CSOR W4231.001 – Spring, 2022
Eleni Drinea

Homework 3 (170 points)

Out: Monday, March 21, 2022
Due: 11:59pm, Monday, April 4, 2022

Homework Instructions.

1. For all algorithms that you are asked to “give" or “design", you should

   - Describe your algorithm clearly in English.
   - Give pseudocode.
   - Argue correctness; you may provide a formal proof or a convincing argument.
   - Provide, with an explanation, the best (smallest) upper bound that you can for the running time. All bounds should be **worst-case** bounds, unless explicitly stated otherwise.

   You are also encouraged to analyze the space required by your algorithm. We will not remove marks if you don’t, unless the problem explicitly asks you to analyze space complexity.

2. **If you give a Dynamic Programming algorithm, the above requirements are modified as follows:**

   (a) Clearly define the subproblems in English.

   (b) Explain the recurrence in English. (This counts as a proof of correctness; feel free to give an inductive proof of correctness too for practice but points will not be deducted if you don’t.) Then give the recurrence in symbols.

   (c) State boundary conditions.

   (d) Analyze time.

   (e) Analyze space.

   (f) If you’re filling in a matrix, explain the order to fill in subproblems in English.

   (g) Give pseudocode.

3. **Full credit will be given to the fastest correct solution.** For example, an algorithm that solves the problem but runs in time $O(n^2)$ will not receive full marks if there is another algorithm that solves the problem and runs in $O(n)$ (possibly at the expense of some additional space).

4. You should submit this assignment as a **pdf** file to Gradescope. Other file formats will not be graded, and will automatically receive a score of 0.

5. I recommend you type your solutions using LaTeX. For every assignment, you will earn 5 extra credit points if you type your solutions using LaTeX or other software that prints equations and algorithms neatly. If you do not type your solutions, make sure that your handwriting is very clear and that your scan is high quality.

6. You should write up the solutions **entirely on your own**. Collaboration is limited to discussion of ideas only. You should adhere to the department’s academic honesty policy (see the course syllabus). Similarity between your solutions and solutions of your classmates or solutions posted online will result in receiving a 0 in this assignment, and possibly further disciplinary actions. There will be no exception to this policy and it may be applied retro-actively if we have reasons to re-evaluate this homework.

Homework Problems

1. (55 points)

   (a) (30 points) Given an unlimited supply of coins of denominations $c_1, c_2, \ldots, c_n$, you wish to make change for a value v; that is, you wish to find a set of coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10, then we can make change for $15 = 10 + 5$ but not for 11.

   Design an $O(nv)$ algorithm to determine whether it is possible to make change for v using coins of denominations $c_1, c_2, \ldots, c_n$. If the answer is yes, also output a way to make change for v.

   (b) (25 points) Consider the following variation of the above problem. You are only allowed to use each denomination *at most once*. For example, if the denominations are 1, 5 and 10, then you can make change for $6 = 1 + 5$ but not for 20 since you cannot use 10 twice.

   Design an $O(nv)$ algorithm to determine whether it is possible to make change for v using each denomination $c_1, c_2, \ldots, c_n$ at most once.

2. (20 points) Problem 24-3, part a. only, from your textbook (p. 679).

3. (30 points) Alice has a directed graph $G = (V, E)$ with possibly negative weights on the edges w_e for $e \in E$, a destination node t and a set of finite values d_v for every $v \in V$. Bob claims that value d_v equals the weight of the shortest path from v to the destination t.

   (a) (14 pts) Give an algorithm that runs in $O(n + m)$ time and verifies or disproves Bob's claim.

   (b) (16 pts) Now assume that Bob's claim is correct and Alice wants to compute distances to a different destination t'. Suggest an algorithm to Alice that runs in time $O(m \log n)$ for computing weights $d'(v)$ of shortest paths for all nodes $v \in V$ to the destination t'.

   If needed, check the hint provided at the end of the recommended exercises.

4. (20 points) Let S and T be two disjoint subsets of size n of a finite universe U. Suppose we select a random subset $R \subseteq U$ by independently including each element of U with probability p. We say that the sample is *good* if it contains an element of T but no element of S.

   Show that when $p = 1/n$, the probability our sample is good is larger than some positive constant (independent of n).

   Remark 1 *You may use the fact that, for all $x, n \in \mathbf{R}$ such that $n \geq 1$ and $|x| \leq n$,*

   $$e^x \left(1 - \frac{x^2}{n}\right) \leq \left(1 + \frac{x}{n}\right)^n \leq e^x \tag{1}$$

5. (45 points) Suppose that time is divided in discrete steps. There are n people and a single shared computer than can only be accessed by one person in a single step: if two or more people attempt to access the computer at the same step, then everybody is "locked out" during that step.

   At every step, each of the n people attempts to access the computer with probability p.

(a) (5 points) Determine the probability that a fixed person i succeeds in accessing the computer during a specific step.

(b) (5 points) How would you set p to maximize the above probability?

(c) (10 points) For the choice of p in part (b), upper bound the probability that person i did not succeed to access the computer in *any* of the first $t = en$ steps.

Hint: Use inequality 1 in Remark 1.

(d) (10 points) What is the number of steps t required so that the probability that person i did not succeed to access the computer in *any* of the first t steps is upper bounded by an inverse polynomial in n?

(e) (15 points) How many steps are required to guarantee that *all* people succeeded to access the computer with probability at least $1 - 1/n$ (that is, with high probability)?

Hint: You may want to first upper bound the probability of the complementary event.

RECOMMENDED exercises: *Do NOT return, they will not be graded.*

1. Problem 7-2 in the textbook.

2. Problem 7-4 in the textbook.
(If necessary, read pages 232-233 to refresh your memory on the definition of a stack.)

3. (30 points) In this problem, you are given $n \times n$ matrices A, B, C and you wish to check whether $AB = C$.

(a) (6 points) Give a $o(n^3)$ algorithm for the above task.

(b) You will now design a faster randomized test for this task.

i. (14 points) Let $\mathbf{x}$ be an n-dimensional vector with entries randomly and independently chosen to be 0 or 1, each with probability $1/2$. Prove that if M is a non-zero $n \times n$ matrix, then $\Pr[M\mathbf{x} = \mathbf{0}] \leq 1/2$.

ii. (10 points) Show that $\Pr[AB\mathbf{x} = C\mathbf{x}] \leq 1/2$ if $AB \neq C$. Then design and analyze a randomized algorithm to check whether $AB = C$.

Hint for Problem 3b: Consider the weight function $w'(e) = w(e) - d(u) + d(v)$ where $e = (u, v)$. Is there a relation between the weights of paths for the two different weight functions w and w'?