CS 6150: HW5 – Randomized algorithms, optimization

Submission date: Monday, Nov 29, 2021 (11:59 PM)

points. Unless otherwise specified, complete and reasoned arguments will be expected for all answers.

Question	Points	Score
Trade-offs in sampling	12	
Satisfying ordering constraints	10	
Writing constraints	6	
LP Basics	10	
Identifying Corners	6	
Checking feasibility vs optimization	6	
Total:	50	

Instructions. For all problems in which you are asked to develop an algorithm, write down the pseudocode, along with a rough argument for correctness and an analysis of the running time (unless specified otherwise). Failure to do this may result in a penalty. If you are unsure how much detail to provide, please contact the instructors on Piazza.

Question 1: Trade-offs in sampling ..... **[12]**

For this problem, you need to run some basic experiments and write down the results you obtained. You **do not need to submit your code**, but if you prefer, you may add a publicly accessible link to the code (e.g., on github).

Suppose we have a population of size 10 million, and suppose 52% of them vote for A and 48% of them vote for B.

(a) [**4**] Randomly pick samples of size (a) 20, (b) 100, (c) 400, and evaluate the probability that A is majority even in your sample (by running the experiment say 100 times and taking the count of times A is the majority in your sample). Write down the values you observe for these probabilities in the cases (a-c).

Solution.

P(A-Majority) for Sample Size 20: 0.51

P(A-Majority) for Sample Size 100: 0.61

P(A-Majority) for Sample Size 400: 0.71

Thus, as the sample size increases, the chances of A getting majority of votes increases as well.

Rubric:

- 4 points for figuring out the probabilities for sample size 20, 100, 400.

(b) [**4**] What is the size of the sample you need for the probability above to become 0.9? (Your answer can be within 20% of the 'best' value.)

Solution.

Sample Size for P(A-Majority) to be 0.9: 1000

Rubric:

- 4 points for defining sample size.

(c) [**4**] Suppose the population is more biased —55% of them vote for A and 45% of them vote for B— and re-solve part (b).

Solution.

Sample Size for P(A-Majority) to be 0.9: 200

Here, with less no. of samples, we get P(A-Majority) = 0.9; Since, the population is biased and there is a larger vote share difference between A and B.

Rubric:

- 4 points for defining sample size.

Question 2: Satisfying ordering constraints ..... **[10]**

This problem is meant to illustrate a rather cool paradigm for solving problems, which is picking a *random solution*, and understanding how well it does. Indeed, there are many problems ("3-SAT", a version of boolean satisfiability, being the chief of them) where doing better than a random solution is actually NP-hard!

Suppose we have n elements, labeled $1, 2, \ldots, n$. Our goal is to find an ordering of these elements that satisfies as many "constraints" as possible, of the kind described below. We are given m constraints, where each constraint is given by a triple (a, b, c), where $a, b, c \in \{1, 2, \ldots, n\}$. The constraint is said to be *satisfied* if in the ordering we output, a does **not** lie "between" b and c (but the order between b and c doesn't matter). For example, if $n = 4$ and we consider the ordering 2431, then the constraint $(1, 4, 3)$ is satisfied, but $(3, 1, 2)$ is not.

Given the constraints, the goal is to find an ordering that satisfies as many constraints as possible (for simplicity, assume in what follows that m **is a multiple of** 3). For large m, n, this problem becomes very difficult.

(a) **[6]** As a baseline, let us consider a *uniformly random* ordering. What is the expected number of constraints that are satisfied by this ordering? [*Hint:* define appropriate random variables whose sum is the quantity of interest, and apply the linearity of expectation.]

