Name & Surname:
ID:

CSE 321 - Quiz #1

October 11th, 2024

In all questions, you must justify your answer to receive credit.

1. ***48 pts.*** For the following functions determine which computational complexity class the function belongs to, by using limit approach. Provide a mathematical proof and make sure you solve any indeterminate forms.

   (a) $f(n) = 3^n + 3$ and $g(n) = 3 \cdot n^3$

   (b) $f(n) = \sum_{x=1}^{n} (x+5)$ and $g(n) = n^2 + 5 \cdot n$

   (c) $f(n) = log_2 n + n^2$ and $g(n) = log_2 n^2 + n$

   (d) $f(n) = \sqrt{n}$ and $g(n) = log_2 n + \sum_{n=1}^{5} n$

   (e) $f(n) = e^n$ and $g(n) = 2^{n+1}$

   (f) $f(n) = 2^{n+1}$ and $g(n) = 3^{n-1}$

Solution 1

$$\boxed{\begin{array}{l} f(n) \in O(g(n)) \longrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \\ f(n) \in \Omega(g(n)) \longrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \\ f(n) \in \Theta(g(n)) \longrightarrow 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} = c < \infty \end{array}}$$

(a) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{3^n+3}{3n^3} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{3^n \cdot ln3}{9n^2} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{3^n \cdot (ln3)^2}{18n}$

$\xrightarrow{L'hospital} \lim_{n\to\infty} \frac{3^n \cdot (ln3)^3}{18} = \infty$

Thus, $f(n) \in \Omega(g(n))$.

(b) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\sum_{x=1}^{n}(x+5)}{n^2+5n} = \lim_{n\to\infty} \frac{\frac{n(n+1)}{2}+5n}{n^2+5n} = \lim_{n\to\infty} \frac{\frac{n^2+11n}{2}}{n^2+5n} = \lim_{n\to\infty} \frac{n^2+11n}{2n^2+10n} \xrightarrow{L'hospital}$

$\lim_{n\to\infty} \frac{2n+11}{4n+10} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{2}{4} = \frac{1}{2}$

Thus, $f(n) \in \Theta(g(n))$.

(c) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{log_2 n+n^2}{log_2 n^2+n} = \lim_{n\to\infty} \frac{log_2 n+n^2}{2log_2 n+n} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{\frac{1}{ln2\cdot n}+2n}{\frac{2}{ln2\cdot n}+1} = \lim_{n\to\infty} \frac{\frac{1+2\cdot ln2\cdot n^2}{ln2\cdot n}}{\frac{2+ln2\cdot n}{ln2\cdot n}} =$

$\lim_{n\to\infty} \frac{1+2\cdot ln2\cdot n^2}{2+ln2\cdot n} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{4\cdot ln2\cdot n}{ln2} = \lim_{n\to\infty} 4\cdot n = \infty$

Thus, $f(n) \in \Omega(g(n))$.

(d) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\sqrt{n}}{log_2 n+\sum_{n=1}^{5} n} = \lim_{n\to\infty} \frac{\sqrt{n}}{log_2 n+15} \xrightarrow{L'hospital} \lim_{n\to\infty} \frac{\frac{1}{2\cdot\sqrt{n}}}{\frac{1}{ln2\cdot n}} =$

$\lim_{n\to\infty} \frac{ln2\cdot n}{2\cdot\sqrt{n}} = \lim_{n\to\infty} \frac{ln2\cdot\sqrt{n}}{2} = \infty$

Thus, $f(n) \in \Omega(g(n))$.

(e) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{e^n}{2^{n+1}} = \lim_{n\to\infty} \frac{e^n}{2 \cdot 2^n} = \lim_{n\to\infty} \frac{1}{2} \cdot (\frac{e}{2})^n = \infty \quad$ (Since $\frac{e}{2} > 1$)

Thus, $f(n) \in \Omega(g(n))$.

