Esra Eryılmaz
171044046

- CSE 321 -

Homework 1

6.11.2020

(1) For each of the following statements, specify whether it is true or not. Explain your reasoning for each of them.

a) $\log_2 n^2 + 1 \in O(n)$

$\log_2 n^2 + 1 \in O(n)$ iff $\boxed{\log_2 n^2 + 1 \leq c \cdot n}$ $n \geq n_0$

with positive constant c and n_0

choose $c=2$

$\log_2 n^2 + 1 \leq 2 \cdot n$

$2 \log_2 n + 1 \leq 2n$

$2 \log_2 n \leq 2n - 1$

$\log_2 n \leq \frac{2n-1}{2}$

logarithm rule

$\log_m a \leq b$

if $m > 1 \rightarrow a \leq m^b$

$n \leq 2^{\frac{2n-1}{2}}$

$n \leq 2^{\frac{2n}{2} - \frac{1}{2}}$

$n \leq 2^n \cdot 2^{-\frac{1}{2}}$

linear ↵ ↳ exponential // It is obvious exponential is bigger.

(where $c=2$) For all $n \geq 1$ this statement is true

b) $\sqrt{n(n+1)} \in \Omega(n)$

$\sqrt{n(n+1)} \in \Omega(n)$ iff $\boxed{\sqrt{n(n+1)} \geq c \cdot n}$ $n \geq n_0$

with positive constant c and n_0

$c=1$

$\left(\sqrt{n^2+n}\right)^2 \geq (1 \cdot n)^2$

$n^2+n \geq n^2$

$n \geq 0$

(where $c=1$). For all $n \geq 0$ this statement is true.

c) $n^{n-1} \in \Theta(n^n)$

$n^{n-1} \in \Theta(n^n)$ iff $\boxed{c_1 \cdot n^n \leq n^{n-1} \leq c_2 \cdot n^n}$ $n \geq n_0$

with positive constant c_1, c_2, n_0

$$\underset{c_1}{\frac{1}{2}} \cdot n^n \leq n^n \cdot n^{-1} \leq \underset{c_2}{2} \cdot n^n$$

Divide all sides with n^n

$$\frac{1}{2} \leq \frac{1}{n} \leq 2$$

For all $n \geq 1$ it provides that side but it doesn't provides that side. So $\Theta(n^n)$ statement is not true.

(we can say $O(n^n)$)

d) $O(\overbrace{2^n + n^3}^{f(n)}) \subset O(\overbrace{4^n}^{g(n)})$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{2^n + n^3}{4^n} = \lim_{n\to\infty} \frac{2^n}{4^n} + \lim_{n\to\infty} \frac{n^3}{4^n}$$

$$= \underbrace{\lim_{n\to\infty} \left(\frac{1}{2}\right)^n}_{0} + \underbrace{\lim_{n\to\infty} \left(\frac{n^3}{4^n}\right)}_{0} = 0 + 0 = 0$$

(Exponential grows faster than cubic)

So $4^n \geq 2^n + n^3$ For all $n \geq 0$ this statement is true

---

e) $O(\overbrace{2 \log_3 \sqrt[3]{n}}^{f(n)}) \subset O(\overbrace{3 \log_2 n^2}^{g(n)})$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{2 \cdot \log_3 \sqrt[3]{n}}{3 \log_2 n^2} = \lim_{n\to\infty} \frac{2 \cdot \log_3 n^{1/3}}{3 \cdot 2 \cdot \log_2 n} =$$

$$\lim_{n\to\infty} \frac{\cancel{2} \cdot \frac{1}{3} \cdot \log_3 n}{3 \cdot \cancel{2} \cdot \log_2 n} = \lim_{n\to\infty} \frac{\log_3 n}{9 \cdot \log_2 n} = \frac{1}{9} \lim_{n\to\infty} \frac{\log_3 n}{\log_2 n}$$

$$= \frac{1}{9} \lim_{n\to\infty} \left(\frac{\frac{\cancel{\log_e n}}{\log_e 3}}{\frac{\cancel{\log_e n}}{\log_e 2}} \right) = \frac{1}{9} \lim_{n\to\infty} \frac{\ln 2}{\ln 3} = \frac{1}{9} \cdot \frac{\ln 2}{\ln 3}$$

