ANALYSIS OF ALGORITHMS – COMP 4040

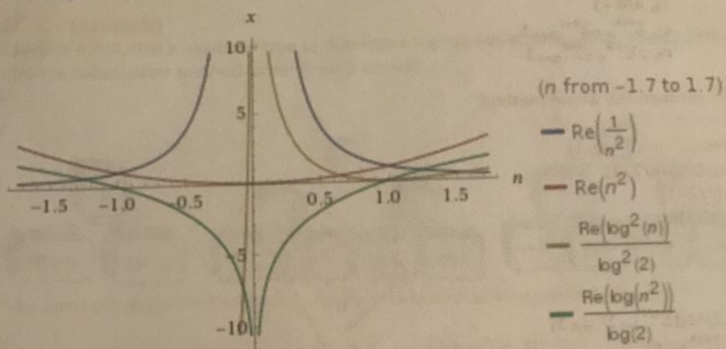
ASSIGNMENT-2 SOLUTIONS

1. Credits: Taylor M. Langlois

List the 4 functions below in non-decreasing asymptotic order of growth.

$(\log n)^2$ $\quad n^{-2} \quad \log_2(2^{\log(n^{\wedge}2)}) \quad n^2$

Justify your answer mathematically by showing values of c and n_0 for each pair of functions that are adja[cent in the] ordering.



(Graph courtesy of Wolfram Alpha)

Non-decreasing asymptotic order of growth: 1. n^{-2}, 2. $\log_2(2^{\log(n^{\wedge}2)})$, 3. $(\log n)^2$, 4. n^2

To prove this, we can take the first 2 values (1 and 2) we've ordered and plug them in c=1/2:

$n^{-2} \le c \log_2(2^{\log(n^{\wedge}2)})$

$1/n^2 \le c \log_2 n^2$

$1/n_0^2 = 2c \log_2 n_0$

$1/n_0^2 = \log_2 n_0$

We get $\sqrt{2}^{-2} = ½ = \log_2 \sqrt{2}$.

So, c= ½, $n_0 = \sqrt{2}$.

Now working with the second pair of values (2 and 3) we can plug in c=2:

$2 \log_2 n \le c (\log_2 n)^2$

$2 \le \log_2 n$

$1 = \log_2 n_0$

$2^1 = 2^{\log\_2 n\_0} = n_0$

Here we get 2 through division of $\log_2 n$.

So, c=2, n_0 = 2.

Finally, we get to the last pairs (3 and 4) and can plug in c=4:

$(\log_2 n)^2 \le cn^2$

$(\log_2 n_0)^2 = 4n_0^2$

Here we get ½ since $(\log_2 (1/2))^2 = 1 = 4(1/2)^2$.

So, c=4, n_0 = ½.

2. Credits: Ryan Cauble

85

2. Pseudocode Analysis (25 points)
For the pseudocode below for procedure Mystery(n), derive tight upper and lower bounds on its asymptotic ***worst-case*** running time $f(n)$. That is, for the set of inputs including those that force Mystery to work its hardest, find $g(n)$ such that $f(n) \in \theta(g(n))$. Assume that the input n is a positive integer. Justify your answer.

Mystery (n)	
1. if n is an even number	$C_1, 1$
2. for $i = 1$ to n	$C_2, n+1$
3. for $j = n$ down to $n/2$	$C_3, \Sigma_{i=1}^{n} \Sigma_{j=\frac{n}{2}}^{n+1} 1$
4. print "even number"	$C_4, \Sigma_{i=1}^{n} \Sigma_{j=\frac{n}{2}}^{n+1} 1$
5. else	$C_5, 1$
6. for $k = 1$ to $n/4$	$C_6, \frac{n}{4} + 1$
7. for $m = 1$ to n	$C_7, \Sigma_{k=1}^{\frac{n}{4}} \Sigma_{m=1}^{n+1} 1$
8. print "odd number"	$C_7, \Sigma_{k=1}^{\frac{n}{4}} \Sigma_{m=1}^{n} 1$

