TheoryPrep **(Due: 7/18/20)**

Assignment #7: Asymptotics and Recurrence

Name: Student name(s)

We've taken some problems/solutions from the course texts and CS 121/125 material. Some problems are marked as (Challenge) or (Additional Problems). Do the rest first, and if you have time, return to these problems. It's **much more important** to have a rigorous understanding of the core problems and how to prove them than to simply finish all the additional problems.

Problem 1: Understanding Asymptotics

Learning goal: The goal for this section is twofold. First, after doing these problems, you will have experience manipulating the mathematical structures of asymptotics. Second, you will build an intuitive understanding that allows you to deduce the asymptotic relationship between two functions without needing to write out all the mathematical formality.

(a) This first problem will help you make sure you understand the definition of Big-Oh. Prove that $n^2 = O(n^3)$.

Solution:
We need to prove that $\exists c, N \in \mathbb{N}.\forall n > N \in \mathbb{N} n^2 \leq cn^3$. Fix $c = 1, N = 0$. Now fix arbitrary $n > N \in \mathbb{N}$. We know that $n^2 \leq n^3 = cn^3$ (since $n > 0$).

(b) This was not in the initial version of the pset so students may need to redownload. In the video, we said that although we define $f(n) = o(g(n))$ as $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) < \epsilon g(n)$, we could have replaced the last predicate with $f(n) \leq \epsilon g(n)$. Why would this be a valid thing to do?

Solution:
These definitions are equivalent. To be fully rigorous, we prove their equivalence below, but the much more important part is just being able to see that intuitively. Since the statement must hold for all ϵ, whether it holds strictly is not meaningful because we can just pick another ϵ. The full proof is considered an Additional Problem. Have students complete the rest of the core problems first.

($\leq \Rightarrow <$) We are given $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) \leq \epsilon g(n)$ and want to prove $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) < \epsilon g(n)$. Fix arbitrary $\epsilon > 0$. We know that $\exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) \leq \frac{\epsilon}{2} g(n) \Rightarrow f(n) < \epsilon g(n)$. Since we chose $\epsilon > 0$ arbitrarily, this holds for all $\epsilon > 0$, completing the proof.

($< \Rightarrow \leq$) We know that $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) < \epsilon g(n)$. To prove $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) \leq \epsilon g(n)$, for any $\epsilon > 0$ can just pick the same N and $f(n) < \epsilon g(n) \Rightarrow f(n) \leq \epsilon g(n)$.

(c) Let $f, g : \mathbb{N} \to \mathbb{R}^+$. Define what it means for $f = \omega(g)$. You should aim for this relationship to capture that $f > g$ in the same way that $f = o(g)$ captures $f < g$. After you do this problem, delete the box on page 3 of the reading and comment on the similarities and differences in your definition.

Solution:
$f = \omega(g)$ if $\forall \epsilon > 0 \exists N \in \mathbb{N}.\forall n > N \in \mathbb{N} f(n) > \epsilon g(n)$.

(d) Fill in the table below to denote the relationship between the functions below. You do not need to prove anything since this question is more about building intuition for the relationships between functions. You should however be able to justify the table to yourself (the last line is an **Additional Problem**)

A	B	O	o	Ω	ω	Θ
n^2	$2n^2$					
n^{137}	$\pi 2^n$					
$(\log\log n)^2$	$\sqrt[3]{\log n}$					
$2^{\sqrt{\log n}}$	n^7					
n^n	$n!$					
$\log(n!)$	$\log(n^n)$					

Solution:

A	B	O	o	Ω	ω	Θ
n^2	$2n^2$	Yes	No	Yes	No	Yes
n^{137}	$\pi 2^n$	Yes	Yes	No	No	No
$(\log\log n)^2$	$\sqrt[3]{\log n}$	Yes	Yes	No	No	No
$2^{\sqrt{\log n}}$	n^7	Yes	Yes	No	No	No
n^n	$n!$	No	No	Yes	Yes	No
$\log(n!)$	$\log(n^n)$	Yes	No	Yes	No	Yes

Here are the learning objectives students should understand from each of these problems

