CSE 5401 Lec. 21

Randomized Rounding of Semidefinite Programs II: Coloring

1. Approximating the minimum # of colors
2. An initial SDP rounding algorithm
3. Improving the algorithm = separating out high-degree vertices

Previously we saw how to use semidefinite programming to obtain approximately optimal solutions to quadratic, PSD objectives and the Max-Cut problem (which had a quadratic objective). We mentioned that similar techniques could obtain state-of-the art algorithms to many problems, where we want to maximize the number of satisfied constraints (e.g., Max 2SAT as we saw on HW). Minimizing the number of colors used in a proper coloring is not of this form. We must satisfy all of the constraints that adjacent vertices receive different colors, perhaps by using extra colors.

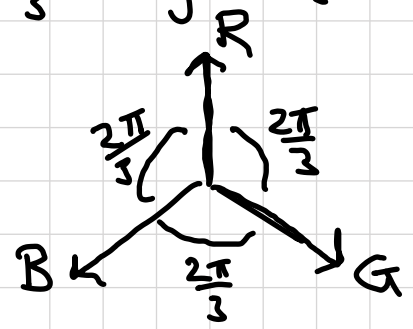
Nevertheless, an algorithm that produces a coloring that satisfies most of the edge constraints can be used to find a proper coloring that uses extra colors: Suppose that given $G=(V,E)$, we can find a coloring $\chi^{(1)}: V \to C$ such that at most $\frac{|V|}{4}$ edges $(u,v) \in E$ are not properly colored (i.e., have $\chi^{(1)}(u) = \chi^{(1)}(v)$). There are at most $\frac{|V|}{2}$ distinct vertices appearing as endpoints to these improperly colored edges. Let $U^{(1)} \subseteq V$ be the set of the endpoints of these edges and $V^{(1)} = V \setminus U^{(1)}$ be the rest. We will recursively find a coloring $\chi^{(2)}$ of the subgraph $G^{(2)}$ of G on $U^{(1)}$, such that at most $\frac{|U^{(1)}|}{4}$ edges in $G^{(2)}$ are not properly colored,

and so on; in general, $G^{(i)}$ has half as many vertices as $G^{(i-1)}$, and on the set $V^{(i)}$ (comprising at least half of the vertices of $G^{(i)}$) $\chi^{(i)}$ is a proper coloring. Thus, after $\log|V|$ iterations, no vertices remain. Now, we recover a coloring to G using $|C| \cdot \log|V|$ colors by considering each $\chi^{(i)}$ to use a distinct set of $|C|$ colors: For every edge $(u,v) \in E(G)$, either u and v receive colors in different iterations, and hence trivially obtain different colors, or else they are among the vertices properly colored by some $\chi^{(i)}$ in iteration i.

So, let's see how we can obtain such a partial coloring using a semidefinite program with randomized rounding. Consider the following semidefinite program (in vector form):

$$\underset{\vec{v}^{(1)},\ldots,\vec{v}^{(n)} \in \mathbb{R}^n,\ \lambda \in \mathbb{R}}{\text{minimize}}\ \lambda \quad \text{s.t.} \quad \langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle \le \lambda \ \ \forall (i,j) \in E, \quad \|\vec{v}^{(i)}\|_2^2 = 1 \ \ \forall i \in V$$

Intuitively, in a proper coloring of the graph using few colors, we can represent the colors with vectors that have a large angle with one another. For example, if the graph is 3-colorable, then we can use vectors (in $\mathbb{R}^2$) with $\frac{2\pi}{3}$ angles (120°) between each pair to represent the colors:



This gives the following feasible solution:

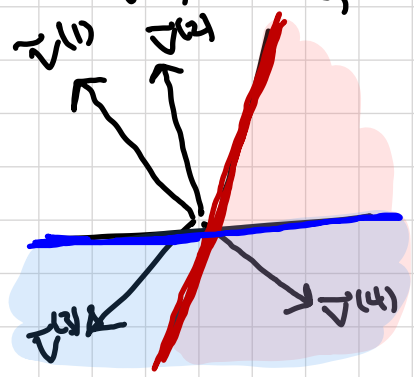
Lemma If $G = (V, E)$ is 3-colorable, then there is a feasible solution to the semidefinite program above with $\langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle = -\frac{1}{2}$ for all $(i,j) \in E$, and hence $\lambda = -\frac{1}{2}$.

Proof: Recall $\langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle = \|\vec{v}^{(i)}\|_2^2 \|\vec{v}^{(j)}\|_2^2 \cos\theta(\vec{v}^{(i)}, \vec{v}^{(j)})$. If G is 3-colorable, then assigning $\vec{v}^{(i)}$ the unit vector corresponding to its

color in same 3-coloring gives $\langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle = 1 \cdot 1 \cdot \cos\left(\frac{2\pi}{3}\right) = -\frac{1}{2}$ for all $(i,j) \in E$ as claimed ▨

Solutions to the semidefinite program generally won't assign such canonical representatives that we can read off as "colors."

What we do instead is use many random hyperplanes to split the vertices into many sets, such that the edges are very likely to cross different sets — i.e., we create many cuts, so that it is likely that any given edge crosses at least one of the cuts. Membership in the t cuts thus defines 2^t sets of vertices, that we interpret as colors.



Precisely, suppose the graph has degree d.

