CSE 321 Homework 1 Solutions

Akif Kortal
171044098

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

By definition

$T(n) = O(f(n))$ such that

$T(n) \leq c.f(n) \quad n \geq n_0$ so;

$$\boxed{\begin{aligned} T(n) &= \log_2 n^2 + 1 \\ f(n) &= n \end{aligned}}$$

we need to show from definition

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

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

*if we take derivative both side

$\Rightarrow (2\log n + 1)' \leq n'$

$\Rightarrow \boxed{\frac{2}{n\ln 2} \leq 1 \text{ is true for } n \geq 2}$

$\boxed{\text{So, This is true.}}$

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

By definition

$T(n) = \Omega(f(n))$ such that

$T(n) \geq c.f(n) \quad n \geq n_0$

So $\boxed{\begin{aligned} T(n) &= \sqrt{n(n+1)} \\ f(n) &= n \end{aligned}}$

we need to show

$\sqrt{n(n+1)} \geq n$ (take square)

$n.(n+1) \geq n^2$

$\boxed{n^2 + n \geq n^2 \text{ is true } n \geq 1.}$

$\boxed{\text{So this is true!}}$

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

By defition

$T(n) = \Theta(f(n))$ if only if

$T(n) = O(f(n))$ and $T(n) = \Omega(f(n))$ so

We need to show $(n^{n-1} \in \Omega(n^n))$ | We need to show $(n^{n-1} \in O(n^n))$

$T(n) \geq c.f(n)$ (Ω-omega)

$T(n) = n^{n-1}$
$f(n) = n^n$

so;

$n^{n-1} \geq n^n$

$n^n . n^{-1} \geq n^n$

$\frac{n^n}{n} \geq n^n$ is not true for $n \geq 1$

$\boxed{\text{So } n^{n-1} \notin \Omega(n^n)}$

$T(n) \leq c.f(n)$ (O (big oh))

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

$n^n . n^{-1} \leq n^n$

$\frac{n^n}{n} \leq n^n$ is true for $n \geq 1$

$\boxed{\text{So } n^{n-1} \in O(n^n)}$

Conclusion

$n^{n-1} \in O(n^n)$ & $n^{n-1} \notin \Omega(n^n) \Rightarrow n^{n-1} \notin \Theta(n^n)$

So, This is not true. (false)

d) $O(2^n + n^3) \subset O(4^n)$

Let $A = O(2^n + n^3)$
$B = O(4^n)$

if A is a subset of B, every element of A is also an element of B.

if $f(n) = 4^n$ then
$T(n) \leq f(n)$ (for Big oh notation)
if $T(n) = O(2^n + n^3) = O(2^n)$ (2^n has bigger degree)
we need to show
$2^n \leq 4^n$ ($f(n) = 4^n$)
$2^n \leq 2^{2n}$ (divide with 2^n)
$\boxed{1 \leq 2^n \text{ for } n \geq 1}$

So since $O(2^n + n^3) \leq O(4^n)$ and every element in $O(2^n + n^3)$ are also elements of $O(4^n)$ as they are less than 4^n, this is true.

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

Let $A = O(2\log_3 \sqrt[3]{n})$ If A is subset of B, every

$B = O(3\log_2 n^2)$ element of A is also an element of B

if $f(n) = 3\log_2 n^2$ and

$T(n) = 2\log_3 \sqrt[3]{n}$ then;

$T(n) \leq f(n)$ $n \geq n_0$ (Bigoh rule)

$2\log_3 \sqrt[3]{n} \leq 3\log_2 n^2$

$\frac{2}{3}\log_3 n \leq 6\log_2 n$ (multiply with 3 and divide with 2)

$\log_3 n \leq 9\log_2 n$

is true for $n \geq 1$ so:

since $O(\log_3 n) < O(\log_2 n)$

(ignore constant terms)

This is true.

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

Let $f(n) = \log_2 \sqrt{n}$

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

$f(n) = \log_2 \sqrt{n} = \frac{\log n}{2} \in O(\log_2 n)$

$g(n) = (\log_2 n)^2 = \log_2 n . \log_2 n \in O(\log_2 n . \log_2 n)$

Since $O(\log_2 n) < O(\log_2 n . \log_2 n)$ (asymptotically)

they are not same asymptotical order.

(false)

2) if we order given functions by growth rate we will get;

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

So Now, we have to show 7 equality.

