Coloring 3-colorable graphs with $o(n^{1/5})$ colors

Ken-ichi Kawarabayashi*[1] and Mikkel Thorup†[2]

1 **National Institute of Informatics, Tokyo Japan / JST ERATO Kawarabayashi Project, Tokyo, Japan**
k_keniti@nii.ac.jp

2 **University of Copenhagen, Copenhagen, Denmark**
mikkel2thorup@gmail.com

Abstract

Recognizing 3-colorable graphs is one of the most famous NP-complete problems [Garey, Johnson, and Stockmeyer STOC'74]. The problem of coloring 3-colorable graphs in polynomial time with as few colors as possible has been intensively studied: $O(n^{1/2})$ colors [Wigderson STOC'82], $\widetilde{O}(n^{2/5})$ colors [Blum STOC'89], $\widetilde{O}(n^{3/8})$ colors [Blum FOCS'90], $O(n^{1/4})$ colors [Karger, Motwani, Sudan FOCS'94], $\widetilde{O}(n^{3/14}) = O(n^{0.2142})$ colors [Blum and Karger IPL'97], $O(n^{0.2111})$ colors [Arora, Chlamtac, and Charikar STOC'06], and $O(n^{0.2072})$ colors [Chlamtac FOCS'07]. Recently the authors got down to $O(n^{0.2049})$ colors [FOCS'12]. In this paper we get down to $O(n^{0.19996}) = o(n^{1/5})$ colors.

Since 1994, the best bounds have all been obtained balancing between combinatorial and semi-definite approaches. We present a new combinatorial recursion that only makes sense in collaboration with semi-definite programming. We specifically target the worst-case for semi-definite programming: high degrees. By focusing on the interplay, we obtained the biggest improvement in the exponent since 1997.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems – Computations on discrete structures, G.2.2 Graph Theory – Graph algorithms

Keywords and phrases Approximation Algorithms, Graph Coloring

Digital Object Identifier 10.4230/LIPIcs.STACS.2014.458

1 Introduction

If ever you want to illustrate the difference between what we consider hard and easy to someone not from computer science, use the example of 2-coloring versus 3-coloring: suppose there is too much fighting in a class, and you want to split it so that no enemies end up in the same group. First you try with a red and a blue group. Put someone in the red group, and everyone he dislikes in the blue group, everyone they dislike in the red group, and so forth. This is an easy systematic approach. Digging a bit deeper, if something goes wrong, you have an odd cycle, and it is easy to see that if you have a necklace with an odd number of red and blue beads, then the colors cannot alternate perfectly. This illustrates both efficient algorithms and the concept of a witness. Knowing that red and blue do not suffice, we might try introducing green, but this is already beyond what we believe computers can do.

* Research partly supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research and by Mitsubishi Foundation.

† Research partly supported by an Advanced Grant from the Danish Council for Independent Research under the Sapere Aude research carrier programme.

© Ken-ichi Kawarabayashi and Mikkel Thorup;
licensed under Creative Commons License CC-BY
31st Symposium on Theoretical Aspects of Computer Science (STACS'14).
Editors: Ernst W. Mayr and Natacha Portier; pp. 458–469
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

Formally a k-coloring of an undirected graph assigns k colors to the vertices. The coloring is only valid if no two adjacent vertices get the same color. The validity of coloring is trivially checked in linear time so the deciding if a graph is k-colorable is in NP.

Three-coloring is a classic NP-hard problem. It was proved hard by Garey, Johnson, and Stockmeyer at STOC'74 [9], and was the prime example of NP-hardness mentioned by Karp in 1975 [13]. Bipartite or 2-colorable graphs are very well-understood. How about tripartite or 3-colorable graphs? How can we reason about them if we cannot recognize them? Three-colorable graphs are obvious targets for any approach to NP-hard problems. With the approximation approach, given a 3-colorable graph, that is a graph with an unknown 3-coloring, we try to color it in polynomial time using as few colors as possible. The algorithm is allowed to fail or give up if the input graph was not 3-colorable. If a coloring is produced, we can always check that it is valid even if the input graph is not 3-colorable. This challenge has engaged many researchers. At STOC'82, Wigderson [17] got down to $O(n^{1/2})$ colors for a graph with n vertices. Berger and Rompel [3] improved this to $O((n/(\log n))^{1/2})$. Blum [4] came with the first polynomial improvements, first to $\widetilde{O}(n^{2/5})$ colors at STOC'89, and then to $\widetilde{O}(n^{3/8})$ colors at FOCS'90.

The next big step at FOCS'94 was by Karger, Motwani, Sudan [12] using semi-definite programming (SDP). This came in the wake of Goemans and Williamson's seminal use of SDP for max-cut at STOC'94 [10]. For a graph with maximum degree $\Delta_{\max}$, Karger et al. got down to $O(\Delta_{\max}^{1/3})$ colors. Combining this with Wigderson's algorithm, they got down to $O(n^{1/4})$ colors. Later Blum and Karger [5] combined the SDP from [12] with Blum's [4] algorithm, yielding an improved bound of $\widetilde{O}(n^{3/14}) = \widetilde{O}(n^{0.2142})$. Later improvements on semi-definite programming have also been combined with Blum's algorithm. At STOC'06, Arora, Chlamtac, and Charikar [1] got down to $O(n^{0.2111})$ colors. The proof in [1] is based on the seminal result of Arora, Rao and Vazirani [2] which gives an $O(\sqrt{\log n})$ algorithm for the sparsest cut problem. At FOCS'07 Chlamtac [6] got down to $O(n^{0.2072})$ colors. Recently, at FOCS'12 [14], we presented a purely combinatorial approach (for the first time since Blum [4]), getting down to $\widetilde{O}(n^{4/11})$ colors. Combining it with Chlamtac's SDP [6], we got down to $O(n^{0.2049})$ colors.