When you sum up the even part lines 1-4 you get:
$T(n) = an^2 + bn + c \in \Theta(n^2)$

When you sum up the odd part lines 5-8 you get:
$T(n) = an^2 + bn + c \in \Theta(n^2)$

So the Worst case is $\Theta(n^2)$

3. Credits: Victoria Albanese

Victoria Albanese
Analysis of Algorithms

October 5, 2017
Homework 2

3. **True or False**: 25 points

25

a. $n\lg^2 n \in O(n^2)$
b. $n\lg^2 n \in \Omega(n^{1.05})$
c. $n^3 \in o(n^3)$
d. The cost of the loop to the right is $O(n)$
e. The cost of the loop to the right is $\Omega(\lg n)$

```
for (i = 1; i <= n; i*=2) // n ≥ 1
{
        constant work;
}
```

a) $n\lg^2 n = (n)(\lg n)(\lg n) \approx (n^1)(n^{0.000\ldots1})(n^{0.000\ldots1}) \approx n^{1.000\ldots2} \in O(n^2)$

→ The power $1.000\ldots2 < 2$, so n^2 is a valid asymptotic upper bound

→ **TRUE**

b) $n\lg^2 n = (n)(\lg n)(\lg n) \approx (n^1)(n^{0.000\ldots1})(n^{0.000\ldots1}) \approx n^{1.000\ldots2} \notin \Omega(n^2)$

