored neighbor $v_j, j < i$, of v_i. Of course, the resulting number of colors may depend not only on the graph G itself, but also on the chosen order of its vertices. A distinctive feature of the greedy algorithm is that it is essentially an *online* algorithm as the color of a vertex is determined by the edges from the vertex to already seen vertices, and once the color is chosen it will remain unchanged (see [27] for a survey on online graph coloring). This fact makes the analysis of the performance of the greedy algorithm on random graphs $G(n,p)$ quite accessible, as we can generate the random graph as the algorithm proceeds, using the so called *vertex exposure* mechanism – once the algorithm reaches vertex i, a p-Bernoulli coin is flipped for each pair $(j, i), 1 \leq j < i$, to decide whether this pair is an edge of $G(n,p)$, and then a color of i is chosen based on the results of coin flips and a coloring of vertices $1, \ldots, i-1$.

The greedy algorithm turns out to be quite successful for most graphs, using about twice as many colors as the chromatic number of a graph – a remarkable achievement taking into account its simplicity and also the hardness results for graph coloring mentioned above!

Theorem 4.1 *[16] Almost all graphs on n vertices are colored by the greedy algorithm in at most $n/(\log_2 - 3\log_2\log_2 n)$ colors.*

Proof. Let $k = \lfloor \frac{n}{\log_2 n - 3\log_2\log_2 n} \rfloor$. We assume that the greedy algorithm colors the vertices of G according to their natural order $1, \ldots, n$. Denote by $\chi_g(G)$ the number of colors used by the greedy algorithm to color G. In the probability space $G(n, 0.5)$ define A_i to be the event "Vertex i is the first to get color $k+1$". Then clearly the event "$\chi_g(G) > k$" is the union of the events $A_i, 1 \leq i \leq n$, which are pairwise disjoint, and thus:

$$Pr[\chi_g(G) > k] = Pr[\bigcup_{i=1}^{n} A_i] = \sum_{i=1}^{n} Pr[A_i]\,,$$

and the theorem will follow if we will prove $Pr[A_i] = o(1/n)$ for all i.

Consider vertex i of G. The probability $Pr[A_i]$ obviously depends only on the coloring of preceding vertices $1, \ldots, i-1$. Moreover, we can assume that exactly k colors have been used by the algorithm to color those vertices, for otherwise $Pr[A_i] = 0$. So we fix a k-coloring $(C_1, \ldots, C_k)$ of $\{1, \ldots, i-1\}$ and estimate the conditional probability $Pr[A_i|(C_1, \ldots, C_k)]$. In order to force vertex i to be colored in color $k+1$ at least one edge should connect i with each of the color classes $C_1, \ldots, C_k$. We therefore get:

$$\begin{aligned}
Pr[A_i|(C_1,\ldots,C_k)] &= \prod_{j=1}^{k}\left(1-\left(\frac{1}{2}\right)^{|C_j|}\right) \le \left(1-\left(\frac{1}{2}\right)^{\sum_{j=1}^{k}|C_j|/k}\right)^k \\
&< \left(1-\left(\frac{1}{2}\right)^{\frac{n}{k}}\right)^k < e^{-\left(\frac{1}{2}\right)^{\frac{n}{k}}k} \\
&\le e^{-\left(\frac{1}{2}\right)^{\log_2 n - 3\log_2\log_2 n}k} = e^{-\frac{k\log_2^3 n}{n}} = e^{-(1+o(1))\log_2^2 n} = o(1/n) ,
\end{aligned}$$

where the first inequality above follows from the convexity of $\log(1-(1/2)^x)$ for $x > 0$.

Grimmett and McDiarmid also showed in [16] that the upper bound on the greedy algorithm from Theorem 4.1 is asymptotically tight – almost every graph in $G(n, 0.5)$ will be colored by at least $(1+o(1))n/\log_2 n$ colors by the greedy algorithm. Moreover, almost surely *all* color classes produced by the greedy algorithm have size at most $(1+o(1))\log_2 n$. Another attractive feature of the greedy algorithm is its extreme robustness when applied to to random graphs, as shown by the following result of McDiarmid [39]:

Theorem 4.2 $Pr[\chi_g(G(n, 0.5) > (1+5\log_2\log_2 n/\log_2 n)n/\log_2 n] < 1/n^n$. *Therefore, almost every graph on* n *vertices is such that no matter which order of the vertices is chosen, the greedy algorithm uses fewer than* $(1+5\log_2\log_2 n/\log_2 n)n/\log_2 n$ *colors.*

When the edge density becomes lower, the greedy algorithm becomes less competitive. Pittel and Weishaar show in [44] that when applied in the probability space $G(n, c/n)$, the greedy algorithm almost surely outputs a coloring with $(1+o(1))\log_2\log n$ colors, while the chromatic number of most of the graphs in this probability space is bounded from above by a constant $C = C(c)$ (see, e.g., Theorem 3.3). As explained in [44], it is quite easy to see why the number of colors used by the greedy algorithm in $G(n, c/n)$ is a.s. unbounded. To show this, define a sequence of trees T_k as follows: T_1 is a single vertex, and for $k \ge 2$ the tree T_k in obtained from T_{k-1} by joining each vertex of T_{k-1} with a new pendant vertex. A standard second moment argument shows that $G(n, c/n)$ contains a.s. $\Theta(n)$ connected components isomorphic to T_k, for each fixed k. Observe that if the vertices of T_k are colored from "outside in", k colors will be required. As the graph a.s. has so many copies of T_k, at least one of them will a.s. be ordered in this adversarial way. One can however reach essentially the same approximation ratio $2+o(1)$ like in the dense case by a simple modification of the greedy algorithm as suggested by Shamir and Upfal in [46]. The algorithm of Shamir and Upfal proceeds in two phases. The first phase is the usual greedy algorithm as described above. In the second phase, called the correction phase in [46], a subgraph of G spanned by the set V_0 of all vertices that received color higher than some predetermined quantity $K(n,p) = (1+o(1))np/\ln(np)$ is considered. This set is then shown to be a.s. colorable by a breadth-first search in a bounded number of colors.

Unfortunately, the greedy algorithm is not always as good as the typical behavior analysis suggests. It is quite easy to construct an example of a bipartite graph G on n vertices for which the greedy algorithm will use a linear in n number of colors for a certain ordering of the vertices of G. Choosing an ordering of vertices at random does not necessarily help much, as Kučera shows in [35] that for every positive $\epsilon > 0$ and all sufficiently large n there exists a graph G on n vertices with chromatic number at most n^{ϵ}, for which the greedy algorithm with a randomly chosen initial vertex ordering uses almost surely at least $(1-\epsilon)n/\log_2 n$ colors.