Only a few lower bounds are known for the coloring of 3-colorable graphs. We know that it is NP-hard to get down to 5 colors [11, 15]. Recently, Dinur, Mossel and Regev [7] showed that it's hard to color with any constant number of colors (i.e., $O(1)$ colors) based on a variant of the Unique Games Conjecture.

Integrality gap results [8, 12, 16] indicates that our understanding of SDP coloring [2, 6, 12] is close to optimal, and it is therefore natural to go back and see if we can improve things combinatorially.

In this paper we show how to color any 3-colorable n vertex graph in polynomial time using only $O(n^{0.19996})$ colors. This is the biggest single improvement in the exponent since 1997 [5], and in particular, we pass the $n^{1/5}$ milestone. Our approach is combinatorial, but aiming at a better combination with SDP, we specifically target the worst-case for SDP: high degrees.

Technical perspective. To appreciate our result, we have to consider the interplay between combinatorial and semi-definite methods in the above mentioned papers. A parameter Δ is picked. Using Blum's notion of progress, it suffices to work with graphs that either have minimum degree Δ or maximum degree Δ. A high minimum degree is good for combinatorial approaches while a low maximum degree is good for semi-definite approaches. The best bounds are obtained choosing Δ to balance between the best semi-definite and combinatorial approaches.

On the combinatorial side, the coloring bounds have followed, for $i = 1, 2, 3, 4$, the sequence $\widetilde{O}((n/\Delta)^{i/(2i-1)})$. Here $i = 1$ is from Wigderson at STOC'82 [17], $i = 2$ is from Blum at STOC'89 [4], $i = 3$ is from Blum at FOCS'90 [4][1], and $i = 4$ is from Kawarabayashi and Thorup at FOCS'12 [14]. For $i \to \infty$, the sequence approaches its limit $\widetilde{O}((n/\Delta)^{1/2})$. Each of the above steps is based on a new combinatorial coloring idea. It is rather curious (1) that the resulting bounds have all been of the form $\widetilde{O}((n/\Delta)^{i/(2i-1)})$, and (2) that none of these STOC/FOCS papers skipped a step in this sequence of bounds.

For a purely combinatorial algorithm, we balance the above bounds with the trivial Δ-coloring that takes out any vertex v with $< \Delta$ neighbors, colors the rest of the graph inductively, and give v the first color not used in its neighborhood.

The first semi-definite solution of Karger et al. from FOCS'94 [12], got $O(\Delta^{1/3})$ colors. Balancing this with yet to be found $\widetilde{O}((n/\Delta)^{1/2})$ coloring, would yield $\widetilde{O}(n^{1/5})$ colors, which have thus been a natural milestone. Later semi-definite approaches of Arora, Chlamtac, and Charikar at STOC'06 [1] and Chlamtac at FOCS'07 [6], have gotten down to $O(\Delta^{1/3-\varepsilon(n,\Delta)})$ colors where $\varepsilon(n, \Delta) > 0$ is a small value that decreases as complicated function of Δ. The integrality gap from [8] implies that $\varepsilon(n, \Delta) = o(1)$ for $\Delta = n^{o(1)}$.

Chlamtac [personal communication] stated that we would pass the $n^{1/5}$ milestone if the combinatorial side could get down around $\widetilde{O}((n/\Delta)^{12/23})$ colors. This, however, is 8 steps away in the current sequence where the first 4 steps have taken 20 years, each introducing a new combinatorial idea.

Our goal is to improve the overall coloring bound in terms of n, and we will indeed get down to $o(n^{1/5})$ colors. Using our previous combinatorial algorithm [14] as a subroutine, we present a novel recursion that gets us down to $\widetilde{O}((n/\Delta)^{12/23})$ colors, but only for the large values of Δ needed for an optimal combination with SDP. In combination with Chlamtac's SDP [6], we get a polynomial time algorithm that colors any 3-colorable graph on n vertices with $O(n^{0.19996})$ colors.

We note that for smaller values of Δ, our new recursion does not offer any improvement over our previous combinatorial bound $\widetilde{O}((n/\Delta)^{4/7})$ from [14]. Instead of adding another independent dot, we connect the dots, improving the combinatorial side only in the parameter range of relevance for combination with SDP.

Contents. The paper is organized as follows. In a preliminary Section 2, we present notations and basic results needed from [4]. In Section 3, we review our previous algorithm from [14] which we shall use here as a subroutine. In Section 4 we present our novel recursion around this subroutine. This completes the description of our new algorithm. Switching to the analysis, in Section 5 we identify the properties of the subroutine from [14] that we need for our recursion. This properties follow from the analysis from [14], as will be verified in a combined journal version. In Section 6 we use these properties for an inductive analysis of our new recursion.

2 Preliminaries

We hide $\log n$ factors, so we use the notation that $\widetilde{O}(x) \leq x \log^{O(1)}(n)$, $\widetilde{\Omega}(x) \geq x/\log^{O(1)}(n)$, $\widetilde{o}(x)) \leq x/\log^{\omega(1)}(n)$, and $\widetilde{\omega}(x) \geq x \log^{\omega(1)}(n)$.

