CS213: Data Structures and Algorithms **Deadline: 12:00 PM, Feb-09 2021**

Task #2

Instructor: Prof. Sharat Chandran

1 Composite Recurrence

Consider two functions f(int n), g(int n) defined as:

```
def f(int n):
    if n == 1:
        return 1
    return f(n-1) + g(n-1)

def g(int n):
    if n == 1:
        return 1
    if (n%2 == 0):
        return g(n/2)
    return g((n+1)/2)
```

What is the big-Oh complexity of $f(n)$?

Put your answer here!!!
Recurrence:

$$f(1) = O(1)$$

$$f(n) = f(n-1) + g(n-1)$$

$$g(n) = g(n/2) + O(1)$$

Note that we are ignoring floor function as we simply wish to find the big-Oh for $f(n)$.
Guess:

$$g(n) = \log n$$

Proof: By induction the base case is trivial, the inductive hypothesis is:

$$g(n) = \log n = \log 2(n/2) = \log n/2 + 1 = g(n/2) + O(1) \quad \square$$

$$\therefore f(n) = f(n-1) + \log n - 1$$

$$f(n) - f(n-1) = \log n - 1$$

$$\vdots$$

$$f(2) - f(1) = 1$$

$$\implies f(n) = \sum_{k=2}^{n-1} \log k + f(1)$$

$$\implies f(n) = O(n \log n) \quad \square$$

2 More Recurrence

Let

$$T(n) = T(\frac{n}{4}) + T(\frac{n}{2}) + 1$$

Take the base case as $T(1) = 1$ and you can assume n to be an even power of 2 so that the inputs to T are always integers.
Find $T(n)$ in terms of Ω notation.

Put your answer here!!!
Since we wish to find a lower bound for the given function, (Ω), note that:

$$T(n/2) \geq T(n/4) \implies T(n) \geq 2T(n/4) + 1$$

Let

$$\begin{aligned} F(n) &\geq 2F(n/4) + 1 \\ \text{by master's theorem:} \\ F(n) &= aF(n/b) + f(n) \\ \text{here } a = 2, b = 4 \text{ and } f(n) &= 1 \end{aligned}$$

for $\epsilon = 1/2 > 0$,

$$f(n) = 1 \in O(n^{\log_b a - \epsilon})$$

$$\implies F(n) \in \Theta(\sqrt{n})$$

$$\implies T(n) \in \Omega(\sqrt{n}) \quad \square$$

Also note that this is not a tight lower bound as the given function can be upper bounded by O(n) by noting that

$$T(n) \leq 2T(n/2) + 1$$

and applying master's theorem on the same.

3 Transformed Recurrences

Let

$$T(n) = 2T(\sqrt{n}) + c$$

and the base case:

$$T(2) = T(1) = 1$$

Give a bound on $T(n)$ in terms of Θ notation. Prove how you obtained the bound.

For simplicity you may consider n to be of a form that ensures only integers are found when you unroll the recursion (that is the inputs to T are always integers).

Put your answer here!!!
(Note that I am making the assumption that $n^{1/2^k}$=1 or =2 for some $k \in \mathbb{N}$)

$$\begin{aligned}
T(n) &= 2T(\sqrt{n}) + c \\
&= 2(2T(n^{1/4}) + c) + c \\
&\vdots \\
&= 2^{\log\log n} + c \cdot (1 + 2 + \cdots + 2^{\log\log n - 1}) \\
&= (c+1) \cdot (2^{\log\log n}) - c
\end{aligned}$$

Therefore, we simply have

$$T(n) \in \Theta(2^{\log\log n})$$

$$T(n) \in \Theta(\log n)$$

4 His Master's Voice

For each of the recurrences below, state whether the master theorem is applicable or not. If yes, state to which of the three cases the recursion belongs to and find the asymptotic bound. If not, state reasons why the theorem is not applicable. In the cases where master theorem is not applicable, can you find the asymptotic bound using other methods? *(this is not necessary but may fetch you bonus marks)*

The base case for each of these recurrences is $T(1) = \Theta(1)$

(i) $T(n) = 4T(\frac{n}{2}) + n^2 \log^4 n$

Put your answer here!!!
Claim: The master's theorem is not applicable
Proof: Here $a = 4$, $b = 2$ and $f(n) = n^2 \log^4 n$. By master's theorem we need to check three cases and see which is applicable to $f(n)$.

