COMPSCI 632: Approximation Algorithms November 1, 2017

Lecture 18

Lecturer: Debmalya Panigrahi *Scribe: Feng Gui*

1 Overview

In this lecture, we examine graph coloring algorithms. We first briefly discuss some general results regarding graph coloring problems. We then focus on 3-colorable graphs and introduce a $O(\sqrt{n})$ coloring algorithm for such graphs. Finally, we present an algorithm by Karger, Motwani and Sudan that improves the above result using semi-definite programming [KMS98].

2 Graph Coloring

A k-coloring on a graph is an assignment of vertices to k colors such that no edge is monochromatic. More specifically, let $G = (V, E)$ be a graph. A k-coloring is a function $f : V \to \{1, ..., k\}$ such that $f(u) \neq f(v)$ for any edge $(u, v) \in E$. A graph G is said to be k-colorable if there exists a k-coloring for G.

One of the most well known results for graph colorability is the Four Color Theorem, which states that every planar graph is 4-colorable. Though this result was proven in the last century, most colorability problems are found to be extremely hard, meaning that many problems don't have any reasonable approximation. Here are some known hardness results that can be found in the literature.

Theorem 1. *It is NP-complete to determine whether a given graph admits a k-coloring for $k \geq 2$. Moreover, approximating k-coloring within $O(n^{1-\varepsilon})$ colors is NP-hard [Zuc07].*

Note that deciding if a given graph is 2-colorable is equivalent to determining whether it is a bipartite graph, which only takes polynomial time. The next easiest case to think about is 3-colorability. In the rest of this lecture, we will focus on 3-colorable graphs.

Theorem 2. *It is NP-hard to color a 3-colorable graph with 4 colors [GK00].*

Theorem 3. *If Unique Games Conjecture is true, then there is no constant factor approximation for 3-coloring.*

3 3-coloring Approximation

Given a 3-colorable graph, our goal is to have a polynomial time algorithm that colors the graph with as few colors as possible. The best known result so far uses $O(n^{\varepsilon})$ colors where $\varepsilon \approx 0.21$. But to begin with, we introduce the long known Widgerson's algorithm which gives an $O(\sqrt{n})$ coloring [Wig83].

3.1 Widgerson's Algorithm

Here is a basic fact that is true for all graphs.

Lemma 4. *Let Δ be the maximum degree of all vertices in graph G. Then G can be colored with $\Delta + 1$ colors in polynomial time.*

Proof. Simple greedy-like arguments will work in this case. We color the vertices in any order. When coloring a vertex, since it has at most Δ neighbors, there's at least one color that will not yield any monochromatic edge if colored on this vertex. So you can continue this process until all vertices are colored. □

Notice that lemma 4 has nothing to do with 3-colorability. To obtain a better result for 3-colorable graphs, we need the following observation.

Lemma 5. *Given a 3-colorable graph G and one of its vertex v, the subgraph containing vertices $\{v\} \cup N(v)$ (i.e. v and its neighbors) can be colored with 3 colors in polynomial time.*

Proof. We first color v with color 1. Then by the 3-colorability condition on graph G, the subgraph containing $N(v)$ can be colored with 2 colors, meaning that $N(v)$ forms a bipartite graph. But we already knew that coloring a bipartite graph takes only polynomial time. □

Based on lemma 4 and 5, Wigderson's algorithm is formulated as follows

1. We repeatedly color the maximum degree vertex and its neighbors with 3 new colors and remove them until the maximum degree of the graph, denote by Δ^*, is not very large.
2. Then we color the the remaining graph with $\Delta^* + 1$ colors.

Suppose our first step is repeated for k times, then the total number of colors we will use is $3k + \Delta^* + 1$. Notice that every time we remove the maximum degree vertex and its neighbors in the graph, we remove at least $\Delta^* + 1$ vertices. Therefore, we have $(\Delta^* + 1)k \le n$. We can lower bound the total number of colors by

$$3k + \Delta^* + 1 \le \frac{3n}{\Delta^* + 1} + \Delta^* + 1 \tag{1}$$

Pick $\Delta^* + 1 \approx \sqrt{n}$, we then obtain an $O(\sqrt{n})$ coloring.