We are given a 3-colorable graph $G = (V, E)$ with $|V| = n = \omega(1)$ vertices. The (unknown) 3-colorings are with red, green, and blue. For a vertex v, we let $N(v)$ denote its set of

[1] The reference is to the joint journal paper

neighbors. For a vertex set $X \subseteq V$, let $N(X) = \bigcup_{v \in X} N(v)$ be the neighborhood of X. If Y is a vertex set, we use N_Y to denote neighbors in Y, so $N_Y(v) = N(v) \cap Y$ and $N_Y(X) = N(X) \cap Y$. We let $d_Y(v) = |N(v) \cap Y|$ and $d_Y(X) = \{d_Y(v) \mid v \in Y\}$. Then $\min d_Y(X)$, $\max d_Y(X)$, and $\operatorname{avg} d_Y(X)$, denotes the minimum, maximum, and average degree from X to Y.

For some color target k depending on n, we wish to find an $\widetilde{O}(k)$ coloring of G in polynomial time. We reuse several ideas and techniques from Blum's approach [4].

Progress

Blum has a general notion of *progress towards an $\widetilde{O}(k)$ coloring* (or *progress* for short if k is understood). The basic idea is that such progress eventually leads to a full $\widetilde{O}(k)$ coloring of a graph. Blum presents three types of progress towards $\widetilde{O}(k)$ coloring:
Type 0: Same color. Finding vertices u and v that have the same color in every 3-coloring.
Type 1: Large independent set. Finding an independent or 2-colorable vertex set X of size $\widetilde{\Omega}(n/k)$.
Type 2: Small neighborhood. Finding a non-empty independent or 2-colorable vertex set X such that $|N(X)| = \widetilde{O}(k|X|)$.

In order to get from progress to actual coloring, we want k to be bounded by a *near-polynomial* function f of n where near-polynomial means that f is non-decreasing and that there are constants $c, c' > 1$ such that $cf(n) \leq f(2n) \leq c'f(n)$ for all n. As described in [4], this includes any function of the form $f(n) = n^\alpha \log^\beta n$ for constants $\alpha > 0$ and β.

▶ **Lemma 1** ([4, Lemma 1]). *Let f be near-polynomial. If we in time polynomial in n can make progress towards $\widetilde{O}(f(n))$ coloring of either Type 0, 1, or 2, on any 3-colorable graph on n vertices, then in time polynomial in n, we can $\widetilde{O}(f(n))$ color any 3-colorable graph on n vertices.*

The general strategy is to identify a small parameter k for which we can guarantee progress. To apply Lemma 1 and get a coloring, we need a bound f on k where f is near-polynomial in n. As soon as we find one progress of the above types, we are done, so generally, whenever we see a condition that implies progress, we assume that the condition is not satisfied.

Our focus is to find a vertex set X, $|X| > 1$, that is guaranteed to be monochromatic in every 3-coloring. This will happen assuming that we do not get other progress on the way. When we have the vertex set X, we get same-color progress for any pair of vertices in X. We refer to this as *monochromatic progress*.

Most of our progress will be made via results of Blum presented below using a common parameter

$$\Psi = n/k^2. \tag{1}$$

A very useful tool we get from Blum is the following multichromatic (more than one color) test:

▶ **Lemma 2** ([4, Corollary 4]). *Given a vertex set $X \subseteq V$ of size at least $\Psi = n/k^2$, in polynomial time, we can either make progress towards an $\widetilde{O}(k)$-coloring of G, or else guarantee that under* every *legal 3-coloring of G, the set X is multichromatic.*

The following lemma is implicit in [4] and explicit in [14].

▶ **Lemma 3** ([14, Lemma 6]). *If the vertices in a set Z on the average have d neighbors in U, then the whole set Z has at least $\min\{d/\Psi, |Z|\}\, d/2$ distinct neighbors in U (otherwise some progress is made).*

Large minimum degree

Our algorithms will exploit a lower bound Δ on the minimum degree in the graph. It is easily seen that if a vertex v has d neighbors, then we can make progress towards $\widetilde{O}(d)$ coloring since this is a small neighborhood for Type 2 progress. For our color target k, we may therefore assume:

$$k \leq \Delta / \log^a n \text{ for any constant } a. \tag{2}$$

However, combining with semi-definite programming (SDP) as in [5], we can assume a much larger minimum degree. The combination is captured by the following lemma, which is proved in [14, §VIII]:

▶ **Lemma 4** ([5, 14]). *Suppose that for some near-polynomial functions d and f, we for any n can make progress towards $\widetilde{O}(f(n))$ coloring for*

- *any 3-colorable graph on n vertices with minimum degree $\geq d(n)$.*
- *any 3-colorable graph on n vertices with maximum degree $\leq d(n)$.*

Then we can make progress towards $\widetilde{O}(f(n))$-coloring on any 3-colorable graph on n vertices.

Using the SDP from [12], we can make progress towards $d(n)^{1/3}$ for graphs with degrees below $d(n)$, so by Lemma 4, we may assume

$$k \leq \Delta^{1/3}. \tag{3}$$

We can do even better using the strongest SDP result of Chlamtac from [6, Theorem 15]:

▶ **Theorem 5** ([6]). *For any $\tau > \frac{6}{11}$ there is a $c > 0$ such that there is a polynomial time algorithm that for any 3-colorable graph G with n vertices and all degrees below $\Delta = n^\tau$ finds an independent set of size $\widetilde{\Omega}\left(n/\Delta^{1/(3+3c)}\right)$. Hence we can make Type 2 progress towards an $\widetilde{O}\left(\Delta^{1/(3+3c)}\right) = \widetilde{O}\left(n^{\tau/(3+3c)}\right)$-coloring.*

The requirement on τ and c is that $c < 1/2$ and $\lambda_{c,\tau}(\alpha) = 7/3 + c + \alpha^2/(1-\alpha^2) - (1+c)/\tau - (\sqrt{(1+\alpha)/2} + \sqrt{c(1-\alpha)/2})^2$ is positive for all $\alpha \in [0, \frac{c}{1+c}]$.