Let us return to the most basic case $p = 0.5$. We know already that almost surely in $G(n, 0.5)$, $\chi_g(G)/\chi(G) = 2 + o(1)$. Is there a better on average coloring algorithm than the greedy coloring? Specifically,

Research Problem 1 *Does there exist a polynomial time algorithm which colors most of the graphs on n vertices in $(1-\epsilon)n/\log_2 n$ colors, for some fixed $\epsilon > 0$?*

This is certainly one of the major problems in algorithmic random graph coloring. It is instructive to observe that any such algorithm would produce also an independent set of size $(1+\epsilon')\log_2 n$ (a largest independent set in such a coloring). Therefore the coloring problem is closely related to a quarter century old question of Karp, who asked [26] for a polynomial time algorithm for finding an independent set of size $(1+\epsilon)\log_2 n$ in almost all graphs on n vertices. As we mentioned already, the greedy algorithm almost surely does not provide such a large set. Jerrum proved in [22] that the Metropolis algorithm, which in this case is a random walk on independent sets of the graph biased towards larger independent sets, also requires almost surely a super-polynomial time to reach an independent set of size $(1+\epsilon)\log_2 n$. So apparently the problem of finding a large independent set in a typical graph is hard algorithmically, the fact which has been used even for cryptographic purposes [23]. An interesting fact is that it follows from the above mentioned expose-and-merge technique of Matula that an algorithm for finding a.s. an independent set of size $(1+\epsilon)\log_2 n$ in $G(n, 0.5)$ can be used as a subroutine in an algorithm for coloring a typical graph in $(1-\epsilon')n/\log_2 n$ colors.

A marginal improvement over the greedy algorithm (Theorem 4.1) has been achieved by Krivelevich and Sudakov [31], who provided a randomized polynomial time algorithm that colors almost every graph on n vertices is $n/(\log_2 n + c\sqrt{\log_2 n})$ colors for every positive constant c, thus outputting a coloring with color classes of average size $\log_2 n + c\sqrt{\log_2 n}$, compared to $\log_2 n - \Theta(\log\log n)$ of the greedy algorithm. Again, the critical task here is to find an independent set of size $\log_2 n + c\sqrt{\log n}$. This is achieved in [31] by running the greedy algorithm for finding an independent set I of size $|I| = \log_2 n - 2c\sqrt{\log n}$ in the first $n/2$ vertices of the graph. The set U of non-neighbors of I in the second half of the graph has then almost surely about $2^{2c\sqrt{\log_2 n}}$ vertices and contains inside an independent set I_1 of size $(1-o(1))2\log_2|U| = (1-o(1))4c\sqrt{\log_2 n}$, which can be found in polynomial time by checking exhaustively all $\binom{|U|}{4c\sqrt{\log_2 n}}$ subsets of U of the appropriate size. The union of I and I_1 forms a desired independent set.

There is (at least) one reason to believe that it would be hard to break the $\log_2 n + \Theta(\sqrt{\log n})$ barrier. Going back to the expectation (1) of the number of independent sets of size k, we can easily check that $f(k)$ is polynomially smaller than the total number of independent sets in $G(n, 0.5)$ only if $k \le \log_2 n + \Theta(\sqrt{\log n})$. This may indicate that finding independent sets of larger size may take superpolynomial time. Further discussion can be found in [31].

5 Approximating the chromatic number in expected polynomial time

As discussed above, the greedy algorithm is quite successful for most of graphs, providing a coloring that uses about twice as many colors as an optimal coloring. But there are some (very rare though) graphs for which the greedy coloring performs rather miserably. On the other hand, one cannot probably expect to design a coloring algorithm that beats significantly the trivial approximation ratio n for all graphs, due to the complexity results stated in Section 2. It is therefore desirable to provide a coloring algorithm with a guaranteed approximation ratio for *all* graphs on n vertices and with running time polynomial on *average* in n. We thus arrive naturally at the concept of algorithms with expected polynomial running time.

Given an algorithm A whose domain is the set of all graphs on n vertices, and a probability distribution $Pr[\cdot]$ on the same set, the *expected running time* of A is defined as $\sum_G Pr[G]R_A(G)$, where the sum runs over all graphs on n vertices, $Pr[G]$ is the probability of G according to the chosen probability measure, and $R_A(G)$ is the running time of A on G. Thus, while looking for an algorithm A whose expected running time is polynomial, we can allow A to spend a superpolynomial time on some graphs on n vertices, but it should be efficient on average.

Observe that if an algorithm A has expected polynomial running time with respect to the probability distribution $P(\cdot)$, then A is polynomial for almost all graphs according to P. Therefore, it is more difficult to develop algorithms with expected polynomial running time than algorithms that perform the same algorithmic task for almost all graphs.

Obviously the problem is quite sensitive to the choice of the underlying probability distribution. In this section we concentrate on the case where the distribution is chosen to be $G(n,p)$, the binomial random graph. There have been a few papers addressing different probability distributions, we will discuss some of them later.

Krivelevich and Vu prove in [32] the following result on the existence of an approximate coloring algorithm with expected polynomial running time:

Theorem 5.1 *For any constant $\epsilon > 0$ the following holds. If the edge probability $p(n)$ satisfies $n^{-1/2+\epsilon} \leq p(n) \leq 0.99$, then there exists a deterministic coloring algorithm, approximating the chromatic number $\chi(G)$ within a factor $O((np)^{1/2}/\log n)$ and having polynomial expected running time over $G(n,p)$.*

Thus in the most basic case $p = 0.5$ we get a coloring algorithm with approximation ratio $O(\sqrt{n}/\log n)$ – a considerable improvement over best known approximation algorithm for the worst case [19], whose approximation ratio is only $O(n/polylog(n))$. Note also that the approximation ratio decreases with the edge probability $p(n)$.

Before describing the basic idea of the algorithm of [32], we would like to say a few words about combinatorial ideas forming the core of its analysis. As is typically the case with developing algorithms whose expected running time is polynomial, we need to distinguish efficiently between "typical" graphs in the probability space $G(n,p)$, for which it is relatively easy to provide a good approximation algorithm, and "non-typical" ones, which are rare but may be hard for approximating a desired quantity. As these rare and possibly hard graphs have an exponentially small probability in $G(n,p)$, this gives us a possibility to spend an exponential time on each of them. This in turn enables to approximate the chromatic number within the desired factor even for these graphs.

