GTU Department of Computer Engineering

CSE 222/505 - Spring 2021

Homework #2

Caner AKIN

151044066

Part 2:

a) for example $7n^2+3n+5$ can be

$7n^2+3n+5 = O(n^4)$

$7n^2+3n+5 = O(n^3)$ $\quad T(N) \le c f(N)$ when $N \ge n_0$

$7n^2+3n+5 = O(n^2)$ least

$7n^2+3n+5 \neq O(n)$

That mean $7n^2+3n+5 \le C \cdot n^2$ when $\forall N \ge n_0$.

This means that it has happened.

b) $f(n) = 7n^2 \Rightarrow \Theta(n^2)$

$g(n) = n \Rightarrow \Theta(n)$

$\max(f(n), g(n)) = \Theta(n^2)$

$\Theta(f(n)+g(n)) = \Theta(7n^2+n) = \Theta(n^2)$

$\max(f(n), g(n)) = \Theta(f(n)+g(n))$

c) I. $\lim_{n\to\infty} \frac{2^{n+1}}{2^n} = \lim_{n\to\infty} \frac{2 \cdot 2^n}{2^n} = 2$ so $2^{n+1} = 2^n$

$2^n = \Theta(2^n) \Rightarrow 2^{n+1} = \Theta(2^n)$ True

II. $\lim_{n\to\infty} \frac{2^{2n}}{2^n} = \lim_{n\to\infty} 2^n = \infty$ then $2^{2n} > 2^n$

$2^n = \Theta(2^n)$ and must be $2^{2n} > \Theta(2^n)$ False

III. for $7n^2 = O(n^2)$ and $3n^2 = \Theta(n^2)$

$7n^2 * 3n^2 = 21n^4$ $\quad 21n^4 = \Theta(n^4)$

$3n^2 = \Theta(n^2)$

But $7n = O(n^2)$ and

$7n * 3n^2 = 21n^3$ $\quad 21n^3 = \Theta(n^3) \neq \Theta(n^4)$

That disprove

Part 3:

$* \lim_{n\to\infty} \frac{3^n}{n.2^n} = \infty \quad 3^n > n2^n$

$* \lim_{n\to\infty} \frac{n2^n}{2^{(n+1)}} = \lim_{n\to\infty} \frac{n.2^n}{2.2^n} = \lim_{n\to\infty} \frac{n}{2} = \infty \quad n2^n > 2^{(n+1)}$

$* \lim_{n\to\infty} \frac{2^{(n+1)}}{2^n} = \lim_{n\to\infty} \frac{2.2^n}{2^n} = \lim_{n\to\infty} 2 = 2 \quad 2^{(n+1)} = 2^n$

$* \lim_{n\to\infty} \frac{2^n}{5^{\log_2 n}} = \frac{n.2^{n-1}}{\frac{1}{2n}.5^{\log_2 n-1}} = \lim_{n\to\infty} \frac{2n^2.2^{n-1}}{5^{\log_2 n-1}} = \infty \quad 2^n > 5^{\log_2 n}$

$* \lim_{n\to\infty} \frac{5^{\log_2 n}}{n^{1.01}} = \lim_{n\to\infty} \frac{\frac{1}{2n}.5^{\log_2 n-1}}{1.01.n^{0.01}} = \infty \quad 5^{\log_2 n} > n^{1.01}$

$* \lim_{n\to\infty} \frac{n^{1.01}}{n.\log^2 n} = \lim_{n\to\infty} \frac{n^{0.01}}{\log^2 n} = \infty \quad n^{1.01} > n.\log^2 n$

$* \lim_{n\to\infty} \frac{n.\log^2 n}{n^{1/2}} = \lim_{n\to\infty} n^{1/2}\log^2 n = \infty \quad n.\log^2 n > \sqrt{n}$

$* \lim_{n\to\infty} \frac{n^{1/2}}{(\log n)^3} = \infty \quad \sqrt{n} > (\log n)^3$

$* \lim_{n\to\infty} \frac{(\log n)^3}{\log n} = \lim_{n\to\infty} (\log n)^2 = \infty \quad (\log n)^3 > \log n$

$\log n < (\log n)^3 < \sqrt{n} < n.(\log^2 n) < n^{1.01} < 5^{\log_2 n} < 2^n = 2^{(n+1)} < n.2^n < 3^n$

Part 4:

