CS 6150: Final Solutions

December 3, 2021

This assignment has 4 questions, for a total of 30 points. Unless otherwise specified, complete and reasoned arguments will be expected for all answers.

Question	Points	Score
Problem A	10	
Problem B	8	
Problem C	4	
Problem D	8	
Total:	30	

Question 1: Problem A ..... **[10]**

Suppose we have n vertices labeled $1, 2, \ldots, n$ arranged as a line. Consider a robot that can hop from one vertex to another. Suppose that hopping from i to j requires *energy* equal to $E(i,j) = \frac{(i-j)^2}{3}$ (thus longer jumps take significantly higher amount of energy).

The goal is to go from vertex 1 to n using the minimum total energy. Let $ans(i)$ denote the minimum energy required to go from vertex i to n.

(a) **[4]** Consider the recurrence:

$$ans(i) = \min_{j>i} E(i,j) + ans(j)$$

for computing the desired answer. Consider a recursive procedure that implements the above recurrence. What is its running time? Next, suppose we remember the answers once computed (and look up instead of making a recursive call). What is the running time of this new procedure?

(a) $O(n^2)$ and $O(n)$ respectively

(b) $\exp(n)$ and $O(n)$ respectively

(c) $O(n^2)$ and $O(n^2)$ respectively

(d) $\exp(n)$ and $O(n^2)$ respectively

In the provided box, give the answer (a-d) along with **one line** of explanation. (No explanation will yield 3/4 points.)

Answer:

(d) $\exp(n)$ and $O(n^2)$ respectively. The top down recursion repeatedly computes the energy from i to all j implying for any given j resulting in $n - i$ computations for each $0 \leq i \leq n$. Since any given j is either selected or jumped over there are a total of 2^n combinations and an exponential running time. Through memoization offered by dynamic programming the bottom up approach allows us to iterate from index i to $n-1$ and at each intermediate index j to record which vertices can be selected to minimize the energy required to reach j. Due to the fact that the minimal energy required to reach some position j contains the minimal path to reach position $j-1$ we can update the minimal path for each $i < j \leq n-1$ as we iterate from i to n. By recording the energies observed at each index i for the remaining $n-i$ vertices we will have the vertex set which minimizes the total energy by $n-i$ where initially at $i = 1$ is $O(n(n-1))$, for $i = 2$ we have $O(n(n-2))$, for $i = 3$ then $O(n(n-3))$, ... As a result the complexity will be $\sum_{i=1}^{n} O(n(n-i)) = O(n^2)$

(b) **[4]** In the recurrence above, suppose instead of searching over all $j > i$, we restrict the search to $j \in [i+1, i+r]$. I.e., the recurrence is:

$$ans(i) = \min_{j \in [i+1, i+r]} E(i,j) + ans(j)$$

What is the smallest value of r for which this yields a correct answer? Explain in a couple of sentences. What is the running time of the new procedure?

Answer.

First note that if we were to search only for $j \in [i+1, i+r]$, then computing $ans(i)$ now only involves r recursive calls (or lookups if we are doing memoization). Thus the running time is $O(nr)$ instead of $O(n^2)$, assuming we did memoization.

As for the minimum value of r, note that the energy function is such that for any $r \geq 2$, taking r steps of length 1 incurs less total energy than taking one step of length r (because it's quadratic)! Thus it is always OK to restrict to steps of length 1. Thus we can pick $r = 1$, which tells us that minimizing total energy is actually pretty trivial – it happens when we take n steps of length 1. The running time is simply $O(n)$ (whether we memoize or not).

(c) [**2**] Can you view the problem above as that of finding the shortest path in an appropriate graph? Answer with a couple of lines of explanation (no need to be very formal).

Answer.

For any energy function $E(i, j)$, we can construct a complete graph where the length of the edge from i to j is $E(i, j)$. The problem then becomes one of finding the shortest path from 1 to n.

Question 2: Problem B ..... [**8**]

Suppose $G = (V, E)$ is an undirected graph. We consider the *coloring* problem, where we are given a set of colors $\{1, 2, \ldots, k\}$, and the goal is to assign colors to vertices so as to minimize the number of "bad edges". An edge ij is said to be bad if the two end points receive the same color.

(a) [**4**] Consider a *random* coloring, i.e., to every vertex, we assign a random color from $\{1, 2, \ldots, k\}$. What is the expected number of bad edges? (You must write down the answer along with a clear description of how you arrived at it. (*Hint:* Define appropriate random variable, express it as sum of "binary" random variables, ...)

Answer.

Let $Y_{u,v}$ be a random variable, $Y_{u,v} = 0$ if edge (u, v) is good, and $Y_{u,v} = 1$ if edge (u, v) is bad (same color for both u and v. Then $Pr(Y_{u,v} = 1) = \frac{1}{k}$, since with the color on u fixed, there is a $1/k$ chance of randomly assigning the same color to v. With the total number of edges as $m = |E|$, then the expected number of bad edges is m/k.