A separation between typical and non-typical instances will be made based on the first eigenvalue of an auxiliary matrix, to be defined later. Then we will apply a large deviation result to show that this eigenvalue deviates from its expectation with exponentially small probability, and thus bad instances are indeed extremely rare. Thus our main tools will come from two seemingly unrelated fields - graph spectral techniques and large deviation inequalities.

We now describe a proof of a somewhat weaker version of Theorem 5.1 for the case $p = 0.5$. Recall that according to Theorem 4.2 the probability that the greedy algorithm fails to color a graph in $G(n, 0.5)$ in $1 + o(1))n/\log_2 n$ colors is less than n^{-n}. Therefore we can run the greedy algorithm, and then apply the exhaustive search for graphs colored by more than, say, $2n/\log_2 n$ colors.

In order to show that the greedy coloring is within $\tilde{O}(\sqrt{n})$ factor from an optimal coloring of a graph G, we need to bound from below the chromatic number of G. As $\chi(G) \geq n/\alpha(G)$, it is enough to certify that $\alpha(G) = \tilde{O}(\sqrt{n})$. We thus need an efficiently computable graph parameter that bounds from above the independence number of G. Given a graph $G = (V, E)$ with vertex set $V = \{1, \ldots, n\}$ define an n-by-n matrix $M = (m_{ij})$ as follows:

$$m_{ij} = \begin{cases} 1, & \text{if } i, j \text{ are non-adjacent in } G, \\ -1, & \text{otherwise ,} \end{cases} \tag{4}$$

Then M is a real symmetric matrix and has therefore n real eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. The connection between $\alpha(G)$ and the first eigenvalue $\lambda_1(M(G))$ is given by the following lemma.

Lemma 5.2 *Let $M = M(G)$ be as defined in (4). Then $\lambda_1(M) \geq \alpha(G)$.*

Proof. Let $k = \alpha(G)$. Then M contains a k by k block of all 1's, indexed by the vertices of an independent set of size k. It follows from interlacing (see, e.g., Chapter 31 of [51]) that $\lambda_1(M) \geq \lambda_1(1_{k\times k}) = k$.

(In fact, $\lambda_1(M(G))$ is an upper bound not only for the independence number of G, but also for its Lovász theta-function [36].)

The spectrum of a real symmetric matrix can be efficiently calculated within any desired precision. So if we calculate $\lambda_1(M(G))$ and see that $\lambda_1(M(G)) = \tilde{O}(\sqrt{n})$, then we have a certificate of the desired lower bound for $\chi(G)$.

What is a typical value of $\lambda_1(M(G))$? Recall that G is distributed according to $G(n, 0.5)$, and therefore $M(G)$ is a random symmetric matrix, each of its entries above the main diagonal is independently 1 or -1 with probability 0.5. This enables us to apply known results on eigenvalues of random symmetric matrices. Füredi and Komlós proved in [14] that

$$E[\lambda_1(M)] = (1 + o(1))2\sqrt{n} \ .$$

This shows that typically $\lambda_1(M(G))$ is of order $\sqrt{n}$.

Now we need to estimate the probability that $\lambda_1(M(G))$ is significantly larger than its expectation. The desired estimate is provided by the following large deviation result from [32] (see also [4] for a more general result):

Lemma 5.3 *Let $M = (m_{ij})$ be an n-by-n random symmetric matrix with all entries bounded by 1 in their absolute values. Then for all $t > 0$,*

$$Pr[\lambda_1(M) > E[\lambda_1(M)] + t] \leq 2e^{-(1+o(1))t^2/32} \ .$$

This lemma is proven by applying the Talagrand concentration of measure inequality, see [32], [4] for details of the proof.

Plugging an estimate of Füredi and Komlös on $E[\lambda_1(M)]$ in the above lemma we conclude that $Pr[\lambda_1(M(G)) \geq 6\sqrt{n \log n}] < n^{-n}$.

Now we have at hand all necessary ingredients to formulate our coloring algorithm and analyze its performance.

Step 1. Run the greedy algorithm on G. Let C be the resulting coloring. If C uses more than $2n/\log_2 n$ colors, go to *Step 3*;
Step 2. Define $M = M(G)$ according to (4) and compute $\lambda_1(M)$. If $\lambda_1(M) \leq 6\sqrt{n \ln n}$, output C;
Step 3. Find an optimal coloring by the exhaustive search and output it.

Let us verify that the above algorithm approximates $\chi(G)$ within a factor of $O(\sqrt{n/\log n})$. If coloring C is output at Step 2, then $|C| \leq 2n/\log_2 n$ and $\chi(G) \geq n/\alpha(G) \geq n/\lambda_1(G) \geq O(\sqrt{n/\log n})$, implying $|C|/\chi(G) = O(\sqrt{n/\log n})$. Of course, if we ever get to Step 3 of the algorithm, an optimal coloring is output.

To estimate the expected running time, observe that Steps 1 and 2 take obviously a polynomial in n number of steps. The probability of getting to Step 3 is at most $O(n^{-n})$ as follows from the above discussion. As the complexity of Step 3 is $O(n^n poly(n))$ the desired expected running time estimate follows.

A more careful implementation of the same basic idea enables to shave off an extra logarithmic factor from the above described result. We refer the reader to [32] for details.

Research Problem 2 *Find a coloring algorithm with approximation ratio $o(\sqrt{n}/\log n)$ and polynomial expected running time over the probability space $G(n, 0.5)$.*

Research Problem 3 *Find coloring algorithms with good approximation ratios and polynomial expected running time in probability spaces $G(n, n^{-a})$ for $a > 0.5$.*

6 Deciding k-colorability in expected polynomial time

In this section we continue our coverage of coloring algorithms with expected polynomial running time in probability spaces $G(n,p)$. The subject here is algorithms for deciding k-colorability.