(f) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{2^{n+1}}{3^{n-1}} = \lim_{n\to\infty} \frac{2 \cdot 2^n}{\frac{1}{3} \cdot 3^n} = \lim_{n\to\infty} 6 \cdot (\frac{2}{3})^n = 0 \quad$ (Since $\frac{2}{3} < 1$)

Thus, $f(n) \in O(g(n))$.

2. ***12 pts.*** Prove that $f(n) = \sum_{x=1}^{n} (x^3) \in \theta(n^4)$ by using the integration method. Make sure you show upper and lower bounds to complete your proof.

Solution 2

Since $f(n)$ is a non-decreasing function, we can use integral as follows:

$\int_0^n g(x)dx \leq \sum_{i=1}^{n} g(i) \leq \int_1^{n+1} g(x)dx$

$\int_0^n x^3 dx \leq f(n) \leq \int_1^{n+1} x^3 dx$

$\frac{x^4}{4}\Big|_0^n \leq f(n) \leq \frac{x^4}{4}\Big|_1^{n+1}$

$\frac{n^4}{4} \leq f(n) \leq \frac{(n+1)^4-1}{4}$

Since $\frac{n^4}{4} \leq f(n)$, $f(n) \in \Omega(n^4)$

Since $f(n) \leq \frac{(n+1)^4-1}{4}$, $f(n) \in O(n^4)$

Therefore, $f(n) \in \Theta(n^4)$

3. ***40 pts.*** The Fibonacci numbers are a sequence defined by the recurrence relation $F(n) = F(n-1) + F(n-2)$ with initial conditions $F(0) = 0$ and $F(1) = 1$, resulting in the series $0, 1, 1, 2, 3, 5, 8, 13, \ldots$

   Following is the pseudocode of an algorithm that determines whether a given non-negative integer is a Fibonacci number. Analyze the pseudocode and provide the best and worst-case time complexities of the algorithm. Explain your answer by referring to the pseudocode.

```
IsFibonacci($x$):
    Input: An non-negative integer $x$.
    Output: Boolean expression indicating whether $x$ is a Fibonacci number or not.
    begin
        if $x == 0$ or $x == 1$ then
            return True
        else
            Fibonacci ← [0, 1]
            Index ← 1
            while $x >$ Fibonacci[Index] repeat
                Index ← Index + 1
                Append Fibonacci[Index - 1] + Fibonacci[Index - 2] to Fibonacci
                if $x ==$ Fibonacci[Index] then
                    return True
                end if
            end while
            return False
        end if
    end
```

Hint: $n \approx \frac{\log(\sqrt{5} \cdot x)}{\log \phi}$ where x is the n^{th} Fibonacci number and ϕ is the golden ratio (≈ 1.618). You can provide an approximate time complexity based on this formula.

Solution 3

Best-Case Complexity: If the given input is 0 or 1, the program will return within the first if statement. Thus, it will take constant time.

The best-case time complexity is $O(1)$.

Worst-Case Complexity: The worst-case scenario is when the input is not a Fibonacci number. In this case, the program should go into the while loop. In each iteration of this loop, a new Fibonacci number will be calculated. If the last element of the Fibonacci array is larger than the input, then the loop is going to end. Since all of the operations in the given pseudo-code (assignments, if statements, etc.) take constant time the loop is going to define the worst-case time complexity. In this case, it is necessary to determine how many iterations the loop will have. Based on the given hint, if x is a Fibonacci number, it is the n^{th} Fibonacci number where $n \approx \frac{\log(\sqrt{5} \cdot x)}{\log \phi}$. If x is not a Fibonacci number, then we will have to check the first n Fibonacci numbers to determine this. After the n^{th} Fibonacci number, the last element of the Fibonacci array is going to be larger than the input (x). In other words, the loop will have n iterations. To be precise, it is going to have $n-2$ iterations, considering the first two Fibonacci numbers are being checked in an if statement. But since -2 doesn't affect the asymptotic complexity, we can ignore it.

Since the loop defines the worst-case time complexity and it has n iterations, we can say that the worst-case time complexity is $O(n) \approx O(\frac{\log(\sqrt{5} \cdot x)}{\log \phi}) \approx O(logx)$.