1) $\log n < \sqrt{n}$ (if we take logarithm both side)

$\Rightarrow \log(\log n) < \log(\sqrt{n})$

$\Rightarrow \log(\log n) < \dfrac{\log(n)}{2}$ is true for $n > 1$

2) $\sqrt{n} < n^2$ (take square both side)

$\Rightarrow n < n^4$ is true $n > 1$

3) $n^2 < n^2 \log n$ (divide both side with n^2)

$\Rightarrow 1 < \log n$ is true $n > 10$

4) $n^2 \log n < n^3$ (divide both side with n^2)

$\Rightarrow \log n < n$ is true for $n > 1$

5) $n^3 = 8^{\log n}$

$\Rightarrow n^3 = n^{\log_2 8}$ (by using logarithm property)

$\Rightarrow n^3 = n^3$ ($\log_2 8 = 3$)

$\Rightarrow n^3 = n^3$ is true for all n values.

6) $8^{\log n} < 2^n$

$\Rightarrow 2^{3\log n} < 2^n$ (since bases are same)

$\Rightarrow 3\log n < n$ is true $n \geq 1$

$\Rightarrow 8^{\log n} < 2^n$ is true $n \geq 1$

---

7) $2^n < 10^n$ (if we divide both side with 2^n)

$\Rightarrow 1 < 5^n$ is true for $n > 0$

---

3) 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];
            }
        }
    }
}
```

Let n = sizeofArray

steps	freq	Total
2	$n+1$	$2n+2$
1	n	n
1	n	n
1	n	n
1	n	n
1	n	n
1	n	n

Total = $8n+2$

Analysis

So if n = Size of Array we have

$\boxed{f(n) = 8n+2}$ since we don't have any condition statement to stop the loop we have;

$\boxed{T(n) = \Theta(f(n)) = \Theta(n)}$ (We ignore constant terms) 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;
      }
   }
}
```

Analyzing of Algorithm

$\Rightarrow$ if we look at the code we will see that in else part loop variable i, increases $i=i^2-i$ so easily we we can say $i=i^2$; instead of i++. Since i changes i^2-i instead of +1 we can easily make following observation:

when $i>=n$ (loop is stop)

$\Rightarrow i=2^{2^k}$ (since i increases exponentially (i^2-i))

$\Rightarrow 2^{2^k} \geq n$

$\Rightarrow 2^{2^k} = n$ (take logarithm both side)

$\Rightarrow 2^k = \log(n)$ (take logarithm again)

$\Rightarrow k = \log_2(\log_2 n)$

$\boxed{\text{Thus, } T(n) = O(\log_2(\log_2 n))}$

4)

a) $\sum_{i=1}^{n} i^2 \log i$ so;

$f(n) = \sum_{i=1}^{n} i^2 \log i$ is non-decreasing function

if we squeeze the function into 2 integral

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

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

To solve integral, if we apply integration by part

$U = \log_2 x \rightarrow du = \frac{1}{x \ln 2} dx$

$dv = x^2 dx \rightarrow \int dv = \int x^2 dx \Rightarrow v = \frac{x^3}{3}$

Rule:

$u.v - \int v.du$

$\ln 2 = 0{,}693$

$\Rightarrow \log x . \frac{x^3}{3} - \int \frac{x^3}{3} . \frac{1}{x \ln 2} dx$

$\Rightarrow \log x . \frac{x^3}{3} - \int \frac{x^2}{3} . \frac{1}{\ln 2} dx$

$\Rightarrow \log x . \frac{x^3}{3} - \frac{1}{3 \ln 2} \int x^2 dx$

$\Rightarrow \log x . \frac{x^3}{3} - \frac{x^3}{9 \ln 2} = \boxed{\frac{x^3(3 \ln 2 \log x - 1)}{9 \ln 2}}$

$$\Rightarrow \left.\frac{x^3(3\ln 2\log x-1)}{9\ln 2}\right|_0^n \le f(n) \le \left.\frac{x^3(3\ln 2\log x-1)}{9\ln 2}\right|_1^{n+1}$$

$$\Rightarrow \frac{n^3(3\ln 2\log n-1)}{9\ln 2} - \frac{0^3(3\ln 2\log 0-1)}{9\ln 2} \le f(n) \le$$

$$\frac{(n+1)^3(3\ln 2\log(n+1)-1)}{9\ln 2} - \frac{1^3(3\ln 2\log 1-1}{9\ln 2} \quad \text{so,}$$

for upper bound we got by taking biggest term elements;
$f(n)=(n+1)^3\log(n+1) = T(f(n)) = O(n^3\log n)$ but,
for lower bound since $\log(0)=\infty$ we have an error.