As stated in Section 2, deciding k-colorability in NP-complete for every fixed $k \geq 3$. Observe however that for absolute most of the graphs G on n vertices the answer to the question whether G is k-colorable is "No":

Proposition 6.1 *For every fixed positive integer k, the random graph $G(n, 0.5)$ is non-k-colorable with probability $1 - 2^{-\Theta(n^2)}$.*

Proof. If G has n vertices and is k-colorable then it contains an independent set of size at least n/k. The probability of the latter event in $G(n, 0.5)$ is at most:

$$\binom{n}{\frac{n}{k}} 2^{-\binom{\frac{n}{k}}{2}} < 2^n \cdot 2^{-\Theta(n^2)} = 2^{-\Theta(n^2)} .$$

The above proposition indicates that it should be probably easy to decide k-colorability quickly on average – the answer is "No" for vast majority of the graphs, and exceptional instances are very rare. And indeed, Wilf showed in [52] that the backtrack algorithm for deciding k-colorability in graphs on n vertices has a *constant* expected running time over $G(n, 0.5)$ as n approaches infinity. For instance, a backtrack search tree for 3-coloring a graph has an average of about 197 nodes only! The backtrack search tree of a graph G with vertex set $\{1, \ldots, n\}$ is the tree whose nodes are on levels $0, \ldots, n$, and in which there is a node on level i corresponding to every proper k-coloring of the subgraph of G induced by its first i vertices. A node v' at level i is connected by an edge to a node v'' at level $i+1$ if the colors of vertices $1, \ldots, i$ are the same at v' and v''. Level 0 contains a single root node, corresponding to the empty coloring.

In fact, it is quite easy to see why there exists an algorithm for deciding k-colorability in constant expected time. To show this, fix $t = C(k)n$ edge disjoint copies $K_1, \ldots, K_t$ of the complete graph K^{k+1} in the complete graph on n vertices, where $C(k)$ is a large enough constant (this is of course feasible, as in fact $\Theta(n^2)$ such copies can be found). The probability of each copy K_i to appear in the random graph $G(n, 0.5)$ is $2^{-\binom{k+1}{2}}$, which is a constant. The appearance of a copy of K^{k+1} is $G(n, 0.5)$ can serve as a certificate for non-k-colorability. Our algorithm scans chosen copies K_i looking for a clique on $k+1$ vertices. If such clique is found, the algorithm rejects the graph G. If no such copy is found, the algorithm decides k-colorability of G by performing the exhaustive search. The correctness of the above algorithm is immediate. To estimate the expected running time, observe the running time of the first phase is the truncated geometric distribution with parameter $p = 2^{-\binom{k+1}{2}} = \Theta(1)$ and has therefore a constant expectation. The probability of ever getting to the second phase can be made much smaller than k^{-n} by choosing the constant $C(k)$ large enough, and hence the expected number of steps spent at the second phase is $o(1)$.

The problem becomes significantly harder as the edge probability $p(n)$ decreases. Bender and Wilf proved [6] that in this case the backtrack algorithm has expected running time $e^{\Theta(1/p)}$, i.e., becomes exponential in n. Also, one can easily show that for every fixed $k \geq 3$, if the edge probability p satisfies $p(n) = o(n^{-2/k})$, then a.s. every subgraph of $G(n,p)$ with a bounded number of vertices is k-colorable, and thus one cannot hope to find a certificate for non-k-colorability by performing local search only.

Here we present an algorithm from [30] for deciding k-colorability in expected polynomial time in $G(n,p)$ for every fixed $k \geq 3$, as long as $p(n) \geq C/n$, where $C = C(k) > 0$ is a sufficiently large constant. Our algorithm can be immediately extended for larger values of $p(n)$. Note that if C is sufficiently large, the random graph $G(n,p)$ is not k-colorable with probability $1 - e^{-\Theta(n)}$. Therefore the algorithm still rejects most of the graphs from $G(n,p)$. In order to be able to reject an input graph, the algorithm needs some graph parameter whose value can serve as a certificate for non-k-colorability. This parameter should be computable in polynomial time. The parameter we will use in our algorithm is the so called *vector* *chromatic number* of a graph [24]. Besides being computable in polynomial time, the vector chromatic number turns out to be extremely robust, and the probability that its value is small is exponentially small in n. This will enable us to invest exponential time in "exceptional" graphs, i.e. those with small vector chromatic number.

Let us now provide necessary background on the vector chromatic number. This concept has been introduced by Karger, Motwani and Sudan in [24]. Suppose we are given a graph $G = (V, E)$ with vertex set $V = \{1, \ldots, n\}$. A *vector k-coloring* of G is an assignment of unit vectors $v_i \in R^n$ to the vertices of G so that for every edge $(i, j) \in E(G)$ the standard scalar product of the corresponding vectors v_i, v_j satisfies the inequality $(v_i, v_j) \le -\frac{1}{k-1}$. The graph G is called *vector k-colorable* if such a vector k-coloring exists. Finally, the *vector chromatic number* of G, which we denote by $v\chi(G)$, is the minimal real $k \ge 1$ for which G is vector k-colorable.

Karger, Motwani and Sudan established the connection between the usual chromatic number of a graph, $\chi(G)$, and its vector chromatic number, $v\chi(G)$. Below we repeat some of their arguments and conclusions.

Lemma 6.2 *If $\chi(G) = k$, then G is vector k-colorable. Thus, $v\chi(G) \le \chi(G)$.*

Proof. The statement will follow easily from the proposition below.

Proposition 6.3 *For every $k \le n + 1$, there exists a family $\{v_1, \ldots, v_k\}$ of k unit vectors is R^n satisfying $(v_i, v_j) = -\frac{1}{k-1}$ for every $1 \le i \ne j \le k$.*

Proof. The existence of such a family can be proven by induction on n, as in [24]. Here we present an alternative proof.

Clearly it is enough to prove the proposition for the case $k = n + 1$ (if $k < n + 1$, find such a family in R^{k-1} and complete the found vectors by zeroes in the last $n - k + 1$ coordinates to get the desired family). Define an n-by-n matrix $A = (a_{ij})$ by setting $a_{ii} = 1$ for $1 \le i \le n$, and $a_{ij} = -1/n$ for all $1 \le i \ne j \le n$. Then A is a symmetric positive definite matrix (the eigenvalues of A are $\lambda_1 = \ldots = \lambda_{n-1} = 1+1/n$, $\lambda_n = 1/n$). Therefore it follows from standard linear algebra results that there exists a family $\{v_1, \ldots, v_n\}$ of n vectors in R^n so that $a_{ij} = (v_i, v_j)$ for all $1 \le i, j \le n$. In particular, $(v_i, v_i) = a_{ii} = 1$, so all members of this family are unit vectors. Also, $(v_i, v_j) = a_{ij} = -1/n$ for all $1 \le i \ne j \le n$. Set now $v_{n+1} = -(v_1 + \ldots + v_n)$. Then $(v_{n+1}, v_{n+1}) = (v_1 + \ldots + v_n, v_1 + \ldots + v_n) = n \cdot 1 + 2\binom{n}{2}(-1/n) = 1$, and v_{n+1} is a unit vector as well. Also, for all $1 \le i \le n$, $(v_i, v_{n+1}) = (v_i, -v_1 - \ldots - v_n) = -1 + (n-1)/n = -1/n$. Hence, $\{v_1, \ldots, v_n, v_{n+1}\}$ forms the desired family.