(b) [**4**] Which of the following is an upper bound on the probability that the number of bad edges (strictly) exceeds $\frac{4|E|}{k}$?

(a) 0
(b) 0.25
(c) 0.5
(d) 1

Say what all apply, and give one line of explanation.

Answer. From Markov's Inequality, we have $Pr(X \geq \frac{4|E|}{k}) \leq \frac{\frac{|E|}{k}}{\frac{4|E|}{k}} = 0.25$. Since 0.5 and 1.0 are also upper bounds for this probability, we have (b), (c), and (d).

Question 3: Problem C ..... [**4**]

Suppose we have a set X of n points divided into k equal sized clusters. Now suppose we sample $2k$ points uniformly at random from X, with replacement. We say that a cluster is *unlucky* if none of its points is selected in the sample.

(a) [**4**] What is the expected number of unlucky clusters?

(a) 0
(b) $O(\sqrt{k})$

(c) $\Theta(k)$
(d) None of the above

Select an answer and provide a couple of lines of reasoning. Not providing any reasoning will yield 2/4 points. *Hint:* you may use $\left(1-\frac{1}{k}\right)^k \approx \frac{1}{e}$.

Answer.

There are $\frac{n}{k}$ points in each cluster because of the equal size of the clusters. Assume X is the number of unlucky clusters, and Y_k is a random variable defined as below:

$Y_k = 1$, if cluster k is unlucky, and $Y_k = 0$, otherwise.

Then we would have $X = \sum_1^k Y_k$.

The probability that cluster k is unlucky after sampling 1 point is $(1 - 1/k)$ and after sampling $2k$ points is $(1 - 1/k)^{2k} \approx 1/e^2$.

By applying *linearity of expectation*, we can write:

$E[X] = \sum_k E[Y_k] = \frac{k}{e^2}$ (note that $\frac{1}{e^2}$ is a constant). Therefore, this is bound by $\Theta(k)$ and option **c** is correct. Θ is appropriate since the bound is exact.

Question 4: Problem D ..... **[8]**

Recall the optimization formulation we saw for the Set Cover problem: we have a set of n candidates and a set of m skills. For each candidate, we are given $S_i \subseteq [m]$ (where $[m]$ is shorthand for $\{1, 2, \ldots, m\}$), which is the set of skills that candidate i possesses. The goal is to choose the fewest number of candidates to hire, while satisfying the requirement that for each skill, at least one person with that skill is hired.

(a) [**4**] Consider a variant of the problem, where each candidate has a minimum "wage requirement". I.e., let $w_1, w_2, \ldots, w_n$ be positive integers such that hiring person i costs w_i. How would you modify the optimization formulation so as to incorporate this? (We now wish to minimize the total wages.) You may state **only the differences** between the new formulation and the one we saw in class.

Answer.

Since the goal is to minimize the cost of hiring while covering all the skills. Note that the constraints in the original formulation ensures that we cover all the skills. We just need to ensure the cost of hiring is minimized. Note that since x_i indicates if ith person is hired, then the cost of hiring is $\sum_{i=1}^n w_i \cdot x_i$. Therefore, the objective would be,

$$\min \sum_{i=1}^{n} w_i \cdot x_i$$

The constraints would remain the same.

(b) [**4**] Let us go back to the standard setting (without w_i's). Consider an instance in which $m = n = 4$ (we have 4 candidates and 4 skills of interest). Suppose that for $1 \leq i \leq 4$, candidate i possesses all but the ith skill.

For this instance, what is the best possible value of the set cover objective?

Now, consider the linear programming *relaxation* for Set Cover for this instance:

minimize $x_1 + x_2 + x_3 + x_4$ subject to: $x_2 + x_3 + x_4 \geq 1$ $x_3 + x_4 + x_1 \geq 1$ $x_4 + x_1 + x_2 \geq 1$ $x_1 + x_2 + x_3 \geq 1$ $0 \leq x_i \leq 1$ for $i = 1, 2, 3, 4$.

Construct a feasible solution to the relaxation with objective value < 1.4. (Slightly larger values can fetch partial credit.) [You just need to write down the x_i values, no need for explanation.]

Answer.

We can see that in the integer linear program, hiring a single person would only cover 3 skills and selecting one other person would cover the missing skill. So the best possible value for the set cover objective is 2.

With the relaxation, we can use the solution, $x_1 = x_2 = x_3 = x_4 = 1/3$. We can see that since $0 \leq x_1, x_2, x_3, x_4 \leq 1$ and $x_2+x_3+x_4 = x_3+x_4+x_1 = x_4+x_1+x_2 = x_1+x_2+x_3 = 1$ the selected solution satisfies the constraints and therefore is feasible. With this, the objective value we get is $x_1 + x_2 + x_3 + x_4 = 4/3 \approx 1.33 < 1.4$.