Solution Let Y_i be a random variable with value 0 if constraint i is not satisfied, and value 1 if constraint i is satisfied. There are multiple arguments to find $\mathbf{E}[Y_i]$:

- For any three elements a, b, c there are 6 possible orderings of these elements in any permutation:

$$\ldots a \ldots b \ldots c \ldots$$
$$\ldots a \ldots c \ldots b \ldots$$
$$\ldots b \ldots a \ldots c \ldots$$
$$\ldots b \ldots c \ldots a \ldots$$
$$\ldots c \ldots a \ldots b \ldots$$
$$\ldots c \ldots b \ldots a \ldots$$

  . Since the permutation is uniformly random, the positions of a, b, c are all random, and each ordering is equally likely. Orderings $\ldots b \ldots a \ldots c \ldots$ and $\ldots c \ldots a \ldots b \ldots$ do not satisfy constraints, and since each ordering is equally likely, the probability of satisfying a constraint is $Pr(Y_i = 1) = \frac{2}{3}$
  By conditional probability, $\mathbf{E}[Y_i] = (1)Pr(Y_i = 1) + (0)Pr(Y_i = 0) = \frac{2}{3}$.
- Let N_i be a random variable, representing the number of elements between b and c for constraint i. By the law of conditional expectation, we can calculate the expected number of elements between b and c in any ordering.

$$\mathbf{E}[N_i] = Pr(N = 0) \cdot 0 + Pr(N = 1) \cdot 1 + \cdots + Pr(N = n-2) \cdot (n-2)$$
$$\mathbf{E}[N_i] = \sum_{i=0}^{n-2} \frac{2(n-1-i)}{n(n-1)} i = \frac{2}{n(n-1)} \sum_{i=0}^{n-2} ni - i - i^2$$
$$\mathbf{E}[N_i] = \frac{2}{n(n-1)} \left(\frac{(n-2)(n-1)n}{2} - \frac{(n-2)(n-1)}{2} - \frac{(n-2)(n-1)(2n-3)}{6} \right)$$
$$\mathbf{E}[N_i] = \frac{n-2}{3}$$

  A constraint is satisfied if a is not between b and c. With $n-2$ slots in our ordering that are not b and c, we know the probability that a is between b and c is $\frac{\mathbf{E}[N_i]}{n-2} = \frac{n-2}{3}\frac{1}{n-2} = \frac{2}{3}$. By conditional probability, $\mathbf{E}[Y_i] = \frac{2}{3}$.

Let X be the random variable which is the number of constraints satisfied by a random ordering. Then $X = \sum_i^m Y_i$.

$$\begin{aligned}\mathbf{E}[X] &= \mathbf{E}[\sum_i^m Y_i] \\ &= \sum_i^m \mathbf{E}[Y_i] = m\frac{2}{3}\end{aligned}$$

Rubric:

- 4 points for an argument showing $\mathbf{E}[Y_i] = \frac{2}{3}$
- 2 points for solving for $\mathbf{E}[X] = m\frac{2}{3}$

(b) [**4**] Let X be the random variable which is the number of constraints satisfied by a random ordering, and let E denote its expectation (which we computed in part (a)). Now, Markov's inequality tells us, for example, that $\Pr[X \geq 2E] \leq 1/2$. But it does not directly imply that $\Pr[X \geq E]$ is "large enough" (which we need if we want to say that generating a few random orderings and picking the best one leads to many constraints being satisfied with high probability). Use the definition of X above to conclude that $\Pr[X \geq E] \geq 1/m$.

[*Hint:* X is a non-negative integer!]

Solution: Let Y denote the random variable which is the number of unsatisfied constraints for a random ordering. Following the logic of part (a), $\mathbf{E}[Y] = \frac{m}{3}$, and $Pr(Y) = \frac{m}{3}$. Since X and Y are complementary events, $Pr[X \geq \frac{2m}{3}] = 1 - Pr[Y > \frac{m}{3}]$. Since m is divisible by 3, $(Y > \frac{m}{3})$ is equivalent to $(Y \geq \frac{m}{3} + 1)$.

We apply Markov's inequality to bound this probability:

$$Pr[Y \geq \frac{m}{3} + 1] = Pr[Y \geq \frac{m+3}{3}] \leq \frac{m}{m+3}$$

Now algebraic manipulation provides a bound:

$$\begin{aligned}Pr[X] &= 1 - Pr[Y \geq \frac{m+3}{3}] \\ &\geq 1 - \frac{m}{m+3} \\ &\geq \frac{3}{m+3}\end{aligned}$$

Since $\frac{3}{m+3} > \frac{1}{m}$ for all integers $m > 1$, $Pr[X \geq \frac{2m}{3}] \geq \frac{1}{m}$.