Returning to the proof of the lemma, we argue as follows. Let $V = C_1 \cup \ldots \cup C_k$ be a k-coloring of G. Based on the above proposition, we can find a family $\{v_1, \ldots, v_k\}$ of unit vectors in R^n so that $(v_i, v_j) = -1/(k-1)$ for all $1 \le i \ne j \le k$. Now, for each color class C_i, every vertex from C_i gets the vector v_i assigned to it. The obtained assignment is clearly a vector k-coloring of G.

The most important algorithmic feature of the vector chromatic number, noticed by Karger et al., is that it is polynomially computable. Formally, if a graph G on n vertices is vector k-colorable, then a vector $(k + \epsilon)$-coloring of the graph can be constructed in time polynomial in k, n and $\log \frac{1}{\epsilon}$. This is due to the fact that vector chromatic number can be represented as a solution of a Semidefinite Program and as such is polynomial time computable (see [17]).

Another concept, needed for the analysis of our algorithm and borrowed again from [24], is that of a semi-coloring. Given a graph $G = (V, E)$ on n vertices and an integer $1 \le t \le n$, a *semi-coloring* of G in t colors is a family $(C_1, \ldots, C_t)$, where each $C_i \subseteq V(G)$ is an independent set in G, the subsets C_i are pairwise disjoint, and $|\bigcup_{i=1}^{t} C_i| \ge \frac{n}{2}$.

Lemma 6.4 *[24] For any $k \ge 3$, there exist $c = c(k) > 0, n_0 = n_0(k) > 0$ so that the following holds. For any $n > n_0$ and for any graph G on n vertices and with $m > n$ edges, if $v\chi(G) \le k$ then there exists a semi-coloring of G in t colors, where*

$$t \le c\left(\frac{m}{n}\right)^{\frac{k-2}{k}} \ln^{1/2}\left(\frac{m}{n}\right) .$$

Thus the assumption that the vector chromatic number of G is small enables to claim the existence of many pairwise disjoint and large on average independent sets in G.

Now we formulate an algorithm for deciding k-colorability. As the reader will see immediately, the algorithm is extremely simple and in a sense just calculates the vector chromatic number of an input graph.

Input: An integer $k \ge 3$ and a graph $G = (V, E)$ on n vertices.

Step 1. Calculate the vector chromatic number $v\chi(G)$ of the input graph G;

Step 2. If $v\chi(G) > k$, output "G is not k-colorable";

Step 3. Otherwise, check exhaustively all k^n potential k-colorings of G. If a proper k-coloring of G is found, output "G is k-colorable", else output "G is not k-colorable".

The correctness of the algorithm is immediate from Lemma 6.2. Let us show that its expected running time is polynomial in $G(n, C/n)$ for $C = C(k)$ large enough. Steps 1 and 2 of the algorithm take polynomial time. Notice that we get to Step 3 only if $v\chi(G) \le k$. At Step 3 we check exhaustively all k^n potential k-colorings of G, and checking each potential coloring costs us time polynomial in n. Therefore it takes at most $k^n poly(n)$ time to perform Step 3. Thus it is enough to prove the following lemma:

Lemma 6.5 *If $C = C(k) > 0$ is large enough, and G is distributed according to $G(n, p)$ with $p = C/n$, then*

$$Pr[v\chi(G) \le k] \le k^{-n} .$$

Proof. The proof is based on the following technical propositions about the probability space $G(n, p)$.

Proposition 6.6 *If $C > 0$ is large enough then*

$$Pr[|E(G)| \le 2n^2 p] \ge 1 - o(k^{-n}) .$$

Proposition 6.7 *For every fixed $c > 0$, $k \ge 3$, if $C > 0$ is large enough then the following is true in $G(n, p)$ with $p = C/n$. Let $t = c(2C)^{\frac{k-2}{k}} \ln^{1/2}(2C)$. Then*

$$Pr[G \textit{ has a semi-coloring in } t \textit{ colors}] = o(k^{-n}) .$$

Assuming the above two propositions hold, we prove now Lemma 6.5. By Proposition 6.6 we may assume that G has at most $2n^2p = 2Cn$ edges. If the vector chromatic number of such a graph is at most k, then by Proposition 6.4 G has a semi-coloring in $t = c(m/n)^{(k-2)/k)} \ln^{1/2}(m/n) \leq c(2C)^{(k-2)/k} \ln^{1/2}(2C)$ colors. However, by Proposition 6.7 this happens in $G(n,p)$ with probability $o(k^{-n})$.

Both Propositions 6.6 and 6.7 are proven by straightforward calculations, quite standard for the probability space $G(n,p)$. We omit the details here, referring the reader to [30].

Research Problem 4 *Find an algorithm for deciding k-colorability whose expected running time is polynomial over $G(n,p)$ for* any *value of the edge probability $p = p(n)$.*

Note that if $p(n) \leq c/n$ with $c = c(k) > 0$ sufficiently small, then $G(n,p)$ is k-colorable almost surely (see, e.g. [43]), and the algorithm should thus accept most of the input graphs. The problem becomes especially challenging when $p(n)$ is close to the threshold probability for non-k-colorability. This is due to the widespread belief that typical instances at the non-colorability threshold are computationally hard (see, e.g., [10] for a relevant discussion).

Research Problem 5 *Find an algorithm for deciding k-colorability in expected polynomial time in $G(n,p)$ when the parameter k is a growing function of n: $k = k(n)$.*

The apparent difficulty here lies in the fact that the vector chromatic number seems no longer be useful as Lemma 6.4 degenerates to a trivial statement already for $k \gg \log n$.

7 Coloring random k-colorable graphs

The somewhat contradictory title of this section should not confuse the reader – of course, a k-coloring of the input graph is not known to an algorithm, and the task it to recover it or to find some k-coloring.

