An Improved Algorithm for Approximating the Chromatic Number of $G_{n,p}$

Amin Coja-Oghlan[1] and Lars Kuhtz[2]

[1] Humboldt-Universität zu Berlin, Institut für Informatik,
Unter den Linden 6, 10099 Berlin, Germany
coja@informatik.hu-berlin.de
[2] Universität des Saarlandes, FR Informatik,
Postfach 15 11 50, 66041 Saarbrücken, Germany
kuhtz@cs.uni-sb.de

Abstract. Answering a question of Krivelevich and Vu [12], we present an algorithm for approximating the chromatic number of random graphs $G_{n,p}$ within a factor of $O(\sqrt{np}/\ln(np))$ in polynomial expected time. The algorithm applies to edge probabilities $c_0/n \leq p \leq 0.99$, where $c_0 > 0$ is a certain constant.
Keywords: approximation algorithms, random graphs, graph coloring.

1 Introduction

In the *Graph Coloring Problem* we are given a graph G, and the goal is to determine the chromatic number $\chi(G)$. With respect to the worst-case complexity of graph coloring, Feige and Kilian [7] proved that unless NP=ZPP, no polynomial time algorithm approximates the chromatic number of a graph on n vertices within a factor of $n^{1-\varepsilon}$, where $\varepsilon > 0$ denotes an arbitrarily small constant. Thus, we do not hope for coloring algorithms that perform well on *all* instances. Therefore, the goal of this paper is to investigate coloring *heuristics*. Hence, we are interested in coloring algorithms that perform well on "most" instances in some meaningful sense.

In order to specify precisely what "most" instances is supposed to mean, we consider the well-known *binomial model* $G_{n,p}$ of *random graphs* pioneered by Erdős and Rényi. The vertex set of $G_{n,p}$ is $V = \{1, \ldots, n\}$, and each of the $\binom{n}{2}$ possible edges is present in $G_{n,p}$ with probability $p = p(n)$ independently of all others. Hence, the number of edges of $G_{n,p}$ is binomially distributed with mean $\binom{n}{2}p$. Of course, the $G_{n,p}$ model does not capture various types of "practical" input distributions. Nevertheless, both the combinatorial structure and the algorithmic theory of $G_{n,p}$ are of fundamental interest [3, 8]. Indeed, $G_{n,p}$ provides an interesting family of benchmarking instances for graph coloring [11].

We say that $G_{n,p}$ enjoys some property *with high probability* ("w.h.p.") if the probability that the property holds tends to 1 as the number n of vertices tends to infinity. Bollobás [2] and Łuczak [15] determined the probable value of the chromatic number:

$$\chi(G_{n,p}) \sim -\frac{n\ln(1-p)}{2\ln(np)} \quad \text{w.h.p. if } n^{-1} \ll p \leq 0.99. \tag{1}$$

However, the proof of (1) does not answer the *algorithmic* question how to actually color $G = G_{n,p}$ in $\chi(G)$ colors efficiently. Concerning the algorithmic issue, Grimmett and McDiarmid [9] proved that on input $G_{n,\frac{1}{2}}$ a greedy algorithm finds a coloring that uses $(1+o(1))n/\log_2 n$ colors w.h.p. Thus, (1) shows that for $G = G_{n,p}$ the greedy algorithm needs at most $(2+o(1)) \cdot \chi(G)$ colors w.h.p.

Though, Kučera [13] observed that there are graphs G on n vertices for which the greedy algorithm needs at least $n^{1-\varepsilon} \cdot \chi(G)$ colors (n arbitrarily large and $\varepsilon > 0$ arbitrarily small but fixed). Of

course, in the light of the hardness result of Feige and Kilian [7] this observation is hardly surprising. But the issue is that the greedy algorithm does not *detect* its failure, because it does not compute a lower bound on the chromatic number. Therefore, Krivelevich and Vu [12] posed the following problem.

> Devise a coloring algorithm that colors *any* input graph with at most $r \cdot \chi(G)$ colors and whose *expected* running time on input $G_{n,p}$ is polynomial for a number $r \geq 1$ as small as possible.

Here the *expected running time* of an algorithm $\mathcal{A}$ on $G_{n,p}$ is $\sum_G R_{\mathcal{A}}(G) \cdot \mathrm{P}[G = G_{n,p}]$, where the sum ranges over all graphs G with vertex set $V = \{1, \ldots, n\}$, and $R_{\mathcal{A}}(G)$ signifies the running time of $\mathcal{A}$ on input G.

