Coloring 3-Colorable Graphs

Charles Jin

April 3, 2015

1 Introduction

Graph coloring in general is an extremely easy-to-understand yet powerful tool. It has wide-ranging applications from register allocation to image segmentation. For such a simple problem, however, the question is surprisingly intractable. In this section I will introduce the problem formally, as well as present some general background on graph coloring.

There are several ways to color a graph, in particular, one can color the vertices, faces, or the edges. These problems turn out to be equivalent - for example, coloring the faces of a graph is the same as coloring the vertices of the dual, while coloring the edges is the same as coloring the vertices of the line graph. Today we consider the problem of vertex coloring. For the sake of formality, here are a few definitions.

Definition A **coloring** for a graph $G = (V, E)$ is a set of colors C along with a function f mapping the vertex set V into C.

Definition A coloring is **legal** if $\forall i, j \in V, (i, j) \in E \Rightarrow f(i) \neq f(j)$. In other words, one cannot color adjacent vertices the same color.

Definition A graph is ***n*-colorable** if there exists a legal coloring on n colors.

Definition The **chromatic number** of a graph is the minimum cardinality over all sets of colors that admit a legal coloring.

Already, we have the following theorem.

Theorem 1.1. *Determining the chromatic number of a graph is NP-complete.*

It turns out the situation is even more dire.

Theorem 1.2. *Let n be the chromatic number of a graph. Then approximating n to within $n^{1-\epsilon}$ for $\epsilon > 0$ is NP-hard.*

With this in mind, we turn to a slightly easier question: assuming we know that a graph is 3-colorable, what's the best we can do?

2 A "Trivial" Solution

Though the results derived in this section may seem a bit weak at first, they turn out to be surprisingly useful in our analysis later. As an aside, we will implicitly assume connected graphs for the rest of this lecture (the results generalize with almost no work.)

2.1 A few propositions

Before we formulate our first solution to the problem itself, it is useful to establish a few facts.

Proposition 2.1. *If a graph is 2-colorable, then we can find a 2-coloring in linear time.*

Proof Because the graph is 2-colorable, we start with any vertex and give it a color. Color all neighbors a second color, and repeat via depth-first search. In general, pick any vertex and color all nodes with odd distance one color, and all nodes with even distance a second color. □

Proposition 2.2. *If a graph has maximum degree Δ, then we can find a $\Delta + 1$ coloring in polynomial time.*

Proof At every step, select any uncolored node, and color it using a color that its neighbors haven't used. Because the node has at most Δ neighbors, and we have $\Delta + 1$ colors, this is sufficient. □

We are now ready to consider our first solution.

2.2 An $O(\sqrt{n})$ algorithm

Theorem 2.3. *Let G be a 3-colorable graph on n vertices. Then there exists a poly-time algorithm that produces an $O(\sqrt{n})$ coloring.*

Proof We make use of Proposition 2.2. Let Δ be the current maximum degree. If $\Delta \leq \sqrt{n}$, then color the graph with the $\Delta + 1$ algorithm. Otherwise, pick a vertex with maximum degree, color it any color, and remove it. The neighbors of the removed vertex can now be 2-colored, since the original graph was 3-colorable. Then by Proposition 2.1, we can color the neighbors with 2 colors in poly-time. Remove the neighbors, and repeat. This loop happens at most $\frac{n}{\sqrt{n}} = \sqrt{n}$ times, since we are removing at least $\sqrt{n}$ vertices at each step. Thus the loop takes at most $3\sqrt{n}$ colors, and the final step also takes $O(\sqrt{n})$ colors. □

3 A Naive SDP Solution

Associate with each vertex i a vector v_i. Then consider the following SDP:

$$\begin{array}{lll} min & \lambda & \\ s.t. & v_i \cdot v_j \leq \lambda & \forall (i,j) \in E \\ & v_i \cdot v_i = 1 & \forall i \\ & v_i \in R^n & \forall i \end{array} \tag{1}$$

The idea here is that we are embedding the vertices in some n-dimensional space, and the SDP is putting vertices that are connected as far away from each other as possible. Therefore, we should be able to pick some number of partitions and color each partition a different color, and show that with high probability, our partition produces a decent coloring. We start by attempting to bound just how well we can expect the embedding to spread out the points.

