Final Exam, Advanced Algorithms 2020-2021

- You are only allowed to have a handwritten A4 page written on both sides.
- Communication, calculators, cell phones, computers, etc... are not allowed.
- Your explanations should be clear enough and in sufficient detail that a fellow student can understand them. In particular, do not only give pseudo-code without explanations. A good guideline is that a description of an algorithm should be such that a fellow student can easily implement the algorithm following the description.
- **You are allowed to use any result stated in class with proving it.**
- **Problems are not necessarily ordered by difficulty.**
- **Do not touch until the start of the exam.**

Good luck!

Name: ______________________ **N° Sciper:** ______________

Problem 1 / 12 points	Problem 2 / 34 points	Problem 3 / 14 points	Problem 4 / 26 points	Problem 5 / 14 points

Total / 100

1 *(12 pts)* **Simplex method.** Suppose we use the Simplex method to solve the following linear program:

$$\begin{aligned} \textbf{minimize} \quad & -x_1 - 4x_2 + 4x_3 \\ \textbf{subject to} \quad & 2x_1 - 5x_2 + 2x_3 \leq 5 \\ & x_2 \leq 1 \\ & x_1 - 3x_2 + 3x_3 \leq 3 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

At the current step, we have the following Simplex tableau:

$$\begin{aligned} s_1 &= 10 - 2x_1 - 2x_3 \\ x_2 &= 1 - s_2 \\ s_3 &= 6 - x_1 - 3x_3 \end{aligned}$$

$$z = -4 - x_1 + 4x_3$$

Write the tableau obtained by executing one iteration (pivot) of the Simplex method starting from the above tableau.

Solution:

At the current step, we have the following Simplex tableau:

$$s_1 = 10 - 2x_1 - 2x_3 \tag{1}$$
$$x_2 = 1 - s_2 \tag{2}$$
$$s_3 = 6 - x_1 - 3x_3 \tag{3}$$

$$z = -4 - x_1 + 4x_3$$
$$x_1 := 0 \; x_2 := 1 \; x_3 := 0 \; s_1 := 10 \; s_2 := 0 \; s_3 := 6$$

Only x_1 has a negative coefficient in z, we will pivot x_1. We have
$\nearrow x_1 \longrightarrow \; x_1 \leq 10/2 \; (1), \; x_1 \leq \; \infty \; (2), \; x_1 \leq 6/1 \; (3) \longrightarrow x_1 := 5, \; s_1 := 0$

$$\begin{aligned} x_1 &= 5 - \frac{s_1}{2} - x_3 \\ x_2 &= 1 - s_2 \\ s_3 &= 1 + \frac{s_1}{2} - 2x_3 \end{aligned}$$

$$z = -9 + \frac{s_1}{2} + 5x_3$$
$$x_1 := 5 \; x_2 := 1 \; x_3 := 0 \; s_1 := 0 \; s_2 := 0 \; s_3 := 1$$

2 *(34 pts)* **Hypergraph cuts.** Let $G = (V, E)$ be a hypergraph with vertex set V and hyperedge set E (every hyperedge $e \in E$ is a subset of V; see Fig. 1 for an illustration). For $S \subseteq V$ the set of hyperedges crossing the cut $(S, V \setminus S)$ is defined as

$$E(S, V \setminus S) = \{e \in E : e \cap S \neq \emptyset \text{ and } e \cap V \setminus S \neq \emptyset\},$$

and the size of the cut $(S, V \setminus S)$ as $|E(S, V \setminus S)|$.

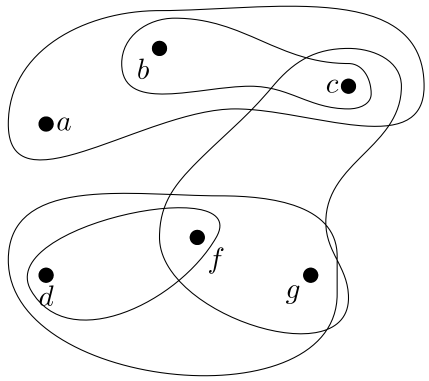

Figure 1. A hypergraph $G = (V, E)$ with $V = \{a, b, c, d, f, g\}$ and hyperedge set $E = \{e_1, e_2, e_3, e_4, e_5\}$, where $e_1 = \{a, b, c\}, e_2 = \{b, c\}, e_3 = \{d, f\}, e_4 = \{c, f, g\}$ and $e_5 = \{d, f, g\}$.

