COS 330: Great Ideas in Theoretical Computer Science

Fall 2025

Problem Set 5

Module: Optimization and Approximation

Below is a reminder of key aspects of the PSet:

- The only goal of this PSet is to help you develop your problem-solving skills in preparation for the exams. Your performance on this PSet will not directly contribute to your grade, but will indirectly improve your ability to do well on the exams.
- Because your performance does not directly impact your grade, you may use any resources you like (collaboration, AI, etc.) to help you complete the PSet.
- We strongly suggest taking a serious stab at the PSet alone, to help self-evaluate where you're at. But, we also suggest collaborating with friends, visiting office hours, asking on Ed, and/or using AI tools to help when stuck. Even when able to complete the entire PSet on your own, you may still find any of these methods useful to discuss the PSet afterwards.
- Throughout the PSet, we've included some general tips to help put these into broader context. Exams will not have these, and future PSets may have fewer.
- We strongly suggest treating this like any other PSet, and writing up your solutions as if you were handing them in for a grade. At minimum, we very strongly suggest writing up sufficiently many solutions to discuss with your Coach.

Aligning Expectations

Recall that the 🗎 symbol implies that the following problem is an "exam-style" problem
We highly recommend that you write-up a full solution to this problem.

Problem 1 : Greedy Set Cover 🗎

Recall the Set Cover problem from PSet 4[1]. You are given a universe of n elements $U = \{1, 2, \ldots, n\}$ and a collection of m subsets $S_1, S_2, \ldots, S_m$ of U. A set cover of U is a collection of these subsets whose union is equal to U, or formally, a set $I \subseteq \{1, 2, \ldots, m\}$ such that $\bigcup_{i \in I} S_i = U$. The goal is to find a set cover of minimum size.

Consider the following greedy algorithm:

[1] You don't need to have solved the problem from PSet 4 to be able to solve this one.

- While there are uncovered elements remaining:
  - Select the set covering the largest number of uncovered elements (breaking ties arbitrarily).
  - Mark all elements in this set as covered.

You will analyze the approximation ratio of this greedy algorithm using duality.
Consider the LP relaxation of the Set Cover problem:

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} x_i \\ \text{s.t.} \quad & \sum_{i:\, v \in S_i} x_i \;\geq\; 1 \qquad \forall v \in U, \\ & x_i \;\geq\; 0 \qquad\quad \forall i \in \{1, \ldots, m\}. \end{aligned}$$

Let OPT denote the size of an optimal (minimum-size) set cover, OPT_{LP} denote the optimal value of the above LP relaxation, and GREEDY denote the size of the set cover returned by the greedy algorithm.

(a) Write the dual program of the LP relaxation above using variables y_v for each element $v \in U$[2].

Now, let $v_1, v_2, \ldots, v_n$ be the order in which the elements of U are covered by the greedy algorithm. Suppose j is the iteration in which element v_j gets covered for the first time. Let c_j be the number of uncovered elements covered by the set chosen in iteration j. So elements $v_1, v_2, \ldots, v_{j-1}$ have already been covered in previous iterations, and elements $v_j, v_{j+1}, \ldots, v_n$ were uncovered at the start of iteration j.

(b) Consider the following dual solution for the iteration j where v_j was covered:

$$y_{v_1} = \cdots = y_{v_{j-1}} = 0, \qquad y_{v_j} = \cdots = y_{v_n} = \frac{1}{c_j}.$$

Show that this assignment is feasible for the dual program. That is, show that for every set S_i, we have $\sum_{v \in S_i} y_v \leq 1$.
(Hint: Consider how many of the elements $v_j, \ldots, v_n$ can be in S_i when iteration j begins.)

(c) Using weak duality and the dual solution from part (b), show that for the iteration j where v_j was covered:

$$\text{OPT} \geq \frac{n - j + 1}{c_j}.$$

(d) Prove that

$$\text{GREEDY} = \sum_{j=1}^{n} \frac{1}{c_j}.$$

[2]We wouldn't ask you to find a dual of a program in an exam - we'd just give you the dual - since it's a mostly mechanical process. But since this is your first time seeing duality, going through the process is a useful exercise. Because you will need this for the next few questions, feel free to check if you got the right answer by asking an AI tool or asking on Ed.

(e) Using parts (c) and (d), prove that the greedy algorithm achieves an H_n-approximation, where $H_n = \sum_{i=1}^{n} \frac{1}{i}$ is the n-th harmonic number.