Lemma 3.1. *If G is 3-colorable, then there exists a feasible solution with $\lambda \leq -\frac{1}{2}$.*

Proof Assume we have some 3-coloring of G. Consider an equilateral triangle centered at the origin, and associate with each color a unit vector that points to one of the vertices of the triangle. Since $v_i \cdot v_j = ||v_i||\,||v_j|| \cos\theta = \cos\theta$, where θ is the angle between v_i and v_j, for $v_i = v_j$ we get exactly 1, and for $v_i \neq v_j$, we get

$$v_i \cdot v_j = \cos\left(\frac{2\pi}{3}\right) = -\frac{1}{2}$$

as desired. □

Thus the lemma shows that every pair of vertices sharing an edge form an angle that is at least $\frac{2\pi}{3}$. It turns out that this is not enough. If we wanted to cover an n-dimensional space with cones of angle $\frac{2\pi}{3}$, because the surface area of an n-sphere grows exponentially with the dimension, this would give an exponential number of colors. Thus, we instead randomly partition the space using some number of hyperplanes, show that on expectations this does not color too many edges incorrectly, and repeat the process to color the illegally-colored vertices.

Before we prove this formally, we will need a new concept.

Definition We call a coloring a **semi-coloring** if at most $\frac{n}{4}$ edges have endpoints that share a color.

Lemma 3.2. *In a semi-coloring, there exists some set of vertices of size at least half the total number of vertices such that the coloring induced on the induced subgraph is legal.*

Proof Remove all vertices that are adjacent and share the same color. Since there are at most $\frac{n}{4}$ such edges, this removes at most $\frac{n}{2}$ vertices. The remaining vertices comprise a legally-colored subgraph. □

Lemma 3.3. *Given an algorithm that produces a semi coloring on G with k colors, then we can color the entire graph legally with only $k \log n$ colors.*

Proof We first semicolor the graph with k colors, and remove the correctly colored vertices. This leaves at most $\frac{n}{2}$ vertices, so we just repeat this process. This terminates in at most $\log n$ steps, meaning the graph is colored with at most $k \log n$ colors. □

Theorem 3.4. *There exists a randomized rounding algorithm that produces a semi-coloring on $\Delta^{\log_3 2}$ colors with probability at least $\frac{1}{2}$.*

I'm going to omit a proof of this theorem because the analysis isn't particularly enlightening. It's extremely similar to the MAXCUT algorithm we saw before: you essentially pick $t = 2 + \Delta^{\log_3 2}$ random hyperplanes and color each of the 2^t regions a different color.

Notice that this still only gives an $O(n^{\log_3 2} \log n)$ solution, which is far worse than the $O(\sqrt{n})$ naive algorithm. However, we can reapply the ideas behind our optimization of the $O(\sqrt{n})$ to improve the bound.

We pick some parameter σ and proceed by removing a vertex of maximum degree and coloring its neighbors with 2 colors as before. When there are no remaining vertices of degree greater than σ, we use our SDP to semicolon the remaining vertices with $O(\sigma^{\log_3 2})$ colors. Optimizing over σ, we find (using basically the same analysis as before) that we can take $\sigma = n^{\log_6 3}$ to give a semi-coloring on $O(n^{0.387})$ colors with probability at least $\frac{1}{2}$. This yields a coloring algorithm using $O(n^{0.387} \log n)$ colors.

Next, we will refine our SDP and rounding algorithm to produce an even better algorithm.

4 A Refined SDP Solution

In order to motivate this solution, we need the concept of an independent set:

Definition Let $G = (V, E)$. We call $S \subset V$ **independent** if $i, j \in S \Rightarrow (i, j) \notin E$.

It is obvious that if $G = (V, E)$ is n-colorable, then V is the disjoint union of at most n independent subsets of V (just take each subset to be the vertices of a certain color). If we reverse this process, we can color a graph by repeatedly finding a large independent set, then coloring and removing the independent set.

In fact, it should be at least believable why this process performs at least as good as randomly picking partitions, because each step depends on previous steps. Therefore, we should be able to look for one large independent set, remove it, then rerun the algorithm

to find the second largest independent set, etc. There was an implicit assumption in the previous section that we really wanted to color the surface of the entire hypersphere. So instead of expecting all the vertices to be distributed across the entire hypersphere, we'd like to exploit clusters of vertices. The following lemma formalizes that such a process actually gives a good coloring.