- Minimum-valued item

```
findMin (array, n)
{
    c = 0
    for (i = 0 to n-1) {
        if (array.get(i) < array.get(c))
        {
            c = i
        }
    }
    return c
}
```

Line	Step	freq	total
c = 0	1	1	1
for (i = 0 to n-1)	1	n+1	n+1
if (array.get(i) < array.get(c))	1	n	n
c = i	1	1	1
return c	1	1	1
			2n+4

(T_1: lines c = 0 through the loop; T_2: return c)

$T_w = T_2 + \max(T_1) = 2n+4 = \Theta(n)$

$T_b = T_2 + \min(T_1) = 2n+3 = \Theta(n)$

$T_n = \Theta(n)$

- Find the median item

```
findMedian (array, n) {
    i = 0
    j = 0
    for (i = 0 to n-2) {
        X1 = i
        for (j = i to n-1) {
            if (array.get(j) < array.get(X1)) {
                X1 = j
            }
        }
        temp = array.get(X1)
        array.set(X1, array.get(i))
        array.set(i, temp)
    }
    print (array.get(n/2))
}
```

Line	Step	freq	total
i = 0	1	1	1
j = 0	1	1	1
for (i = 0 to n-2)	1	n	n
for (j = i to n-1)		n . n+1	

(T_1: i = 0 through X1 = i; T_2: inner for loop; T_3: temp, swap and print)

$T_1 = 2 + \sum_{i=0}^{n-2} 1 = n \quad \Theta(n)$

$T_2 = \sum_{i=0}^{n-2} \sum_{j=i}^{n-1} 1 + \frac{n.(n+1)}{2} \quad \frac{2n^2+2n+3}{2} \quad \Theta(n^2)$

$T_3 = 3n+1 \quad \Theta(n)$

$T_n = \Theta(n^2)$

- Find two elements whose sum is equal to a given value

```
FindSum(array, n, Find)
{
  for(i=0 to n-2) {                          1      n-1          n-1
    for(j=0 to n-1) {                        1      (n-2).(n-1)  n²-3n-2
      if(array.get(i)==array.get(j)) {       1      n-1          n-1
        print(i, i)                          1      1            1
      }
    }
  }
}
```

$$T_n = n^2 - n - 5 = \Theta(n^2)$$

- Merge two lists

```
MergeList(arr1, arr2, n)
{ i=0
  j=0
  temp = ArrayList<>();                      } Θ(2n) = Θ(n)
  while((i+j) to 2n-2)
  {
    if(arr1.get(i) < arr2.get(j)) { } T3
      temp.add(arr1.get(i))                  } T1
      i = i+1
    }
    else
    {
      temp.add(arr2.get(j))                  } T2
      j = j+1
    }
  }
}
```

$$T_n \le T_3(n) + \max(T_1(n), T_2(n))$$

$$T_n \ge T_3(n) + \min(T_1(n), T_2(n))$$

$T_n \le 2n+2$ $\quad$ $T_n \ge 2n+2$

$$T_n = 2n+2 = \Theta(n) = O(n) = \Omega(n)$$

Part 5:

a)

```
int p_1 (int array[3])
{
  return array[0] * array[2]      1     |     2
}
```

$T(N) = 1 \Rightarrow \Theta(1) = O(1) = \Theta(1)$ time

$O(1)$ space

b)

```
int p_2 (int array[], int n)
{
  sum = 0
  for (int i=0 ; i<n ; i+=5) {
      sum += array[i] * array[i]
  }
  return sum
}
```

$T(N) = \frac{n}{5} + 1 + \frac{n}{5} = \frac{2n+5}{5} = \Theta(n)$ time

$\Theta(n)$ space

c)

```
void p_3 (int array[], int n)
{
  for(int i=0; i<n; i++)
      for(int j=1; j<i; j=j*2)
          printf("%d", array[i] * array[j])
}
```

$T_1 = \Theta(n)$ $T_2 = n \times (\log n) = \Theta(n \log n)$ time

$\Theta(n)$ space

d)

```
void p_4 (int array[], int n)
{
    if ( p_2 (array, n) > 1000)        Θn
        p_3 (array, n)                 Θ(n log n)
    else
        printf("%d", p_1(array) * p_2 (array, n))   Θ(n)
}
```

$T_{best} = \Theta(n)$ time

$T_{worst} = \Theta(n \log n)$ time

$\Theta(n)$ space