▶ **Corollary 6.** *In polynomial time, for any 3-colorable graph with n vertices, and all degrees below $\Delta = n^{0.61674333}$, we can make progress towards an $\widetilde{O}(n^{0.19996})$-coloring.*

Proof. We apply Theorem 5 with $\tau = 0.6167433$ and $c = 0.02811113$. Then For $\alpha \in [0, \frac{c}{1+c}]$, it is easily verified that $\lambda_{c,\tau}(\alpha)$ is minimized and positive with $\alpha = 0.0273425$. Then $\tau/(3+3c) = 0.19996$. ◀

By Lemma 4, we may thus assume

$$k = n^{0.19996} \text{ and } \Delta = n^{0.61674333}. \tag{4}$$

With this setting k is slightly smaller than $(n/\Delta)^{12/23}$. Our original algorithm from [14] only assumes the combinatorial bound (2), to make progress towards $\widetilde{O}((n/\Delta)^{4/7})$ coloring. It is only in our new developments that we need the higher degrees that can be assumed via SDP.

Two-level neighborhood structure

The most complex ingredient we get from Blum [4] is a certain regular second neighborhood structure. Let Δ be the smallest degree in the graph G. In fact, we shall use the slightly modified version described in [14].

Unless other progress is made, for some $\Delta_1 = \widetilde{\Omega}(\Delta)$, in polynomial time [4, 14] identifies a 2-level neighborhood structure $H_1 = (r_1, S_1, T_1)$ in G consisting of:

- A root vertex r_1. We assume r_1 is colored red in any 3-coloring.
- A first neighborhood $S_1 \subseteq N(r_1)$ of size at least Δ_1.
- A second neighborhood $T_1 \subseteq N(S_1)$ of size at most n/k. The sets S_1 and T_1 may overlap.
- The edges between vertices in H_1 are the same as those in G.
- The vertices in S_1 all have degrees at least Δ_1 into T_1.
- For some δ_1 the degrees from T_1 to S_1 are all between δ_1 and $5\delta_1$.

3 Review of our FOCS'12 coloring

Our algorithm makes internal use of the coloring algorithm from [14], which we review below. It uses the above 2-level neighborhood structure $H_1 = (r_1, S_1, T_1)$, and works on induced subproblems $(S, T) \subseteq (S_1, T_1)$ defined in terms of a subsets $S \subseteq S_1$ and $T \subseteq T_1$. The edges considered in the subproblem are exactly those between S and T in G. This edge set is denoted $E(S, T)$.

With r_1 red in any 3-coloring, we know that all vertices in $S \subseteq S_1 \subseteq N(r_1)$ are blue or green. We say that a vertex in T has *high S-degree* if its degree to S is bigger than $\delta_1/4$, and we will make sure that any subproblem (S, T) considered satisfies:

(i) We have more than Ψ vertices of high S-degree in T.

In [14, §IV] we implemented a subroutine `cut-or-color`(t, S, T) which for a problem $(S, T) \subseteq (S_1, T_1)$ starts with an arbitrary high S-degree vertex $t \in T$. It has one of the following outcomes:

- Some progress toward a $\widetilde{O}(k)$-coloring. Then we are done, so we assume that this does not happen.
- A guarantee that if r_1 and t have different colors in a 3-coloring C_3 of G, then S is monochromatic in C_3.
- Reporting a "sparse cut around a subproblem $(X, Y) \subseteq (S, T)$" satisfying the following conditions:
  - **(i)** The original high S-degree vertex t has all its neighbors from S in X, that is, $N_S(t) \subseteq X$.
  - **(ii)** All edges from X to T go to Y, so there are no edges between X and $T \setminus Y$.
  - **(iii)** Each vertex $s' \in S \setminus X$ has $|N_Y(s')| < \Psi$.
  - **(iv)** Each vertex $t' \in T \setminus Y$ has $|N_Y(N_S(t'))| < \Psi$.

Assuming `cut-or-color`, we now review the main recursive algorithm, `monochromatic`, from [14]. It takes as input a subproblem (S, T). The pseudo-code is presented in Algorithm 1.

Algorithm 1: `monochromatic`(S, T)

```
let U be the set of high S-degree vertices in T;
check that U is multichromatic in G with Lemma 2;      // if not, progress found and we are done
if there is a t ∈ U such that cut-or-color(S, T, t) returns "sparse cut around (X, Y)" then
    recursively call monochromatic(X, Y)
else
    return "S is monochromatic in every 3-coloring"
```

Let U be the set of high S-degree vertices in T. By 1 we have $|U| \geq \Psi$, so we can apply Blum's multichromatic test from Lemma 2 to U in G. Assuming we did not make progress,

we know that U is multichromatic in every valid 3-coloring. We now apply `cut-or-color` to each $t \in U$, stopping only if a sparse cut is found or progress is made. If we make progress, we are done, so assume that this does not happen. If a sparse cut around a subproblem (X, Y) is found, we recurse on (X, Y).

The most interesting case is when we get neither progress nor a sparse cut.

