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).

Answer:

(a) I pick 20 samples from 10 million, then run 100 times, use Java random system this is what I get

(10, 9, 11, 12, 11, 11, 4, 13, 12, 12, 10, 10, 12, 8, 11, 6, 11, 7, 13, 12, 11, 13, 8, 11, 13, 9, 10, 11, 7, 9, 11, 10, 10, 9, 11, 11, 12, 12, 11, 15, 11, 12, 10, 10, 14, 13, 12, 11, 9, 15, 9, 10, 9, 9, 8, 8, 10, 12, 6, 11, 11, 8, 9, 11, 10, 11, 15, 9, 14, 9, 12, 11, 11, 11, 8, 12, 10, 12, 10, 9, 13, 10, 10, 12, 11, 12, 8, 10, 10, 12, 15, 8, 12, 11, 12, 12, 10, 12, 13, 10)

Each of this number is "how many A in 20 samples", and we can see from my 100 try times, there are 56 times that A is the majority (A appears times > 10 in samples).

(b) This time, pick 100 sample in 10 million, also Java random system

(54, 53, 55, 52, 48, 47, 45, 54, 49, 51, 50, 54, 53, 41, 46, 45, 64, 48, 51, 56, 57, 58, 60, 48, 52, 49, 50, 43, 54, 56, 46, 48, 56, 55, 49, 50, 53, 53, 46, 55, 45, 54, 51, 55, 55, 59, 57, 52, 56, 55, 49, 47, 59, 50, 52, 54, 55, 46, 50, 60, 58, 48, 52, 52, 60, 61, 63, 54, 58, 54, 47, 48, 56, 59, 51, 54, 52, 53, 49, 50, 52, 54, 54, 51, 50, 51, 42, 40, 65, 52, 53, 52, 51, 52, 46, 52, 49, 55, 47, 47)

Also, each of this number is "how many A in 100 samples, and 100 tries, we get 64 times that A is the majority (A appears times > 50 in 100 samples).

(c) This time, pick 400 sample in 10 million, also Java random system

(201, 231, 217, 212, 206, 227, 203, 221, 203, 210, 206, 218, 224, 210, 229, 204, 212, 190, 205, 211, 208, 220, 219, 186, 210, 209, 197, 211, 191, 198, 211, 226, 213, 220, 215, 201, 206, 226, 198, 207, 214, 209, 192, 209, 217, 211, 206, 215, 208, 213, 207, 210, 177, 213, 216, 202, 199, 211, 208, 222, 201, 202, 197, 210, 198, 203, 212, 214, 222, 198, 212, 204, 211, 203, 211, 205, 217, 213, 187, 214, 209, 208, 191, 215, 202, 217, 211, 224, 203, 192, 211, 202, 200, 195, 212, 197, 207, 199, 210, 213)

Also, each of this number is "how many A in 400 samples, and 100 tries, we get 81 times that A is the majority (A appears times > 200 in 100 samples).

(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.)

Answer:
I notice with big samples we have, the more probability can be above 0.9, so I tried several sample size, for example, I try sample size 800 with 10 times run, I get (89,89,88,91,83,82,80,83,86,87), those numbers are " how many times A is majority experiment 100 times". So clearly, within 10 tries, only 1 time, the probability is above 0.9, which is 91/100.

Let us keeping trying, I set up sample size to be 1000, we get (91,88,89,91,89,89,89,87,84,90), we can see, there are 3 times that probability is above 0.9, let us keep try.

I set up sample size to be 1500, I get (92,97,96,99,90,96,92,95,94,99), so from my experience, I see that when sample size become 1500, there are 10 times that probability is above 0.9 with 10 experiences, so I can see, 1500 sample size is pretty great.

(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).
Answer:
Let us reset A and B, others remain the same, let us adjust sample size to be 800 first, I get (100,100,99,99,99,100,100,99,99,99), this is pretty good sample size, but I want to get it smaller sample size.

Let us set sample size to 500, I get (100,100,98,100,100,99,100,97,98,100), this is also good, but let me make it smaller again.

