

# CS 473 (Spring 2025)

## Homework 4 (due Mar 6 Thu 10am)

**Instructions:** As previous homeworks.

**Problem 4.1:** Suppose we have a Las Vegas randomized algorithm A that runs in  $\mu$  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  $2\mu$  steps, and if it fails, rerun A for another  $2\mu$  steps (with an independent sequence of random bits), etc., for  $\lfloor t/2 \rfloor$  rounds. This gives a Monte Carlo algorithm with the total number of steps still bounded by  $t\mu$ , and the error probability reduced to  $1/2^{\lfloor t/2 \rfloor} = O(1/2^{t/2}) \leq O(0.708^t)$ , which decays exponentially!

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

- (a) (25 pts) Let  $X \geq 0$  be a random variable with  $\mathbb{E}[X] = \mu$ . Given  $\alpha_1, \alpha_2, \dots, \alpha_k \geq 0$ , let  $p_1 = \Pr[X \geq \alpha_1\mu]$ ,  $p_2 = \Pr[X \geq (\alpha_1 + \alpha_2)\mu]$ ,  $\dots$ ,  $p_k = \Pr[X \geq (\alpha_1 + \dots + \alpha_k)\mu]$ . Prove the following inequality:

$$\alpha_1 p_1 + \alpha_2 p_2 + \dots + \alpha_k p_k \leq 1.$$

[Hint: modify the proof of Markov’s inequality and use telescoping sum...]

- (b) (25 pts) Now, consider a strategy with three iterations where we run A 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.

With this strategy, we obtain a Monte Carlo algorithm with the total number of steps upper-bounded by  $(3\alpha_1 + 2\alpha_2 + \alpha_3)\mu$ . Bound the error probability purely as a function of  $\alpha_1$ ,  $\alpha_2$ , and  $\alpha_3$ .

[Hint: you may use the fact that  $x + y + z \leq 1$  ( $x, y, z \geq 0$ ) implies  $xyz \leq 1/27$ . (This follows from the standard inequality that the “geometric mean” is at most the “arithmetic mean”.)]

- (c) (25 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  $O(0.545^t)$  (thus, improving  $O(0.708^t)$ ).

[Hint: repeat for  $\lfloor \frac{t}{3\alpha_1 + 2\alpha_2 + \alpha_3} \rfloor$  rounds; set  $\alpha_1 = c/3$ ,  $\alpha_2 = c/2$ ,  $\alpha_3 = c$  for some choice of  $c$  (you might need a calculator)...]

- (d) (25 pts) Generalize part (b) from 3 iterations to  $k$  iterations for a larger constant  $k$ , and show how to reduce the error probability further to below  $O(0.4^t)$ .

[Hint: choose some  $k$  below 50, and choose  $c = 1$ ...]

**Problem 4.2:** In a popular form of logic puzzles, you land on an island that has three types of inhabitants: “knights”, who always tell the truth; “knaves”, who always lie; and “spies”, who sometimes lie and sometimes tell the truth.

Suppose there are  $n$  inhabitants, where 60% are known to be decent folks, i.e., knights. The remaining 40% are bad, i.e., knaves or spies. You want to know who are the good/bad guys, i.e., determine the types of all  $n$  inhabitants. You are allowed to ask only questions of the form, “is person A a knight/knave/spy?”, to another person B. (All the inhabitants know each other.) Obviously, if you can find a person who you know is a knight, the problem can be solved after asking  $n$  additional questions.

- (a) (10 pts) Give a (very) efficient Monte Carlo algorithm that finds a knight. State the probability of error. (Hint: this is supposed to be very easy!)
- (b) (90 pts) Give a Las Vegas algorithm that finds a knight by asking  $O(n)$  expected number of questions. Analyze the constant factor in the big-Oh and make it smaller than 1.5. [Hint: use (a). How can you confirm whether a specific person is a knight by asking  $O(n)$  questions?]  
[Note: there is also a deterministic algorithm that requires  $O(n)$  questions, but it is more complicated and has a larger constant factor.]

**Problem 4.3:** We are given a set  $S$  of strings of total length  $N$  over the binary alphabet  $\{0, 1\}$ . We are also given a number  $\ell$ . We want to find a palindrome  $p$  of length  $\ell$  and two distinct strings  $s, s' \in S$  such that their concatenation  $ss'$  contains  $p$  as a substring.

For example, for  $S = \{0111, 1111010, 10110, 11011\}$  and  $\ell = 7$ , the concatenation of 1111010 and 11011 contains the palindrome 1101011.

Describe a randomized Las Vegas algorithm for this problem, by using the Karp–Rabin fingerprint technique (to be covered in Feb 28’s lecture). The expected running time should be  $O(N \log N)$  or better.

[Hint: for every string  $s \in S$ , pre-compute fingerprints of each prefix/suffix of  $s$  and the reverse of  $s$ . You may find subtraction useful...]