Coloring 3-Colorable Graphs using SDPs[1]

- In this lecture we look at a graph coloring problem. The input is an undirected graph $G = (V, E)$ and the goal is to "color" every vertex from a "palette" of as few colors so that any two neighboring vertices obtain different colors. More precisely, we want to find the smallest c and a map $\chi : V \to \{1, 2, \ldots, c\}$ such that for any $(u, v) \in E$, we have $\chi(u) \neq \chi(v)$. This smallest c is called the *chromatic number* of the graph G.

  Coloring is inherently related to the independent set question in a graph. Indeed, observe that if χ is a valid coloring then all the vertices obtaining the same color must be an independent set. Therefore, the graph coloring problem is simply the question of "packing" the graph into as few independent sets as possible. In particular, if $\chi(G)$ of a graph is small, then the independent sets must be large. More precisely, the graph must have an independent set of size $\geq \frac{n}{\chi(G)}$ where n is the number of vertices.

- Finding a large independent set, and therefore the coloring problem, is a notoriously hard problem to approximate. One therefore looks at special cases where one can design non-trivial algorithms. One "simple" class of graphs where the independent set problem can be solved *exactly* are bi-partite graphs. Now note that a bipartite graph is *precisely* graphs which have $\chi(G) = 2$. Therefore, coloring a graph G with $\chi(G) = 2$ with two colors is easy, and so is finding the maximum independent set in such graphs.

  In this lecture we look at the "next hardest case". Suppose someone promises that $\chi(G) = 3$, can we color it in 3 colors? In other words, given the promise that graph G is tripartite, can we find the tri-partition? Turns out this problem is NP-hard. So, we ask what is the *fewest* number of colors one can color such a promised graph in? Or, and this is the question we will focus on, what's the largest independent set we can find in such graphs.

  Exercise: *Find an independent set of size $\Omega(\sqrt{n})$ in a 3-colorable graph, and then use it to color a 3-colorable graph with $O(\sqrt{n})$ colors. To make it precise, given an arbitrary graph G, either find an independent set of size $\Omega(\sqrt{n})$ or prove that the graph is **not** 3-colorable.*

- We further assume we have an upper bound on the maximum degree of G is d and our answer will be in terms of this parameter. One can always think of $d = n$. Note that a simple greedy algorithm returns an independent set of size $\Omega(n/d)$. More relevant to us, and something that we saw in the randomized rounding lectures, one can obtain such sized independent set using randomized rounding + alteration method. It is good to remind oneself of that as we will use a similar method. However, our randomized process will be guided by an SDP. To kill the suspense, we will be able to return an independent set of size $\tilde{\Omega}(n/d^{1/3})$ where the $\tilde{}$ is hiding logarithmic factors. So, we would be able to obtain an independent set of size roughly $n^{2/3}$ instead of $\sqrt{n}$ which was asked for in the above exercise. In turn, we would be able to color the 3-colorable graph using $\tilde{O}(d^{1/3})$ colors. And if you have done the above exercise, then armed with the above result you would also be able to color 3-colorable graphs using $\tilde{O}(n^{1/4})$ colors.

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 15th Mar, 2022*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

- ***An SDP for certifying 3-colorable graphs.*** We start by writing an SDP which would return a feasible solution if G is 3-colorable. Contrapositively, if the SDP doesn't return a feasible solution, we can assert that the graph is not 3-colorable. Then, we use the SDP solution to find a large independent set.

  The main observation is this. If G is 3-colorable, then there are three independent subsets A, B, C of vertices which partition V. We can therefore embed the vertices on to a unit circle on the 2-dimensional plane via $\phi : V \to \mathbb{R}^2$ such that for every edge $(i, j) \in E$ we have that the angle between $\phi(i)$ and $\phi(j)$ is precisely $120°$. See Figure 1 for an illustration.