- $n, 2n^2$: Constant factors don't matter for Big-Oh/Big-Omega. They do not characterize Little-Oh or Little-Omega relationships.
- $n^{137}, \pi 2^n$: Exponential is always larger than polynomial
- $(\log\log n)^2$, $\sqrt[3]{\log n}$: Try substituting $x = \log\log n$. Then you can conclude the result since exponentials are much larger than polynomials (no mater the base or degree). It's probably worth discussing with students that this substitution is "allowed" under the definition of asymptotics only because we're substituting monotonic transformations (and if you really want you can talk about how the domain changes from $\mathbb{N}$ to $\log(\log(\mathbb{N}))$, but that's pretty tangential).
- $2^{\sqrt{\log n}}, n^7$. This can also be solved with substitution $x = \log(n)$. We can then compare the bases and exponents on the powers.
- $n^n, n!$. n^n is the product of n ns, but $n!$ is the product of n and a bunch of numbers smaller than it. Thus n^n is a lot bigger (discuss with students how this gap continues to grow, so it's little-omega not just big-omega).
- Clearly $\log(n^n) > \log(n!)$ by applying the monotonic transformation log to the last row. However

$$\begin{aligned}
4\log(n!) &= 4(\log(n) + \ldots + \log(1)) \\
&\geq 4(\log(n) + \ldots + \log(\lceil \frac{n}{2} \rceil)) \\
&\geq 4(\lceil \frac{n}{2} \rceil \log(\lceil \frac{n}{2} \rceil)) \\
&\geq n\log(n)
\end{aligned} \tag{0.1}$$

  An interesting takeaway is that $f(n) = o(g(n))$ does not imply $h(f(n)) = o(h(g(n)))$ even for monotonic h.

(e) This question will help you build some understanding for properties of asymptotics and fluency with the definitions. Let $f, g : \mathbb{N} \to \mathbb{R}^+$.

1. As you may recall from the video, $f(n) = O(g(n))$ corresponds to $f \leq g$ and $g(n) = \Omega(f(n))$ corresponds to $g \geq f$, so we might expect these to be equivalent conditions. Thus, prove that $f(n) = O(g(n))$ if and only if $g(n) = \Omega(f(n))$.
2. We associate $f(n) = O(g(n))$ with $f \leq g$ and $f(n) = o(g(n))$ with $f < g$. Thus, we might guess that $f(n) = o(g(n)) \Rightarrow f(n) = O(g(n))$. This is true–prove it (note that it's also true that $f(n) = \omega(g(n)) \Rightarrow f(n) = \Omega(g(n))$ but that proof is similar so we will not make you do both).

Solution:

1. ($\Rightarrow$) If $f(n) = O(g(n))$ then $\exists c, N \in \mathbb{N}. \forall n > N \in \mathbb{N} f(n) \leq cg(n)$. Let $c' = \frac{1}{c}$, so then $\forall n > N \in \mathbb{N} g(n) \geq c'f(n)$ so $g(n) = \Omega(f(n))$.

   ($\Leftarrow$) If $g(n) = \Omega(f(n))$ then $\exists c, N \in \mathbb{N}. \forall n > N \in \mathbb{N} g(n) \geq cf(n)$. Let $c' = \frac{1}{c}$ so then $\forall n > N \in \mathbb{N} f(n) \leq c'g(n)$ so $f(n) = O(g(n))$.
2. If $f(n) = o(g(n))$ then $\forall \epsilon > 0 \exists N \in \mathbb{N}. \forall n > N \in \mathbb{N} f(n) < cg(n)$. Fix some ϵ and the N such that is true and set $c = \epsilon$. Then we see that $\forall n > N \in \mathbb{N} f(n) \leq cg(n)$, so $f(n) = O(g(n))$ by definition.

(f) Below is a false proof that $2^n = O(1)$. Explain why it is false.

Letting $f(n) = 2^n$, we prove by induction on n that $f(n) = O(1)$. The base case of $f(0) = 2^0 = 1 \leq 1$ holds. In the inductive step, we assume that $f(n) \leq c \cdot 1$. We know that

$$\begin{aligned} f(n+1) &= 2^{n+1} \\ &= 2f(n) \\ &\leq (2c) \cdot 1 \end{aligned} \tag{0.2}$$

Thus, $2^{n+1} = O(1)$. This completes the inductive step and therefore $f(n) = O(1)$.