Problem 2 : Maximum Acyclic Subgraph

In the Maximum Acyclic Subgraph problem, we are given a directed graph $G = (V, E)$. A subset $E' \subseteq E$ is called acyclic if the graph (V, E') contains no directed cycles. The goal is to find an acyclic subset of maximum size.

Let OPT denote the size of a maximum acyclic subset of edges.

(a) Prove that for any directed graph $G = (V, E)$, we have $\text{OPT} \geq |E|/2$.

(b) Design a 2-approximation algorithm for the Maximum Acyclic Subgraph problem and prove that it achieves a 2-approximation ratio.

(Hint: Use the intuition from the proof of part (a).)

Problem 3 : Approximating MAX-2-SAT

In the MAX-2-SAT problem, we are given m clauses $C_1, \ldots, C_m$ over n Boolean variables $x_1, \ldots, x_n$, where each clause contains at most 2 literals. A literal is either a variable x_i or its negation $\neg x_i$. The goal is to find an assignment to the variables that maximizes the number of satisfied clauses.

For a clause C_j, let $\ell(C_j)$ denote the set of literals in C_j (so $|\ell(C_j)| \in \{1, 2\}$). A clause is satisfied by an assignment if at least one of its literals evaluates to true.

Consider the following LP relaxation for MAX-2-SAT:

$$\begin{aligned} \max \quad & \sum_{j=1}^{m} z_{C_j} \\ \text{s.t.} \quad & z_{C_j} \leq \sum_{\ell \in \ell(C_j)} y(\ell) && \forall j \in \{1, \ldots, m\}, \\ & 0 \leq y_i \leq 1 && \forall i \in \{1, \ldots, n\}, \\ & 0 \leq z_{C_j} \leq 1 && \forall j \in \{1, \ldots, m\}, \end{aligned} \tag{LP}$$

where $y(\ell) = y_i$ if $\ell = x_i$ and $y(\ell) = 1 - y_i$ if $\ell = \neg x_i$.

In this formulation, y_i represents the "fractional truth value" of variable x_i. The variable z_{C_j} represents the "fractional satisfaction" of clause C_j.

Let OPT denote the maximum number of clauses that can be satisfied by any assignment, and let OPT_{LP} denote the optimal value of the LP relaxation above.

Consider the following randomized rounding algorithm: Solve (LP) to obtain an optimal solution (y^*, z^*). Independently set each variable x_i to true with probability y_i^* and to false with probability $1 - y_i^*$. In the following parts you will analyze the expected performance of this algorithm and show that it is a 3/4-approximation algorithm in expectation.

(a) Consider a clause C_j with exactly 1 literal. Show that the probability that the randomized rounding algorithm satisfies C_j is at least $z_{C_j}^*$.

(b) Consider a clause C_j with exactly 2 literals ℓ_1 and ℓ_2. Let $p = y(\ell_1)^*$ and $q = y(\ell_2)^*$ denote the fractional truth values of these literals in the optimal LP solution. Show that:

$$\Pr[C_j \text{ is satisfied}] \geq p + q - (p+q)^2/4.$$

(Hint: The following inequality might be useful[3]: for any $p, q \in [0,1]$, we have $pq \leq \frac{(p+q)^2}{4}$.)

(c) Show that in an optimal LP solution, for a clause C_j with two literals, we have $z^*_{C_j} = \min\{1, p+q\}$ where $p = y(\ell_1)^*$ and $q = y(\ell_2)^*$.

(d) Let $s = p + q$. Suppose $z^*_{C_j} = s$ (i.e., $s \leq 1$). Using part (b), show that:

$$\Pr[C_j \text{ is satisfied}] \geq \frac{3}{4} z^*_{C_j}.$$

(e) Let $s = p + q$. Suppose $z^*_{C_j} = 1$ (i.e., $s > 1$). Using part (b), show that:

$$\Pr[C_j \text{ is satisfied}] \geq \frac{3}{4} z^*_{C_j}.$$

(f) Using parts (a) through (e), prove that the randomized rounding algorithm is a 3/4-approximation algorithm for MAX-2-SAT in expectation.

[3]This is a special case of a really popular inequality known as AM-GM (arithmetic mean-geometric mean) inequality, which says that the geometric mean is always at most the arithmetic mean. For k non-negative numbers $a_1, \ldots, a_k$, we have $\sqrt[k]{a_1 \cdots a_k} \leq \frac{a_1 + \cdots + a_k}{k}$, with equality when all the a_i are equal.