2a *(20 pts)* Give an algorithm that finds the size of the minimum cut in a given hypergraph G, i.e. outputs

$$\min_{S \subset V, S \neq \emptyset} |E(S, V \setminus S)|.$$

For example, the size of the minimum cut in the hypergraph G in Fig. 1 is 1. There are two minimum cuts: $(\{a\}, \{b, c, d, f, g\})$ and $(\{a, b, c\}, \{d, f, g\})$.

Your algorithm should run in time polynomial in the number of vertices and hyperedges in G.

Hint: use submodularity.

Solution: Let $f(S) = E(S, V \setminus S)$. We first observe that f is submodular. We verify the diminishing returns property. For every $S \subseteq T \subseteq V$ and $u \in V$ we have

$$\begin{aligned} f(u|S) &= f(S \cup \{u\}) - f(S) \\ &= |\{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}| \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \end{aligned} \tag{1}$$

Since $S \subseteq T$, we have

$$\{e \in E : u \in e, e \cap T = \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) \neq \emptyset\} \subseteq \{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}$$

and

$$\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\} \subseteq \{e \in E : u \in e, e \cap T \neq \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) = \emptyset\}.$$

We thus get by (1)

$$\begin{aligned} f(u|S) &= |\{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}| \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \\ &\leq |\{e \in E : u \in e, e \cap T = \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) \neq \emptyset\}| \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \\ &\geq f(u|T) \end{aligned}$$

and therefore $f(S)$ is submodular.

Now we would like to use the fact that unconstrained submodular function minimization can be done in polynomial time. However, we do have constraints $S \neq \emptyset$ and $S \neq V$. Thus, instead we pick an element $u \in V$ and then for every other $v \in V \setminus \{u\}$ define

$$g(S) = f(S \cup \{u\})$$

and find

$$\min_{S \subseteq V \setminus \{u,v\}} g(S).$$

Note that $g(S)$ is submodular, since for every $w \in V \setminus \{u, v\}$ and $S \subseteq T \subseteq V$ one has

$$g(w|S) = f(w|S \cup \{u\}) \geq f(w|T \cup \{u\}) = g(w|T)$$

by the diminishing returns property of f.

(This page is intentionally left blank.)

2b *(14 pts)* Give a randomized polynomial time algorithm that outputs, given a hypergraph G where every hyperedge contains three vertices, a cut $(S, V \setminus S)$ such that

$$\mathbb{E}[|E(S, V \setminus S)|] \geq (3/4)OPT, \qquad (*)$$

where

$$OPT = \max_{S \subseteq V} |E(S, V \setminus S)|.$$

Note that unlike **2a**, here we are interested in the **maximum** cut. Your algorithm should run in time polynomial in the number of vertices and hyperedges in G, and you should prove that the expected size of the cut that it outputs satisfies $(*)$.

Hint: consider a random cut.

Solution: Let S include every vertex $u \in V$ independently with probability $1/2$. Then

$$\mathbb{E}[|E(S, V \setminus S)|] = \sum_{e \in E} \Pr[e \in E(S, V \setminus S)].$$

Let $e = \{u, v, w\}$, and suppose that $u \in S$ (this is without loss of generality, as $E(S, V \setminus S) = E(V \setminus S, S)$). Then

$$\Pr[e \in E(S, V \setminus S)] = 1 - \Pr[v, w \in S] = 3/4.$$

Thus, a random cut cuts at least $3/4$ of the hyperedges in expectation. By Markov's inequality applied to $|E| - |E(S, V \setminus S)|$ we have

$$\Pr[|E| - |E(S, V \setminus S)| > 1/4(1 + 1/n)|E|] \leq 1/(1 + 1/n) = 1 - O(1/n).$$

3 (14 pts) **Finding heavy elements in data streams.** Consider a data stream $\sigma = (a_1, \ldots, a_m)$, with $a_j \in [n]$ for every $j = 1, \ldots, m$, where we let $[n] := \{1, 2, \ldots, n\}$ to simplify notation. For $i \in [n]$ let f_i denote the number of times element i appeared in the stream σ.