3.2 KMS Algorithm with SDP Formulation

In 1994, Karger, Motwani and Sudan (KMS) improved this result using semi-definite programming (SDP) for the second step in Wigderson's algorithm.

Suppose for each vertex, we have a unit length vector, often referred as vertex vector, as a variable. The goal is to make two vertex vectors are as far as possible if there's an edge connect the two vertices. Then we want to color vertices with the same color when they are "close enough" on the unit circle. More specifically, we have SDP formulation:

$$\begin{aligned} \min \quad & t && && (2) \\ \text{subject to} \quad & v_i \cdot v_j \le t && \forall\, (i,j) \in E && (3) \\ & \|v_i\| = v_i \cdot v_i = 1 && \forall\, i && (4) \end{aligned}$$

We need to understand what 3-colorability property implies about this SDP. To that end, let's first introduce some definitions.

Definition 1. *Suppose the above SDP has optimal solution t^*. Then the Lovasz number of a graph G (sometimes referred as Lovasz theta function) is defined as*

$$\vartheta(G) = 1 - \frac{1}{t^*} \tag{5}$$

Definition 2. *The chromatic number $\chi(G)$ of a graph G is the smallest value of k for which G admits a k-coloring.*

Definition 3. *The clique number $\eta(G)$ is the number of vertices in a maximum clique of G.*

If graph G has a clique of size k, then any two of these k vertices must be colored with different colors. Therefore $\chi(G) \geq \eta(G)$. Moreover, it turns out that the Lovasz number is bounded by chromatic number and clique number of a graph: $\chi(G) \geq \vartheta(G) \geq \eta(G)$.

Lemma 6. *If $\eta(G) = k$, then $\vartheta(G) \geq k$.*

Proof. Without the loss of generality, we can assume that vertex v_1, v_2, ..., v_k forms a clique. Then by the nonnegativity of Euclidean norm, we have

$$\left\|\sum_{j=1}^{k} v_j\right\|^2 = \sum_{j=1}^{k} \|v_j\|^2 + 2 \sum_{1\leq i<j\leq k} v_i \cdot v_j \geq 0 \tag{6}$$

Since $\|v_i\| = 1$ for all $j = 1,2,...,k$,

$$k(k-1) \cdot \max_{1\leq i<j\leq k} v_i \cdot v_j \geq 2 \sum_{1\leq i<j\leq k} v_i \cdot v_j \geq -\sum_{j=1}^{k} \|v_j\|^2 = -k \tag{7}$$

Also as these k vertices forms a clique in G, we have $t^* \geq v_i \cdot v_j$ for all $i, j \in \{1,2,...,k\}$ with $i \neq j$. From equation 7,

$$t^* \geq \max_{1\leq i<j\leq k} v_i \cdot v_j = -\frac{k}{k(k-1)} = -\frac{1}{k-1} \tag{8}$$

Then it follows that $\vartheta(G) \geq k$ □

Lemma 7. *If $\chi(G) = k$, then $\vartheta(G) \leq k$.*

Proof. Given a k-coloring on a graph G, we denote V_i be the set of vertices that are colored with color i. Notice that by definition of Lovasz number, it suffices to show $t^* \leq \frac{-1}{k-1}$. We will prove this result by induction.

For $k = 2$, set

$$\vec{v} = v_1 = (1,0,...,0) \qquad \forall\, v \in V_1 \tag{9}$$

$$\vec{u} = v_2 = (-1,0,...,0) \quad \forall\, u \in V_2 \tag{10}$$

Since all edges are between V_1 and V_2, we have $t^* \leq t = v_1 \cdot v_2 = -1$. This implies that our statement is true for the base case $k = 2$.

Now suppose the graph G is k-colorable and we can construct a feasible solution which sets $\vec{v} = v_k \in \mathbb{R}^k$ for all $v \in V'_k$ such that $v'_i \cdot v'_j \leq -\frac{-1}{k-1}$. Then for any $(k+1)$-colorable graph, set

$$\vec{v} = v_{k+1} = (0,...,0,1) \in \mathbb{R}^{k+1} \quad \forall\, v \in V_{k+1} \tag{11}$$

$$\vec{u} = v_i = \left(\alpha v'_i, -\frac{1}{k}\right) \in \mathbb{R}^{k+1} \quad \forall\, u \in V_i, \quad 1 \leq i \leq k \tag{12}$$

with $\alpha = \sqrt{1 - 1/k^2}$.