Rubric:

- 4 points for logical argument, which may be as follows:
  - 1 point for using complementary event Y
  - 2 point for showing $(Y > \frac{m}{3})$ is equivalent to $(Y \geq \frac{m}{3} + 1)$
  - 1 point for showing lower bound through algebraic manipulation

Question 3: Writing constraints ..... **[6]**

We will see how writing down constraints can be tricky when formulating problems as optimization.

Recall the traveling salesman problem (TSP): we are given a directed graph $G = (V, E)$ with edge lengths $w(e)$. The goal is to find a *directed cycle* that (a) visits every vertex exactly once, and (b) minimizes the total length (sum of lengths of the edges of the cycle).

Consider the following optimization formulation. We have variables $x_{uv} \in \{0, 1\}$, one for each edge (u, v) (remember that the graph is directed). The objective is to minimize $\sum_{(u,v)\in E} w(uv)x_{uv}$. The constraints are as follows: let $N_+(u)$ and $N_-(u)$ denote the out-neighbors and in-neighbors of u; then we consider the constraints:

$$\forall\ u, \quad \sum_{v\in N_+(u)} x_{uv} = 1,$$
$$\forall\ u, \quad \sum_{w\in N_-(u)} x_{wu} = 1.$$

(Intuitively, the constraints say that exactly one incoming edge and one outgoing edge must be chosen — as in a directed cycle.) Does an optimal solution to this optimization problem *always yield* an optimal solution to TSP? Provide a proof or a counterexample with a full explanation.

[*Hint:* consider a feasible solution to the optimization problem and focus on the edges with $x_{uv} = 1$. Do they have to form a *single* cycle?]

Solution.

This approach will not always yield an optimal solution to TSP because there might be more than a single cycle (disjoint cycles are possible). For example, if we have a graph with 10 vertices and weights: $w(v_1 \to v_2) = w(v_2 \to v_3) = w(v_3 \to v_4) = w(v_4 \to v_1) = 10$ and $w(v_5 \to v_6) = w(v_6 \to v_7) = w(v_7 \to v_8) = w(v_8 \to v_5) = 10$. Other edges (e.g., the edges between vertices $v_2 \to v_5$ and $v_8 \to v_3$) have weights much larger than 10 (e.g., 500). By running the optimization we get two disjoint cycles: v_1, v_2, v_3 and v_4 is one cycle and the other cycle is v_5, v_6, v_7 and v_8. This is a valid optimal solution because every vertex in the graph has exactly one incoming edge and one outgoing edge, and it minimizes the total length to 80. But we do not have a single cycle covering all vertices (this could happen when the graph is not fully connected as well); hence, an optimal solution to the given optimization problem does not always yield an optimal solution to the TSP.

Rubric:

- 6 points figuring the approach could yield disjoints cycles

Question 4: LP Basics . . . . . **[10]**

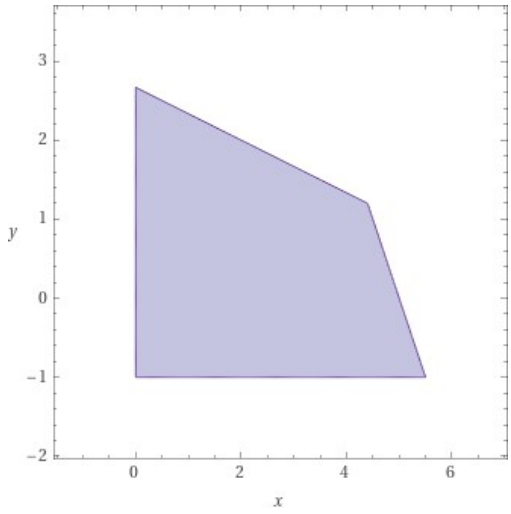
Consider the following two-variable linear program. The variables are $x, y \in \mathbb{R}$. And the constraints are:

$$\begin{aligned} 2x + y &\le 10, \\ x &\ge 0, \\ y &\ge -1, \\ x + 3y &\le 8. \end{aligned}$$

(a) [**4**] Plot the feasible region for the linear program. (Diagram can be drawn approximately to scale + scanned).