We say that a stream σ is *approximately sparse* if there exists $i^* \in [n]$ such that $f_{i^*} = \lceil n^{1/4} \rceil$ and for all $i \in [n] \setminus \{i^*\}$ one has $f_i \leq 10$. We call i^* the *dominant* element of σ. Give a single pass streaming algorithm that finds the dominant element i^* in the input stream as long as the stream is approximately sparse. Your algorithm should succeed with probability at least 9/10 and use $O(n^{1/2} \log^2 n)$ bits of space. You may assume knowledge of n.

Hint: use $O(n^{1/2})$ AMS sketches.

Solution: We partition the universe into $\sqrt{n}$ disjoint blocks $[n] = B_1 \cup \ldots \cup B_{\sqrt{n}}$ each of size $\sqrt{n}$ and apply the AMS sketch with ε a sufficiently small constant and $\delta = 1/n^2$. Denote the corresponding frequency vectors by $f^1, \ldots, f^{\sqrt{n}} \in \mathbb{R}^{\sqrt{n}}$. The algorithm is as follows. For every $i \in [\sqrt{n}]$ and every $j \in B_i$ we use the AMS sketch to obtain a $(1 \pm \varepsilon)$-approximation to

$$||f^i||_2^2$$

and

$$||f^i - \lceil n^{1/4} \rceil \cdot e_j||_2^2.$$

Since blocks are of size $\sqrt{n}$, when we subtract an incorrect element, the corresponding Euclidean norm squared goes up by at least a $(1+\Omega(1))$ factor, which we can detect with the AMS sketch as long as ε is a small constant. If we subtract a correct element, the Euclidean norm squared reduces by at least a $(1-\Omega(1))$ factor, which we can again detect with the AMS sketch with constant ε. The setting of $\delta = 1/n^2$ ensures that we can afford a union bound over all possible elements to subtract.

4 *(26 pts)* **Spectral graph theory.** For a d-regular graph $G = (V, E), |V| = n$, let $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ be the eigenvalues of its normalized adjacency matrix $M = \frac{1}{d}A$.

4a *(8 pts)* Let $G = (V, E)$ be a cycle graph: $V = \{0, 1, 2, \ldots, n-1\}$, $n \geq 3$, and there is an edge between vertex $a \in V$ and vertex $b \in V$ if and only if $a \equiv b \pm 1 \pmod{n}$. Prove that $\lambda_2 = 1 - O(1/n)$.

Solution: Let $S = \{0, 1, \ldots, \lfloor n/2 \rfloor - 1\}$. The edge expansion of G is at most

$$h(S) = \frac{2}{2 \cdot (\lfloor n/2 \rfloor - 1)} = O(1/n).$$

Thus, by Cheeger's inequality we have $\lambda_2 \leq 2h(S) = O(1/n)$.

4b *(10 pts)* Let $G = (V, E)$ be a cycle graph: $V = \{0, 1, 2, \ldots, n-1\}$, $n \geq 3$, and there is an edge between vertex $a \in V$ and vertex $b \in V$ if and only if $a \equiv b \pm 1 \pmod{n}$. Prove that $\lambda_n = -1$ when n is even and $\lambda_n = -1 + O(1/n)$ when n is odd. For the latter you may use the fact that $\lambda_n = \min_{x \in \mathbb{R}^n \setminus \{0^n\}} \frac{x^T M x}{x^T x}$.

Solution: Since

$$\lambda_n = \min_{x \in \mathbb{R}^n \setminus \{0^n\}} \frac{x^T M x}{x^T x},$$

it suffices to exhibit one choice of $x \in \mathbb{R}^n \setminus \{0^n\}$ for which $\frac{x^T M x}{x^T x} = -1 + O(1/n)$. Let $x_u = (-1)^u$ for $u \in V$. We have

$$\frac{x^T M x}{x^T x} = \frac{\frac{1}{2} \sum_{u,v:\{u,v\} \in E} x_u \cdot x_v}{n}.$$

If n is even, then $x_u \cdot x_v = -1$ for all $\{u, v\} \in E$, and the right hand side above is -1. If n is odd, then we get $x_u \cdot x_v = -1$ unless $u = 0$ and $v = n-1$, in which case we have $x_u \cdot x_v = 1$, and therefore the rhs above is $\frac{1}{2}(-1 \cdot 2(n-1) + 1 \cdot 2)/n = -1 + O(1/n)$, as required.