It is clear that $\|v_{k+1}\| = \|v_k\| = 1$. By induction hypothesis,

$$v_i \cdot v_{k+1} = -\frac{1}{k} \qquad \forall\, 1 \leq i \leq k \tag{13}$$

$$v_i \cdot v_j = \alpha^2 v'_i \cdot v'_j + \frac{1}{k^2} \leq -\frac{\alpha^2}{(k-1)} + \frac{1}{k^2} = -\frac{1}{k} \qquad \forall\, 1 \leq i < j \leq k \tag{14}$$

Therefore, this construction is a feasible solution for the SDP and we obtain $t^* \leq t = -\frac{1}{k}$ which implies our statement is also true for case $k+1$. □

With the aid of lemma 6 and 7, we now can improve step 2 in Wigderson's algorithm.

Theorem 8. *Given 3-colorable graph G with maximum degree Δ, it takes polynomial time to color G with $O(\Delta^{\log_3 2} \cdot \log n)$ colors.*

Proof. We first solve the SDP associated with graph G and denote the optimal solution by v_i^* and t^*. From lemma 7, we know that

$$v_i^* \cdot v_j^* \leq -\frac{1}{2} \tag{15}$$

Since dot product for two unit length vectors is just the cosine value of the angle between them, equation 15 tells us that the angle between v_i^* and v_j^* is greater than $2\pi/3$ if $(i,j) \in E$. Now if we take a random hyperplane and color the vertices on one side with one color and the vertices on the other side with a second color, then

$$\mathbb{P}[(i,j) \in E \text{ is monochromatic}] \leq \frac{1}{3}. \tag{16}$$

If we take l random hyperplanes instead, the unit ball will be partitioned into 2^l regions. We use the same color for the vertices in the same region. Then we need 2^l colors to color all vertices in G. The probability that a certain edge in G is monochromatic is $1/3^l$ as these hyperplanes are taken independently. We want to repeat this process on the vertices with monochromatic edges. So our algorithm is formulated as follows:

1. Solve SDP associated with graph G.
2. Take l random hyperplanes and color the vertices with 2^l new colors. Remove vertices that have no monochromatic edges.
3. Repeat step 2 until all vertices are colored.

As the maximum degree of graph G is Δ. The total number of edges $m \leq \frac{n\Delta}{2}$. Every time we execute step 2, we are expecting less than

$$\frac{n\Delta}{2} \cdot \frac{1}{3^l} = \frac{n\Delta}{2 \cdot 3^l} \tag{17}$$

edges that go to the next round. Pick $l = \log_3 \Delta + 1$, then the expected number of vertices that are remained after each round is less than $n/3$. That implies our algorithm will terminate in $\log_3 n$ steps.

So the total number of color we used is

$$2^l \log_3 n = 2^{\log_3 \Delta + 1} \log_3 n \approx O(\Delta^{\log_3 2} \cdot \log n) \tag{18}$$

□

Taking theorem 8 back to equation 1, we have a polynomial time algorithm that uses less than

$$\frac{3n}{\Delta^* + 1} + (\Delta^*)^{\log_3 2} \cdot \log n \tag{19}$$

colors. Let $(\Delta^*)^{1+\log_3 2} \approx n$, then

$$\frac{3n}{\Delta^* + 1} + (\Delta^*)^{\log_3 2} \cdot \log n \approx (\Delta^*)^{\log_3 2}(3+n) \approx O(n^{0.39}) \tag{20}$$

Therefore, the expected number of colors we will use is $O(n^{0.39})$. A smarter preprocessing presented by Karger, Motwani and Sudan gives an $O(n^{0.25})$-coloring, which we will not discuss in class.

References

[GK00] Venkatesan Guruswami and Sanjeev Khanna. On the hardness of 4-coloring a 3-colorable graph. In *In Proceedings of the 15th Annual IEEE Conference on Computational Complexity*, pages 188–197. IEEE Computer Society Press, 2000.

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

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

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007.