▶ **Lemma 7.** *If* `cut-or-color` *does not find progress nor a sparse cut for any high S-degree $t \in U$, then S is monochromatic in every 3-coloring of G.*

Proof. Consider any 3-coloring C_3 of G. With Lemma 2 we checked that U is multichromatic in every 3-coloring of G including C_3, so there is some $t \in U$ that has a different color than r_1 in C_3. With this t, `cut-or-color`(S, T, t) guarantees that S is monochromatic in C_3. Note that different 3-colorings may use a different vertex t for the guarantee, and our algorithm does not need to know which t are used. ◀

Thus, unless other progress is made, `monochromatic` ends up with a set S that is monochromatic in every 3-coloring, and then monochromatic progress can be made. However, the correctness demands that we respect 1 and never apply `monochromatic` to a subproblem (S, T) where T has less than Ψ high S-degree vertices (otherwise Lemma 2 cannot be applied to U).

In [14] it is proved that 1 is respected when the recursion `monochromatic`(S_1, T_1) starts in the initial two-level structure (S_1, T_1) from Section 2. In each recursive step, we take the subproblem (X, Y) returned by `cut-or-color`, and recurse on $(S, T) = (X, Y)$. The analysis from [14] has the following points:

- If the average degree from Y to X is at least $\delta_1/2$, then $(S, T) = (X, Y)$ satisfies 1.
- The set Y is of size at least $\Delta_1^2 k^2/(2n)$ and has at least $\delta_1|Y| \geq \delta_1\Delta_1^2 k^2/(2n)$ edges to S_1.
- The set Y has at most $(40\delta_1 n^2)/(\Delta_1^2 k^4) \cdot |T_1|$ edges to $T_1 \setminus X$ (this is the hard part).
- We pick $k = \Theta((n/\Delta_1)^{4/7})$ such that

$$\frac{40\delta_1 n^2}{\Delta_1^2 k^4}|T_1| = \delta_1\Delta_1^2 k^2/(4n) \leq \delta_1|Y|/2.$$

Then the average degree from Y to X is at least $\delta_1/2$, implying 1 for $(S, T) = (X, Y)$.

4 A novel outer loop for high degree graphs

As described above, the first call to Algorithm 1 is with the initial problem (S_1, T_1) that is *regular* in the sense that the degrees from T_1 to S_1 are all between δ_1 and $5\delta_1$ for some δ_1. However, with a color target k below $\Theta((n/\Delta_1)^{4/7})$, we can no longer guarantee that the average degree from Y to X remains above $\delta_1/2$. To preserve the correctness, we will stop our recursive Algorithm 1 if we get to a subproblems (X, Y) where the average degree from Y to X is less than $\delta_1/2$. Inside (X, Y) we find a new regular subproblem (S_2, T_2) where the degrees are between δ_2 and $5\delta_2$ for some δ_2. Again we apply Algorithm 1 until the average drops below $\delta_2/2$. We continue this new outer loop, generating a sequence of regular subproblems $(S_1, T_1) \supset (S_2, T_2) \supset (S_3, T_3) \supset \cdots$, until we somehow end up either making progress, or some error event happens. In combination with SDP, our analysis will show that this outer loop can be used to give error-free progress towards $\tilde{O}(n^{0.19996})$ coloring.

The regularization is described in Algorithm 2, and it is, in itself, fairly standard. Blum [4] used several similar regularizations.

Algorithm 2: `regularize`(S,T)

Let $d_\ell = (4/3)^\ell$;
Partition the vertices of T into sets $U_\ell = \{v \in T \mid d_S(v) \in [d_\ell, d_{\ell+1})\}$;
Subject to $d_\ell \geq \operatorname{avg} d_S(T)/2$ let ℓ maximize $|E(U_\ell, S)|$;
$\delta^r \leftarrow d_\ell/4$; $\Delta^r \leftarrow \operatorname{avg} d_{U_\ell}(S)/4$;
Repeatedly remove vertices $v \in S$ with $d_{U_\ell}(v) \leq \Delta^r$ and $w \in U_\ell$ with $d_S(w) \leq \delta^r$;
$S^r \leftarrow S$; $T^r \leftarrow U_\ell$;
return $(S^r, T^r, \Delta^r, \delta^r)$

▶ **Lemma 8.** *When* `regularize`(S,T) *in Algorithm 2 returns* $(S^r, T^r, \Delta^r, \delta^r)$ *then* $\Delta^r \geq \operatorname{avg} d_T(S)/(30 \lg n)$ *and* $\delta^r \geq \operatorname{avg} d_S(T)/8$. *The sets* S^r *and* T^r *are both non-empty. The degrees from* S^r *to* T^r *are at least* Δ^r *and the degrees from* T^r *to* S^r *are between* δ^r *and* $5\delta^r$.

Proof. Below S and U_ℓ refers to the sets before vertices are removed. We have S^r and T^r denoting the sets after the vertices have been removed.

To prove $\Delta^r \geq \operatorname{avg} d_T(S)/(30 \lg n)$, we first note that the sets U_ℓ with $d_\ell < \operatorname{avg} d_S(T)/2$ only contain vertices of degree below $(4/3)\operatorname{avg} d_S(T)/2 = (2/3)\operatorname{avg} d_S(T)$, so at least $1/3$ of the edges from $E(S,T)$ leave vertices from sets U_ℓ satisfying the condition $d_\ell \leq \operatorname{avg} d_S(T)/2$. There are only $\log_{4/3} n < (5/2) \lg n$ possible values of ℓ, and subject to the condition, we picked ℓ maximizing $E(S, U_\ell)$. Therefore $|E(S,U_\ell)| > (1/3)|E(S,T)|/((5/2)\lg n) = (2/15)|E(S,T)|/\lg n$. It follows that $\Delta^r \geq \operatorname{avg} d_{U_\ell}(S)/4 > \operatorname{avg} d_T(S)/(30 \lg n)$.