(a) Case 1: Assume $\exists \epsilon > 0 : f(n) \in O(n^{\log_b a - \epsilon})$.
But $f(n) = n^2 \log^4 n \implies f(n) \geq n^2 (\forall n > 2)$
So $f(n) \notin O(n^{2-\epsilon})$ for any $\epsilon > 0$ $\perp$.

(b) Case 2: We know already that $f(n) \notin \Theta(n^2)$ so Case 2 isn't applicable.

(c) Case 3: Observe that

$$\begin{aligned} \exists n_0 \in \mathbb{N} : n^\epsilon > \log^4 n \ \forall n > n_0 \\ \implies f(n) \notin \Omega(n^{2+\epsilon}) \end{aligned}$$

Case 3 fails as well. So master's theorem fails.

(ii) $T(n) = T(\frac{n}{2}) + \tanh n$

Put your answer here!!!
Note:

$$-1 \leq \tanh n \leq 1 \implies \tanh n \in \Theta(1)$$

Also, for the masters theorem here $a = 1$, $b = 2$, $f(n) = \tanh n \in \Theta(n)$.
Therefore we observe that Case 2 of the master's theorem is applicable as we have:

$$f(n) \in \left(\Theta(1) = \Theta(n^{\log_b a})\right)$$
$$\therefore T(n) \in \Theta(\log n) \quad \square$$

(iii) $T(n) = T(\frac{n}{2}) + n(2 - \cos n)$

Put your answer here!!!
Claim: The master's theorem is not applicable
Proof: Here $a = 1$, $b = 2$ and $f(n) = n(2 - \cos n)$. By master's theorem we need to check three cases and see which is applicable to $f(n)$.

(a) Case 1: Assume $\exists \epsilon > 0 : f(n) \in O(n^{\log_b a - \epsilon})$.
But $f(n) = n(2 - \cos n) \implies n \leq f(n) \leq 3n (\forall n \in \mathbb{N})$
So $f(n) \in \Theta(n)$

$$\implies f(n) \notin O(n^{0-\epsilon})$$

A contadiction. So case 1 fails.

(b) Case 2: We know already that $f(n) \notin \big(\Theta(n^0) = \Theta(1)\big)$ so Case 2 isn't applicable.

(c) Case 3: $f(n) \in \Omega(n^{0+\epsilon})$ for any $\epsilon > 1$ however we are unable to satisfy the second condition that is:

$$\exists c < 1 : af(n/b) \leq cf(n) \; \forall n > n_0$$

For n=6, we need $c > 1.45$ to find any suitable threshold n_0. Thus case 3 fails as well.

5 I Hate Loops!!!

Algorithm 1: : I Hate Loops!!!

```
int a = 0;
for i=1; i≤n; i++ do
    for j=i; j≤n; j+=i do
        a++;
    end
end
```

Find the asymptotic complexity of the above code in terms of n.

Put your answer here!!!
We shall break this problem step by step to observe a pattern and make a guess:

$$
\begin{aligned}
i = 1 &\implies \text{inner loop runs for } n \text{ iterations} \\
i = 2 &\implies \text{inner loop runs for } \left\lfloor \frac{n}{2} \right\rfloor \text{ iterations} \\
i = 3 &\implies \text{inner loop runs for } \left\lfloor \frac{n}{3} \right\rfloor \text{ iterations} \\
&\vdots \\
i = n &\implies \text{inner loop runs for } \left\lfloor \frac{n}{n} \right\rfloor \text{ iterations}
\end{aligned}
$$

Ignoring floors and summing each inner loop run gives:

$$= n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)$$

It is also well know that the upper bound of the harmonic series upto n terms is

$$\log n + 1$$

The asymptotic complexity is therefore,

$$\Theta(n \log n) \quad \square$$