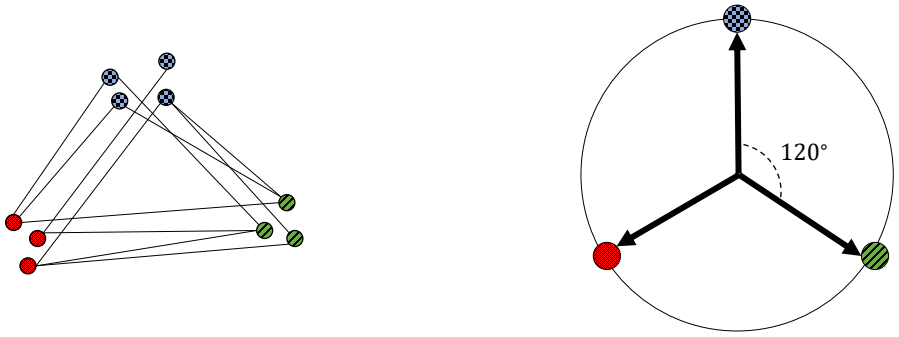

Figure 1: *The graph can be 3-colored using red (trellis), blue (checkered), and green (diagonal). The embedding on to the unit circle is shown on the right.*

  In other words, if G is 3-colorable, then $\exists \mathbf{u}_1, \ldots, \mathbf{u}_n \in \mathbb{R}^2$ with $\|\mathbf{u}_i\|_2 = 1$ and $\mathbf{u}_i^\top \mathbf{u}_j = -\frac{1}{2}$, $\forall (i, j) \in E$. Noting that the $n \times n$ matrix $\mathbf{X}_{ij} = \mathbf{u}_i^\top \mathbf{u}_j$ is a PSD matrix[2], we get that if G is 3-colorable,

$$\{\mathbf{X} \in \mathbb{R}^{n \times n} \ : \ \mathbf{X}_{ii} = 1, \quad \mathbf{X}_{ij} = -\frac{1}{2}, \ \forall (i, j) \in E, \ \ \mathbf{X} \succcurlyeq 0\} \quad \text{is non-empty} \qquad \text{(3Col SDP)}$$

  This can be checked using an SDP. In other words, we can write an SDP to check if the RHS is empty or not, and if it is empty, we can assert G is *not* 3-colorable. Next, we show what to do with the feasible $X \succcurlyeq 0$ that is returned by the SDP.

- ***SDP rounding.*** Given the PSD matrix $\mathbf{X}$ we can obtain n unit vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{R}^d$ such that $\mathbf{v}_i^\top \mathbf{v}_j = -0.5$ for all edges $(i, j) \in E$. If d were 2, then we would be done as that would imply all the vectors are clumped at the three endpoints of an equilateral triangle, and all the coincident points must be independent. However, d could be as large as n, and then it is not immediately clear how to obtain a large independent set.

  We first try to describe the idea qualitatively. As in the case of maximum cut, we have the vertices spread on the surface of a high-dimensional orange. This time, we know that endpoints of any edge are indeed "quite far" from each other. In particular, if we focus on the plane containing the center and the endpoints of a single edge, then they form the 120 degree angle between them. Now suppose we slice this orange using a random hyperplane and pick the vertices S on one of the sides uniformly at random. The probability a vertex is picked in S is then $1/2$. Furthermore, for any fixed edge, the probability of picking both endpoints in S, a la the Goemans-Williamson analysis, is $1 - \frac{120}{180} = 1/3$. The issue, however, is that there could be many edges (d many) incident to a vertex i, and so we could

[2] $\mathbf{X} = UU^\top$ where U is the $n \times 2$ matrix formed by taking $\mathbf{u}_i^\top$ as the ith row; and therefore for any $\mathbf{v} \in \mathbb{R}^n$, we have $\mathbf{v}^\top \mathbf{X} \mathbf{v} = \sum_{r=1}^{2} (\mathbf{v}^\top U_r)^2$ where U_r is the rth column of U. And so, $\mathbf{v}^\top \mathbf{X} \mathbf{v} \geq 0$ implying $\mathbf{X} \succcurlyeq 0$.

pick $\approx d/3$ many neighbors of i in S as well. If we then tried "fixing" by deleting a vertex per edge sampled, we may end up deleting almost the whole set S.

The fix is not to slice the orange through the center but through a "lesser circle". Rethinking the orange as our earth, instead of slicing through the equator, slice through the tropic of cancer (or, perhaps more closer to the truth, at much higher latitudes), and then pick the smaller region. Now, the probability of picking a vertex will drop; larger the latitude, smaller is this probability. However, and we show this precisely, since endpoints of an edge are "far away", the chances of picking both end points of an edge becomes *much* smaller. And indeed, the correct latitude to slice at is determined by the parameter d, and is figured so as to balance out the alteration step and the sampling step. We now give the formal description.

procedure KMS ROUNDING(G, $\mathbf{X}$ solution to (3Col SDP)):
  Obtain vectors $\mathbf{v}_i \in \mathbb{R}^d$ such that $\mathbf{X}_{ij} = \mathbf{v}_i^\top \mathbf{v}_j$.
  Sample a random Gaussian vector $\mathbf{g} \in \mathbb{R}^d$.
  ▷ *Sample $g_i \in \mathcal{N}(0,1)$ for $1 \le i \le d$ and $\mathbf{g} = (g_1, \ldots, g_d)$*
  Let $I_1 := \{i \in V \ : \ \mathbf{v}_i^\top \mathbf{g} \ge c\}$. ▷ *Parameter c is the "latitude" and is set to $\sqrt{\frac{2\ln(2d)}{3}}$.*
  For any $(i,j) \in E$ if both i and j are in I_1, add them to D.
  return $I \leftarrow I_1 \setminus D$.