Result is constant. So we can say that

$f(n) \in \Theta(g(n)) \iff g(n) \in \Theta(f(n))$ also we can say

$f(n) \in O(g(n)) \iff O(f(n)) \subset O(g(n))$

$O(2 \log_3 \sqrt[3]{n}) \subset O(3 \log_2 n^2)$

That statement is true.

f) $\log_2\sqrt{n}$ and $(\log_2 n)^2$ are of the same asymptotical order

$\log_2\sqrt{n} = \log_2 n^{1/2} = \frac{1}{2}\log_2 n \xrightarrow{\text{get rid of constants}} \log_2 n$

$(\log_2 n)^2 = \log_2^2 n$

If assume $n \geq 1$ we have $\log_2 n \geq 1$

we have $\log_2^2 n = \boxed{\log_2 n \cdot \log_2 n \geq \log_2 n}$

$O(\log n)$ is faster than $O(\log^2 n)$

So statement is not true.

2nd way

Also we can write $\lim_{n\to\infty} \frac{(\log_2 n)^2}{\log_2\sqrt{n}} = \frac{\infty}{\infty} \xRightarrow{\text{L. Hospital}} \lim_{n\to\infty} \frac{\frac{2\log_2 n}{(\ln 2)x}}{\frac{1}{2(\ln 2)x}}$

$= \lim_{n\to\infty} 4\log_2 n = 4\lim_{n\to\infty}\log_2 n = \infty$

$\log^2 n$ time complexity is slower.

---

(2) Order the following functions by growth rate and explain your reasoning for each of them

$n^2, n^3, n^2\log n, \sqrt{n}, \log n, 10^n, 2^n, 8^{\log n}$

$n^2, n^3 \Rightarrow \lim_{n\to\infty}\frac{n^2}{n^3} = \frac{\infty}{\infty} \xrightarrow{\text{L Hospital}} \lim_{n\to\infty}\frac{2n}{3n^2} = 0$ so $\boxed{n^3 > n^2}$

$8^{\log n}, n^3 \Rightarrow 8^{\log n} = n^{\log 8} = n^{\log_2 2^3} = n^{3\cdot\log_2 2} = n^3$ so $\boxed{8^{\log n} = n^3}$

$n^2, n^2\log n$ ⇒ $\lim_{n\to\infty} \frac{n^2}{n^2\log n} = 0$ so $\boxed{n^2\log n > n^2}$

$n^2\log n, n^3$ ⇒ $\lim_{n\to\infty} \frac{n^2\log n}{n^3} = \lim_{n\to\infty} \frac{\log n}{n} = 0$ so $\boxed{n^3 > n^2\log n}$

(linear grows faster than logarithmic)

$n^2, \sqrt{n}$ ⇒ $n^2, n^{1/2}$ we can easily see that $\boxed{n^2 > \sqrt{n}}$

$\lim_{n\to\infty} \frac{\sqrt{n}}{n^2} = \lim_{n\to\infty} \frac{n^{1/2}}{n^2} = \lim_{n\to\infty} \frac{1}{n^{3/2}} = 0$ so

$\sqrt{n}, \log n$ ⇒ Let $n = 2^{2k}$ $\rightarrow \sqrt{n} = \sqrt{2^{2k}} = 2^k$ $\quad 2^k$ vs $2k$

$\rightarrow \log n = \log_2 2^{2k} = 2k$

$\lim_{n\to\infty} \frac{2k}{2^k} = 0$ so $\boxed{\sqrt{n} > \log n}$

$2^n, 10^n$ ⇒ $\lim_{n\to\infty} \frac{2^n}{10^n} = \lim_{n\to\infty} \frac{1^n}{5^n} = \lim_{n\to\infty} \frac{1}{5^n} = 0$ so $\boxed{10^n > 2^n}$

$n^3, 2^n$ ⇒ $\lim_{n\to\infty} \frac{n^3}{2^n} = \frac{\infty}{\infty} \xrightarrow{\text{L'Hospital}} \lim_{n\to\infty} \frac{3n^2}{2^n \cdot \ln 2} \xrightarrow{\text{L'Hospital}}$