Solution.

Rubric:



- x points for TODO
- x points for TODO

(b) [**2**] Suppose the objective function is to *maximize* $x + y$. Find the maximum value and the point at which this is attained.

Solution. We can see that given the objective, *maximize* $x + y$, the corner point that does this is, $(22/5, 6/5)$, which leads to $x + y = 28/5$.

Rubric:

- 1 point for finding the maximum value
- 1 point for finding the corner point

(c) [**4**] Say the maximum value found in part (b) is C. Then could you have concluded that $x+y \leq C$ by just "looking at" the equations? [*Hint:* does adding equations, possibly with constants, yield a bound?]

Solution. We can use the inequalities (α) $2x + y \leq 10$ and (β) $x + 3y \leq 8$ and compute $2\alpha + \beta$ to get $5x + 5y \leq 28$, which implies $x + y \leq 28/5$.

Rubric:

- 2 points for adding equations with constant factors
- 2 points for computing the optimal value

Question 5: Identifying Corners . . . . . **[6]**

Consider the following linear program, with variables $x_1, x_2, \ldots, x_n \in \mathbb{R}$. Suppose that the constraints are:

$$0 \leq x_i \leq 1 \text{ for all } i,$$
$$x_1 + x_2 + \cdots + x_n \leq k,$$

where k is some integer between 1 and n. Consider any point $(a_1, a_2, \ldots, a_n)$ where $a_i \in \{0, 1\}$, and exactly k of the $\{a_i\}$ are 1. Prove that any such point is (a) a feasible point for the LP, and (b) a corner point of the polytope defining the feasible set.

[*Hint:* To prove that a point is a corner point, we use the definition we mentioned in class: a feasible point x is a corner point if and only if for *all* nonzero vectors z, either $x + z$ or $x - z$ is infeasible.]

[*Remark:* It turns out that these are the only corner points of the polytope, and so it is an example of a polytope defined by $2n+1$ constraints and having $\binom{n}{k}$ corner points.]

Solution.

(a) By construction only k of the a_i are equal to 1 and all other a_i are zero. For a to be a feasible point then it must satisfy LP conditions (1) and (2). Assume the first k elements of a (i.e. $(a_0, \cdots, a_k)$ are equal to 1) and the subsequent a_i are 0. We then have that $\sum_{i=0}^{n} a_i = \sum_{i=0}^{k} 1 + \sum_{i=k+1}^{n} 0 = k$ showing a satisfies LP condition (2). By construction it is easy to see that a must also satisfy LP condition (1).

Rubric:

- 1 point for proving the a_is satisfy the 1st constraint.
- 1 point for proving the a_is satisfy the 2nd constraint.

(b) We will show a is necessarily a corner point and at most only one of the perturbations $a^+ = a + z$ and $a^- = a - z$ are in the feasible set. We can see this by contradiction. Assume that a is in the feasible set, that a is not a corner point (but an interior point) and both of the a^+ or a^- are in the feasible set. Consider the perturbed point in the feasible set a^+. With a in the interior (because not a corner by assumption) then by being in the feasible set a satisfies LP constraint (1) and (2). By (1) and the construction of a there are k of the a_i of value 1 with all other a_i of value 0. For our assumption to be true it must be that $a^+ = a + z$ and $a^- = a - z$ are both in the feasible set. For this to be true then for all $z_i \in (z_1, \cdots, z_n)$ we must have $0 \leq a_i^{\pm} \leq 1$ for all i. Consider some z with a non-zero element z_j . Now considering the perturbation of a. Since only k of a_i are non-zero and of value 1 then there is some $a_j \in (a_1, \cdots, a_n)$ that for either a^+ after perturbation is $a_j^+ = a_j + (\pm z_j) = 1 \pm z_j$ or $a_j^- = a_j - (\pm z_j) = 0 \pm z_j$ since $a_j \in \{0, 1\}$. We can see then either a_j^+ or a_j^- are greater than 1 or less than 0 which is a violation of LP constraint (1). Therefore for any z a perturbation of a would break LP condition (1) contradicting our assumption both a^+ and a^- are in the feasible set and a is in the interior. It must be that a is then a corner point. From this we can also see that for any interior point x there must exists some z such that $x + z$ and $x - z$ are in the feasible region, similarly, if x is a corner point at most one of these perturbations are feasible.