$\Rightarrow$ if we play with boundries to get rid off the error we will get;

$$\underbrace{1^2\log 1}_{0} + \sum_{i=2}^{n} i^2\log i = \sum_{i=2}^{n} i^2\log i$$

if we use integration method for this;

$$\Rightarrow f(n) \le \int_1^n g(x)dx$$

$$f(n) \le \int_1^n x^2\log x$$

$\Rightarrow$ if we take integral by using integration by part as we did before we will get;

$$\Rightarrow \left.\frac{x^3(3\ln 2\log x-1)}{9\ln 2}\right|_1^n \ge f(n)$$

$$\frac{n^3(3\ln 2\log n-1)}{9\ln 2} - \frac{1(3\ln 2\log 1-1)}{9\ln 2} \ge f(n)$$

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

As a result if we ignore constant terms we will get;

$f(n) \in O(n^3 \log n)$ $\Rightarrow f(n) \in \Theta(n^3 \log n)$
$f(n) \in \Omega(n^3 \log n)$

b) $\sum_{i=1}^{n} i^3$ is a non-decreasing function

if we squeeze the function into 2 integral

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

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

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

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

So if we ignore constant terms we will get;

$$\left.\begin{array}{l} f(n) \in O(n^4) \\ f(n) \in \Omega(n^4) \end{array}\right\} \Rightarrow \boxed{f(n) = \Theta(n^4)}$$

---

c) $\sum_{i=1}^{n} \frac{1}{2\sqrt{i}}$ is non-increasing (decraising) function

So let obtain a closed form formula;

$$\int_1^{n+1} g(x)dx \le f(n) \le \int_0^n g(x)dx$$ (since f(n) is decraising, we changed the order.)

$$\Rightarrow \int_1^{n+1} \frac{1}{2} \cdot x^{-\frac{1}{2}} dx \le f(n) \le \int_0^n \frac{1}{2} \cdot x^{-\frac{1}{2}} dx$$ (take constant outside)

$$\Rightarrow \frac{1}{2} \int_1^{n+1} x^{-\frac{1}{2}} dx \le f(n) \le \frac{1}{2} \int_0^n x^{-\frac{1}{2}} dx$$

$$\Rightarrow \frac{1}{2} \cdot \left(\frac{x^{1/2}}{1/2} \Bigg|_1^{n+1} \right) \le f(n) \le \frac{1}{2} \left(\frac{x^{1/2}}{1/2} \Bigg|_0^n \right)$$

$$\Rightarrow \frac{1}{2} \cdot (2(n+1)^{1/2} - 2) \le f(n) \le \frac{1}{2} (2n^{1/2})$$

$$\Rightarrow (n+1)^{1/2} - 1 \le f(n) \le n^{1/2}$$

$$\Rightarrow \boxed{\sqrt{n+1} - 1 \le f(n) \le \sqrt{n}}$$

So both upper & lower bounds are defined in terms of $\sqrt{n}$ therefore;

$$\boxed{f(n) \in \Theta(\sqrt{n})}$$

---

d) $\sum_{i=1}^{n} 1/i$ is non-increasing (decraising) function

if we look the function if we take integral with this border we will get $\boxed{\ln(0) = \infty}$ so we need to change borders. if we change the borders we will get;

$$\boxed{f(n) = 1 + \sum_{i=2}^{n} \frac{1}{i}}$$ (Change borders to avoid errors)

$$\Rightarrow 1 + \int_{2}^{n+1} g(x)\,dx \leq f(n) \leq 1 + \int_{1}^{n} g(x)\,dx$$

$$\Rightarrow 1 + \int_{2}^{n+1} \frac{1}{x}\,dx \leq f(n) \leq 1 + \int_{1}^{n} \frac{1}{x}\,dx$$

$$\Rightarrow 1 + \left(\ln(x)\Big|_{2}^{n+1}\right) \leq f(n) \leq 1 + \left(\ln(x)\Big|_{1}^{n}\right)$$

$$\boxed{1 + \ln(n+1) - \ln 2} \leq f(n) \leq \boxed{1 + \ln(n)}$$

So if we ignore constant terms both upper & lower bounds are defined in terms of $\ln(n)$.

$$\Rightarrow \boxed{f(n) \in \Theta(\log(n))}$$

5) if we write pseudocode for this algorithm;

```
int linearsearch(int my_array, int key){
    for(int i=0; i<sizeofArray; i++){
        if(my_array[i]==key){
            return i;
        }
    }
    return -1;
}
```

Analysis

Best Case:

if searched element key is the first element of the List we will get best case with 1 comparison.

$\boxed{B(n)=\Theta(1)}$

Worst Case:

In order to get worst case we have 2 option;

1) if searched element key is not in the list
2) if searched element key is last element of the list.

In both cases we will get worst case such that

$\boxed{W(n)=\Theta(n)} \Rightarrow$ (n is size of list)