CS 574, Spring 2021
Timothy Chan

Homework 1 (due Feb 22 Monday 5pm (CT))

Instructions: You may work in groups of at most 3; submit one set of solutions per group. Always acknowledge any discussions you have with other people and any sources you have used (although most homework problems should be doable without using outside sources). In any case, *solutions must be written entirely in your own words.*

1. [*38 pts*] Suppose we have a Las Vegas randomized algorithm A that runs in μ expected number of steps. As mentioned in class, for any given $t > 1$, we can obtain a new Monte Carlo algorithm A′ that runs within $t\mu$ steps by outputting "garbage" if the time limit is exceeded. The error probability is at most $1/t$ by Markov's inequality.

   But we can do better! For example, we can run A for up to 2μ steps, and if it fails, rerun A for another 2μ steps (with an independent sequence of random bits), etc., for $\lfloor t/2 \rfloor$ iterations. This gives a Monte Carlo algorithm with the total number of steps still bounded by $t\mu$, and the error probability reduced to $1/2^{\lfloor t/2 \rfloor} = O(1/2^{t/2}) = O(0.7072^t)$, which decays exponentially!

   In this question, you will analyze further improvements to the base of the exponential:

   (a) [*8 pts*] Let $X \geq 0$ be the random variable denoting the number of steps made by one execution of A. We know $\mathbb{E}[X] = \mu$. Let $p_\alpha = \Pr[X \geq \alpha\mu]$.
   Markov's inequality states that $p_\alpha \leq 1/\alpha$ for any $\alpha > 0$.
   First prove the following generalization: for any $\alpha_1, \alpha_2, \alpha_3, \ldots \geq 0$,

   $$\alpha_1 p_{\alpha_1} + \alpha_2 p_{\alpha_1+\alpha_2} + \alpha_3 p_{\alpha_1+\alpha_2+\alpha_3} + \cdots \ \leq \ 1.$$

   (b) [*10 pts*] Now, consider a strategy with three iterations where we run for $\alpha_1\mu$ steps in the first iteration, $(\alpha_1+\alpha_2)\mu$ steps in the second iteration, and $(\alpha_1+\alpha_2+\alpha_3)\mu$ steps in the third iteration.
   Prove that with this strategy, we obtain a Monte Carlo algorithm with the total number of steps bounded by $(3\alpha_1 + 2\alpha_2 + \alpha_3)\mu$, and error probability at most some function $f(\alpha_1, \alpha_2, \alpha_3)$. Specify your function f.
   [Hint: the maximum of xyz over $x+y+z = 1$ $(x, y, z \geq 0)$ occurs when $x = y = z = 1/3$ (by the standard inequality on arithmetic means and geometric means)...]

   (c) [*10 pts*] Using part (b), show how to obtain a Monte Carlo algorithm with the total number of steps bounded by $t\mu$, and the error probability at most $O(0.606^t)$ (thus, improving 0.7072^t).
   [Hint: choose $\alpha_1, \alpha_2, \alpha_3$ appropriately, and repeat $\left\lfloor \frac{t}{3\alpha_1+2\alpha_2+\alpha_3} \right\rfloor$ times...]

   (d) [*10 pts*] Generalize part (b) from 3 iterations to ℓ iterations for an arbitrary constant ℓ, and further improve the error probability in part (c). What does the base of the exponential converge to as ℓ increases?
   [Hint: Stirling's formula for $\ell!$ may be useful...]

2. [*20 pts*] We are given an undirected graph $G = (V, E)$ with n vertices. For a permutation π of V, a vertex $v \in V$ is said to be *good* if the number of v's neighbors that appear before v and the number of v's neighbors that appear after v differ by at most 1. (It is NP-hard to determine whether a graph has a permutation in which all vertices are good, even when the maximum degree is 6.)

   Using the probabilistic method, prove that for any graph with maximum degree D, there always exists a permutation with at least $\Omega(n/D)$ good vertices.

3. [*20 pts*] Given a set P of n points in 2D, we want to find the smallest-area triangle $\triangle pqr$ whose vertices p, q, r are in P.

   Prof. X claims to have discovered an $O(n^{1.9})$-time algorithm[1] to solve the following related problem: given a set P of n points in 2D and a real number x, decide whether there exists a triangle $\triangle pqr$ with $p, q, r \in P$ whose area is less than x.

   Your mission is to provide an efficient Las Vegas algorithm for the original smallest-area triangle problem, using Prof. X's algorithm as a black box.[2]

   Specifically, consider the following pseudocode for some constants b and c:

   ```
   min-area-triangle(P):
       if |P| ≤ c then return answer by brute force
       partition P into subsets P_1, ..., P_b each with at most ⌈n/b⌉ points
       t = ∞
       for each (i, j, k) ∈ {1, ..., b}^3 in random order do
           if there exists a triangle in P_i ∪ P_j ∪ P_k of area less than t then
               t = min-area-triangle(P_i ∪ P_j ∪ P_k)
       return t
   ```

   Explain why the algorithm is always correct, and prove that its expected running time is still $O(n^{1.9})$ for an appropriate choice of b (and determine the smallest such b).

4. [*22 pts*] Given an undirected graph $G = (V, E)$ with n vertices, we want to decide whether there exists an isolated vertex, i.e., a vertex of degree 0. Assume that G is stored as an adjacent matrix, i.e., given any two vertices u and v, we can determine whether $uv \in E$ in unit time. The problem can be solved in $O(n^2)$ time by a trivial (deterministic) algorithm. You will prove that randomization doesn't help much for this problem.

   (a) [*10 pts*] Consider a random graph G with n vertices, where each edge is included independently with probability $p = \frac{100 \log n}{n}$. First prove that the probability that G has no isolated vertices is at least 0.99 for a sufficiently large n.

   (b) [*12 pts*] Now, using Yao's principle, prove a randomized $\Omega(n^2/\log n)$ lower bound on the expected running time of any Las Vegas algorithm for the above problem.

[1] Actually, this is still not known...

[2] Note: binary search is not applicable here since the number of candidates for the optimal area is $O(n^3)$, which is too many to enumerate. Also, even if applicable, the running time would worsen by a logarithmic factor.