CSC2411 - Linear Programming and Combinatorial Optimization*
Lecture 13: Coloring of Graphs

Notes taken by Waqas ur Rehman

April 19, 2007

Summary: In this lecture, we talk about coloring of graphs. First we discuss the randomized rounding technique of Karger, Matwani and Sudan that can color a 3-colorable graph in n vertices with maximum degree $\triangle$ in polynomial time using $O(\triangle^{0.631})$ colors. Next we discuss more generalized algorithm of Karger, Matwani and Sudan that combines this randomized technique with Wigderson's algorithm to yield an $O(n^{0.386})$-coloring of graph. Finally, we discuss Lovász Theta function, a SDP relaxation that led to finding clique and chromatic number of perfect graphs.

1 Background

We talked about coloring of graphs in previous lecture and have discussed two things.

Wigderson's Algorithm [Wig83] It is a totally combinatorial algorithm and allows us to color any 3-colorable graph in $O(\sqrt{n})$.

Relevant SDP Relaxation if $\chi(G) \leq 3$ then the following SDP is feasible

$$\forall_i \ \|v_i\| = 1$$

$$\forall_{ij} \in E \ \langle v_i, v_j \rangle \ \leq -\frac{1}{2}$$

This implies that if we have three coloring then the vertices of a graph can be partitioned into three disjoined sets such that vertices in the same set have no edges between them. So If we map these sets to three different points on a 2D sphere as shown in Figure 1, we get a maximum separation of 120^o between these points so the inner product of vectors is exactly $-\frac{1}{2}$.

* Lecture Notes for a course given by Avner Magen, Dept. of Computer Sciecne, University of Toronto.

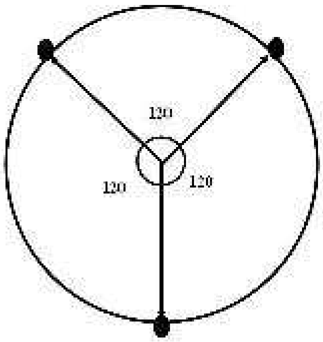

Figure 1: SDP relaxation for $\chi(G) \leq 3$

2 Karger-Matwani-Sudan's Algorithm (KMS)

Overview We use Wigderson's algorithm and SDP relaxation mentioned above to define KMS. As with the Geomans-Williamson's case we have some spread of points on unit sphere as our starting points. Points associated with the neighboring vertices have a large separation, 120^o at least. We can think Geomans-Williamson as a variant of coloring problem so that we try to color in two colors but not to satisfy all edges but to satisfy as many edges as possible. We use something similar here with the exception that now we use many more colors instead of just using two colors. Getting more and more colors will help us getting less and less monochromatic edges.

We take a random hyperplane and the color given to the vertices on one side of hyperplane is different from the color on other side. Next we take another random hyperplane and do the same, so in general we can do this for t times and at the end we partition the sphere into at most 2^t sections. In general we have t such random unit vectors and a point on the sphere is mapped to a ± 1 vector of dimension t depending on its $sign$ vector as shown in Figure 2.

Algorithm 2.1 (KMS Part I)**.**

1. Solve the VP and let $\upsilon_i, \upsilon_j, ...\upsilon_n$ be the solution.
2. Choose $t = \lceil \log_3 \triangle + 1 \rceil$ random hyperplanes $r_i, r_j, ..., r_n$ where $\triangle$ is the maximum degree of graph G.
3. Assign υ_i with a color that correspond to is $sign$ vector as follows

$$\upsilon_i \longrightarrow (sign(\langle \upsilon_i, r_i \rangle), sign(\langle \upsilon_i, r_j \rangle), ..., sign(\langle \upsilon_i, r_t \rangle)$$

Since there are 2^t distinct $sign$ vectors so 2^t colors are used in this step. Now we analyze the number of monochromatic edges when we use the mapping as

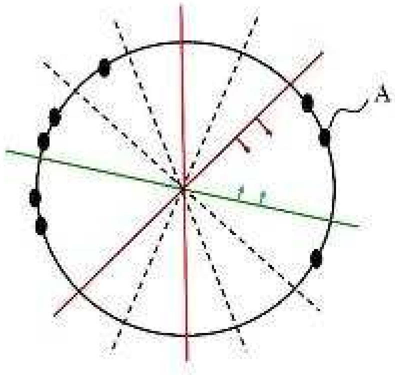

Figure 2: Point A is positive since all its $sign$ vectors are positives

our colors. The mapping can be thought of as a coloring where the name of color is $\pm$ vector in t-dimension. We foucs on one edge, let v_i and v_j be the vertices so the probability that they get the same color is exactly the probability that each and and every hyperplane will keep them on the same side, if any of t hyperplanes separate these vertices, they will get the different color. Suppose the angle between vertices is β then the probability that they are not separated is $\frac{\Pi-\beta}{\Pi}$. We know if there is an edge between vertices then they have a good separation and β is at least 120^o. So far an edge ij and for any coordinate s we have

$$Pr[sign(\langle v_i, r_s\rangle) = sign(\langle v_j, r_s\rangle)] \leq \frac{1}{3}$$