The only other slightly non-trivial statement is that the sets S^r and T^r do not end up empty. When we remove vertices from S, we remove at most $|S| \operatorname{avg} d_{U_\ell}(S)/4 \leq |E(S,U_\ell)|/4$ edges, and likewise for the vertices removed from U_ℓ, so these removals take away at most half the edges. It follows that some edges remain hence that $S^r, T^r \neq \emptyset$. ◀

Our new coloring is described in Algorithm 3. Except for the possible regularization, each round j is an iterative version of the recursive Algorithm 1. Moreover, we have made it self-checking in the sense that we report an error if the set U of high degree vertices is too small for 1 ("Error B" below). Also, we report an error if the set S is too small for monochromatic progress which requires at least two same-color vertices ("Error A" below). With $k = \Theta((n/\Delta)^{4/7})$, the analysis from [14] shows that we never get an error and that we never get $|E(S,T)| \leq \delta_1 |T|/2$, so the regularization never happens.

The outer loop in Algorithm 3 continues until it either makes an error, or makes progress. The progress can either be explicit with a monochromatic set, or it can happen implicitly as part of the multichromatic test from Lemma 2. Ensuring that we make progress and no errors happen will require a very careful choice of parameters, and we will only gain over [14] when large minimum degree vertices are guaranteed from SDP as in (3) or (4).

5 A good round

When we start round j of Algorithm 3 with a problem $(S_j, T_j, \Delta_j, \delta_j)$, it follows directly from Lemma 8 that the degrees from T_j to S_j are between δ_j and $5\delta_j$, and that the degrees from S_j to T_j are all at least Δ_j.

The journal version of this conference paper will also cover [14] and there we will make a simple generalization of the analysis from [14] so that it applies to an arbitrary round j

Algorithm 3: Seeking progress towards $\widetilde{O}(k)$ coloring

```
let (S_1, T_1, Δ_1, δ_1) be the initial two-level structure from Section 2;
for j ← 1, 2, ... do        // outer loop, round j
    (S, T) ← (S_j, T_j);
    repeat        // iterative version of recursive monochromatic(S_j, T_j)
        if |S| ≤ 1 then return "Error A";
        U ← {v ∈ T | d_S(v) ≥ δ_j/4};
        if |U| < Ψ then return "Error B";
        check U multichromatic with Lemma 2;        // if not, progress was found and we are
        done
        if ∃t ∈ U such that cut-or-color(S, T, t) returns "sparse cut around (X, Y)" then
            (S, T) ← (X, Y)
        else return "S is monochromatic in every 3-coloring, so monochromatic progress
        found"
    until |E(S, T)| < δ_j|T|/2;
    (S_j, T_j, Δ_j, δ_j) = regularize(S, T);
```

of the outer loop in Algorithm 3—not just round 1. Below we describe the outcome of the analysis.

The basic requirements for the analysis is that the following *pre-conditions* are satisfied:

$$\Delta_j = \omega(\Psi) \tag{5}$$

$$\delta_j \geq 4\Delta_j/\Psi \tag{6}$$

Based on the pre-conditions, it will follow that no error is made in the round. Also, for any subproblem (X, Y) considered, it will follow that

$$\min d_Y(X) \geq \Delta_j \tag{7}$$

$$|X| \geq \delta_j/4 \tag{8}$$

$$|Y| \geq \Delta_j^2/(2\Psi) \tag{9}$$

If no progress is made in the round, we will get to a subproblem (X, Y) where the average degree from Y to X is smaller than $\delta_j/2$. This is where we terminate the round and regularize. Let (X_j, Y_j) denote this final subproblem of round j. The most interesting part of the analysis is to argue

$$|Y_j| \leq |T_j|(80n^2)/(\Delta_j^2 k^4). \tag{10}$$

Note here that if the upper bound from (10) is smaller than the lower bound in (9), then we can conclude that we never get to the last subproblem (X_j, Y_j), hence progress must have be made in round j. With the parameters from [14], we get this contradiction already for the first round $j = 1$. However, with a smaller k, the upper bound is higher, and then more rounds may happen.

We say round j is *good* if

- the pre-conditions (5) and (6) are satisfied at the beginning of the round.
- No error is made during the round.
- (7)–(10) are satisfied as long as no progress is made.

A simple generalization of the analysis from [14] implies

▶ **Lemma 9** ([14]). *Round j is good if and only if pre-conditions (5) and (6) are satisfied.*

6 Analysis of outer loop

Using Lemma 9 we will prove

▶ **Theorem 10.** *Suppose for some integer $c = O(1)$ that*

$$k = \widetilde{\omega}\left((n/\Delta)^{\frac{2c+2}{4c+3}}\right). \tag{11}$$

and for all $j = 1, ..., c-1$,

$$(\Delta/k)(\Delta k/n)^j(\Delta k^2/n)^{j(j+1)} = \Delta^{j^2+2j+1}k^{2j^2+3j-1}/n^{j^2+2j} = \widetilde{\omega}(1). \tag{12}$$

Then Algorithm 3 will make only good rounds, and make progress towards an $\widetilde{O}(k)$ coloring no later than round c.