$= \lim_{n\to\infty} \frac{6n}{\ln^2(2)\cdot 2^n} \xrightarrow{\text{L'Hospital}} \lim_{n\to\infty} \frac{6}{\ln^3(2)\cdot 2^n} = \frac{6}{\ln^3(2)} \lim_{n\to\infty} \frac{1}{2^n}$

$= 0$ so $\boxed{n^3 < 2^n}$

$$\boxed{\log n < \sqrt{n} < n^2 < n^2\log n < 8^{\log_2 n} = n^3 < 2^n < 10^n}$$

logarithmic   quadratic   Cubic   Exponential   Exponential

(3) What is the time complexity of the following programs?
Explain by giving details.

a)
```
void f( int my_array[]){
        for(int i=0;i<sizeofArray;i++){
            if(my_array[i]<first_element){
                second_element=first_element;
                first_element=my_array[i];
            }
            else if(my_array[i]<second_element){
                if(my_array[i]!= first_element){
                    second_element= my_array[i];
                }
            }
```

n

"To find second smallest element in the array" I think that is the algorithm of the code above.

walk through all "n" elements and find the second smallest element.

→ In other words traverse the array twice.
In the first traversal find the minimum element.
In the second travelsal find the smallest element but greater than the first traversal.

Time complexity = $O(n)$ //

b)
```
void f(int n){
    int count=0;
    for(int i=2;i<=n;i++){
        if(i%2==0){
            count++;
        }
        else{
            i=(i-1)i;
        }
    }
}
```

Constant

There are two conditions in for loop. But all conditions keeps constant time to execute. So we can write that piece of code like that:

```
for (int i=2 ; i<n ; X)
    count++;
```

To find X.

$i=2$
$i=3$
$i=6$
$i=7$
$i=42$
$i=43$
⋮

we can get rid of these Because constant time

$2, 6, 42 \ldots \longrightarrow X = \boxed{i^2+i}$

So to find time complexity we can ignore $+i$ part. Because i^2 have control on $+i$ part.

In the for loop $i \Rightarrow 2, 2^t, 2^{t^t}, \ldots = 2, \ldots 2^{t^{\log_t(\log n)}}$

At the end of the for loop i must be equal or less than n

So we can write; $2^{t^{\log_t(\log n)}} = n \rightarrow 2^{\log n^{\log_t t}} = n$

$2^{\log_2 n} = n^{\log_2 2} = n$

the value of the last i (proof)

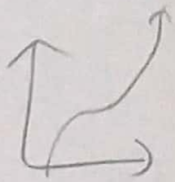
➜ So there are $\log_t(\log n)$ iterations at the execution.
And every iteration takes constant time
Because of that reasons the time complexity of the function is

$O(\log(\log n))$ //

4) Find the complexity classes of the following functions using the integration method

a) $\sum_{i=1}^{n} i^2 \log i \Rightarrow$ Let $f(n) = \sum_{i=1}^{n} g(n)$ where

$g(n)$ is a nondecreasing function



we can write

$$\boxed{\int_0^n x^2 \log x \, dx \leq f(n) \leq \int_1^{n+1} x^2 \log x \, dx}$$

$$\int x^2 \log x \, dx \Rightarrow \begin{array}{l} u = \log_2 x \\ v' = x^2 \end{array} \Rightarrow \left[\frac{1}{3} x^3 \log_2 x - \int \frac{x^2}{3 \ln 2} dx \right]$$

$$\int \frac{x^2}{3 \ln 2} dx = \frac{x^3}{9 \ln 2} \Rightarrow \int x^2 \log x \, dx = \left[\frac{1}{3} x^3 \log_2 x - \frac{x^3}{9 \ln 2} \right]$$

$$\left[\frac{1}{3} x^3 \log_2 x - \frac{x^3}{9 \ln 2} \right] \Big|_0^n \leq f(n) \leq \left[\frac{1}{3} x^3 \log_2 x - \frac{x^3}{9 \ln 2} \right] \Big|_1^{n+1}$$

$$\frac{1}{3} n^3 \log_2 n - \frac{n^3}{9 \ln 2} \leq f(n) \leq \frac{1}{3} \log_2 (n+1) (n+1)^3 - \frac{(n+1)^3}{9 \ln 2} + \frac{1}{9 \ln 2}$$

Get rid of constants and slower functions.. ?