Krivelevich and Vu obtained a coloring algorithm with approximation ratio $r = O(\sqrt{np}/\ln(np))$ whose expected running time on $G_{n,p}$ is polynomial, provided that $p \gg n^{-1/2}$. This algorithm combines the greedy coloring algorithm with a spectral technique for bounding the chromatic number from below. Moreover, Krivelevich and Vu asked for an algorithm that achieves a similar approximation ratio in polynomial expected time for smaller values of p (cf. also Research Problem 3 in [11]).

Partial answers were obtained by Coja-Oghlan, Moore, Sanwalani, and Taraz [5, 6], who presented algorithms that approximate $\chi(G_{n,p})$ within a factor of $r = O(\sqrt{np})$ in polynomial expected time if $p \geq c_0/n$ for a sufficiently large constant $c_0 > 0$. However, the approximation ratio obtained by these results is by a $\ln(np)$-factor worse than that obtained by Krivelevich and Vu. The main result of this paper is the following theorem, which answers the question of Krivelevich and Vu completely, giving an approximation ratio of $O(\sqrt{np}/\ln(np))$ for $p \geq c_0/n$.

Theorem 1. *Suppose that $c_0/n \leq p \leq 0.99$ for a sufficiently large constant c_0. There exists a coloring algorithm `ApxColor` that guarantees an approximation ratio of $O(\sqrt{np}/\ln(np))$ on any input graph and runs in polynomial expected time on input $G_{n,p}$.*

The algorithm `ApxColor` consists of two parts: a procedure `GreedyCoreColor` that yields an upper bound on the chromatic number, and a semidefinite progamming based method for bounding the chromatic number from below. In comparison to [5, 6], the significant new aspect is the procedure `GreedyCoreColor`, which yields the improved approximation ratio. In Section 2 we present and analyze this procedure. Then, we prove Theorem 1 in Section 3.

2 The Algorithm `GreedyCoreColor`

2.1 Outline of the Algorithm

The goal of this section is to establish the following result.

Theorem 2. *There is an algorithm `GreedyCoreColor` that enjoys the following properties.*

1. *If G is a graph on n vertices and $0 < p < 1$, then `GreedyCoreColor`(G, p) colors G with at most $\frac{30np}{\ln(np)} + \chi(G)$ colors.*
2. *If $c_0 \leq np \leq 0.99n$, where $c_0 > 0$ denotes a sufficiently large constant, then the expected running time of `GreedyCoreColor`$(G_{n,p}, p)$ is linear.*

The algorithm `GreedyCoreColor` employs the following procedure `CoreColor` from [6].

Proposition 3. *There is an algorithm `CoreColor` that has the following properties.*

1. *For any input graph G and any $k \geq 3$, `CoreColor`(G, k) needs at most $\chi(G) + k$ colors.*
2. *If $k \geq 10np$, then the expected running time of `CoreColor`$(G_{n,p}, k)$ is linear.*

In addition to `CoreColor`, `GreedyCoreColor` makes use of the following well-known linear time algorithm `GreedyMIS` for computing an independent set.

Algorithm 4. `GreedyMIS`(G)
Input: A graph $G = (V, E)$. *Output:* An independent set S of G.

1. Let $S = \emptyset$.
2. For all $v \in V$ do: if there is no v-S-edge in G, then add v to S.
3. Output S.

Thus, `GreedyMIS` goes through the vertices in a fixed order and includes each vertex v into S if $S \cup \{v\}$ remains independent. Observe that `GreedyMIS`(G) actually computes a *maximal* independent set S; i.e., there is no independent set T in G that contains S properly.

Recall that the well-known greedy coloring algorithm just calls `GreedyMIS` repeatedly as follows. On input $G = (V, E)$, `GreedyColor` calls `GreedyMIS` to find an independent set S. Then, `GreedyColor` assigns one new color to the vertices in S, and proceeds recursively to obtain a coloring of $G - S$.