If we did not have the j-bound (12), we would just make c very large, with (11) converging to $(n/\Delta)^{1/2}$. The j-bound (12) is rather unattractive, but we need it to make sure that no errors are made when $c > 1$. As an example, our previous bound $k = \widetilde{O}((n/\Delta)^{4/7})$ from [14] corresponds to the case $c = 1$ in (11). To improve this bound, we need $c > 1$. In particular, we need to satisfy (12) for $j = 1$ which becomes $\Delta^4 k^4/n^3 = \widetilde{\omega}(1)$. Now $k \leq (n/\Delta)^{4/7}$ implies $\Delta^4(n/\Delta)^{4/7\cdot 4}/n^3 = \Delta^{12/7}/n^{5/7} = \widetilde{\omega}(1) \iff \Delta = \widetilde{\omega}(n^{5/12})$. Thus we can only make improvements over [14] if we restrict ourselves to sufficiently high degrees, e.g., relying on SDP for lower degrees.

Note that if $\sqrt{n/\Delta} < k < n/\Delta$, then (12) must be minimized for some unique $j \in \mathbb{R}$. However, since $c = O(1)$, we can easily check (12) for all $j = 1, .., c-1$ with a computer.

In the rest of this section, we will prove Theorem 10 by induction assuming (11) and (12). Also, from Lemma 9, we know that round j is good if the pre-conditions (5) and (6) are satisfied. First, for the base case, we will show (a) that the pre-conditions are satisfied for round 1, and hence that round 1 is good. Next, assuming the first j rounds are good, but no progress is made, we will show (b) that $j < c$ and (c) that the pre-conditions of round $j+1$ are satisfied. By induction, (a), (b), and (c) imply Theorem 10.

First round

For the pre-conditions of the first round, we need

▶ **Lemma 11.** $\Psi = n/k^2 = \Delta/n^{\Omega(1)}$

Proof. Since c is constant, (11) implies $k > (n/\Delta)^{1/2+\Omega(1)}$, hence $\Psi = n/k^2 = \Delta/(n/\Delta)^{\Omega(1)}$. Finally we need to argue $(n/\Delta) = n^{\Omega(1)}$. This follows because the neighborhood of any vertex is 2-colorable, hence we always make Type 2 progress towards $\widetilde{O}(n/\Delta)$ coloring. We are only aiming for $k = n^{\Omega(1)}$ coloring, so we would be done if $(n/\Delta) = n^{o(1)}$. ◀

Since $\Delta_1 = \widetilde{\Omega}(\Delta)$, we get $\Psi = o(\Delta_1)$. Pre-condition (5) for round 1 thus follows Lemma 11.

For pre-condition (6), we note that $\Delta_1/\Psi = \Delta_1 k^2/n$ and $5\delta_1 \geq \Delta_1|S_1|/[T_1| = \widetilde{\Omega}(\Delta_1\Delta k/n)$. Moreover, by (3) we have $k \leq \Delta^{1/3}$. Hence $\delta_1 \gg 4\Delta_1/\Psi$, so pre-condition (6) is also satisfied for round 1. Thus we conclude that both pre-conditions are satisfied for round 1, hence by Lemma 9, round 1 is good.

General rounds

Now for $j \le c$, we assume that the first j rounds are good but that no progress is made. We want to prove that $j < c$ and that the pre-conditions of round $j+1$ are satisfied.

Since no progress is made, round j ends up regularizing. As in Section 5, we let X_j and Y_j denote the last values of X and Y. Then

$$(S_{j+1}, T_{j+1}, \Delta_{j+1}, \delta_{j+1}) = \texttt{regularize}(X_j, Y_j).$$

We will derive inductive bounds on $|T_{j+1}|$, Δ_{j+1}, and δ_{j+1}. From (7) and Lemma 8 with $(S,T) = (X_j, Y_j)$, we get

$$\Delta_{j+1} \ge \operatorname{avg} d_{Y_j}(X_j)/(30 \lg n) \ge \Delta_j/(30 \lg n) \ge \Delta_1/(30 \lg n)^j = \widetilde{\Omega}(\Delta). \tag{13}$$

From (10) we also have

$$\begin{aligned} |T_{j+1}| \;\le\; |Y_j| \;&\le\; |T_j|\,(80n^2)/(\Delta_j^2 k^4) = \widetilde{O}(|T_j|\, n^2/(\Delta^2 k^4)) \\ &=\; \widetilde{O}\left(|T_1| \left(\frac{n^2}{\Delta^2 k^4}\right)^j\right) = \widetilde{O}\left((n/k)\left(\frac{n^2}{\Delta^2 k^4}\right)^j\right). \end{aligned}$$

From (9), we know that any Y considered, including Y_j, is of size at least $\Delta_j^2/(2\Psi) = \widetilde{\Omega}(\Delta^2/\Psi) = \widetilde{\Omega}(\Delta^2 k^2/n)$, so we must have

$$\Delta^2 k^2/n = \widetilde{O}\left((n/k)\left(\frac{n^2}{\Delta^2 k^4}\right)^j\right) \iff k = \widetilde{O}\left((n/\Delta)^{\frac{2j+2}{4j+3}}\right).$$

By (11) this implies that $j < c$.

Next we need to argue that preconditions (5) and (6) are satisfied for round $j+1$. By (13), we get that (5) follows from Lemma 11.