- ***Analysis.*** It should be clear that I is an independent set by design. We prove the following theorem.

Theorem 1. If G has maximum degree d and $c = \sqrt{\frac{2}{3}\ln(2d)}$, then the expected size of I returned by KMS ROUNDING is $\Omega\left(\frac{n}{d^{1/3}\sqrt{\ln d}}\right)$.

Before we dive into this, we state some facts about Gaussian random variables.

Fact 1 (Gaussian Facts)**.**

a. If $\mathbf{g}$ is a random Gaussian vector in $\mathbb{R}^d$ and $\mathbf{v}$ is another vector in $\mathbb{R}^d$, then $G := \mathbf{v}^\top \mathbf{g}$ satisfies $G \sim \mathcal{N}(0, \|\mathbf{v}\|_2)$, that is, it is a standard Gaussian whose standard deviation is the Euclidean length of $\mathbf{v}$.

b. Let $G \sim N(0,1)$ and define, for $z \ge 0$, $\mathsf{erf}(z) := \mathbf{Pr}[G \ge z] = \frac{1}{\sqrt{2\pi}} \int_z^\infty e^{-\frac{t^2}{2}}\, dt$. This is just 1 minus the CDF of the Gaussian. Then, it can be shown for all $t > 0$,

$$\left(\frac{1}{t} - \frac{1}{t^3}\right) e^{-t^2/2} \le \sqrt{2\pi} \cdot \mathsf{erf}(t) \le \frac{1}{t} \cdot e^{-t^2/2} \tag{1}$$

If $G \sim N(0, \sigma)$, then $\mathbf{Pr}[G \ge z] = \mathsf{erf}(z/\sigma)$.

- The main observation is this: if (i,j) is an edge, then

$$\|\mathbf{v}_i + \mathbf{v}_j\|_2 = \|\mathbf{v}_i\|_2 + \|\mathbf{v}_j\|_2 + 2\mathbf{v}_i^\top \mathbf{v}_j = 1$$

That is, the sum of these two vectors are also unit vectors. In particular, they are "decently" far away. Now we have all the ingredients to prove Theorem 1.

First we lower bound the size of $|I_1|$ as follows.

$$\mathbf{Exp}[|I_1|] = \sum_{i \in V} \mathbf{Pr}[i \in I_1] = \sum_{i \in V} \mathbf{Pr}[\underbrace{\mathbf{v}_i^\top \mathbf{g}}_{\sim \mathcal{N}(0,1)} \geq c] = n \cdot \mathsf{erf}(c)$$

where we used Fact 1(a) to say that $\mathbf{v}_i^\top \mathbf{g}$ is a standard gaussian, since $\mathbf{v}_i$ is a unit vector.

Next we upper bound the size of $|D|$ as follows.

$$\begin{aligned}
\mathbf{Exp}[|D|] &= 2 \cdot \sum_{(i,j) \in E} \mathbf{Pr}[i \in I_1 \text{ and } j \in I_1] \\
&= 2 \cdot \sum_{(i,j) \in E} \mathbf{Pr}[\underbrace{\mathbf{v}_i^\top \mathbf{g} \geq c \text{ and } \mathbf{v}_j^\top \mathbf{g} \geq c}_{\Rightarrow\ (\mathbf{v}_i + \mathbf{v}_j)^\top \mathbf{g} \geq 2c}] \\
&\leq 2 \cdot \sum_{(i,j) \in E} \mathbf{Pr}[\underbrace{(\mathbf{v}_i + \mathbf{v}_j)^\top \mathbf{g}}_{\sim \mathcal{N}(0,1) \text{ since } \|\mathbf{v}_i + \mathbf{v}_j\|_2 = 1} \geq 2c] \\
&\leq nd \cdot \mathsf{erf}(2c)
\end{aligned}$$

where last inequality follows since the number of edges is $\leq nd/2$ and Fact 1(a).

