CS 3000: Algorithms & Data — Spring '20 — Jonathan Ullman

Homework 1
Due Friday January 17 at 11:59pm via Gradescope

Name:
Collaborators:

- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Friday January 17 at 11:59pm via Gradescope. No late assignments will be accepted. Make sure to submit something before the deadline.
- Solutions must be typeset in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly forbidden.

Problem 1. *Inductive Proofs*

(a) Prove the following statement by induction: For every $n \in \mathbb{N}$, $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

Solution:

(b) In class I asserted without proof that "polynomials are smaller than exponentials." Specifically, that $n^a = O(b^n)$ for every $a > 0$ and $b > 1$. In this problem you will use induction to prove a special case of this fact, that $n^2 = O(2^n)$, by induction.

Prove by induction that, for every $n \geq 4$, $n^2 \leq 2^n$.

Solution:

Problem 2. *Asymptotic Order of Growth*

(a) Rank the following functions in increasing order of asymptotic growth rate. That is, find an ordering $f_1, f_2, \dots, f_{10}$ of the functions so that $f_i = O(f_{i+1})$. No justification is required.

$$\begin{array}{ccccc} n^{5/2} & 4^{\log_2 n} & n! & 7^n & \log_2(n!) \\ 2^{3n} & n^2 \log_2(n) & 8n & 3^{\log_5 n} & \log_2(n^3) \end{array}$$

Solution:

(b) Prove that the following somewhat unusual function f satisfies $f(n) = \Theta(n)$.

$$f(n) = \begin{cases} 10n - 10 & \text{if } n \text{ is even} \\ n + 10 & \text{if } n \text{ is odd} \end{cases}$$

Solution:

(c) Consider the following piece of code.

Algorithm 1: Waste some time

Function $A(n)$**:**
- Let m be the smallest power of 2 that is at least n ($m = 2^{\lceil \log_2 n \rceil}$)
- **For** $i = 1, \dots, m^3$: Do an operation

Give an asymptotic expression for the number of operations done by $A(n)$ as a function of n in $\Theta(\cdot)$ notation. Justify your answer. Your expression should be as simple as possible—for example, $\Theta(n)$ would be a better than $\Theta(100n + 10)$.

Solution:

Problem 3. *What Does This Code Do?*

You encounter the following mysterious piece of code.

Algorithm 2: Mystery function

```
Function C(a, n):
    If n = 1 :
        Return (1, a)
    ElseIf n = 2 :
        Return (a, a · a)
    ElseIf n is odd :
        (u, v) ← C(a, ⌊(n+1)/2⌋)
        Return (u · u, u · v)
    ElseIf n is even :
        (u, v) ← C(a, ⌊(n+1)/2⌋)
        Return (u · v, v · v)
```

(a) What are the results of $C(a,3)$, $C(a,4)$, and $C(a,5)$. You do not need to justify your answers.

Solution:

(b) What does the code do in general? Prove your assertion by induction on n.

Solution:

(c) In this problem you will analyze the running time of C as a function of n. Prove that, for every $n \in \mathbb{N}$, the number of multiplication operations performed in evaluating $C(a,n)$ is at most $2 \cdot \log_2(n-1) + 1$ (where we use the convention that $\log_2(0) = 0$).

Solution:

Problem 4. *Karatsuba Example*

Carry out Karatsuba's Algorithm to compute $24 \cdot 82$. What are the inputs for each recursive call, what does that recursive call return, and how do we compute the final product?

Solution: