CMPT409/815: Advanced Algorithms

Midterm Exam

Instructor: Igor Shinkar *November 2, 2020*

- Write your name and SFU ID (numerical and alphabetical) at the top of your solution.
- Submit your solution to Coursys before 12:30.
- Allowed formats for solutions: word, pdf, or any other readable format in a zip file.
- Coursys may give you a warning that you submitted only one file. That's ok.

- **This is an open book exam. You may use books, lecture notes, the internet for hints/solutions**.
- If you find the solution on the internet, you should add the reference. (No penalty for that)
- No matter how you find the solution, you need to explain all your answers.
- Provide enough details to convince me that you understand the solution.

- **You may not go to chegg.com or similar websites and ask for solutions.**
- **You may not discuss your solutions with other students.**

- The exam consists of four (4) problems. The sum of all points in the four problems is 125.
- Explain all your answers. Provide enough details to convince me that you understand the solution.

Question 1 (30 points)

(a) **[10 points]** *Consider the* Minimum Vertex Cover *problem*[1] *for 3-uniform hypergraphs: Given a 3-uniform hypergraph* $H = (U, F)$ *the goal is to find a vertex cover* $C \subseteq U$ *of* H *such that* $|C|$ *is minimized. Let* $\textsf{min-VC}(H) = \min\{|C| : C \text{ is a vertex cover in } H\}$.

Design a polynomial time 3-approximation algorithm for the problem of finding a minimum vertex cover in a hypergraph. That is, the algorithm must output a vertex cover C *such that* $|C| \leq 3 \cdot \textsf{min-VC}(H)$.

(b) **[15 points]** *Given an undirected graph* $G = (V, E)$*, denote by* $Rem_{\Delta}(G)$ *the minimum number of edges required to remove from* G *so that the remaining graph contains no triangle.*

Design a polynomial time 3-approximation algorithm for $Rem_{\Delta}(G)$*. That is, the algorithm outputs a subset of edges* $E' \subseteq E$ *such that* $G = (V, E \setminus E')$ *contains no triangles, and* $|E'| \leq 3 \cdot Rem_{\Delta}(G)$.

---

[1] Recall, a *vertex cover* in H is a collection of vertices $C \subseteq U$ that touches every hyperedge in H, i.e., for every $e \in F$ it holds that $e \cap C \neq \emptyset$.

Question 2 (40 points) *In the* MAX-CUT *problem, we are given graph $G = (V, E_G)$, and our goal is to find a partition of the vertices into two parts $V = V_1 \cup V_2$ such that the number of edges with exactly one end point in each V_i is maximized. Denote the* MAX-CUT *of G by*

$$\mathsf{MAX-CUT}(G) = \max_{V=V_1 \cup V_2} \frac{|\{(u,v) \in E_G : (u \in V_1, v \in V_2) \text{ or } (v \in V_1, u \in V_2)\}|}{|E|}.$$

Let G be a graph with m edges and n vertices such that $m > n^{1.5}$. Let $H = (V, E_H)$ be a random sub-graph of G obtained by keeping each edge of G with some probability $p \in (0,1]$.

Fix $\varepsilon \in (0, 0.1)$. Prove that there exists an absolute constant $C > 0$ such that if we remove each edge with probability $p > \frac{Cn}{\varepsilon^2 m}$, then the following holds:

(a) **[5 points]** *If* $\mathsf{MAX-CUT}(G) = 1$, *then* $\Pr[\mathsf{MAX-CUT}(H) = 1] = 1$.

(b) **[10 points]** *Let $m_G = |E_H|$ be the number of edge in H. Then $\Pr[|m_H - pm| < \varepsilon pm] > 1 - 2^{-n}$.*

(c) **[10 points]** $\Pr[\mathsf{MAX-CUT}(H) \geq \mathsf{MAX-CUT}(G) - \varepsilon] > 1 - 2^{-n}$.

(d) **[15 points]** $\Pr[\mathsf{MAX-CUT}(H) \leq \mathsf{MAX-CUT}(G) + \varepsilon] > 1 - 2^{-n}$.
(Hint: use Chernoff bound)

Question 3 (35 points) *Consider the following linear problem.*

$$\text{minimize}_{x,y \in \mathbb{R}} \quad 3x + 2y$$

$$\text{subject to} \begin{cases} 2x + 4y & \geq 1 \\ y - 2x & \leq 2 \\ -5 \leq x \leq 5 & \\ -5 \leq y \leq 5 & \end{cases}$$

(a) **[15 points]** *Find the optimal solution for the LP.*

(b) **[10 points]** *Convert the LP into a canonical form. Write explicitly the matrix A and the vectors b and c representing the LP.*

(c) **[10 points]** *Write the dual LP of the canonical form.*

Question 4 (20 points) *Generalizing on the notion of a cut, we define a k-cut in an undirected graph as follows. In the min-k-cut problem, the input is a graph $G = (V, E)$ and the goal is to find a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$ into k non-empty disjoint sets that minimizes the number of edges of G that do not belong to the same V_i.*

Recall Karger's min-cut algorithm we saw in class. Show that the algorithm can be modified to find a minimum k-cut in time $n^{O(k)}$ with high probability.