4c *(8 pts)* Let $G = (V, E)$ be the hypercube graph: $V = \{0, 1\}^d$ for some integer $d \geq 1$, $n = 2^d$, and there is an edge between vertex $x \in V$ and vertex $y \in V$ if and only if the Hamming distance between x and y is exactly one (recall that the Hamming distance between x and y is the number of coordinates on which they differ). Prove that $\lambda_n = -1$.

Solution: We note that the hypercube is a bipartite graph, with the bipartition given by vertices with even Hamming weight on one side and vertices with odd Hamming weight on the other side. Thus, $\lambda_n = 2$ by the results proved in class.

5 *(14 pts)* **Approximate k-center.** In the k-center problem you are given n points $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$, and your task is to find k centers $C = \{c_0, c_1, \ldots, c_{k-1}\} \subset \mathbb{R}^d$ that best summarize the dataset in the following formal sense. For a collection C of centers and $p \in P$ we first define

$$d(p, C) = \min_{c \in C} ||p - c||_2,$$

where $|| \cdot ||_2$ stands for the Euclidean norm. Then we define the cost of a collection C of centers as

$$\text{cost}(C) = \max_{p \in P} d(p, C).$$

In this problem you will analyze the approximation ratio of a natural algorithm for the k-center problem:

Algorithm 1 Approximate k-center.

procedure ApproxKCenter(P, k)
  $c_0 \leftarrow$ arbitrary point in P
  for $i = 1$ to k **do**
    $c_i \leftarrow$ a point in P furthest from $\{c_0, \ldots, c_{i-1}\}$ ▷ Breaking ties arbitrarily
    ▷ Formally, $c_i = \text{argmax}_{p \in P} d(p, \{c_0, \ldots, c_{i-1}\})$
  end for
  return $\{c_0, \ldots, c_{k-1}\}$ ▷ Note that c_k is not returned
end procedure

Let $C = \{c_0, \ldots, c_{k-1}\}$ denote the collection of centers returned by ApproxKCenter, and note that $\text{cost}(C) = d(c_k, C)$. Prove that

$$\text{cost}(C) \leq 2 \cdot OPT,$$

where OPT is the cost of the optimal solution, i.e.,

$$OPT = \min_{C' = \{c'_0, \ldots, c'_{k-1}\} \subset \mathbb{R}^d} \text{cost}(C').$$

Hint: note that at least two of $\{c_0, \ldots, c_k\}$ must be closest to the same center in the optimal solution, and derive a lower bound on OPT based on this observation.

Solution: Let $\{c'_0, \ldots, c'_{k-1}\}$ be the optimal collection of centers. At least two of $\{c_0, \ldots, c_k\}$ share a closest optimal center. Suppose that $a, b \in \{0, 1, \ldots, k\}$ are such that $a \neq b$ and

$$\min_{i=0,\ldots,k-1} \|c_a - c'_i\|_2 = \|c_a - c'_{i^*}\|_2 = \|c_b - c'_{i^*}\|_2 = \min_{i=0,\ldots,k-1} \|c_b - c'_i\|_2.$$

Suppose that $b > a$, and note that by triangle inequality

$$OPT \geq \max\{\|c_a - c'_{i^*}\|_2, \|c_b - c'_{i^*}\|_2\} \geq \frac{1}{2}\|c_b - c_a\|_2.$$

At the same time

$$\|c_b - c_a\|_2 \geq d(c_b, \{c_0, \ldots, c_{b-1}\}) \geq d(c_k, C) = ALG,$$

where $d(c_k, C) = ALG$ is the cost of the solution obtained by ApproxKCenter. We thus get $ALG \leq 2OPT$, as required.

(This page is intentionally left blank.)