Lemma 4.1. *Given an algorithm that finds an independent set of expected size γn, then with high probability we can find an $O(\frac{1}{\gamma}\ln n)$ coloring.*

Proof Assume for the time being that the algorithm deterministically returns an independent set of size γn. We color this independent set some color, remove the set, and repeat. It should be obvious that this produces a legal coloring. Then after k iterations, at most $(1-\gamma)^k n$ vertices remain. We can use the identity $1 - x \leq e^{-x}$ to get $(1-\gamma)^k \leq e^{-\gamma k}$, so letting $k = \frac{1}{\gamma}\ln n$, this yields

$$(1-\gamma)^k n \leq e^{-\gamma k} n = e^{-\ln n} n = 1$$

vertex left after k iterations, which we can just assign its own color.

Now we use Markov's inequality to provide a similar result for a randomized algorithm. Let X be a random variable corresponding to the number of vertices not in the selected independent set. If the expected size of the independent set is at least γn, then it follows that $\mathbb{E}[X] \leq n(1-\gamma)$. By Markov's inequality, then

$$\Pr\left[X \geq n\left(1 - \frac{\gamma}{2}\right)\right] \leq \frac{\mathbb{E}[X]}{n\left(1-\frac{\gamma}{2}\right)} \leq \frac{n(1-\gamma)}{n\left(1-\frac{\gamma}{2}\right)} \leq 1 - \frac{\gamma}{2}$$

So with probability at least $\frac{\gamma}{2}$, we get an independent set of size $\frac{\gamma n}{2}$, and for a given constant c, running the algorithm $t = \frac{2c}{\gamma}\ln n$ times yields an independent set of size at least $\frac{\gamma n}{2}$ with probability at least

$$1 - \left(1 - \frac{\gamma}{2}\right)^{\frac{2c}{\gamma}\ln n} \geq 1 - e^{-c\ln n} \geq 1 - \frac{1}{n^c}$$

□

Now we are ready to formally state our new SDP. As with before, associate with each vertex i a vector v_i, and consider the following SDP:

$$\begin{array}{lll} min & 0 & \\ s.t. & v_i \cdot v_j \leq -1/2 & \forall (i,j) \in E \\ & v_i \cdot v_i = 1 & \forall i \\ & v_i \in R^n & \forall i \end{array} \tag{2}$$

Notice that our objective function is simply a constant. This is because we only care about a feasible solution, and by Lemma 3.1 such a solution always exists for 3-colorable graphs.

Recall from the discussion before that our main intuition in seeking this algorithm is that we don't want or need to color the entire surface of the hypersphere. So instead of taking a partition, we just pick some large slice of the space that is somewhat concentrated, and hope that our slice doesn't contain too many edges.

Consider the following coloring algorithm, based on the feasible solution to the SDP. First generate a random vector $r = (r_1, r_2, ..., r_n)$ by sampling each component independently from $\mathcal{N}(0, 1)$, the normal distribution. Instead of producing a cut as we did before in MAXCUT by partitioning the space with a hyperplane, instead we are going to consider the set of vertices that are at least a distance ϵ from the given vector.

$$S(\epsilon) = \{i \in V \mid v_i \cdot r \geq \epsilon\}$$

Since this does not necessarily give us an independent set, we just throw away internal edges in the crudest way possible.

$$S'(\epsilon) = \{i \in S(\epsilon) \mid \forall j \in S(\epsilon), (i, j) \notin E\}$$

This might seem overly confident, however, realize that if we had a better way of removing edges, we could just construct an independent set that way. The purpose of the algorithm is to reduce the number of internal edges so that we can throw them out without affecting our independent set too much.

First, some notation:

Fact 4.2. *The probability density function of* $\mathcal{N}(0, 1)$ *is* $p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$.

Fact 4.3. *The cumulative density function of* $\mathcal{N}(0, 1)$ *is* $\Phi(x) = \int_{-\infty}^{x} p(s)ds$. *We can further define* $\bar{\Phi}(x) = 1 - \Phi(x) = \int_{x}^{\infty} p(s)ds$.

Now we will begin to analyze the rounding algorithm given. The purpose of this analysis is to pick a good value for ϵ by finding bound on the size of S' in terms of ϵ. We expect that if we can pick ϵ just right, then we can still capture a large portion of the vertices without running into too many bad edges. Then Lemma 4.1 completes the proof.