How does `GreedyColor` behave in input $G_{n,p}$? As mentioned earlier, Grimmett and McDiarmid [9] pointed out that `GreedyColor`$(G_{n,\frac{1}{2}})$ uses $(1 + o(1))\frac{n}{\log_2 n} \sim 2\chi(G_{n,\frac{1}{2}})$ colors w.h.p. (cf. also [3, pp. 297–298] for some more detailed results). However, since our goal is to achieve a coloring algorithm with polynomial expected running time that satisfies the first condition in Theorem 2 on *all* inputs, we need to take a closer look. Let $G = G_{n,p}$, and let S_j be the color class determined during the j'th iteration of Step 3 of `GreedyColor`(G).

Clearly, the odds that in the j'th iteration `GreedyMIS`$(G' = G - \bigcup_{i=1}^{j-1} S_i)$ will find a "large" color class S_j depend on the number $\#V(G')$ of remaining vertices. For the larger $\#V(G')$ is, the more candidates for inclusion into the new color class there are. In fact, in Lemma 6 below we shall prove that the probability that the first, say, $20np/\ln(np)$ color classes cover less than a $(1 - 1/\ln(np))$-fraction of the vertices is $\leq \exp(-n)$.

On the other hand, suppose that $np = d > 0$ is a large constant, and let us try to lower bound the probability that `GreedyColor`$(G = G_{n,p})$ behaves "badly". Consider a graph H on d^2 vertices such that `GreedyColor`(H) uses at least d colors, while $\chi(H) \leq \sqrt{d}$ (cf., e.g., [13] for a proof that such graphs H exist). Then the probability that the subgraph of $G = G_{n,p}$ induced on the first d^2 vertices is precisely H is at least $p^{d^4} \geq n^{-O(1)}$. Since $\chi(H) = \sqrt{d}$, by (1) we have

$$\mathrm{P}[\texttt{GreedyColor}(G) \text{ needs} > d \text{ colors, but } \chi(G) \leq d/\ln d] \geq n^{-O(1)}. \tag{2}$$

In summary, we have indicated that with probability $\geq 1 - \exp(-n)$, the "first few" color classes produced by `GreedyColor`$(G_{n,p})$ will cover a large fraction of the vertices. Though, by (2) there is a moderate chance that `GreedyColor`$(G_{n,p})$ produces a bad coloring of the entire input graph. Therefore, the basic idea behind the next algorithm `GreedyCoreColor` is to follow the greedy strategy while the number of uncolored vertices is still fairly large. But as soon as there are $< n/\ln(np)$ vertices left, the algorithm calls the procedure `CoreColor` from Proposition 3. If, however, the greedy phase needs too many colors (more than, say, $20np/\ln(np)$), then `GreedyCoreColor` calls an exact coloring algorithm `Lawler` from [14]. This algorithm computes an optimal coloring of a graph on n vertices in time $O(2.443^n)$.

Algorithm 5. `GreedyCoreColor`(G, p)
Input: A graph $G = (V = \{1, \ldots, n\}, E)$, a number $0 < p < 1$. *Output:* A coloring of G.

1a. Let $G' = G$.
1b. While $\#V(G') \geq n/\ln(np)$
1c. Let $S =$ `GreedyMIS`(G').
Color the vertices in S with one new color and remove S from G'.
2. If Step 1 has used $\leq 20np/\ln(np)$ colors in total, then
Call `CoreColor`$(G', 10np/\ln(np))$ to color the remaining graph G' with new colors and output the resulting coloring of G.
3. Otherwise run `Lawler`(G) to compute an optimal coloring.

The analysis of `GreedyCoreColor` relies on the following lemma, which we shall prove in Section 2.2.

Lemma 6. *Suppose that $c_0 \leq np \leq 0.99n$ for some sufficiently large constant $c_0 > 0$. Let $\mathcal{E}$ be the event that $G = (V, E) = G_{n,p}$ has $l = \frac{10np}{\ln(np)}$ pairwise disjoint maximal independent sets $S_1, \ldots, S_l$ such that*

$$\sum_{i=1}^{l} \#S_i \leq n\left[1 - \frac{1}{\ln(np)}\right] \text{ and } \#S_i \leq \frac{\ln(np)}{10p} \text{ for } i = 1, \ldots, l. \tag{3}$$

Then $\mathrm{P}(\mathcal{E}) \leq \exp(-n)$.

