CSCI 3104 Fall 2022
Profs Josh Grochow and Chandra Kanth Nagesh

Problem Set 5

Due Date .... TODO
Name .... **Your Name**
Student ID .... **Your Student ID**
Collaborators .... **List Your Collaborators Here**

Contents

Instructions **2**

Honor Code (Make Sure to Virtually Sign the Honor Pledge) **3**

1 Standard 12- Asymptotics I (Calculus I techniques) **4**
1.1 Problem 1(a) 5
1.2 Problem 1(b) 6

2 Standard 13- Asymptotics II (Calculus II techniques): **7**
2.1 Problem 2(a) 7
2.2 Problem 2(b) 8

3 Standard 14- Analyzing Code I: (Independent nested loops) **10**

4 Standard 15- Analyzing Code II: (Dependent nested loops) **11**

Instructions

- The solutions **must be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Useful links and references on LaTeXcan be found here on Canvas.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding**

of the material. If there is any confusion about this policy, it is your responsibility to clarify before the due date.

- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section Honor Code). Failure to do so will result in your assignment not being graded.

Honor Code (Make Sure to Virtually Sign the Honor Pledge)

Problem HC. On my honor, my submission reflects the following:

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

In the specified region below, clearly indicate that you have upheld the Honor Code. Then type your name.

Honor Pledge. □

1 Standard 12- Asymptotics I (Calculus I techniques)

Problem 1. For each part, you will be given a list of functions. Your goal is to order the functions from **slowest growing** to **fastest growing**. That is, if your answer is $f_1(n), \ldots, f_k(n)$, then it should be the case that $f_i(n) \in O(f_{i+1}(n))$ for all i. If two adjacent functions have the same order of growth (that is, $f_i(n) \in \Theta(f_{i+1}(n))$), clearly specify this. **Show all work, including Calculus details.** Plugging into WolframAlpha is not sufficient.

You may find the following helpful.

- Recall that our asymptotic relations are transitive. So if $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$, then $f(n) \in O(h(n))$. The same applies for little-o, Big-Theta, etc. Note that the goal is to order the growth rates, so transitivity is very helpful. We encourage you to make use of transitivity rather than comparing all possible pairs of functions, as using transitivity will make your life easier.
- You may also use the Limit Comparison Test. However, you **MUST** show all limit computations at the same level of detail as in Calculus I-II. Should you choose to use Calculus tools, whether you use them correctly will count towards your score.
- You may **NOT** use heuristic arguments, such as comparing degrees of polynomials or identifying the "high order term" in the function.
- If it is the case that $g(n) = c \cdot f(n)$ for some constant c, you may conclude that $f(n) = \Theta(g(n))$ without using Calculus tools. You must clearly identify the constant c (with any supporting work necessary to identify the constant- such as exponent or logarithm rules) and include a sentence to justify your reasoning.

You may also find it helpful to order the functions using an `itemize` block, with the work following the end of the `itemize` block.

- This function grows the slowest: $f_1(n)$
- These functions grow at the same asymptotic rate and faster than $f_1(n)$: $f_2(n), f_3(n), \ldots$
- These functions grow at the same asymptotic rate, but faster than $f_2(n)$: $f_k(n)$.

Also below is an example of an `align` block to help you organize your work.

$$\begin{aligned}
\lim_{n\to\infty} \frac{n^2}{2^n} &= \lim_{n\to\infty} \frac{2n}{\ln(2)\cdot 2^n} \\
&= \lim_{n\to\infty} \frac{2}{(\ln(2))^2 \cdot 2^n} \\
&= 0.
\end{aligned}$$

1.1 Problem 1(a)

(a) $n^2 + 200n, \quad n + 1000000, \quad n^3 - 20n^2, \quad \frac{n^2}{\sqrt{n}}.$

Answer.

- $n + 1000000 \in O(n^2 + 200n)$.
- $\frac{n^2}{\sqrt{n}} \in O(n^2 + 200n)$ or $n^2 + 200n \in \Omega(\frac{n^2}{\sqrt{n}})$.
- $n^3 - 20n^2 \in \Omega(n^2 + 200n)$.

We use the Limit Comparison Test to establish the claims above.