We should first define models or probability spaces we will be working with. The first one, which we denote by $G(n,p,k)$ is formed as follows. The vertex set is a union of k disjoint subsets of $V_1, \ldots, V_k$ of size n each, and for every pair of vertices $u \in V_i, v \in V_j, i \neq j$, (u,v) is an edge of $G(n,p,k)$ independently and with probability $p = p(n)$. The second model $G_S(n,p,k)$, usually called the *semi-random model*, is more complicated – first a random graph G is generated according to the distribution $G(n,p,k)$, and then the adversary can for every non-edge (u,v) of G, where u and v belong to different color classes, add this pair to the set of edges. Adding extra edges to the otherwise random graph $G(n,p,k)$ can spoil its random structure, making the task of recovering its k-coloring significantly more difficult. Of course, the resulting graphs in both models are guaranteed to be k-colorable with proper coloring $(V_1, \ldots, V_k)$, but this k-coloring is not necessarily unique, and in fact the resulting graph can even have chromatic number smaller than k. It should be clear to the reader that those two models are not the most general ones, and quite a few other models of random k-colorable graphs exist (see, e.g., Section 1 of [49] for a detailed discussion). In this survey we will mostly restrict ourselves with the case of a constant $k \geq 3$, although some of the results we will describe work for growing $k = k(n)$ as well.

Just as in the previous sections, we will consider here two algorithmic tasks. The first task is to provide an algorithm that k-colors in polynomial time almost all graphs in a chosen probability space. The second, more challenging task is to give a k-coloring algorithm with expected polynomial running time.

Let us start with the random model $G(n,p,k)$ and the edge probability $p = 0.5$. This value of the edge probability is the most natural choice as most k-colorable graphs are easily seen to have a quadratic number of edges. Turner [50] proposed the following very simple algorithm for finding a.s. a k-coloring in this case. The algorithm of Turner starts with finding a clique $K = \{v_1, \ldots, v_k\}$ of size k in G and coloring its vertices arbitrary in k distinct colors. Then the algorithm repeatedly searches for an uncolored vertex v that has neighbors in exactly $k-1$ colors, and colors such a vertex in a unique available color. It is rather easy to see that for a constant k such a vertex can almost surely be found at each step. Turner proves in fact that the above algorithm works as long as the number of colors k satisfies $k \le (1-\epsilon)\log_2 n$. The result of Turner has been strengthened by Dyer and Frieze [11], who proposed an algorithm for finding a k-coloring in $G(n, 0.5, k)$ in $O(n^2)$ expected time. As the number of edges in $G(n, 0.5, k)$ is almost surely quadratic in n, the algorithm of Dyer and Frieze colors this random graph in linear in the number of edges expected time.

An equally simple algorithm has been proposed by Kučera [34] for the case of $k \le Cn/\log n$. (It could be helpful for the reader to note here that a formal statement of Kučera's result in his paper appears different; this is due to the fact that he considers the probability space of k-colorable graphs on n vertices, so if $k = \Theta(\sqrt{n/\log n})$ in his result, this approximately translates to $\Theta(n/\log n)$ colors in our setting). Observe that if vertices u, v belong to the same color class V_i of $G(n,p,k)$, then the number of their common neighbors is binomially distributed with parameters $(k-1)n, p^2)$, while if u and v come from distinct color classes $u \in V_i, v \in V_j, i \ne j$, the number of their common neighbors is again binomially distributed, but this time with parameters $(k-2), n, p^2)$. Therefore we expect a pair of vertices in the same color class to have more common neighbors than a pair from different color classes. Using standard bounds on the tails of the binomial distribution, one can easily show that for the case $p = 0.5$, $k \le cn/\log n$, almost surely the number of common neighbors of any two vertices in the same color class is strictly larger than the number of common neighbors in different color classes. We can thus use the number of common neighbors to classify vertices into the same of different color classes. Technical details of the proof can be easily filled or alternatively found in [34].

Research Problem 6 *Find an algorithm that almost surely k-colors a random graph $G(n, 0.5, k)$ for $k \gg n$.*

However, as the edge probability $p = p(n)$ decreases, it becomes harder and harder to find a k-coloring in $G(n,p,k)$ even for fixed k. This should not be surprising – the more random edges we have, the more evident becomes the prefixed coloring scheme. Still, algorithms are known even for very sparse random graphs. The best achievement belongs to Alon and Kahale [1], who gave an algorithm for k-coloring $G(n,p,k)$ for $p \ge C/n$, where $C = c(k)$ is a large enough constant. Notice that if $p = C/n$, then the random graph has typically only a linear in n number of edges, and a linear number of vertices are isolated.

Let us describe briefly the main idea of the algorithm of [1] for the case of $k = 3$ colors. Denote $d = pn$, then d is the expected number of neighbors of every vertex $v \in V_i$ in every other color class. Let us assume for simplicity that every vertex v has indeed exactly d neighbors in every other color class in G. Consider the adjacency matrix $A = A(G)$ of G. Let $\lambda_1 \geq \ldots \geq \lambda_{3n-1} \geq \lambda_{3n}$ be the eigenvalues of A, and $e_1, \ldots, e_{3n-1}, e_{3n}$ be the corresponding orthonormal basis of eigenvectors. The largest eigenvalue of A is then $\lambda_1 = d$, and the spectrum of A is in the interval $[-d, d]$. Let F be the 2-dimensional subspace of all vectors $x = (x_v : v \in V)$ that are constant on every color class, and whose sum is zero: $\sum_{v \in V} x_v = 0$. A simple calculation shows that any non-zero vector from F is an eigenvector of A with eigenvalue $-d$. One can also show that almost surely the multiplicity of the eigenvalue $-d$ is two, and thus F is in fact the eigenspace of $-d$. Therefore any linear combination t of the vectors e_{3n-1} and e_{3n} (both can be efficiently computed) is constant on every color class. Now we find a non-zero linear combination t of e_{3n-1} and e_{3n}, whose median is zero, that is, the numbers of positive and negative components of t both do not exceed $3n/2$. (It is easy to see that such a combination always exists and can be found efficiently.) Normalizing such t to have it with l_2-norm $\sqrt{2n}$, we get a vector t' whose coordinates take values 0,1 or -1 depending on the color class. Defining now

$$\begin{aligned} V_1 &= \{v \in V : t'_v = 0\}; \\ V_2 &= \{v \in V : t'_v = 1\}; \\ V_3 &= \{v \in V : t'_v = -1\}, \end{aligned}$$

we get a proper coloring of G in three colors. The real algorithm of Alon and Kahale is of course much more complicated, it starts from defining an approximate coloring (V_1, V_2, V_3) according to the last two eigenvectors e_{3n-1}, e_{3n} as described above, and then refines it to get a proper coloring.