Rubric:

- 2 points for considering the 2 perturbation points a^+, a^-
- 2 points for proving at most only of the perturbations can lie in the feasible set.

Question 6: Checking feasibility vs optimization ..... **[6]**

Some of the algorithms for linear programming (e.g. simplex) start off with one of the corner points of the feasible set. This turns out to be tricky in general. In this problem, we will see that **in general**, finding one feasible point is as difficult as actually performing the optimization!

Consider the following linear program (in n variables $x_1, \ldots, x_n$, represented by the vector x):

$$\begin{gathered}\text{minimize } c^T x \text{ subject to} \\ a_1^T x \geq b_1 \\ a_2^T x \geq b_2 \\ \cdots \\ a_m^T x \geq b_m.\end{gathered}$$

Suppose you know that the optimum value (i.e. the minimum of $c^T x$ over the feasible set) lies in the interval $[-M, M]$ for some real number M (this is typically possible in practice). Suppose also that you have an **oracle** that can take any linear program and say whether it is feasible or not. Prove that using $O(\log(M/\epsilon))$ calls to the oracle, one can determine the optimum value of the LP above up to an error of $\pm\epsilon$, for any given accuracy $\epsilon > 0$. [*Hint:* can you write a new LP that is feasible only if the LP above has optimum value $\leq z$, for some z?]

Solution. Let $\hat{C}$ be the set of constraints, of the original LP. Now define $\tilde{C}_z = \hat{C} \cup \{c^\top x \leq z\}$.

Algorithm 1 Approximate optimum

Input: Set of constraints $\hat{C}$ and the objective function definition $c^\top x$ and ϵ.
Output: Approximate optimal OPT_{apx}.

```
Set ub = M and lb = −M
while ub − lb > 2ε do
    z = (ub + lb)/2
    Check compatibility of the LP with the set of constraints C̃_z.
    if The new LP is compatible then
        set ub = z
    else
        set lb = z
    end if
end while
return z = (ub + lb)/2
```

Note that if the modified LP is feasible according to the oracle, then $\exists x'$ such that $c^\top x' \leq z$ and the other constraints also hold. Since the objective is a minimization, we can see that if there is such an x, then the $OPT = \min_x c^\top x \leq c^\top x' \leq z$ which means the optimum has to lie within the interval of the lower bound lb to z. Otherwise, if there is no such x', we can see that the $z \leq OPT \leq ub$ where ub is the current upperbound. We can see that in either case, the algorithm chooses the bounds such that $lb \leq OPT \leq ub$. At the termination we can see that $ub - lb \leq 2\epsilon$ and therefore $|OPT - OPT_{apx}| \leq OPT_{apx} - lb = (ub + lb)/2 - lb \leq \epsilon$. At the same time we can follow a line of arguments similar to binary search. Since we are reducing the search space by $1/2$ each time the number of steps till we get $ub - lb \leq 2\epsilon$, we can see that, given the number of steps k, we can infer that $(M - (-M))/2^{k-1} \geq 2\epsilon$ and $(M - (-M))/2^k \leq 2\epsilon$. We can solve this and get $k = \Theta(\log(M/\epsilon))$

Rubric:

- 2 points for correctness
- 2 points for running time
- 2 points for accuracy