- To show that $n + 1000000 \in O(n^2 + 200n)$, we compute the following limit:

$$\begin{aligned} L := &\lim_{n\to\infty} \left(\frac{n^2 + 200n}{n + 1000000} \right) \\ &= \lim_{n\to\infty} \left(\frac{n + 200}{1 + \frac{1000000}{n}} \right) \\ &= \frac{\lim_{n\to\infty}(n + 200)}{\lim_{n\to\infty}\left(1 + \frac{1000000}{n}\right)} \\ &= \frac{\infty}{1} = \infty \end{aligned} \tag{1}$$

  As $L = \infty$, we have $n + 1000000 \in O(n^2 + 200n)$.

- To show that $n^2 + 200n \in \Omega(\frac{n^2}{\sqrt{n}})$, let's show that $\frac{n^2}{\sqrt{n}} \in O(n^2 + 200n)$ as follows.

$$\begin{aligned} L := &\lim_{n\to\infty} \left(\frac{n^2 + 200n}{\frac{n^2}{\sqrt{n}}} \right) \\ &= \lim_{n\to\infty} \left(\frac{n + 200}{\sqrt{n}} \right) \\ &= \lim_{n\to\infty} \left(\sqrt{n} + \frac{200}{\sqrt{n}} \right) \\ &= \infty + 0 = \infty \end{aligned} \tag{2}$$

  As $L = \infty$, the Limit Comparison Test provides that $\frac{n^2}{\sqrt{n}} \in O(n^2 + 200n)$. Equivalently, $n^2 + 200n \in \Omega(\frac{n^2}{\sqrt{n}})$.

- To show that $n^3 - 20n^2 \in \Omega(n^2 + 200n)$, we compute the following limit:

$$\begin{aligned} L := &\lim_{n\to\infty} \left(\frac{n^2 + 200n}{n^3 - 20n^2} \right) \\ &== \lim_{n\to\infty} \left(\frac{\frac{1}{n} + \frac{200}{n^2}}{1 - \frac{20}{n}} \right) \\ &= \frac{\lim_{n\to\infty}\left(\frac{1}{n} + \frac{200}{n^2}\right)}{\lim_{n\to\infty}\left(1 - \frac{20}{n}\right)} \\ &= \frac{0}{1} = 0 \end{aligned} \tag{3}$$

  As $L = 0$, the Limit Comparison Test provides that $n^3 - 20n^2 \in \Omega(n^2 + 200n)$.

$\square$

1.2 Problem 1(b)

(b) $(\log_7 n)^7$, $10\log_5 n$, $\log_7(n^7)$, $n^{1/1000}$.

Hint: Recall change of logarithmic base formula $\log_a x = \log_b x \cdot \log_a b$

Answer. We claim that the functions grow in order:

- $10\log_5 n, \log_7 n^7 \in \Theta(\log n)$. Note that by the change of logarithm base formula, we have that $\log_b n \in \Theta(\log_a n)$ for all $a, b > 0$ where $a \neq 1$ and $b \neq 1$. So the result is really independent of the logarithm base.
- $(\log_7 n)^7 \in \Omega(\log n)$.
- $n^{1/1000} \in \Omega((\log_7 n)^7)$.

We have the following.

- We first note that $10\log_5 n \in \Theta(\log n)$. We now show that $\log_7 n^7 \in \Theta(\log n)$. To see this, observe that:

$$\log_7(n^7) = 7\log_7(n)$$

  Using the base change rule for logarithms, we can write $\log_7(n) = \frac{log_2(n)}{log_2(7)}$. Thus by taking the constant $c = \frac{7}{\log_2(7)}$ as in the definition of Big-Theta, we find that $\log_7(n^7) \in \Theta(\log n)$, as desired.
- We now claim that $(\log_7 n)^7 \in \Omega(\log n)$. To show that this is true, let's compare $(\log_7 n)^7$ to $\log_7 n$. We have that:

$$\begin{aligned} L &:= \lim_{n\to\infty} \frac{(\log_7 n)^7}{\log_7 n} \\ &= \infty. \end{aligned}$$