Very recently, McSherry [40] described a very general spectral algorithm, applicable to several partitioning problems in random graphs. This algorithm differs significantly from the one of Alon and Kahale, and when applied to the k-coloring problem works for the edge probability $p(n)$ down to $p(n) \geq c \log^3(n)/n$.

Research Problem 7 *Find an algorithm that k-colors almost every graph in $G(n, c/n, k)$ for a fixed $k \geq 3$, for all values of the constant $c > 0$.*

If the constant c in the above problem is small enough, the random graph $G(n, c/n, k)$ almost surely does not contain a subgraph with minimal degree k and hence can be easily colored by a greedy-type algorithm. The problem is most challenging for moderate values of c.

The expected time version of the problem has been considered by Subramanian in [48], who proposed a k-coloring algorithm with expected running time polynomial in n as long as $p(n) \geq n^{-a}$ and $a < 3/4$. It would be interesting to obtain coloring algorithms with polynomial expected time for smaller values of the edge probability $p(n)$.

Let us now switch to the semi-random model $G_S(n, p, k)$. This model has been considered by Blum and Spencer [7], who applied the so-called forced coloring approach. To explain this approach, assume that vertices u, v of G are fully connected to a common clique of size $k-1$. Then every proper k-coloring of G should put u, v in the same color class. Blum and Spencer implemented this approach as follows: (a) Given a graph $G = (V, E)$, define a new graph $G' = (V, F)$ where $(u, v) \in F$ if u and v are both adjacent to a common $(k-1)$-clique; (b) Find all connected components of G'. If G' contains exactly k connected components, then they coincide with color classes of the original graph G. Returning to the probability space $G_S(n,p,k)$, observe that adding edges to the random graph $G \sim G(n,p,k)$ can only add edges in the corresponding auxiliary graph G', and hence it is enough to show that already in $G(n,p,k)$ almost surely the graph G' has exactly k connected components. The expected number of copies of $K^{k+1} - e$ becomes linear in n for $p(n) = n^{-\frac{2k}{(k-1)(k+2)}}$, and Blum and Spencer were able to show that increasing this probability a bit (say, by factor n^ϵ for any $\epsilon > 0$) suffices to get almost surely k connected components in G'. Later, Subramanian, Fürer and Veni Madhavan [49] extended the result of Blum and Spencer by giving a k-coloring algorithm for $G_S(n,p,k)$ with expected polynomial time for the same range of edge probabilities.

The best result for the semi-random coloring problem has been obtained by Feige and Kilian [13]:

Theorem 7.1 *For every constant k, there is a polynomial time algorithm that k-colors almost graphs in the probability space $G_S(n,p,k)$ for $p(n) \geq (1+\epsilon)k \ln n/n$.*

Feige and Kilian observed also that the above result is close to optimal, as given by the next theorem:

Theorem 7.2 *Let $\epsilon > 0$, $k \geq 3$ be constants and let $p(n) \leq (1-\epsilon)\ln n/n$. Then unless $NP \subseteq BPP$, every random polynomial time algorithm will fail almost surely to k-color a semi-random graph from $G_S(n,k,p)$.*

We do not intend to cover a rather complicated algorithm of Feige and Kilian here. Very briefly, it starts by finding a large independent set in $G_S(n,p,k)$, using Semidefinite Programming in the spirit of Lemma 6.4. This independent set I is shown to belong almost entirely to one of the color classes V_i. Then I is purified to get rid of the vertices outside V_i, and then remaining vertices from V_i are found to recover one color class completely; the algorithm then proceeds to recovering the next color class and so on.

Finally we note that a random graph $G(n,p,k)$ can have in fact chromatic number less than k. Subramanian in [47] provides algorithms that color $G(n,p,k)$ and $G_S(n,p,k)$ in minimal possible number of colors in expected polynomial time, as long as $p(n) \geq n^{-\gamma(k)+\epsilon}$ for the random model and $p(n) \geq n^{-\alpha(k)+\epsilon}$ for the semi-random model, where $\gamma(k) = \frac{2k}{k^2-k+2}$ and $\alpha(k) = \frac{2k}{(k-1)(k+2)}$.

References

[1] N. Alon and N. Kahale, *A spectral technique for coloring random 3-colorable graphs*, Proc. of the 26^{th} Annual ACM Symposium on Theory of Computing (STOC'94), 346–355.

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

[3] N. Alon, M. Krivelevich and B. Sudakov, *List coloring of random and pseudo-random graphs*, Combinatorica 19 (1999), 453–472.

[4] N. Alon, M. Krivelevich and V. H. Vu, *On the concentration of eigenvalues of random symmetric matrices*, Israel Journal of Mathematics, in press.

[5] N. Alon and J. Spencer, **The probabilistic method**, 2^{nd} ed., Wiley, New York, 2000.

[6] E. Bender and H. Wilf, *A theoretical analysis of backtracking in the graph coloring problem*, J. Algorithms 6 (1985), 275–282.

[7] A. Blum and J. Spencer, *Coloring random and semirandom k-colorable graphs*, J. Algorithms 19 (1995), 204–234.

[8] B. Bollobás, *The chromatic number of random graphs*, Combinatorica 8 (1988), 49–55.

[9] B. Bollobás, **Random graphs**, 2^{nd} ed., Cambridge Univ. Press, Cambridge, 2001.

[10] J. Culberson and I. Gent, *Well out of reach: why hard problems are hard*, Technical report APES-13-1999, APES Research Group, 1999. Available from: http://www.cs.strath.ac.uk/ apes/apesreports.html.

[11] M. Dyer and A. Frieze, *The solution of some random NP-hard problems in polynomial expected time*, J. Algorithms 10 (1989), 451–489.

[12] U. Feige and J. Kilian, *Zero knowledge and the chromatic number*, Proc. 11^{th} IEEE Conf. Comput. Complexity, IEEE (1996), 278–287.

[13] U. Feige and J. Kilian, *Heuristics for semirandom graph problems*, J. Computer and System Sciences, to appear.