Lemma 4.4. *If* $\bar{\Phi}(\sqrt{3}\epsilon) \leq \frac{1}{2\Delta}$, *then* $\mathbb{E}[|S'|] \geq \frac{n}{2}\bar{\Phi}(\epsilon)$.

This isn't exactly what we need, since our epsilon is still wrapped in some function of $\bar{\Phi}(x)$, so we need the following bounds.

Lemma 4.5. *For* $x > 0$,

$$\frac{x}{1 + x^2} p(x) \leq \bar{\Phi}(x) \leq \frac{1}{x} p(x)$$

We can now stitch our results together to give the final theorem:

Theorem 4.6. *There exists a polynomial-time algorithm that produces an $\tilde{O}(\Delta^{1/3}\sqrt{\ln \Delta})$ coloring algorithm for 3-colorable graphs.*

Proof Recall the SDP for finding an independent set given in (2). I claim the rounding algorithm above produces an independent set of expected size $\Omega(n\Delta^{-1/3}(\ln \Delta)^{-1/2})$. Then by Lemma 4.1 this yields an $O(\Delta^{1/3}(\ln \Delta)^{1/2} \ln n)$ coloring.

Then set $\epsilon = \sqrt{\frac{2}{3} \ln \Delta}$. By Lemma 4.5,

$$\begin{aligned}
\bar{\Phi}\sqrt{3}\epsilon &\leq \frac{1}{\sqrt{3}\epsilon} \frac{1}{\sqrt{2\pi}} e^{-3\epsilon^2/2} \\
&= \frac{1}{\sqrt{2 \ln \Delta}} \frac{1}{\sqrt{2\pi}} e^{-\ln \Delta} \\
&= \frac{1}{2\Delta} \frac{1}{\sqrt{\pi \ln \Delta}} \\
&\leq \frac{1}{2\Delta}
\end{aligned}$$

By Lemma 4.4, this is enough to guarantee that

$$\mathbb{E}[|S'(\epsilon)|] \geq \frac{n}{2} \bar{\Phi}(\epsilon)$$

Furthermore, by Lemma 4.5, if we can guarantee $\epsilon \geq 1$, then

$$\begin{aligned}
\bar{\Phi}(\epsilon) &\geq \frac{\epsilon}{1+\epsilon^2} \frac{1}{\sqrt{2\pi}} e^{-\epsilon^2/2} \\
&\geq \frac{1}{2\epsilon} \frac{1}{\sqrt{2\pi}} e^{-\epsilon^2/2} \\
&= \frac{1}{2\sqrt{\frac{2}{3} \ln \Delta}} \frac{1}{\sqrt{2\pi}} e^{-\ln \Delta/3} \\
&= \Omega(\Delta^{-1/3}(\ln \Delta)^{-1/2}).
\end{aligned}$$

Combining these results, we get that the expected size of the independent set is at least

$$\Omega(n\Delta^{-1/3}(\ln \Delta)^{-1/2})$$

as claimed.

All that remains is to guarantee that $\epsilon \geq 1$ during execution. To accomplish this, we use a familiar technique: if ϵ ever falls below 1, then we color the remaining graph with the greedy algorithm from Proposition 2.2 with $\Delta + 1 \leq e^{3/2} + 1 \leq 6$ colors. This adds a constant number of colors, which does not affect the asymptotic behavior. □

Finally, we return to proving our lemmas.

To prove Lemma 4.4, we're going to use 3 steps. First, bound the size of S, then find the probability that $v_i \in S$ gets thrown out, and finally use this to find a bound on the size of S'. Since S is defined in terms of $v_i \cdot r$, we first consider how $v_i \cdot r$ is distributed.

Proposition 4.7. *$v_i \cdot r$ is normally distributed.*

Proof We've seen this before, but in case you forgot, the weighted sum of gaussians is again a gaussian, and we are free to rotate v_i to be a unit vector along one component without changing the distribution, so that in particular $v_i \cdot r \sim \mathcal{N}(0, 1)$. □

Lemma 4.8. $\Pr[i \in S(\epsilon)] = \bar{\Phi}(\epsilon)$, $\forall i \in V$.