  As $L = \infty$, we have that $(\log_7 n)^7 \in \Omega(\log n)$, as desired.
- We finally show that $n^{1/1000} \in \Omega((\log_7 n))^7$. To prove this, we use the Limit Comparison Test. We have that:

$$\begin{aligned} L &:= \lim_{n\to\infty} \frac{n^{1/1000}}{(\log_7 n)^7} \\ &= \frac{1}{1000} \cdot \lim_{n\to\infty} \frac{n \cdot n^{-1+1/1000}}{7\ln(7)\cdot \log_7(n)} \\ &= \frac{1}{1000 \cdot 7\ln(7)} \cdot \lim_{n\to\infty} \frac{n^{1/1000}}{\log_7(n)} \\ &= \frac{1}{1000^2 \cdot 7\ln(7)} \cdot \lim_{n\to\infty} \frac{n \cdot n^{-1+1/1000}}{\ln(7)} \\ &= \infty. \end{aligned}$$

  As $L = \infty$, we have that $n^{1/1000} \in \Omega((\log_7 n)^7)$.

□

2 Standard 13- Asymptotics II (Calculus II techniques):

Problem 2. *For each of the following questions, put the growth rates in order, from slowest-growing to fastest. That is, if your answer is $f_1(n), f_2(n), \ldots, f_k(n)$, then $f_i(n) \leq O(f_{i+1}(n))$ for all i. If two adjacent ones are asymptotically the same (that is, $f_i(n) = \Theta(f_{i+1}(n))$), you must specify this as well. Justify your answer (show your work). You may assume transitivity: if $f(n) \leq O(g(n))$ and $g(n) \leq O(h(n))$, then $f(n) \leq O(h(n))$, and similarly for little-oh, etc. The same instructions as for Problem 1 apply.*

2.1 Problem 2(a)

(a) $n^{\log_5 n}$, $\quad 4$, $\quad n^{\log_3 n}$, $\quad n^{\log_n(n^2)}$, $\quad n^{\log_n 5}$.

Answer. Before we order the functions, we first perform some simplifications. Observe the following:

$$n^{\log_n(n^2)} = n^2,$$
$$n^{\log_n 5} = 5.$$

We now claim that the functions grow in order:

- 4 and $n^{\log_n 5} = 5$ grow at the same asymptotic rate of $\Theta(1)$. Namely, take $C = 5/4$. So $C \cdot 4 = n^{\log_n 5}$, which yields the result.
- $n^2 \in \omega(1)$.
- $n^{\log_5 n} \in \omega(n^2)$.
- $n^{\log_3 n} \in \omega(n^{\log_5 n})$.

We now justify our claim.

- We show that $n^2 \in \omega(1)$. To see this, observe that:

  $$\lim_{n \to \infty} \frac{n^2}{1} = \infty.$$

  So by the Limit Comparison Test, we have that $n^2 \in \omega(1)$.
- We show that $n^{\log_5(n)} \in \omega(n^2)$. Observe that:

  $$\begin{aligned} L &:= \lim_{n \to \infty} \frac{n^{\log_5(n)}}{n^2} \\ &= \lim_{n \to \infty} n^{\log_5(n) - 2} \\ &= \infty. \end{aligned}$$

  As $L = \infty$, we have by the Limit Comparison Test that $n^{\log_5(n)} \in \omega(n^2)$, as desired.
- We now show that $n^{\log_3(n)} \in \omega(n^{\log_5(n)})$. Observe that:

  $$\begin{aligned} L &:= \lim_{n \to \infty} \frac{n^{\log_3(n)}}{n^{\log_5(n)}} && (4) \\ &= \lim_{n \to \infty} \frac{n^{\log_3(n)}}{n^{\log_3(n)/\log_3(5)}} && (5) \\ &= \lim_{n \to \infty} n^{\log_3(n) - \log_3(n)/\log_3(5)} && (6) \\ &= \lim_{n \to \infty} n^{\log_3(n) \cdot (1 - 1/\log_3(5))} && (7) \\ &= \infty. && (8) \end{aligned}$$

  Here, (5) follows by applying the change of logarithm base formula to $\log_5(n)$. Now as $L = \infty$, we have by the Limit Comparison Test that: $n^{\log_3(n)} \in \omega(n^{\log_5(n)})$.

$\square$

2.2 Problem 2(b)

(b) $n!$, 2^n , $2^{n/3}$, n^n, 2^{n-2}, $\sqrt{n^{2n+1}}$. (*Hint:* Recall Stirling's approximation, which says that $n! \sim \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$, i.e. $\lim_{n\to\infty} \frac{n!}{\left(\frac{n}{e}\right)^n \sqrt{2\pi n}} = 1$.).

