GEORGIA TECH S'22 CS 6550/8803: ADVANCED ALGORITHMS & UNCERTAINTY

Homework 1

Out: *Jan 10* Due: ~~*Jan 28*~~ *Feb 1*

Instructions:

- Upload your solutions to the **non-extra-credit** problems as a single PDF on Gradescope. Please anonymize all your submissions (i.e., do not list your name in the PDF), but if you forget, it's OK.
- You may collaborate with any classmates, textbooks, Internet, etc. Please upload a brief "collaboration statement" listing any collaborators as the last page of your non-extra-credit solutions PDF on Gradescope. But after the collaboration, always **write your solutions individually**.
- If you choose to do extra credit, upload your solution to the extra credits as a single separate PDF file to Gradescope. Please again anonymize your submission.
- For each problem, you should aim to keep your writeup below one page. For some problems, this may be infeasible, and for some problems you may write significantly less than a page. This is not a hard constraint, but part of the assignment is figuring out how to easily convince the grader of correctness, and to do so concisely. "One page" is just a guideline: if your solution is longer because you chose to use figures (or large margins, display math, etc.) that's fine.
- Each problem is worth twenty points (even those with multiple subparts).

Problems:

§1 (Matroids) A *matroid* on $[n]$ elements is a collection of sets that generalized the concept of linear independence for vectors. Specifically, a matroid $\mathcal{I}$ satisfies:

- **Non-trivial:** $\emptyset \in \mathcal{I}$.
- **Downwards-closed:** If $S \in \mathcal{I}$, then $T \in \mathcal{I}$ for all $T \subseteq S$.
- **Augmentation:** If $S, T \in \mathcal{I}$, and $|S| > |T|$, then there exists an $i \in S \setminus T$ such that $T \cup \{i\} \in \mathcal{I}$.[1]

(a) Prove that the following collections are matroids:

i. (Uniform matroid) Sets of size at most k (that is, the elements are $[n]$, and $\mathcal{I} = \{X \subseteq [n] | \ |X| \leq k\}$).

ii. (Graphic matroid) Acyclic subgraphs of any undirected graph $G = (V, E)$ (that is, the elements are E and $\mathcal{I} = \{X \subseteq E | X \text{ contains no cycles}\}$).

[1] Think of this as a generalization of linear independence: if I give you a set S of k linearly independent vectors, and T of $< k$ linearly independent vectors, then there is some vector in S not spanned by T.

iii. (Transversal matroid) Let $G = (L, R, E)$ be a bipartite graph. The elements are L, and $\mathcal{I} = \{X \subseteq L |\ |N(S)| \geq |S|\ \forall S \subseteq X\}$ ($N(S)$ are the neighbors of S: $\{x \in R |\ \exists y \in S, (x, y) \in E\}$). That is, $X \in \mathcal{I}$ if and only if all nodes in X can be simultaneously matched to R.

(b) (Greedy Algorithm) Given weights $w_i \geq 0, i \in [n]$, and some collection of feasible sets $\mathcal{I}$ where $\emptyset \in \mathcal{I}$, your goal is to find the max-weight feasible set: $\arg\max_{S \in \mathcal{I}} \{\sum_{i \in S} w_i\}$. Consider a greedy algorithm that first sorts the elements in decreasing order of w_i (i.e. picks a permutation σ such that $w_{\sigma(i)} \geq w_{\sigma(i+1)}$ for all i), then iteratively does the following (initializing $A = \emptyset$, $i = 1$, go until $i > n$): Check if $A \cup \{\sigma(i)\} \in \mathcal{I}$. If so, add $\sigma(i)$ to A. Update $i := i + 1$. Prove that this greedy algorithm finds the max-weight feasible set no matter what non-negative weights are input *if and only if* $\mathcal{I}$ is a matroid (that is, prove that the algorithm succeeds whenever $\mathcal{I}$ is a matroid. Also, if $\mathcal{I}$ is not a matroid, provide an instance of weights for which the algorithm fails).

§2 (Submodularity) In this problem we study properties of a submodular function $f : 2^{\Omega} \to \mathbb{R}_{\geq 0}$ where Ω is a set of m elements.