→ The power $1.000\ldots2 < 2$, so n^2 is NOT a valid asymptotic lower bound

→ **FALSE**

c) $n^3 \notin o(n^3)$

→ The o-notation requires the functions to be loosely upper bounded

→ This function would be tightly bound, which is not allowed

→ **FALSE**

d) loop cost = $\log n = n^{0.000\ldots1} \in O(n)$

→ The power $0.000\ldots1 < 1$, so n is a valid asymptotic upper bound

→ **TRUE**

e) loop cost = $\lg n \in \Omega(\lg n)$

→ The Ω allows for tight bounds, so this is a valid asymptotic lower bound

→ **TRUE**

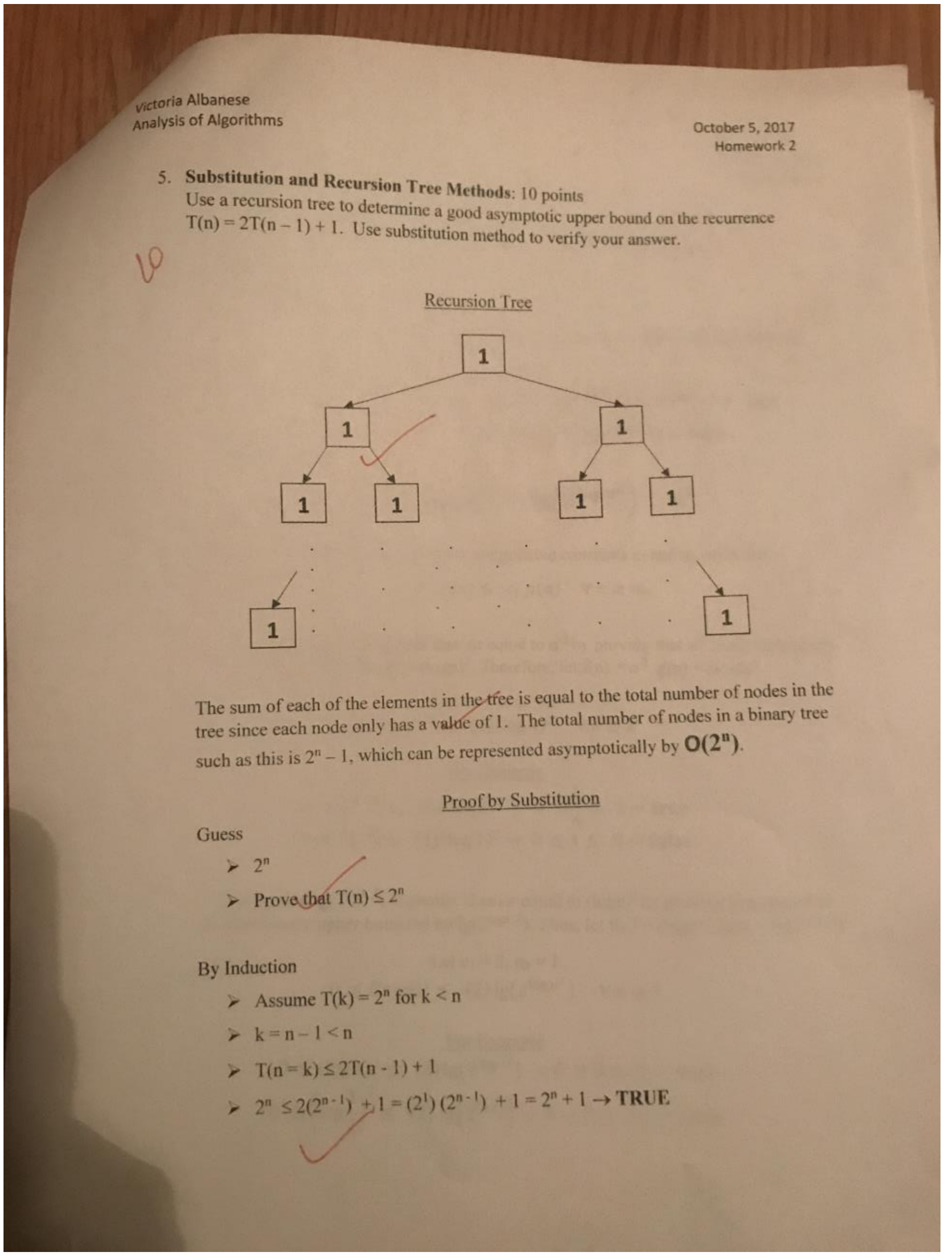
4. Credits: Taylor M. Langlois

4. (20 points)

For each of the following recurrences, give an expression for the runtime $T(n)$ if the recurrence can be solved with the Master Theorem. Otherwise, explain why the Master Theorem does not apply. Justify your answer.

1. $T(n) = 3^n T(n/3) + n^3$ — NO, a is not a constant since its value is 3^n.
2. $T(n) = 5T(n/2) + \sqrt{10n^3}$ — $T(n) = \Theta(n^3)$ due to case 1 in the master theorem.
3. $T(n) = 1/4T(n/4) + n \log n$ — NO, the a value of ¼ for a doesn't work because it must be $a \geq 1$.
4. $T(n) = T(n-1) + 2n$ — NO, our value over b is n-1, not just n like the master theorem uses.
5. $T(n) = 16T(n/4) + n^2$ — $T(n) = \Theta(n^2 \log n)$ due to case 2 in the master theorem.

5. Credits: Victoria Albanese



Victoria Albanese
Analysis of Algorithms

October 5, 2017
Homework 2

5. **Substitution and Recursion Tree Methods**: 10 points
Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n) = 2T(n-1) + 1$. Use substitution method to verify your answer.

10

Recursion Tree

The sum of each of the elements in the tree is equal to the total number of nodes in the tree since each node only has a value of 1. The total number of nodes in a binary tree such as this is $2^n - 1$, which can be represented asymptotically by $\mathbf{O(2^n)}$.

Proof by Substitution

Guess

- 2^n
- Prove that $T(n) \le 2^n$

By Induction

- Assume $T(k) = 2^n$ for $k < n$
- $k = n - 1 < n$
- $T(n = k) \le 2T(n - 1) + 1$
- $2^n \le 2(2^{n-1}) + 1 = (2^1)(2^{n-1}) + 1 = 2^n + 1 \rightarrow$ **TRUE**