Solution:
The issue with this proof is that you need to fix a c and prove that $\forall n > N \in \mathbb{N}$ $f(n) \leq cg(n)$, but here in the inductive case we assume $f(n) \leq cg(n)$ and then prove $f(n+1) \leq (2c)g(n)$. We need to prove that $f(n+1) \leq cg(n)$. This is an example where the order of quantifiers is important!

(g) (Additional Problem) Since we say $f(n) = O(g(n))$ is like $f \leq g$ and $f(n) = \Omega(g(n))$ is like $f \geq g$, we might expect one or the other to hold. However, this is not true–prove that there is an $f, g : \mathbb{N} \to \mathbb{R}^+$ such that $f \neq O(g)$ and $f \neq \Omega(g)$. We can't have $n, m \in \mathbb{N}$ such that $n \not\leq m$ and $n \not\geq m$, so why is this possible for functions?

Solution:
Counterexample: Let $f(n) = n$ and define $g(n)$ as a piecewise function

$$g(n) = \begin{cases} 1 & n \text{ is odd} \\ n^2 & n \text{ is even} \end{cases}$$

We see f is not in $O(g)$ by contradiction. Suppose that f is $O(g)$, so for some N, c, $\forall n > N \in \mathbb{N} f(n) \leq cg(n)$. In particular, let n' be an odd number greater than c. We see that $f(n') = n' \leq cg(n') = c$ is not true and thus a contradiction so f is not $O(g)$. We see by similar contradiction that f is not $\Omega(g)$. Suppose towards a contradiction that f is $\Omega(g)$. Then for some N, c, $\forall n > N \in \mathbb{N} g(n) \leq cf(n)$. In particular, let m be an even number greater than c. We see that $g(m) = m^2 \leq cf(m) = cm$ is not true–thus a contradiction so f is not $\Omega(g)$. Thus, we see that this is a counterexample since f is not $O(g)$ and f is not $\Omega(g)$.

The issue here is that we can have oscillating functions (like g) that are hard to categorize as larger or smaller. This doesn't happen with natural numbers, which is why inequalities are a complete relation on $\mathbb{N}$ (complete relation means $n \leq m$ or $n \geq m$).

(h) (Additional Problem) Part of the motivation for our definition for asymptotics is that they are invariant with respect to constant factors–if we multiply by a factor of 2 it's still about the same runtime. In other words $2f(n) = O(f(n))$. However, this may not be true when we move the factor of 2 to the input, considering $f(n), f(2n)$. Prove that there exists some $f : \mathbb{N} \to \mathbb{R}^+$ such that $f(n)$ is $o(f(2n))$ (so $f(n)$ is much smaller than $f(2n)$ and therefore multiplying the input by 2 makes a big difference).

Solution:
Let $f(n) = 2^n$. In this case I claim that $f(n) = o(f(2n))$. We see that

$$\begin{aligned}\lim_{n\to\infty} \frac{f(n)}{f(2n)} &= \lim_{n\to\infty} \frac{2^n}{2^{2n}} \\ &= \lim_{n\to\infty} \frac{1}{2^n}\end{aligned} \tag{0.3}$$

This limit evaluates to 0. We proved this in the first video for this week.

(i) (Additional Problem) We know that big-oh is impervious to constant factors, but we will show an even stronger fact about little-oh; that it is impervious to muliplying by functions. Let $f, g, h : \mathbb{N} \to \mathbb{R}^+$. Prove that if $f = o(g)$ then $f \cdot h = o(g \cdot h)$ (where $f \cdot h(x) = f(x)h(x)$, the product not composition).

Solution:
Since f is $o(g)$, we know that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, which in particular by the definition of limits means that for any $\epsilon > 0$ there exists and N such that $\forall n > N \in \mathbb{N}\frac{f(n)}{g(n)} < \epsilon$. I claim that (we will prove this with the definition of limits)