Let us set sample size to 250, I get (89,95,97,95,97,92,94,96,94,93), I notice there is 1 time below 0.9, let me increase sample size a little bit

Let us set sample size to 280, I get (93,94,95,97,94,96,93,96,94,98), there are 10 times that probability is above 0.9 with 10 experiences, so I believe, 280 sample size is really good sample size

https://github.com/prosperousZ/CS6150-HW5-Question-1.git

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

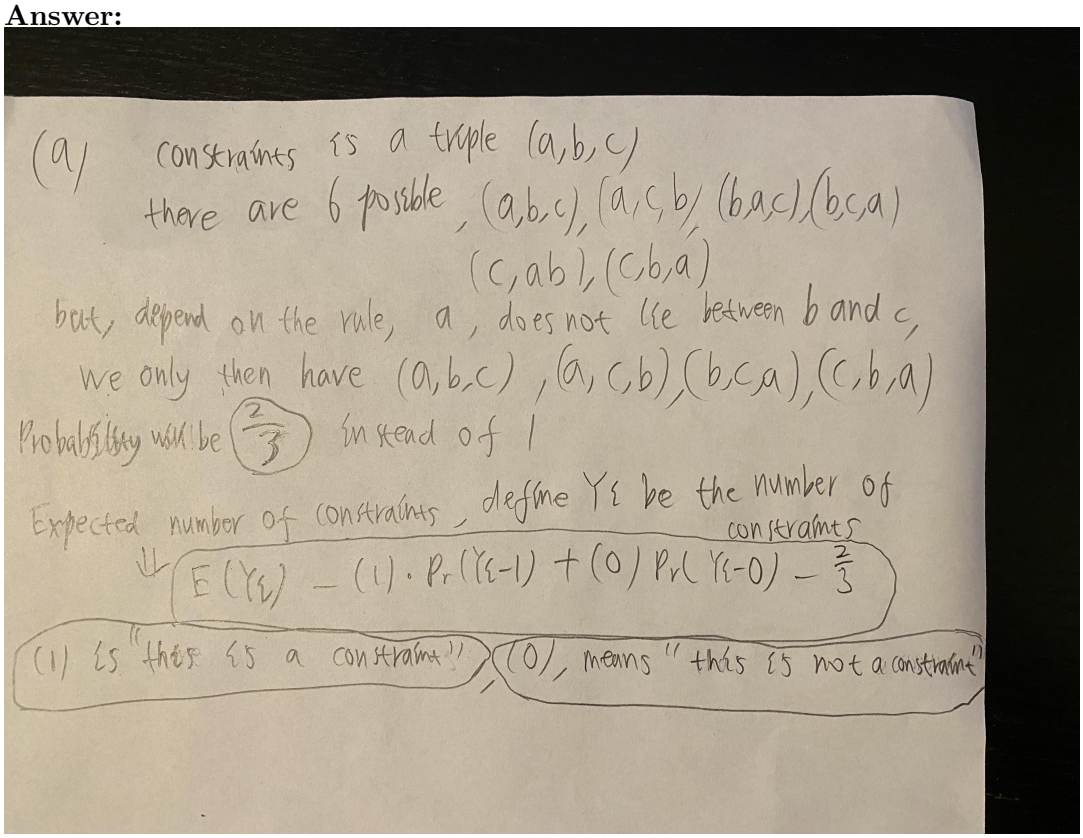
This problem is meant to illustrate a rather cool paradigm for solving problems, which is picking a *random solution*, and understanding how well it does. Indeed, there are many problems ("3-SAT", a version of boolean satisfiability, being the chief of them) where doing better than a random solution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