(a) Prove that the following two definitions of submodular functions are equivalent:

i. For all sets $A, B \subseteq \Omega$, we have $f(A \cap B) + f(A \cup B) \leq f(A) + f(B)$.

ii. For every set $X \subseteq Y \subseteq \Omega$ and $x \in \Omega \setminus Y$, we have $f(X \cup \{x\}) - f(X) \geq f(Y \cup \{x\}) - f(Y)$.

(b) In the submodular welfare problem, there are n bidders and a set Ω of m items. The value of bidder $i \in \{1, \ldots n\}$ for a subset of items $S \subseteq \Omega$ is given by a monotone (i.e., $f(A) \leq f(B)$ for every $A \subseteq B$) submodular function $f_i(S)$ where $f_i(\emptyset) = 0$. We want to allocate the m items to the n bidders, i.e., find an item partitioning where bidder i gets susbset $S_i \subseteq \Omega$ and $S_i \cap S_j = \emptyset$ for $i \neq j$, and the goal is to maximize the welfare $\sum_{i \in \{1,\ldots,n\}} f_i(S_i)$. Show that the greedy algorithm which considers the items one-by-one in an arbitrary order and allocates the next item j to the bidder which has the highest marginal value (i.e., which maximizes $f_i(S \cup j) - f_i(S)$ where S is the currently allocated set of items to bidder i), gives a 2-approximation algorithm for the submodular welfare problem.

Hint: First prove that a submodular function remains submodular even if you "contract" a set, i.e., $f_S(A) := f(S \cup A) - f(S)$ is submodular on $\Omega \setminus S$.

§3 (a) (Concentration) Suppose there are n coupons $\{1, \ldots, n\}$. Each step, you draw a uniformly random coupon independently with replacement (i.e., you may redraw coupons), and you repeat this until you have drawn *all n coupons* at least once. Prove that with probability at least $1 - 1/n$, the process takes $O(n \log n)$ steps.

(b) Consider the following process for matching n jobs to n processors. In each step, every job picks a processor at random. The jobs that have no contention on the processors they picked get executed, and all the other jobs *back off* and then try again. Jobs only take one round of time to execute, so in every round all the processors are available. Show that all the jobs finish executing w.h.p. after $O(\text{loglog}\, n)$ steps.

Hint: Try to argue that if there are currently ϵn unmatched jobs, then in the next round roughly $\epsilon^2 n$ jobs remain unmatched.

§4 (Basic Feasibility & Iterated Rounding) Suppose you are given a matrix $A \in \{0,1\}^{n \times n}$ such that each column of A has at most s ones, i.e., column sparsity is at most s. We will analyze the following polynomial time algorithm to find a coloring $\vec{\varepsilon} \in \{-1,+1\}^n$ such that $\|A\vec{\varepsilon}\|_\infty = \|\sum_t \vec{a}_t \varepsilon_t\|_\infty = O(s)$, where $\vec{a}_t$ is the t-th column of A.

The algorithm will iteratively color more and more columns of A. Initially, all columns are "uncolored" and a column $\vec{a}_t$ gets "colored" when ε_t is defined.

- In any iteration, if the number of remaining uncolored columns is at most $2s$, color them arbitrarily and halt.
- Otherwise, we solve an LP with fractional variables denoting the fractional colors of the uncolored columns. For every row i that contains more than $2s$ uncolored ones, put an LP constraint that its fractional discrepancy is zero, i.e., $\sum_t a_t(i) Y_t = 0$ where Y_t denotes LP variable $x_t \in [-1,1]$ for uncolored columns t, and Y_t denotes constant ε_t for colored columns t.
- Find a basic solution x^* of the above LP, and for any uncolored column t that gets $x^*_t \in \{-1,+1\}$ in this basic solution, we permanently set its ε_t to that color.
- Repeat.

(a) Show that in any iteration, the number of row constraints that we put is at most half the number of currently uncolored columns. Deduce that a basic solution will integrally color at least half the columns in each iteration.

(b) Show that this algorithm obtains a solution with discrepancy $O(s)$. This means, e.g., that if each column has $O(1)$ non-zero entries then the discrepancy is $O(1)$.

Remark: There is nothing special about $A \in \{0,1\}^{n \times n}$ vs. $A \in [0,1]^{n \times T}$, i.e., having $T \gg n$ columns and real entries. We can easily extend the above result using the idea of finding a basic solution in the beginning with at most n uncolored columns.