$$\lim_{n\to\infty} \frac{f(n)h(n)}{g(n)h(n)} = 0$$

Fix arbitrary $\epsilon > 0$. Since $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, we see that there exists a N such that $\forall n > N \in \mathbb{N}.\frac{f(n)}{g(n)} < \epsilon$. Fix this N. Now fix an arbitrary $n > N$, and let us consider $\frac{f(n)h(n)}{g(n)h(n)}$. Since $h(n)$ is a positive function, so in particular $h(n) \neq 0$, we have $\frac{f(n)h(n)}{g(n)h(n)} = \frac{f(n)}{g(n)}$. By choice of N, we know that $\frac{f(n)}{g(n)} < \epsilon$, so our previous equality implies $\frac{f(n)h(n)}{g(n)h(n)} < \epsilon$. Since $n > N$ was chosen arbitrarily, this holds for all $n > N$, so $\forall n > N\frac{f(n)h(n)}{g(n)h(n)} < \epsilon$. Since $\epsilon > 0$ was chosen arbitrarily, this holds for all $\epsilon > 0$ so we have that $\forall \epsilon > 0 \exists N \in \mathbb{N} : \forall n > N \in \mathbb{N}\frac{f(n)h(n)}{g(n)h(n)} < \epsilon$. Thus, we have that

$$\lim_{n\to\infty} \frac{f(n)h(n)}{g(n)h(n)} = 0$$

This implies that $f \cdot h = o(g \cdot h)$.

Problem 2: Recurrence Relations

Learning goal: This section contains two proofs that will require you to use induction to deduce the asymptotic runtime of a recurrence relation. This technique will likely come up in 124 and will also help you build your induction skills (which will come up everywhere!).

(a) You may have heard of the mergesort algorithm before (if you have not and want to learn more about it you can see this video). Mergesort basically splits the problem of sorting an array of length n into the problem of sorting each half, and the problem of merging those two arrays. If we let $T(n)$ be the time for mergesort to sort a list of length n, we thus get the recurrence relation that $T(n) = T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + n$ (mergesort the first half of the list, then the second half of the list which we give the extra element if it exists, and then a linear time algorithm to merge them–the new symbols here are the floor and ceiling function, meaning round down or up to the nearest integer). Prove that $T(n) = O(n \log(n))$. You may assume $T(1) = 1$. The log in this problem is considered to have base 2.

Solution:
Set $N = 1, c = 2$. We will prove that $\forall n > N \in \mathbb{N}. T(n) \leq 2 \cdot n \log(n)$ by induction on n. For the base case, $n = 2$ and we see $T(2) = T(1) + T(1) + 2 = 4 \leq 2 \cdot 2 \log(2) = 4$.

Now for the inductive step. We consider $T(n)$ knowing that $\forall m < n. T(m) \leq 2m \log(m)$. We see that (first line by definition, second line by inductive hypothesis, third line by combining the first two terms)

$$\begin{aligned} T(n) &\leq T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + n \\ &\leq 2\lfloor \frac{n}{2} \rfloor \log(\lfloor \frac{n}{2} \rfloor) + 2\lceil \frac{n}{2} \rceil \log(\lceil \frac{n}{2} \rceil) + n \\ &\leq 2n \log(\lceil \frac{n}{2} \rceil) + n \end{aligned} \tag{0.4}$$

Since $\forall n > 2. \lceil \frac{n}{2} \rceil \leq \frac{n}{1.5}$ (for even numbers the ceiling function does nothing so this holds trivially; for odd numbers $\lceil \frac{n}{2} \rceil = \frac{n}{2} + 0.5 \leq \frac{n}{1.5} \Rightarrow 0.5 \leq \frac{n}{6} \Rightarrow 3 \leq n$), we can write (the last line holds since $2 \log(1.5) > 1$

$$\begin{aligned} T(n) &\leq 2n \log(\frac{n}{1.5}) + n \\ &= 2n \log(n) - 2n \log(1.5) + n \\ &\leq 2n \log(n) \end{aligned} \tag{0.5}$$

This completes the proof.

(b) (Additional Problem) Let $T(n) = 4T(n-2) + n$. Find an $f(n)$ such that $T(n) = \Theta(f(n))$ and prove that $T(n) = O(f(n))$. If you have time, try to prove that $T(n) = \Omega(f(n))$. You may assume $T(1) = 1$ and that n is odd (in other words, instead of proving that $\exists c, N \in \mathbb{N}. \forall n > N \in \mathbb{N} f(n) ? cg(n)$ you can prove that $\exists c, N \in \mathbb{N}. \forall n > N \in$ odd natural numbers$f(n) ? cg(n)$–where ? represents either $\leq$ or $\geq$)

Solution:
TF Note: This is a little bit of a longer proof and the solution doesn't really explain the process students may go through. The first step is to just try some $T(n)$s to get a sense of what the asymptotic runtime might be. Talking about the $4T(n-2)$ structure may suggest an exponential runtime. From there, students can try an asymptotic proof, but they will likely find that it doesn't work out of the gate. They may have trouble proving it for small numbers (which suggests setting a large N) and may find that the runtime isn't strictly less than 2^n even though it grows at that rate (suggests a $c > 1$).