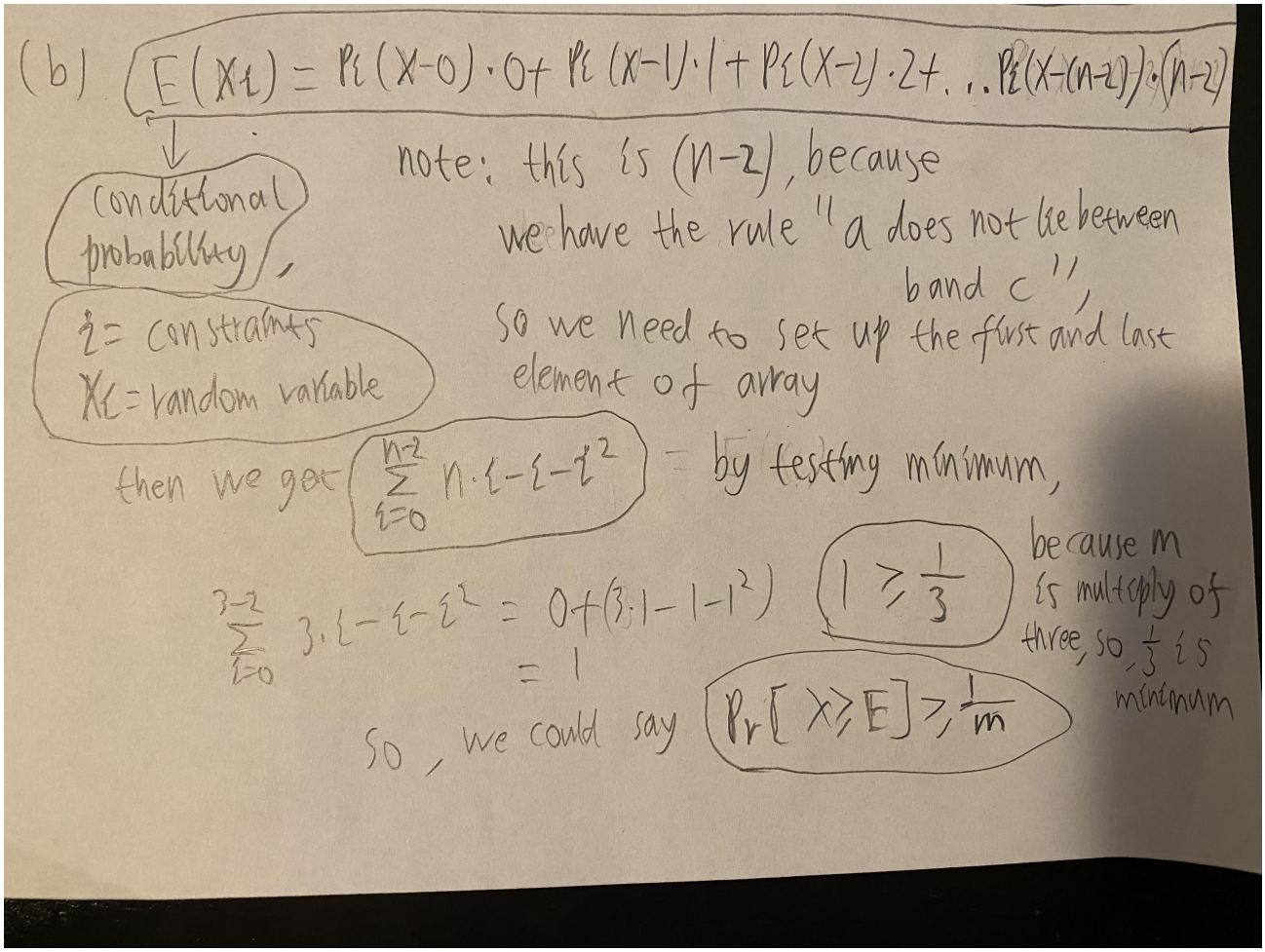
Given the constraints, the goal is to find an ordering that satisfies as many constraints as

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.]
Answer:



(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!]
Answer:



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:

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

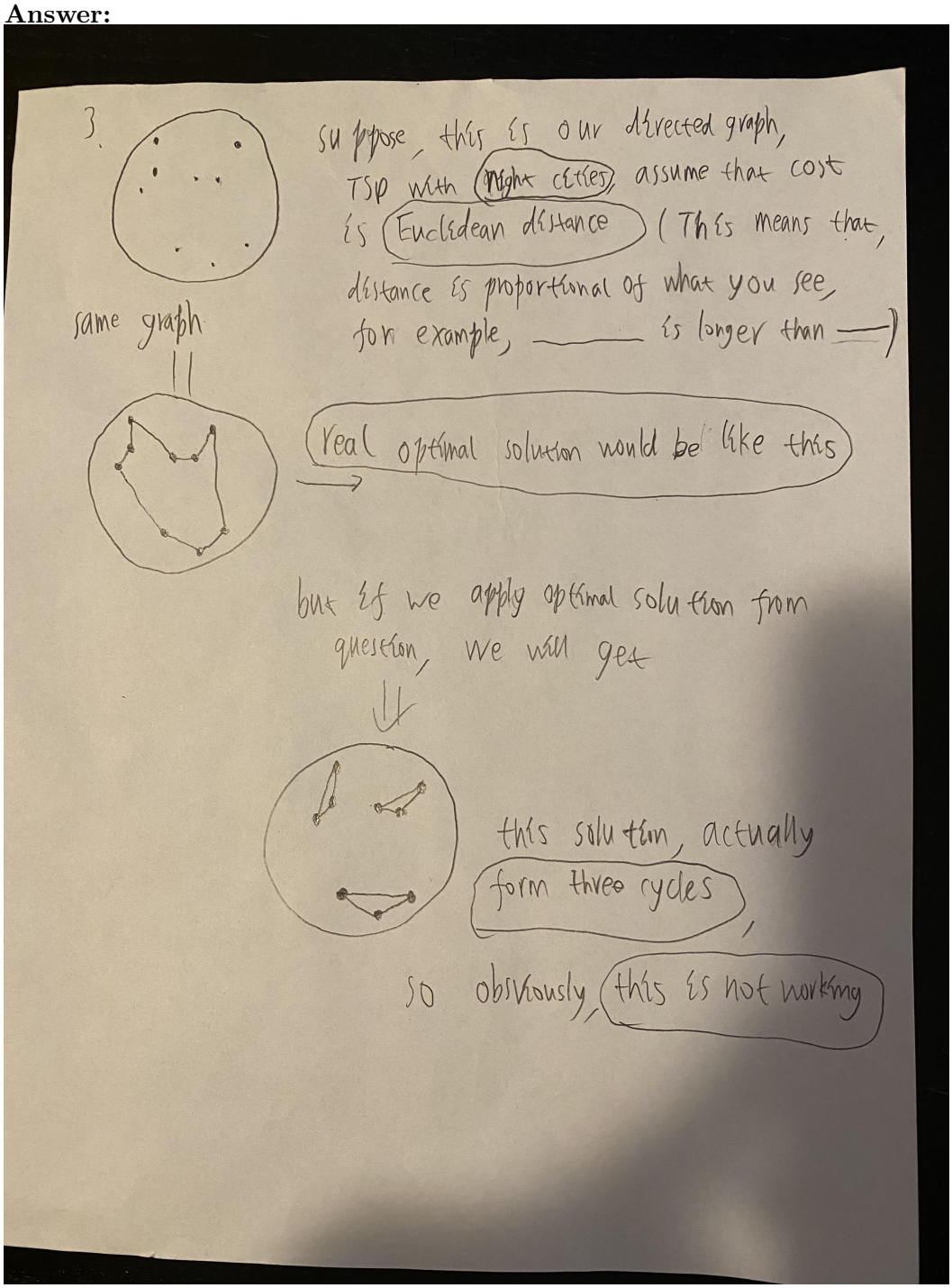
$$\forall\ u, \quad \sum_{w \in N_-(u)} x_{wu} = 1.$$

(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?]

Answer:



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

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

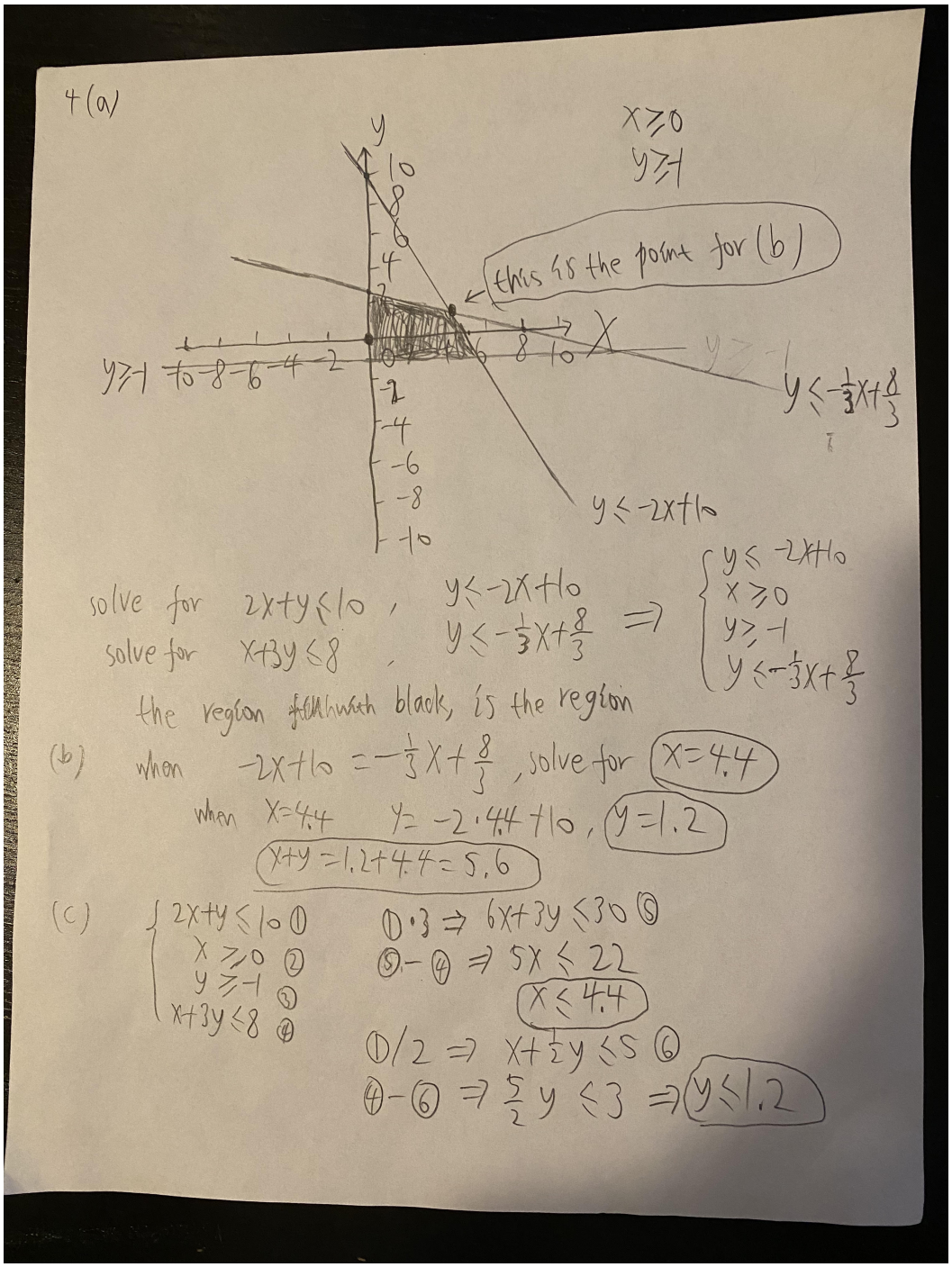
constraints are:

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

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

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

(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?]
Answer:



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.]

Answer:

5. (a) $0 \le x_i \le 1$ for all i,

$x_1 + x_2 + \dots + x_n \le k, \quad 1 \le k \le n$

maximum: every $x_i = 1$, there will be n x_i, so

$n \cdot 1 = n$, which is maximum of k

minimum: every $x_i = 0$, there will be n x_i,,

so $n \cdot 0 = 0$, minimum value of k will be 1,

so $0 < 1$

we have any point $(a_1, a_2, \dots, a_n)$ where $a_i \in \{0, 1\}$

let's prove corner point,

minimum: when $k = 1$, this is minimum corner point,

we have a_1 to a_n, so in n-dimension, with $k=1$

we have $(1, 0, \dots, 0)$ (n number), or $(0, 1, \dots, 0)$ (n number), or

$(0, 0, 1, \dots 0)$, ... or $(0, 0, \dots, 1)$, only

one $a_i = 1$, (little example here, $n = 3$, $k = 1$, $a = (1, 0, 0)$

a is feasible point or $a = (0, 1, 0)$ or $a = (0, 0, 1)$)

there is a nonzeros vector, z, any $a + z$, is over 1,

but our a_i is $= 1$, so $a + z$ over 1, so clearly infeasible

prove maximum: let $k = n$, this is maximum corner point,

we have $(1, 1, 1, \dots, 1)$, for example, $n = 3$, then $k = 3$

we have $a = (a_1, a_2, a_3) = (1, 1, 1)$, maximum point

is all $a_i = 1$, for $\forall$ direction vector z, nonzero,

no matter which direction, there will be one a_i, that is $a + z$

is bigger than 1, so clearly is infeasible,

so we could say, we prove minimum and maximum,

we could say, any point is feasible point for the LP,

I also prove the corner point, define corner point, case close

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?]

Answer:

6. $c^T x$ means, dot product between vectors c and x,
so, as well as a, $a_1^T x$ means dot product between a_1 and x,
same as the rest of a, the minimum of $c^T x$, lies in the
interval $[-M, M]$, suppose I have an oracle,
prove using $O(\log(M/\varepsilon))$ calls to the oracle

First, M cannot be zero, because if $M = 0$, there will not be an interval

$\log M$, M must > 0, because, $\log M$ is error if $M \leq 0$

we let $f(n) = \log n$ and $g(n) = n$, $f'(n) = \frac{1}{n}$, $g'(n) = 1$,
since when M approach to 0^+, $f'(n) \leq g'(n)$ for $n > 0$,
we must have $f(n) < g(n)$ for all $n > 0$
in this case, n is M

so $M > \log(M)$, which means, $\log(M)$ must be feasible.

But, if ε approach to 0^+, which means ε is really close to 0,
$\frac{M}{\varepsilon}$ will be infinite bigger, $\frac{M}{\varepsilon} \to \infty$, $\left(\log \frac{M}{\varepsilon}\right) = \infty$,
$\infty > M$, clearly, this is not feasible