So from independence

$$Pr[col(v_i) = col(v_j)] \leq (\frac{1}{3})^t \leq \frac{1}{3\triangle}$$

So the expected number of monochromatic edges is

$$E[number\ of\ monochromatic\ edges] \leq m\frac{1}{3\triangle} \quad where\ m\ is\ number\ of edges$$

bounding m as a function of $\triangle$ we get

$$E[number\ of\ monochromatic\ edges] \leq \frac{n\triangle}{2}.\frac{1}{3\triangle} \leq \frac{n}{6}$$

4. After step 3 we have at most $\frac{n}{3}$ bad vertices that are involved in $\frac{n}{6}$ monochromatic edges. We remove the colors from the vertices with monochromatic edges and continue the process for uncolored vertices with fresh colors. To color all the vertices we require $\log(n)$ iterations and each round requires 2^t colors so in total we need $2^t.(\log(n))$ colors. See Figure 3.

$$Total\ colors\ used\ = 2^t.(\log(n)) \approx \triangle^{\log_3(2)} \approx \triangle^{0.631}$$

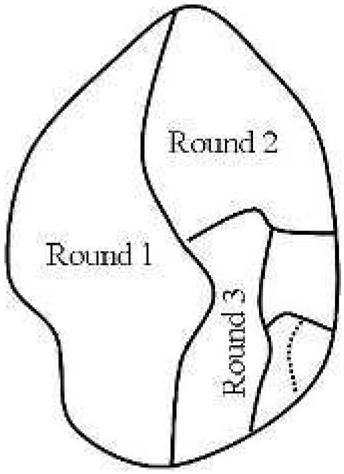

Figure 3: Graph Coloring with KMS Part 1

As we can see that there is no improvement in the number of colors used when compared to Wigderson's algorithm. To improve the number of colors Karger, Matwani and Sudan suggested another algorithm that combines Algorithm 2.1 with Wigderson's algorithm.

Algorithm 2.2 (KMS Part 2)**.**

1. As long as there exists vertex υ with degree $\geq \delta$, color the neighbors of υ with two colors. Afterwards we discard the neighbors of υ and the two colors used.
2. For the remaining graph apply Algorithm 2.1.

Since we cannot have more than $\frac{n}{\delta}$ iterations so total number of colors used $\leq \frac{n}{\delta} + \delta^{0.631}$. The number of colors used is minimized when $\frac{n}{\delta} = \delta^{0.631}$ which implies $\delta = n^{\frac{1}{1.631}}$. Thus, total colors used $= O(n^{\frac{1}{1.631}})$

3 Lovász Theta Function

Definition 3.1. Vector chromatic number is the smallest k for which the $\bar{G}$ is k-vector colorable. G is k-vector colorable if the following SDP is feasible.

$$\langle \upsilon_i, \upsilon_j \rangle \leq -\frac{1}{k-1} \ \forall_{ij} \ \epsilon \ E$$

$$\upsilon_i \ \epsilon \ \mathbb{R}^n$$

$$\langle \upsilon_i, \upsilon_i \rangle = 1$$

Lavász [Lov99] defined the vector chromatic number of a graph G and it is named Lovász Theta function theta $\theta(G)$, where

$$\omega(G) \leq \theta(\bar{G}) \leq \chi(G)$$

where $\omega(G)$ is size of maximal clique of a graph G and $\theta(\bar{G})$ is vector coloring of G.

Lemma 3.2. *If G is k-colorable then it is also k-vector colorable.*

Proof. Karger showed that for all positive integers k and n with $k \leq n+1$, there exist K unit vectors in $\Re^n$ such that the dot product of any distinct pair is $-\frac{1}{k-1}$. Bijectively mapping the k colors to these k vectors we can prove the lemma. For details refer to [KMS98]. □

Definition 3.3. Perfect graph is a graph G for which

$$\omega(H) = \chi(H)$$

$$\forall \text{ subgraph } H \text{ of } G$$

For perfect graphs $\theta(\bar{G})$ allows us to compute $\omega(G) = \chi(G)$

Lemma 3.4. $\omega(G) = \Theta(\bar{G})$

Proof. Let $\upsilon_i, \upsilon_j, ...\upsilon_n$ be an optimal vector chromatic solution of G.

$$\|\upsilon_i\| = 1$$

$$\langle \upsilon_i, \upsilon_j \rangle \ \leq -\frac{1}{k-1} \quad , \quad k = \Theta(\bar{G})$$

Suppose $\exists$ a clique of size ω in G ($\omega = \omega(G)$). Let I be that clique.

$$0 \leq \langle \sum_{i \epsilon I} \upsilon_i, \sum_{i \epsilon I} \upsilon_i \rangle = \sum_i \langle \upsilon_i, \upsilon_i \rangle + \sum_{i \neq j} \langle \upsilon_i, \upsilon_j \rangle$$

$$\Rightarrow \sum_i \langle \upsilon_i, \upsilon_j \rangle \geq -\omega$$

$$\Rightarrow \max_{i \neq j} \langle \upsilon_i, \upsilon_j \rangle \geq -\frac{\omega}{\omega(\omega-1)} = -\frac{1}{\omega-1}$$

$$\Rightarrow \Theta(\bar{G}) = k \geq \omega$$

□

References

[KMS98] David R. Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45(2):246–265, 1998.

[Lov99] L Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7, 1999.

[Wig83] A. Wigderson. Improving the performance guarantee for approximate graph coloring. *Journal of the ACM*, 30(4):729–735, 1983.