I claim $T(n) = \Theta(2^n)$. First to prove that $T(n) = O(2^n)$. In particular, I will prove that $T(n) = O(2^n - n)$, which implies that $T(n) = O(2^n)$ since $2^n - n \leq 2^n$. Fix $N = 8$ and $c = 2$. First to show the base case for $n = 9$. We see that $T(9) = 565$ and $2(2^9 - 9) = 1006$, so clearly, $T(9) \leq 2(2^9 - 9)$.

Now for the inductive step, so fix n odd. We know by the inductive hypothesis since $n - 2$ is also odd that $T(n-2) \leq 2(2^{n-2} - (n-2)) = 2^{n-1} - 2n + 4$. We know from the recurrence relation that

$$\begin{aligned} T(n) &= 4T(n-2) + n \\ &\le 4(2^{n-1} - 2n + 4) + n \\ &\le 2 \cdot 2^n - 8n + 16 + n \\ &\le 2 \cdot 2^n - 7n + 16 \end{aligned} \tag{0.6}$$

In particular, since $N = 8, n > 8$, so $2n > 16$, so $-2n < -16$, so

$$T(n) \le 2 \cdot 2^n - 5n$$

Furthermore, $n > 8 \Rightarrow -3n \le 0$, so

$$T(n) \le 2 \cdot 2^n - 2n = 2(2^n - n)$$

Thus, we see that this completes the inductive step so $T(n) = O(2^n - n)$, so $T(n) = O(2^n)$

Now to show that $T(n) = \Omega(2^n)$. Let $N = 5$ and $c = 1$. We will show that for all odd $n > N$, $2^n \le T(n)$. Let us start with the base case of $n = 5$

$$\begin{aligned} T(5) &= 4T(3) + 5 \\ &= 4(4T(1) + 3) + 5 \\ &= 4(4 + 3) + 5 \\ &= 28 + 5 \\ &= 33 \end{aligned} \tag{0.7}$$

Since $2^5 = 32$ and $32 < 33$, this completes the proof of the base case. Now for the inductive step assume that for some odd n, $2^n \le T(n)$. Let us consider bounding 2^{n+2}, and using our inductive hypothesis, we have

$$2^{n+2} = 2^n 2^2 \le T(n) 2^2 = 4T(n) < 4T(n) + n + 2$$

Since $4T(n) + n + 2 = T(n+2)$, we have that

$$2^{n+2} \le T(n+2)$$

This completes the inductive proof, and we have that $T(n) = \Omega(2^n)$, so $T(n) = \Theta(2^n)$.

(c) (Challenge Problem) Prove the master theorem, which says that if $T(n) = aT(n/b) + cn^k$ for $a \ge 1, b \ge 2$ integers and $c, k > 0$ then

$$T(n) = \begin{cases} \Theta(n^{\log_b a}) & a > b^k \\ \Theta(n^k \log n) & a = b^k \\ \Theta(n^k) & a < b^k \end{cases} \tag{0.8}$$

Solution:
This is a great explanation. It might be worth just going through this with students instead of having them solve the problem because it is a fairly technical problem.

Problem 3: Review

Learning goal: The goal for this section is to reflect on how you can improve your proof skills!

(a) Select the problem from a previous week that you think you would learn the most from redoing. Copy the problem and your instructor's feedback below, rewrite the proof, and add a short reflection on the changes you made.

(b) (Additional Question) Choose another problem you received feedback on and copy the problem and feedback you received, rewrite the proof, and write a sentence about why you made your changes (so do Part A again) OR solve a problem you left blank from a previous week (try to pick from a week you felt least secure on).

Problem 5: Logistics

Purpose: This helps us make sure the course is going at the right speed!

(a) How long did you spend on the videos and readings this week?

(b) How long (including time in problem sessions) did you spend on this problem set?

(c) Do you have any feedback about the course in general (did the videos and readings sufficiently prepare you for the problem set)?