Proof of Theorem 2. To prove the first assertion, let $G = (V, E)$ be a graph on n vertices. If Step 1 of `GreedyCoreColor`(G, p) needs more than $20np/\ln(np)$ colors, then `GreedyCoreColor` executes Step 3 and therefore outputs an optimal coloring of G. Furthermore, if Step 1 uses at most $20np/\ln(np)$ colors, then Proposition 3 entails that Step 2 yields a coloring of G' that needs at most $\chi(G) + 10np/\ln(np)$ colors. Thus, in total `GreedyCoreColor` needs at most $\frac{30np}{\ln(np)} + \chi(G)$ colors, as desired.

With respect to the second assertion, we observe that Step 1 runs in linear time. Furthermore, we claim that the expected time that `GreedyCoreColor`$(G = G_{n,p}, p)$ spends on executing Step 2 is linear as well. Indeed, let G' be the graph with which Step 2 is encountered, let $W = V(G) \setminus V(G')$, and let $n' = \#V(G')$. We claim that G' is distributed as a random graph $G_{n',p}$. Indeed, Step 1 has only inspected the neighborhoods of the vertices in W. That is, Step 1 only depends on edges of the form $\{v, w\}$ with $v \in V(G)$ and $w \in W$, which are independent of the internal edges of the resulting graph G'. Hence, as the input graph G is a random graph $G_{n,p}$, we conclude that $G' = G_{n',p}$. Therefore, Proposition 3 entails that the expected running time of Step 2 is linear.

As Step 3 runs in time $O(2.443^n)$, to establish the second part of the theorem the remaining task is to show that

$$\mathrm{P}\left[\text{Step 1 of } \texttt{GreedyCoreColor}(G_{n,p}, p) \text{ uses} > \tfrac{20np}{\ln(np)} \text{ colors}\right] \leq 2.443^{-n}. \tag{4}$$

Thus, let $G = (V, E) = G_{n,p}$. Then `GreedyCoreColor`(G, p) iterates Step 1b $k \leq n$ times. In each iteration, `GreedyMIS` exhibits an independent set S_j of cardinality $s_j = \#S_j$ $(j = 1, \ldots, k)$. We say that the j'th iteration *fails* if $s_j < \frac{\ln(np)}{10p}$. Note that if Step 1 uses more than $\frac{20np}{\ln(np)}$ colors, then more than $l = \frac{10np}{\ln(np)}$ iterations of Step 1b fail. Therefore, in order to prove (4), it suffices to establish the following:

$$\mathrm{P}\left[\text{at least } l \text{ iterations of Step 1b fail}\right] \leq \exp(-n). \tag{5}$$

To prove (5), assume that there are indices $1 \leq j_1 < j_2 < \cdots < j_l < k$ such that the j_i'th iteration of Step 1b fails. Then $S_{j_1}, \ldots, S_{j_l}$ are pairwise disjoint maximal independent sets of G that satisfy (3). Thus, (5) follows from Lemma 6. □

2.2 Proof of Lemma 6

Let us call a set S of vertices of $G = (V, E)$ *closed* if every $v \in V \setminus S$ has a neighbor in S. Let

$$\nu = \frac{n}{\ln(np)},\ l = \frac{10np}{\ln(np)}, \text{ and } s = \frac{\ln(np)}{10p}.$$

Suppose that $S_1, \ldots, S_l \subset V$ are pairwise disjoint sets that satisfy (3). If $S_1, \ldots, S_l$ are maximal independent sets, then in particular $S_1, \ldots, S_l$ are closed. Let $s_i = \#S_i$ for $i = 1, \ldots, l$. We claim that

$$\begin{aligned}\mathrm{P}\left[S_1, \ldots, S_l \text{ are maximal independent sets in } G = G_{n,p}\right] &\leq \mathrm{P}\left[S_1, \ldots, S_l \text{ are closed in } G = G_{n,p}\right] \\ &\leq \prod_{i=1}^{l} \left(1 - (1-p)^{s_i}\right)^{\nu}. \end{aligned} \tag{6}$$

Indeed, for each $v \in V \setminus \bigcup_{j=1}^{l} S_j$ we let $e(v, S_i)$ denote the number of v-S_i-edges; then we have $\mathrm{P}\left[e(v, S_i) > 0\right] = 1 - (1-p)^{s_i}$, because each of the s_i possible v-S_i-edges is present with probability p independently. Furthermore, since the random variables $e(v, S_i)$ are mutually independent for $v \in V \setminus \bigcup_{j=1}^{l} S_j$ and because $\#V \setminus \bigcup_{j=1}^{l} S_j \geq \nu$, we obtain

$$\mathrm{P}\left[\forall v \in V \setminus \bigcup_{j=1}^{l} S_j : e(v, S_i) > 0\right] \leq \left(1 - (1-p)^{s_i}\right)^{\nu}. \tag{7}$$

Finally, as the sets $S_1, \ldots, S_l$ are pairwise disjoint, the random variables $(e(v, S_i))_{1 \leq i \leq l}$ are mutually independent ($v \in V \setminus \bigcup_{j=1}^{l} S_j$). Therefore, we obtain

$$\mathrm{P}\left[S_1, \ldots, S_l \text{ are closed in } G = G_{n,p}\right] \leq \prod_{i=1}^{l} \mathrm{P}\left[\forall v \in V \setminus \bigcup_{j=1}^{l} S_j : e(v, S_i) > 0\right],$$

so that (6) follows from (7).

Now, let $s_1, \ldots, s_l$ be integers such that

$$0 \leq s_1, \ldots, s_l \leq s \text{ and } \sum_{i=1}^{l} s_i \leq n - \nu. \tag{8}$$

Let $\mathcal{E}(s_1, \ldots, s_l)$ signify the event that $G = G_{n,p}$ admits pairwise disjoint closed sets S_i of cardinalities $\#S_i = s_i$ ($i = 1, \ldots, l$). Then by (6)

$$\begin{aligned}\mathrm{P}\left[\mathcal{E}(s_1, \ldots, s_l)\right] &\leq \prod_{i=1}^{l} \binom{n}{s_i} \left(1 - (1-p)^{s_i}\right)^{\nu} \leq \left[\binom{n}{s} \left(1 - (1-p)^{s}\right)^{\nu}\right]^{l} \\ &\leq \left(\frac{\mathrm{e}n}{s}\right)^{ls} \exp\left[-l \cdot \nu (1-p)^{s}\right] \leq \exp\left[n\left(1 + \ln\left(\frac{n}{s}\right) - \frac{\nu}{s}(1-p)^{s}\right)\right], \end{aligned} \tag{9}$$

because $ls = n$. Since $p \leq 0.99$, we have $\ln(1-p)/p \geq -4.7$. Hence,

$$\frac{\nu}{s}(1-p)^{s} = \frac{\nu}{s} \exp\left[\ln(np) \frac{\ln(1-p)}{10p}\right] \geq \frac{10np}{\ln(np)^2} \cdot (np)^{-0.47} \geq 10\sqrt{np}, \tag{10}$$

because $np \geq c_0$ for a large constant $c_0 > 0$. Moreover,

$$\ln\left(\frac{n}{s}\right) = \ln\left(\frac{10np}{\ln(np)}\right) \leq \ln(np). \tag{11}$$

Plugging (10) and (11) into (9), we get

$$\mathrm{P}\left[\mathcal{E}(s_1, \ldots, s_l)\right] \leq \exp\left[n\left(1 + \ln(np) - 10\sqrt{np}\right)\right] \leq \exp\left[-9n^{3/2}p^{1/2}\right]. \tag{12}$$

Finally, let $\mathcal{S}$ be the set of all tuples $(s_1, \ldots, s_l)$ that satisfy (8). By the union bound,

$$\begin{aligned}\mathrm{P}\left[\mathcal{E}\right] &\leq \sum_{(s_1, \ldots, s_l) \in \mathcal{S}} \mathrm{P}\left[\mathcal{E}(s_1, \ldots, s_l)\right] \overset{(12)}{\leq} (s+1)^{l} \exp\left[-9n^{3/2}p^{1/2}\right] \\ &\leq \exp\left[l \ln(n) - 9n^{3/2}p^{1/2}\right] \leq \exp\left[-8n^{3/2}p^{1/2}\right] \leq \exp(-2n),\end{aligned}$$

because $l \ln(n) = o(n^{3/2}p^{1/2})$.

3 The Algorithm `ApxColor`

`ApxColor` combines the procedure `GreedyCoreColor` from Section 2 with computing the *vector chromatic number* $\hat{\chi}(G)$ of its input graph G. The vector chromatic number is a semidefinite programming relaxation of the chromatic number and was introduced by Karger, Motwani, and Sudan [10]. For all graphs G we have $\hat{\chi}(G) \leq \chi(G)$; moreover, $\hat{\chi}(G)$ can be computed in polynomial time (cf. [10]). Furthermore, the following result from [4] bounds $\hat{\chi}(G_{n,p})$.