We will choose $t = 2 + \log_2 d$ unit vectors $\vec{r}^{(1)}, \ldots, \vec{r}^{(t)} \in \mathbb{R}^n$ uniformly at random. Our 2^t colors correspond to $c_1, \ldots, c_t \in \{-1, 1\}$ such that $c_j \langle \vec{v}, \vec{r}^{(j)} \rangle \geq 0$ for $j = 1, \ldots, t$.

Theorem With probability at least $1/2$, all but $\frac{n}{4}$ edges are properly colored by this coloring if G was 3-colorable.

Proof: First, what is the probability that an edge (i,j) is properly colored by the coloring in terms of the $\{\vec{v}^{(i)}\}$ in a given solution? Recall that for each cut $\vec{r}^{(k)}$, i & j are on different sides with probability $\frac{2\theta(\vec{v}^{(i)}, \vec{v}^{(j)})}{2\pi}$. Since the t cuts are independent, the probability that i & j are on the same side for all t (& hence, the edge is not properly colored) is $\left(1 - \frac{1}{\pi}\theta(\vec{v}^{(i)}, \vec{v}^{(j)})\right)^t \leq \left(1 - \frac{1}{\pi}\frac{2\pi}{3}\right)^t$ since $\theta(\vec{v}^{(i)}, \vec{v}^{(j)}) \geq \frac{2\pi}{3} \Leftrightarrow \langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle \leq -\frac{1}{2}$, where by the lemma, for a 3-colorable graph, $\langle \vec{v}^{(i)}, \vec{v}^{(j)} \rangle \leq -\frac{1}{2}$ in

an optimal solution to the semidefinite program for all $(i,j) \in E$. This is $\leq \frac{1}{3}t = \frac{1}{3^{2+\log_3 d}} = \frac{1}{9d}$ by our choice of t. Note that $|E| \leq \frac{d|V|}{2}$, so $\frac{1}{9d} \leq \frac{|V|}{9 \cdot 2 \cdot |E|}$. The expected number of improperly colored edges is thus

$$\mathbb{E}\left[\sum_{(i,j)\in E} I[(i,j) \text{ improperly colored}]\right] = \sum_{(i,j)\in E} \Pr[(i,j) \text{ improperly colored}] \leq |E| \cdot \frac{|V|}{9 \cdot 2 \cdot |E|}$$

The number of improperly colored edges is nonnegative, so we can use Markov's inequality to conclude

$$\Pr\left[\#\text{ improperly colored } (i,j) \in E > \frac{|V|}{4}\right] < \frac{\frac{|V|}{9 \cdot 2}}{\frac{|V|}{4}} = \frac{2}{9} < \frac{1}{2}$$ ▨

We can check if an adequately large number of edges were properly colored, and resample the cuts if so. Thus, each iteration gives a $2^{2+\log_3 d} = 4d^{\log_3 2}$-coloring, for $4d^{\log_3 2}\log_2 n$ colors overall. Since $\log_3 2 = 0.631\ldots$ and $\log_2 n = O(n^\varepsilon)$ for all $\varepsilon > 0$, using the trivial bound $d \in n$ gives that this is $O(n^{0.631\ldots})$ colors. This is a nontrivial guarantee, but we can easily do much better by coloring high-degree vertices separately.

Observe first that we can easily find a 2-coloring of a 2-colorable graph. How? Consider any vertex; its neighbors must all receive the other color. The vertices reachable by paths of length 2 must receive the same color as the original vertex. Vertices reachable by paths of length 3 must receive the other color, and so on: we see by induction on the length of the path that vertices reachable by even length paths must receive the same color, and vertices reachable by odd length paths must receive the other color. Thus, we can find the coloring in $O(|V| + |E|)$ time during

a breadth-first search.

Now, in a 3-colorable graph, the neighbors of any given vertex v are 2-colorable since none of them can use v's color. If a vertex v has high degree, we can use this to find a coloring of a large number of vertices: supposing we again use a new set of colors for each high-degree vertex (degree $\geq d$) and remove all of these vertices from the graph, we see that we can find at most $\frac{n}{d}$ such degree-d vertices and use $3\frac{n}{d}$ colors before the remaining graph contains only vertices of degree $< d$. Thus, if we use our semidefinite program rounding algorithm to color the remaining low-degree portion of the graph, overall we will use $3\frac{n}{d} + 4d^{\log_3 2}\log_e n$ colors. An optimal choice of d minimizes the maximum of these two terms, occurring when they are equal: $3\frac{n}{d} = 4d^{\log_3 2}\log_e n \Leftrightarrow \frac{3}{4}\frac{n}{\log_e n} = d^{1+\log_3 2} \Leftrightarrow d = \left(\frac{3n}{4\log n}\right)^{\frac{1}{\log_3 6}}$

Then asymptotically, $d^{\log_3 2} = O\left(n^{\frac{\log_3 2}{\log_3 6}}\right) = O(n^{\log_6 2}) = O(n^{0.387\ldots})$

Unlike Max-Cut and similar problems, the simple semidefinite program rounding is not state-of-the-art for coloring (let alone optimal). The state-of-the-art algorithms combine rounding of more elaborate semidefinite programs with much more complicated combinatorial algorithms (than just 2-coloring neighborhoods of high-degree vertices) and obtain $o(n^{1/5})$ colors for 3-colorable graphs. It is strongly believed that no constant number of colors suffices, but this still leaves a large gap. Also note, the algorithm made strong use of

the assumption that the graph was 3-colorable.
Generalizations to other small numbers of colors are possible, but in general the problem of approximating the minimum number of colors is as hard as Clique/Independent Set: no $n^{1-\varepsilon}$-approximation algorithms exist unless P=NP.