Answer. We claim that the functions grow in order:

- $2^{n/3}$
- $2^{n-2} \in \Theta(2^n)$
- $n!$
- n^n
- $\sqrt{n^{2n+1}}$.

We have the following.

- We first show that $2^{n/3} \in o(2^n)$. To establish this claim, we use the Limit Comparison Test:

$$\begin{aligned} L &:= \lim_{n\to\infty} \frac{2^{n/3}}{2^n} \\ &= \lim_{n\to\infty} \frac{1}{2^{2n/3}} \\ &= 0. \end{aligned}$$

  As $L = 0$, we have that $2^{n/3} \in o(2^n)$, as desired.

- We next establish the claim that $2^{n-2} \in \Theta(2^n)$. Observe that $2^2 \cdot 2^{n-2} = 2^n$. So taking $C = 2^2$ as in the definition of Big-Theta, we have that $2^{n-2} \in \Theta(2^n)$.

- We next establish the claim that $2^n \in o(n!)$. To this end, we apply the Ratio Test. Consider the infinite series:

$$\sum_{n=1}^{\infty} \frac{2^n}{n!}.$$

  We take:

$$a_n := \frac{2^n}{n!}.$$

  Now the Ratio Test tells us to consider the following limit:

$$\begin{aligned} L' &:= \lim_{n\to\infty} \left| \frac{a_{n+1}}{a_n} \right| \\ &= \lim_{n\to\infty} \frac{2^{n+1}}{(n+1)!} \cdot \frac{n!}{2^n} \\ &= \lim_{n\to\infty} \frac{2}{n} \\ &= 0. \end{aligned}$$

  As $L' = 0$, the Ratio Test provides that:

$$\begin{aligned} L &= \lim_{n\to\infty} |a_n| \\ &= \lim_{n\to\infty} \frac{2^n}{n!} \\ &= 0. \end{aligned}$$

  As $L = 0$, we have by the Limit Comparison Test that $2^n \in o(n!)$, as desired.

- We now show that $n! \in o(n^n)$. We have that:

$$\begin{align}
L &:= \lim_{n\to\infty} \frac{n!}{n^n} \tag{9}\\
&= \lim_{n\to\infty} \frac{(n/e)^n\sqrt{2\pi n}}{n^n} \tag{10}\\
&= \lim_{n\to\infty} \frac{\sqrt{2\pi}\cdot\sqrt{n}}{e^n} \tag{11}\\
&= \sqrt{2\pi}\cdot \lim_{n\to\infty} \frac{1}{2\sqrt{n}\cdot e^n} \tag{12}\\
&= 0. \tag{13}
\end{align}$$

Here, (10) follows by applying Stirling's approximation to $n!$, and (12) follows by L'Hopital's rule. As $L = 0$, we have by the Limit Comparison Test that $n! \in o(n^n)$.

- We now establish the claim that $n^n \in o(\sqrt{n^{2n+1}})$. We first observe that:

$$\begin{aligned}
\sqrt{n^{2n+1}} &= (n^{2n+1})^{1/2}\\
&= n^{(1/2)\cdot(2n+1)}\\
&= n^{n+1/2}.
\end{aligned}$$

Now observe that:

$$\begin{aligned}
L &:= \lim_{n\to\infty} \frac{n^n}{n^{n+1/2}}\\
&= \lim_{n\to\infty} \frac{1}{n^{1/2}}\\
&= 0.
\end{aligned}$$

As $L = 0$, we have by the Limit Comparison Test that $n^n \in o(\sqrt{n^{2n+1}})$, as desired.

$\square$

3 Standard 14- Analyzing Code I: (Independent nested loops)

Problem 3. Analyze the *worst-case* runtime of the following algorithms. Clearly derive the runtime complexity function $T(n)$ for this algorithm, and then find a tight asymptotic bound for $T(n)$ (that is, find a function $f(n)$ such that $T(n) \in \Theta(f(n))$). Avoid heuristic arguments from 2270/2824 such as multiplying the complexities of nested loops.