Lemma 7. *There are constants $c_0, c_1 > 0$ such that the following holds. If $c_0/n \leq p \leq 0.99$, then* $\mathrm{P}\left[\hat{\chi}(G_{n,p}) \geq c_1\sqrt{np}\right] \geq 1 - \exp(-n)$.

Algorithm 8. `ApxColor`(G, p)
Input: A graph $G = (V, E)$ on n vertices and a number $0 < p < 1$. *Output:* A coloring of G.

1. If $\hat{\chi}(G) \geq c_1\sqrt{np}$ for a certain constant $c_1 > 0$ (cf. Lemma 7),
2. Run `GreedyCoreColor`(G, p) and output the resulting coloring.
3. Otherwise run `Lawler`(G) to compute an optimal coloring in time $O(2.443^n)$.

Proof of Theorem 1. Suppose that $c_0/n \leq p \leq 0.99$ for a sufficiently large constant $c_0 > 0$. Step 1 of `ApxColor` has polynomial running time. Furthermore, by Lemma 7 the probability that `ApxColor`$(G_{n,p}, p)$ executes Step 3 is $\leq \exp(-n)$. Thus, the expected time spent on executing Step 3 is polynomial. Finally, by Theorem 2, the expected running time of Step 2 is polynomial.

To prove that `ApxColor` achieves an approximation ratio of $O(\sqrt{np}/\ln(np))$, let G be any input graph. If $\hat{\chi}(G) < c_1\sqrt{np}$, then Step 3 of `ApxColor` outputs an optimal coloring. Thus, assume that $\hat{\chi}(G) \geq c_1\sqrt{np}$. By Theorem 2, `GreedyCoreColor`$(G, p)$ uses at most $30np/\ln(np) + \chi(G)$ colors. Therefore, we obtain an approximation ratio of

$$\frac{\chi(G) + 30np/\ln(np)}{\chi(G)} \leq 1 + \frac{30np}{\ln(np) \cdot \hat{\chi}(G)} \leq 1 + \frac{30\sqrt{np}}{c_1 \ln(np)} = O\left(\frac{\sqrt{np}}{\ln(np)}\right),$$

thereby proving the theorem. □

References

1. Achlioptas, D., Naor, A.: The two possible values of the chromatic number of a random graph. Proc. 36th STOC (2004) 587–593
2. Bollobás, B.: The chromatic number of random graphs. Combinatorica **8** (1988) 49–55
3. Bollobás, B.: Random graphs. 2nd edition. Cambridge University Press (2001)
4. Coja-Oghlan, A.: The Lovász number of random graphs. To appear in Combinatorics, Probability and Computing. A preliminary version appeared in the Proc. 7th RANDOM (2003) 228–239
5. Coja-Oghlan, A., Moore, C., Sanwalani, V.: MAX k-CUT and approximating the chromatic number of random graphs. Proc. 30th ICALP (2003) 200–211
6. Coja-Oghlan, A., Taraz, A.: Exact and approximative algorithms for coloring $G(n, p)$. Random Struct. Alg. **24** (2004) 259–278
7. Feige, U., Kilian, J.: Zero knowledge and the chromatic number. J. Comp. Syst. Sci. **57** (1998) 187–199
8. Frieze, A., McDiarmid, C.: Algorithmic theory of random graphs. Random Struct. Alg. **10** (1997) 5–42
9. Grimmett, G., McDiarmid, C.: On colouring random graphs. Math. Proc. Camb. Phil. Soc. **77** (1975) 313–324
10. Karger, D., Motwani, R., Sudan, M.: Approximate graph coloring by semidefinite programming. J. ACM **45** (1998) 246–265
11. Krivelevich, M.: Coloring random graphs – an algorithmic perspective. Proc. 2nd Colloquium on Mathematics and Computer Science (2002) 175-195.
12. Krivelevich, M., Vu, V.H.: Approximating the independence number and the chromatic number in expected polynomial time. J. Comb. Opt. **6** (2002) 143–155
13. Kučera, L.: The greedy coloring is a bad probabilistic algorithm. J. of Algorithms **12** (1991) 674–684
14. Lawler, E.L.: A note on the complexity of the chromatic number problem. Inf. Proc. Lett. **5** (1976) 66–67
15. Łuczak, T.: The chromatic number of random graphs. Combinatorica **11** (1991) 45–54