[14] Z. Füredi and J. Komlós, *The eigenvalues of random symmetric matrices*, Combinatorica 1 (1981), 233–241.

[15] M. Fürer, C. R. Subramanian and C. E. Veni Madhavan, *Coloring random graphs in polynomial expected time*, Algorithms and Comput. (Hong Kong 1993), Lecture Notes Comp. Sci. 762, Springer, Berlin, 1993, 31–37.

[16] G. Grimmett and C. McDiarmid, *On colouring random graphs*, Math. Proc. Cam. Phil. Soc. 77 (1975), 313–324.

[17] M. Grötschel, L. Lovász and A. Schrijver, **Geometric algorithms and combinatorial optimization**, Algorithms and Combinatorics 2, Springer Verlag, Berlin, 1993.

[18] V. Guruswami and S. Khanna, *On the hardness of 4-coloring a 3-colorable graph*, Proc. 15^{th} Annual IEEE Conf. on Computational Complexity, IEEE Comput. Soc. Press, Los Alamitos, 2000, 188–197.

[19] M. M. Halldórsson, *A still better performance guarantee for approximate graph coloring*, Inform. Process. Letters 45 (1993), 19–23.

[20] S. Janson, T. Łuczak and A. Ruciński, **Random graphs**, Wiley, New York, 2000.

[21] T. Jensen and B. Toft, **Graph coloring problems**, Wiley, New York, 1995.

[22] M. Jerrum, *Large cliques elude the metropolis process*, Random Structures and Algorithms 3 (1992), 347–359.

[23] A. Juels and M. Peinado, Hiding cliques for cryptographic security, *Proc. of the Ninth Annual ACM-SIAM SODA*, ACM Press (1998), 678–684.

[24] D. Karger and R. Motwani and M. Sudan, *Approximate Graph coloring by semidefinite programming*, Journal of the ACM, 45 (1998), 246–265.

[25] R. Karp, Reducibility among combinatorial problems, in: *Complexity of computer computations* (E. Miller and J. W. Thatcher, eds.) Plenum Press, New York, 1972, 85–103.

[26] R. M. Karp, Probabilistic analysis of some combinatorial search problems, In: *Algorithms and Complexity: New Directions and Recent Results*, J. F. Traub, ed., Academic Press, New York, 1976, pp. 1–19.

[27] H. Kierstead, *Coloring graphs on-line*, Online algorithms (Schloss-Dagstuhl 1996), Lecture Notes in Computer Science 1442, Springer, Berlin, 1998, 281–305.

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

[29] M. Krivelevich, *The choice number of dense random graphs*, Combinatorics, Probability and Computing 9 (2000), 19–26.

[30] M. Krivelevich, *Deciding k-colorability in expected polynomial time*, Information Processing Letters 81 (2002), 1–6.

[31] M. Krivelevich and B. Sudakov, *Coloring random graphs*, Inform. Process. Letters 67 (1998), 71–74.

[32] M. Krivelevich and V. H. Vu, *Approximating the independence number and the chromatic number in expected polynomial time*, Proc. 27^{th} Int. Colloq. on Automata, Languages and Programming (ICALP'2000), Lecture Notes in Computer Science 1853, Springer, Berlin, 13–24.

[33] M. Krivelevich and V. H. Vu, *Choosability in random hypergraphs*, Journal of Combinatorial Theory Ser. B 83 (2001), 241–257.

[34] L. Kučera, *Graphs with small chromatic numbers are easy to color*, Inform. Process. Letters 30 (1989), 233–236.

[35] L. Kučera, *The greedy coloring is a bad probabilistic algorithm*, J. Algorithms 12 (1991), 674–684.

[36] L. Lovász, *On the Shannon capacity of a graph*, IEEE Trans. Inform. Theory 25 (1979), 1–7.

[37] T. Łuczak, *The chromatic number of random graphs*, Combinatorica 11 (1991), 45–54.

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

[39] C. J. H. McDiarmid, *Colouring random graphs badly*, Graph Theory and Combinatorics (R. J. Wilson, Ed.), Pitman Research Notes in Mathematics, vol. 34, 1979, 76–86.

[40] F. McSherry, *Spectral partitioning of random graphs*, Proc. 42^{nd} IEEE Symposium on Found. of Comp. Science (FOCS'01), IEEE Comp. Society, 529–537.

[41] D. W. Matula, *On the complete subgraph of a random graph*, Combinatory mathematics and its applications, Chapel Hill, North Carolina (1970), 356–369.

[42] D. Matula, *Expose-and-merge exploration and the chromatic number of a random graph*, Combinatorica 7 (1987), 275–284.

[43] M. Molloy, *Thresholds for colourability and satisfiability in random graphs and Boolean formulae*, in: Surveys in Combinatorics 2001, London Math. Soc. Lect. Note Ser. 288, Cambridge Univ. Press, Cambridge, 2001.

[44] B. Pittel and R. S. Weishaar, *On-line coloring of sparse random graphs and random trees*, J. Algorithms 23 (1997), 195–205.

[45] E. Shamir and J. Spencer, *Sharp concentration of the chromatic number of random graphs $G_{n,p}$*, Combinatorica 7 (1987), 124–129.

[46] E. Shamir and E. Upfal, *Sequential and distributed graph coloring algorithms with performance analysis in random graph spaces*, J. Algorithms 5 (1984), 488–501.

[47] C. R. Subramanian, *Minimum coloring k-colorable graphs in polynomial average time*, J. Algorithms 33 (1999), 112–123.

[48] C. R. Subramanian, *Algorithms for colouring random k-colourable graphs* Combinatorics, Probability and Computing 9 (2000), 45–77.

[49] C. R. Subramanian, M. Fürer and C. E. Veni Madhavan, *Algorithms for coloring semi-random graphs*, Random Struct. Algorithms 13 (1998), 125–158.

[50] J. S. Turner, *Almost all k-colorable graphs are easy to color*, J. Algorithms 9 (1988), 63–82.

[51] J. H. van Lint and R. M. Wilson, **A course in combinatorics**, Cambridge Univ. Press, Cambridge, 1992.

[52] H. Wilf, *Backtrack: a $O(1)$ expected time algorithm for the graph coloring problem*, Inform. Proc. Letters 18 (1984), 119–121.

Michael Krivelevich
Department of Mathematics
Faculty of Exact Sciences
Tel Aviv University
Tel Aviv, 69978
Israel.
krivelev@post.tau.ac.il