CS 170 HW 13

Due **2021-05-04, at 10:00 pm**

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, write "none".

2 $\sqrt{n}$ coloring

(a) Let G be a graph of maximum degree Δ. Show that G is $(\Delta + 1)$-colorable.

(b) Suppose G is a 3-colorable graph. Let v be any vertex in G. Show that the graph induced on the neighborhood of v is 2-colorable. *Clarification: the graph induced on the neighborhood of v refers to the subgraph of G obtained from the vertex set V' comprising vertices adjacent to v (but not v itself) and edge set comprising all edges of G with both endpoints in V'.*

(c) Give a polynomial time algorithm that takes in a 3-colorable n-vertex graph G as input and outputs a valid coloring of its vertices using $O(\sqrt{n})$ colors. Prove that your algorithm is correct and also analyze its runtime.
Hint: think of an algorithm that first colors "high-degree" vertices and their neighborhoods, and then colors the rest of the graph. The previous two parts might be useful.

3 Cuts from Colors

Given a graph $G = (V, E)$ on n vertices and m edges, and a vector $x \in \{\pm 1\}^n$, we say

$$\mathsf{Cut}(G, x) := \frac{1}{m} \sum_{\{i,j\} \in E} \left(\frac{x_i - x_j}{2} \right)^2$$

and define

$$\mathsf{MaxCut}(G) := \max_{x \in \{\pm 1\}^n} \mathsf{Cut}(G, x).$$

For every algorithmic question below, please analyze your runtime and prove that your algorithm is correct.

(a) (Warmup; ungraded) Let G be any graph. Prove that

$$\mathsf{MaxCut}(G) \geq \frac{1}{2}$$

always, and give a polynomial time randomized algorithm that outputs $\boldsymbol{x} \in \{\pm 1\}^n$ satisfying $\mathbf{E}[\mathsf{Cut}(G, \boldsymbol{x})] \geq \frac{1}{2}$.
What if each x_i is chosen to be $+1$ or -1 uniformly independently?

(b) Let G be any graph. Prove that

$$\mathsf{MaxCut}(G) \geq \frac{1}{2} + \Omega\left(\frac{1}{n}\right)$$

always, and give a polynomial time randomized algorithm that outputs $\boldsymbol{x} \in \{\pm 1\}^n$ satisfying $\mathbf{E}[\mathsf{Cut}(G, \boldsymbol{x})] \geq \frac{1}{2} + \Omega\left(\frac{1}{n}\right)$.
Hint: try to construct a simple distribution over ± 1 vectors such that for $\boldsymbol{x}$ drawn from this distribution any $i, j \in [n]$, $\mathbf{E}[\boldsymbol{x}_i \boldsymbol{x}_j] = -\frac{c}{n}$ for some absolute constant $c > 0$.

(c) Let G be any k-colorable graph. Prove that

$$\mathsf{MaxCut}(G) \geq \frac{1}{2} + \Omega\left(\frac{1}{k}\right).$$

Note that we are not asking you to give an algorithm to find such a cut, but instead just asking you to prove existence. Try to reduce this problem to the previous part.

(d) Let G be any 3-colorable graph. Give a polynomial time randomized algorithm to find a $\boldsymbol{x} \in \{\pm 1\}^n$ satisfying:

$$\mathbf{E}[\mathsf{Cut}(G, \boldsymbol{x})] \geq \frac{1}{2} + \Omega\left(\frac{1}{\sqrt{n}}\right).$$

Hint: Part (b) of Question 2 may be useful.

(e) Let G be any graph with maximum degree Δ. Prove that

$$\mathsf{MaxCut}(G) \geq \frac{1}{2} + \Omega\left(\frac{1}{\Delta}\right).$$

Give a polynomial time randomized algorithm to find a $\boldsymbol{x} \in \{\pm 1\}^n$ satisfying:

$$\mathbf{E}[\mathsf{Cut}(G, \boldsymbol{x})] \geq \frac{1}{2} + \Omega\left(\frac{1}{\Delta}\right).$$

Hint: Part (a) of Question 2 might be useful here.

4 Reservoir Sampling

(a) Design an algorithm that takes in a stream $z_1, \ldots, z_M$ of M integers in $[n]$ and at any time t can output a uniformly random element in $z_1, \ldots, z_t$. Your algorithm may use at most polynomial in $\log n$ and $\log M$ space. Prove the correctness and analyze the space complexity of your algorithm. Your algorithm may only take a single pass of the stream.
Hint: $\frac{1}{t} = 1 \cdot \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{3}{4} \cdots \frac{t-1}{t}$.

(b) For a stream $S = z_1, \ldots, z_{2n}$ of $2n$ integers in $[n]$, we call $j \in [n]$ a *duplicative element* if it occurs more than once. Prove that S must contain a duplicative element, and design an algorithm that takes in S as input and with probability at least $1 - \frac{1}{n}$ outputs a duplicative element. Your algorithm may use at most polynomial in $\log n$ space. Prove the correctness and analyze the space complexity of your algorithm. Your algorithm may only take a single pass of the stream.