Algorithm 1 Nested Algorithm 1

```
procedure Foo2(Integer n)
    for i ← 1; i ≤ n; i ← i + 2 do
        for j ← 1; j ≤ n; j ← 3 * j do
            print "Hi"
```

Answer. We begin by analyzing the j-loop.

- The initialization step $j \leftarrow 1$ takes 1 step.
- Let k denote the number of iterations of the j-loop. The j-loop terminates when $3^k > n$. So $k > \log_3(n)$. Thus, the loop takes $k = \lceil \log_3(n) \rceil + 1$ iterations.
- At each iteration, we have:
  - 1 step for the comparison $j \leq n$
  - 2 steps for the update $j \leftarrow 3 * j$ (multiply and assign)
  - 1 step for the print statement.

  Thus, the complexity of the j-loop is:

$$1 + \sum_{j=1}^{\lceil \log_3(n) \rceil + 1} (1 + 2 + 1) = 1 + 4 \cdot (\lceil \log_3(n) \rceil + 1)$$

We now analyze the i-loop.

- The initialization step $i \leftarrow 1$ takes 1 step.
- The i-loop takes $\lceil \frac{n-1}{2} \rceil + 1$ iterations.
- At each iteration of the i-loop:
  - 1 step for the comparison $i \leq n$
  - 2 steps for the update $i \leftarrow i + 2$ (sum and assign)
  - $1 + 4 \cdot (\lceil \log_3(n) \rceil + 1)$ steps for the execution of the j-loop.

  Thus, the total complexity of the algorithm is:

$$\begin{aligned} T(n) &= 1 + \sum_{i=1}^{\lceil \frac{n-1}{2} \rceil + 1} (1 + 2 + 4 \cdot (\lceil \log_3(n) \rceil + 1)) \\ &= 1 + \sum_{i=1}^{\lceil \frac{n-1}{2} \rceil + 1} (7 + 4 \lceil \log_3(n) \rceil) \\ &= 1 + 7 \cdot \left(\left\lceil \frac{n-1}{2} \right\rceil + 1 \right) + 4 \lceil \log_3(n) \rceil \cdot \left(\left\lceil \frac{n-1}{2} \right\rceil + 1 \right) \end{aligned}$$

  Thus, $T(n) \in \Theta(n(\log(n)))$.

□

4 Standard 15- Analyzing Code II: (Dependent nested loops)

Problem 4. Analyze the *worst-case* runtime of the following algorithms. Clearly derive the runtime complexity function $T(n)$ for this algorithm, and then find a tight asymptotic bound for $T(n)$ (that is, find a function $f(n)$ such that $T(n) \in \Theta(f(n))$). Avoid heuristic arguments from 2270/2824 such as multiplying the complexities of nested loops.

Algorithm 2 Nested Algorithm 2

```
procedure Boo1(Integer n)
    for i ← 1; i ≤ n; i ← i + 1 do
        for j ← 1; j ≤ n; j ← j + 1 do
            print "Hi"
```

Answer. We begin by analyzing the j-loop.

- The initialization step $j \leftarrow 1$ takes 1 step.
- The j-loop takes $n - i$ iterations.
- At each iteration, the j-loop takes:
  - 1 step for the comparison $j \leq n$.
  - 2 steps for the update: $j \leftarrow j + 1$
  - 1 step for the print statement.

  So the complexity of the j-loop is:

$$1 + \sum_{j=1}^{n-i} (1 + 2 + 1) = 1 + 4(n - i).$$

We now analyze the i-loop.

- The initialization step $i \leftarrow 1$ takes 1 step.
- The i-loop takes n iterations.
- At each iteration of the i loop, we have:
  - 1 step for the comparison $i \leq n$.
  - 2 steps for the update $i \leftarrow i + 1$.
  - $1 + 4(n - i)$ steps for the j-loop.

  Thus, the runtime of the algorithm is:

$$\begin{aligned} T(n) &= 1 + \sum_{i=1}^{n} (1 + 2 + 1 + 4(n - i)) \\ &= 1 + \sum_{i=1}^{n} 4 + \sum_{i=1}^{n} 4(n - i) \\ &= 1 + 4n + 4 \cdot \sum_{i=1}^{n} (n - i) \\ &= 1 + 4n + 4 \cdot \left(\frac{n(n-1)}{2} \right). \end{aligned}$$

So $T(n) \in \Theta(n^2)$.

□