CS 6150: HW0 – Introduction and background

Submission date: Friday, August 22, 2025, 11:59 PM

This assignment has 6 questions, for a total of 50 points. Unless otherwise specified, complete and reasoned arguments will be expected for all answers.

Question	Points	Score
Big oh and running times	10	
Square vs. Multiply	5	
Graph basics	8	
Background: Probability	12	
Tossing coins	7	
Array Sums	8	
Total:	50	

Question 1: Big oh and running times . . . . **[10]**

(a) [**4**] Write down the following functions in big-oh notation:

1. $f(n) = n^2 + 5n + 20$.

$$\mathbf{O(n^2)}$$

2. $g(n) = \frac{1}{n^2} + \frac{2}{n}$.

$$\mathbf{O(\frac{1}{n})}$$

(b) [**6**] Consider the following algorithm to compute the GCD of two positive integers a, b. Suppose a, b are numbers that are both at most n. Give a bound on the running time of $\textsc{Gcd}(a, b)$. (You need to give a formal proof for your claim.)

Algorithm 1 $\textsc{Gcd}(a, b)$

```
if (a < b) return GCD(b, a);
if (b = 0) return a;
return GCD(b, a%b); (Recall: a%b is the remainder when a is divided by b)
```

Question 2: Square vs. Multiply . . . . **[5]**

Suppose I tell you that there is an algorithm that can square any n digit number in time $O(n \log n)$, for all $n \geq 1$. Then, prove that there is an algorithm that can find the product of *any two* n digit numbers in time $O(n \log n)$. [*Hint:* think of using the squaring algorithm as a subroutine to find the product.]

Question 3: Graph basics . . . . **[8]**

Let G be a *simple*[1] undirected graph. Prove that there are at least two vertices that have the same degree.

Question 4: Background: Probability . . . . **[12]**

(a) [**3**] Suppose we toss a fair coin k times. What is the probability that we see heads precisely once? **To solve this we realize that there will be k "slots" one for each coin flip. For each toss we have a 1/2 probability of getting what we want (heads or tails) After that we simply need to decide which "slot" we want the heads to be in using a combination**

$$\mathbf{(\frac{1}{2})^k * k}$$

(b) [**4**] Suppose we have k different boxes, and suppose that every box is colored uniformly at random with one of k colors (independently of the other boxes). What is the probability that all the boxes get distinct colors? **To solve this problem we can think about how many valid choices we have at each**

$$\mathbf{\frac{k!}{k^k}}$$

(c) [**5**] Suppose we repeatedly throw a fair die (with 6 faces). What is the expected number of throws needed to see a '1'? How many throws are needed to ensure that a '1' is seen with probability $> 99/100$?

Question 5: Tossing coins . . . . **[7]**

Suppose we have two coins, one of which is *fair* (i.e. prob[heads] = prob[tails] = 1/2), and another of which is slightly biased. More specifically, the second coin has prob[heads] = 0.51. Suppose we toss the coins N times, and let H_1 and H_2 be the number of heads observed (respectively).

(a) [**3**] Intuitively, how large must N be, so that we have $H_2 > H_1$ with "reasonable certainty"?

[1] I.e., there are no self loops or multiple edges between any pair of vertices.

(b) [**2**] Suppose we pick $N = 25$. What is the expected value of $H_2 - H_1$?

(c) [**2**] Can you use this to conclude that the probability of the event $(H_2 - H_1 \geq 1)$ is small? [It's OK if you cannot answer this part of the problem.]

Question 6: Array Sums .... [**8**]

Given an array $A[1 \dots n]$ of integers, find if there exist indices i, j, k such that $A[i] + A[j] + A[k] = 0$. Can you find an algorithm with running time $o(n^3)$? [NOTE: this is the little-oh notation, i.e., the algorithm should run in time $< cn^3$, for any constant c, as $n \to \infty$.] [*Hint:* aim for an algorithm with running time $O(n^2 \log n)$.]