Therefore,

$$\mathbf{Exp}[|I|] = \mathbf{Exp}[|I_1|] - \mathbf{Exp}[|D|] \geq n \cdot \mathsf{erf}(c) - nd \cdot \mathsf{erf}(2c) \geq n \cdot \left(\frac{1}{2c} e^{-c^2/2} - \frac{d}{2c} e^{-2c^2} \right)$$

where we have used Fact 1(b) for the last inequality.

Substituting $c = \sqrt{\frac{2}{3} \ln (2d)}$, we get

$$\mathbf{Exp}[|I|] \geq \frac{n}{4\sqrt{\frac{2}{3} \ln (2d)}} \cdot e^{-\frac{\ln(2d)}{3}} = \Omega\left(\frac{n}{d^{1/3} \sqrt{\ln d}} \right)$$

- It is worthwhile comparing the above randomized algorithm to find an independent set with an older randomized algorithm we saw in the course: sampling vertices independently and then fixing. More precisely, suppose we sampled every vertex *independently* with probability p to get the set I_1 which has cardinality np in expectation. Then, for every edge (i, j) the probability both end points are sampled is p^2, and thus I_1, in expectation, contains $\approx ndp^2$ edges. When we fix the solution (by deleting endpoints of edges in I_1), we get the resulting independent set of size $\approx np - ndp^2$. This gives us the correct sampling probability of $p = \Theta(1/d)$ and an independent set of size $\Theta(n/d)$.

  The KMS algorithm is a ***dependent*** randomized rounding algorithm. The probability of picking a vertex $i \in I_1$ is $p \approx e^{-c^2/2}$ (ignoring the $2c$ in the denominator). However, the probability that two endpoints of an edge (i, j) are *both* picked in I_1 is not p^2 but $\approx p^4 \approx e^{-2c^2}$. In particular there is a significant *negative* correlation between the events $i \in I_1$ and $j \in I_1$. And this is what really leads to the improvement over naive independent rounding.

Exercise: ☕☕ *Generalize the above algorithm for k-colorable graphs. In particular, if we know G is k-colorable, what is the SDP which would have a feasible solution? And how would you modify the KMS algorithm to round the solution of the SDP? What is the number of colors you get as a function of k?*

- To get the coloring result we use a standard peeling argument. We start by finding an independent set of size as prescribed above. We delete this independent set and all edges incident. Note the maximum degree upper bound d can only go down. We can keep doing so till all the vertices are picked in some independent set. We leave it as a simple exercise to show that the number of independent sets is $\tilde{O}(d^{1/3})$.

Notes

Coloring 3-colorable graphs is an outstanding problem in approximation algorithms. The $O(\sqrt{n})$-color algorithm is from the paper [7] by Wigderson. This was improved in this paper [1] by Blum to $\tilde{O}(n^{3/8})$ colors. The algorithm described in these notes using SDPs is from the paper [4] by Karger, Motwani, and Sudan (hence the name KMS), and when combined with Wigderson's algorithm gives an $\tilde{O}(n^{1/4})$-coloring algorithm. A chain of improvements, both combinatorial and using SDP methods followed, and the current best algorithm is one from the paper [5] by Kawarabayashi and Thorup, and it uses $\tilde{O}(n^{0.19996})$ colors.

On the hardness front, it is known that it is NP-hard to color 3-colorable graphs with 4 colors; this result was first proved in the paper [6] by Khanna, Linial, and Safra, and then proved using elementary means in the paper [3] by Guruswami and Khanna. This is the best NP-hardness known! If one assumes a variant of the Unique Games Conjecture, then the paper [2] proved that it is NP-hard to color 3-colorable graphs with $O(1)$ colors.

References

[1] A. Blum. New approximation algorithms for graph coloring. *Journal of the ACM*, 41(3):470–516, 1994.

[2] I. Dinur, E. Mossel, and O. Regev. Conditional hardness for approximate coloring. *SIAM Journal on Computing (SICOMP)*, 39(3):843–873, 2009.

[3] V. Guruswami and S. Khanna. On the hardness of 4-coloring a 3-colorable graph. *SIAM Journal on Discrete Mathematics (SIDMA)*, 18(1):30–40, 2004.

[4] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45(2):246–265, 1998.

[5] K.-i. Kawarabayashi and M. Thorup. Coloring 3-colorable graphs with less than $n^{1/5}$ colors. *Journal of the ACM*, 64(1):1–23, 2017.

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

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