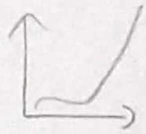
$$\underbrace{n^3 \log n}_{\text{lower bound}} \leq f(n) \leq \underbrace{(\log n) \cdot n^3}_{\text{upper bound}}$$

$$f(n) \in \Theta(n^3 \log n)$$

$$\boxed{\sum_{i=1}^{n} i^2 \log i \in \Theta(n^3 \log n)}$$

b) $\sum_{i=1}^{n} i^3 \Rightarrow$ Let $f(n) = \sum_{i=1}^{n} g(n)$ where

$g(n)$ is a non decreasing function



we can write

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

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

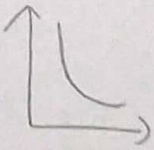
$$\frac{n^4}{4} - 0 \leq f(n) \leq \frac{(n+1)^4 - 1}{4}$$

$$f(n) \in \Theta(n^4) \longrightarrow \boxed{\sum_{i=1}^{n} i^3 \in \Theta(n^4)}$$

//

c) $\sum_{i=1}^{n} \frac{1}{2\sqrt{i}} \Rightarrow$ Let $f(n) = \sum_{i=1}^{n} g(n)$ where

$g(n)$ is a non increasing function



we can write

$$\boxed{\int_1^{n+1} \frac{1}{2\sqrt{x}}\,dx \leq f(n) \leq \int_0^n \frac{1}{2\sqrt{x}}\,dx}$$

$$\int \frac{1}{2\sqrt{x}}\,dx \Rightarrow \frac{1}{2}\int x^{-1/2}\,dx = \frac{1}{2}\left[\frac{x^{-1/2+1}}{-1/2+1}\right] = \frac{1}{2}\left[\frac{x^{1/2}}{1/2}\right] = \sqrt{x}$$

$$\left.\sqrt{x}\right|_1^{n+1} \leq f(n) \leq \left.\sqrt{x}\right|_0^n$$

$$\sqrt{n+1} - 1 \leq f(n) \leq \sqrt{n} - 0$$

$$f(n) \in \Theta(\sqrt{n}) \longrightarrow \sum_{i=1}^{n} \frac{1}{2\sqrt{i}} \in \Theta(\sqrt{n})$$

//

d) $\sum_{i=1}^{n} \frac{1}{i}$ $\Rightarrow$ Let $f(n) = \sum_{i=1}^{n} g(n)$ where

$g(n)$ is a nonincreasing function

we can write

$$\boxed{\int_{1}^{n+1} \frac{1}{x}\,dx \leq f(n) \leq \int_{0}^{n} \frac{1}{x}\,dx}$$

$$\int \frac{1}{x}\,dx = \ln|x|$$

$$\ln|x| \Big|_{1}^{n+1} \leq f(n) \leq \ln|x| \Big|_{0}^{n}$$

$$\underbrace{\ln|n+1| - \underset{0}{\ln 1}}_{\text{lower bound } \checkmark} \leq f(n) \leq \underbrace{\ln|n| - \underset{-\infty}{\ln|0|}}_{\text{upper bound } \times}$$

So this works to find a lower bound. We cannot find upper bound with this method.

But we can write;

$$\boxed{\sum_{i=1}^{n} \frac{1}{i} \in \Omega(\log n)}$$

//

5) Find the best case and worst case complexities of linear search with repeated elements, that is the elements in the list need not be distinct. Show your analysis.

```
function Linear Search (L[ ], x)
    for i=1 to n do
        if (L[i] == x) then
            return i;
        end if
    end for
    return
end
```

→ Pseudocode

<u>Best Case</u>: If the searched element is at the first place

$x = L[1]$

Then best occurs $\boxed{\Theta(1)}$ //

<u>Worst Case</u>: If the searched element is at the end

$x = L[n]$

Then worst occurs $\boxed{\Theta(n)}$ //