§5 (a) (Scheduling) Suppose we are given n machines and m jobs where the jth job takes time $p_{ji} \in \mathbb{R}_{\geq 0}$ to be executed on machine i. The goal of this problem is to find an assignment $\sigma : [m] \to [n]$ of all the jobs onto the n machines to minimize the maximum load, i.e., $\min_\sigma \max_{i \in [n]} \sum_{j:\sigma(j)=i} p_{ji}$.

i. Consider the special case of the problem where job sizes don't depend on the machines, i.e., $p_{ji} = p_j$ for all j (this is called the "identical machines" scheduling). Use NP-Hardness of Subset-Sum[2] to prove that this problem is NP-Hard even for $n = 2$ machines.

ii. Consider the greedy algorithm for scheduling: consider the jobs in an arbitrary order and schedule the next job on the machine that has the least load currently, breaking ties arbitrarily. Prove that this algorithm gives an $O(1)$ approximation on identical machines.

[2]Given a set of m integers, the goal is to partition them into two parts such that the sum of the numbers in each part is equal (which equals half of the sum of all numbers).

iii. Prove that the LP relaxation below is valid for the problem, i.e., it gives a lower bound on the optimum (think x_{ji} means job j is assigned to machine i). Next, give a family of identical machines problem instances where for every n the integrality gap (ratio of optimum to LP value) is $\Omega(n)$:

$$\begin{array}{lll} \min & L & \\ \text{s.t.} & \sum_i x_{ji} \geq 1 & \forall j \in [m] \\ & \sum_j x_{ji} p_{ji} \leq L & \forall i \in [n] \\ & x_{ji} \geq 0 & \forall j \in [m], i \in [n] \end{array}$$

Remark: In the next assignment we will use a modification of this LP to design $O(1)$ approximation algorithms for general (non-identical) scheduling.

(b) (Separation Oracle) Describe separation oracles for the following convex sets. Your oracles should run in linear time, assuming the given oracles run in linear time (so you can make a constant number of black-box calls to the given oracles).

i. The ℓ_1 ball, $\{x : \|x\|_1 \leq 1\}$. Recall that $\|x\|_1 = \sum_i |x_i|$.

ii. Any convex set A that we have a projection oracle for. I.e. we have an oracle to compute $\arg\min_{x \in A} \|x - y\|_2$ for any y.

iii. The ϵ-neighborhood, E, of any convex set A: $E = \{x : \exists y \in A \text{ with } \|x - y\|_2 \leq \epsilon\}$, given a projection oracle for A.

Extra Credit:

§6 (extra credit) In a *combinatorial auction* there are n bidders and m items. Bidder i has a monotone valuation $v_i(\cdot)$ where $v_i(S)$ is their value for item set S (and $v_i(S \cup T) \geq v_i(S)$ for all S, T). A *Walrasian Equilibrium* is a price for each item $\vec{p}$ such that:

- Each buyer i selects to purchase a set $B_i \in \arg\max_S \{v_i(S) - \sum_{j \in S} p_j\}$.
- The sets B_i are disjoint, and $\cup_i B_i = [m]$.

Prove that a Walrasian equilibrium exists for $v_1, \ldots, v_n$ if and only if the optimum of the LP relaxation below (called the *configuration LP*) is achieved at an integral point (i.e. where each $x_{i,S} \in \{0, 1\}$). Hint: use strong duality!

$$\begin{aligned} \max & \sum_i \sum_S v_i(S) \cdot x_{i,S} \\ \forall i, & \sum_S x_{i,S} = 1 \\ \forall j, & \sum_{S \ni j} \sum_i x_{i,S} \leq 1 \\ \forall i, \forall S, & \ x_{i,S} \geq 0 \end{aligned}$$

Also, find an example of 2 items & 2 bidders where Walrasian equilib. doesn't exist.

§7 (Open Problem) Can you extend the discrepancy result in Problem 4 to every prefix, i.e., find a coloring $\vec{\varepsilon}$ such that $\max_t \|\sum_{i \leq t} \vec{a}_i \varepsilon_i\|_\infty = O(s)$? Can you prove this result for some larger function $f(s)$ that does not depend on n and T (like $f(s) = 2^s$)?