The critical issue is to make sure that pre-condition (6) is satisfied with $\delta_{j+1} \ge 4\Delta_j/\Psi$. Using (7)–(10), we get that

$$\begin{aligned} \operatorname{avg} d_{X_j}(Y_j) \;&\ge\; \Delta_j |X_j|/|Y_j| = \Delta_j(\delta_j/4)/\widetilde{O}\left((n/k)\left(\frac{n^2}{\Delta^2 k^4}\right)^j\right) \\ &=\; \widetilde{\Omega}\Big(\Delta_j \delta_j (k/n)\left(\Delta^2 k^4/n^2\right)^j\Big). \end{aligned}$$

By Lemma 8, $\delta_{j+1} \ge \operatorname{avg} d_{X_j}(Y_j)/8$, so $\delta_{j+1} = \delta_j\, \widetilde{\Omega}\Big(\Delta_j (k/n)\left(\Delta^2 k^4/n^2\right)^j\Big)$. Inductively, since $j = O(1)$ and $\Delta_h = \widetilde{\Omega}(\Delta)$ for all $h \le j$, it follows that

$$\delta_j = \widetilde{\Omega}\Big(\delta_1 (\Delta k/n)^{j-1}\left(\Delta^2 k^4/n^2\right)^{j(j-1)/2}\Big).$$

Therefore $\delta_{j+1} = \widetilde{\Omega}\Big(\delta_1 (\Delta_j k/n)(\Delta k/n)^{j-1}\left(\Delta^2 k^4/n^2\right)^{j(j+1)/2}\Big)$. Here $5\delta_1 \ge |S_1|\Delta_1/|T_1| = \widetilde{\Omega}(\Delta^2 k/n)$, so we get

$$\delta_{j+1} = \widetilde{\Omega}\Big(\Delta_j (\Delta k/n)^{j+1}\left(\Delta k^2/n\right)^{j(j+1)}\Big)$$

By (12) we have $(\Delta/k)(\Delta k/n)^j(\Delta k^2/n)^{j(j+1)} = \widetilde{\omega}(1)$, so

$$(\Delta k/n)^{j+1}\left(\Delta k^2/n\right)^{j(j+1)} = \widetilde{\omega}(k^2/n) = \widetilde{\omega}(1/\Psi).$$

Thus $\delta_{j+1} = \widetilde{\omega}(\Delta_j/\Psi) > 4\Delta_j/\Psi$, so pre-condition (6) is indeed satisfied for round $j+1$. This completes our proof of Theorem 10. Our main coloring result follows.

▶ **Theorem 12.** *In polynomial time, we can color any 3-colorable n vertex graph using $\widetilde{O}(n^{0.19996})$ colors.*

Proof. We use Chlamtac's SDP [6] for low degrees, so as stated in (4), for progress towards an $k = \widetilde{O}(n^{0.19996})$ coloring, we may assume the minimum degree is at least $\Delta = n^{0.61674333}$. Then $(n/\Delta)^{14/27} < k < (n/\Delta)^{12/23}$, so to satisfy (11) in Theorem 10, we set $c = 6$. It is easily verified that (12) is satisfied for $j = 1, ..., 5$. By Theorem 10, we conclude that progress is made within the first $c = 6$ rounds.

Incidentally, with our particular values of k and Δ, for an integer j, (12) reaches its minimum with $j = 5$. This implies that our bounds also hold with any larger c. ◀

References

1. S. Arora, E. Chlamtac, and M. Charikar. New approximation guarantee for chromatic number. In *Proc. 38th STOC*, pages 215–224, 2006.
2. S. Arora, S. Rao, and U. Vazirani. Expanders, geometric embeddings and graph partitioning. *J. ACM*, 56(2):1–37, 2009. Announced at STOC'04.
3. B. Berger and J. Rompel. A better performance guarantee for approximate graph coloring. *Algorithmica*, 5(3):459–466, 1990.
4. A. Blum. New approximation algorithms for graph coloring. *J. ACM*, 41(3):470–516, 1994. Announced at STOC'89 and FOCS'90.
5. A. Blum and D.R. Karger. An $\tilde{O}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Inf. Process. Lett.*, 61(1):49–53, 1997.
6. E. Chlamtac. Approximation algorithms using hierarchies of semidefinite programming relaxations. In *Proc. 48th FOCS*, pages 691–701, 2007.
7. I. Dinur, E. Mossel, and O. Regev. Conditional hardness for approximate coloring. *SIAM J. Comput.*, 39(3):843–873, 2009. Announced at STOC'06.
8. U. Feige, M. Langberg, and G. Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. *SIAM J. Comput.*, 33(6):1338–1368, 2004. Announced at FOCS'02.
9. M.R. Garey, D.S. Johnson, and L.J. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. Announced at STOC'74.
10. M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. Announced at STOC'94.
11. V. Guruswami and S. Khanna. On the hardness of 4-coloring a 3-colorable graph. *SIAM Journal on Discrete Mathematics*, 18(1):30–40, 2004.
12. D.R. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998. Announced at FOCS'94.
13. R. Karp. On the computational complexity of combinatorial problems. *Networks*, 5:45–68, 1975.
14. K. Kawarabayashi and M. Thorup. Combinatorial coloring of 3-colorable graphs. In *Proc. 53rd FOCS*, pages 68–75, 2012.
15. S. Khanna, N. Linial, and S. Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000.
16. M. Szegedy. A note on the θ number of Lovász and the generalized Delsarte bound. In *Proc. 35th FOCS*, pages 36–39, 1994.
17. A. Wigderson. Improving the performance guarantee for approximate graph coloring. *J. ACM*, 30(4):729–735, 1983. Announced at STOC'82.