Proof We know that $i \in S(\epsilon) \iff v_i \cdot r \geq \epsilon$. By Proposition 4.7, $v_i \cdot r$ is normally distributed so by definition $\Pr[v_i \cdot r \geq \epsilon] = \bar{\Phi}(\epsilon)$. □

Lemma 4.9. $\Pr[i \in S(\epsilon) \text{ and } i \notin S'(\epsilon)] \leq \Delta\bar{\Phi}(\sqrt{3}\epsilon)$.

Proof The only way this can be the case is if there is another $j \in S(\epsilon)$ such that $(i, j) \in E$. Thus we seek to bound

$$\Pr[j \in S \mid (i, j) \in E \text{ and } i \in S] \tag{3}$$

We know that $v_i \cdot v_j = -1/2$. Now the plane spanned by v_i and v_j has dimension 2, which means that any two orthogonal vectors form a basis. Therefore, there must be some unit vector u orthogonal to v_i lying in this plane such that $v_j = av_i + bu$, for some choice of constants a and b. However, we also know that $v_j \cdot v_i = v_i \cdot (av_i) + v_i \cdot (bu) = a$ since v_i and u are orthogonal, which implies that $a = -\frac{1}{2}$. Furthermore, $||v_j||^2 = ||av_i + bu||^2 = \frac{1}{4} + b^2 = 1$ so that $b = \frac{\sqrt{3}}{2}$.

We can thus rewrite this as $u = \frac{2}{\sqrt{3}}\left(\frac{1}{2}v_i + v_j\right)$ which yields

$$u \cdot r = \frac{2}{\sqrt{3}}\left(\frac{1}{2}v_i + v_j\right) \cdot r \geq \frac{2}{\sqrt{3}}\left(\frac{1}{2}\epsilon + \epsilon\right) = \sqrt{3}\epsilon$$

Again, u is a unit vector that is the sum of gaussians, so it is normally distributed. Furthermore, it is distributed as an independent gaussian, since v_i and u are orthogonal. This means that the event happens for a particular neighbor j with probability at most $\bar{\Phi}(\sqrt{3}\epsilon)$. Since i has at most Δ neighbors, by the union bound, we get the probability that i is in S but not S' is at most $\Delta\bar{\Phi}(\sqrt{3}\epsilon)$, as desired. □

Proof (of Lemma 4.4) Since $\Phi(\sqrt{3}\epsilon) \leq \frac{1}{2\Delta}$, then the probability that $i \notin S'$ given that $i \in S$ is at most $\frac{1}{2}$ by Lemma 4.9. By Lemma 4.8 and linearity of expectation, this means that we can expect $|S'| \geq \frac{|S|}{2} \geq \frac{n}{2}\bar{\Phi}(\epsilon)$. □

Finally, we prove our bounds on $\bar{\Phi}(x)$.

Proof (of Lemma 4.5) This proof isn't particularly enlightening. We have that $p'(x) = -xp(x)$, so that $\left(-\frac{1}{x}p(x)\right)' = \left(1 + \frac{1}{x^2}\right)p(x)$. To get the lower bound,

$$\begin{aligned}\left(1 + \frac{1}{x^2}\right)\bar{\Phi}(x) &= \int_x^\infty \left(1 + \frac{1}{x^2}\right)p(s)ds \\ &\geq \int_x^\infty \left(1 + \frac{1}{s^2}\right)p(s)ds \\ &= -\frac{1}{s}p(s)\Big|_x^\infty = \frac{1}{x}p(x)\end{aligned}$$

For the upper bound,

$$\begin{aligned}\bar{\Phi}(x) &= \int_x^\infty p(s)ds \\ &\leq \int_x^\infty \left(1 + \frac{1}{s^2}\right)p(s)ds \\ &= \frac{1}{x}p(x)\end{aligned}$$

□

5 Conclusion

Just to give you an idea of how good this algorithm actually is, using an improved rounding scheme apparently gives a slightly better asymptotic bound, while a more clever SDP relaxation gives an even better algorithm still. However, $O(n^c)$ is still the best bound we know of, and improving upon that is still an open question.

References

[1] D. P. Williamson, and D. B. Shmoys. *The Design of Approximation Algorithms*. Electronic copy, to be published by Cambridge University Press. URL: http://www.designofapproxalgs.com/book.pdf. Date: 2010.