Cleveland State University

CIS 606 – Analysis of Algorithms

Quiz – 1

Name: Prudhvi Reddy Araga

Login ID: praraga

33/70

Question 1:

a) Answer: True

F(n) = θ (g(n))

➔ $c_1 . g(n) \leq f(n) \leq c_2 g(n)$

-7

➔ $c_1 . g(n) \leq f(n)$ – **Step 1**

?

➔ $f(n) \geq g(n)$ where is c1?

i.e., $g(n) = \Omega(f(n))$ – **Step 2**

$f(n) \leq c_2 . g(n)$ – **step 3** which implies

?

g. f(n) = O (f(n)) – **Step 4**

By combining step 3 and step 4

$g(n) = \theta(f(n))$

3^3n <= c 3^n
divide 3^n both sides
3^2n <= c
No such constant c exists

b) Answer: False

$3^{3n} = (3^3)^n = 27^n$

$27^n >= 3^n$ for n >1 but O notation is upper bound notation and as per the definition $27^n \leqslant 3^n$.

-6

Hence the answer is false.

Question 2:

Use Master Method

T(n) = T(5n/7) + n. – **Step 1**

-9

Substitute n = 5n/7 in Step 1

T(5n/7) = T(5/7 *5n/7) + 5n/7 – **Step 2**

Substituting Step 2 in Step 1

$T(n) = T(5^2/7^2*n) + 5n/7 + n$

$T(n) = T[(5/7)^2*n] + 5n/7 + n$ – **Step 3**

Substitute n = 5n/7 in Step 2

$T[(5/7)^{2}*n] = T[(5/7)^{3}n] + (5/7)^{2}n$ – **Step 4**

Substituting Step 4 in Step 3

$T(n) = T[(5/7)^{3}n] + [(5/7)^{2}*n] + [5/7*n] + n$

$$T(n) = T\left[\left(\frac{5}{7}\right)^{k}.n\right] + n.\sum_{i=1}^{k}\left(\frac{5}{7}\right)^{k}$$

$\Sigma x^{i} = (x^{n+1} - x) / (x - 1)$

➔ $\Sigma(5/7)^{k} = (5/7)^{k}$ by ignoring the lower order terms

Let $n = (7/5)^{k}$

Log n = k

$T(n) = T(1) + (n.\ 5^{\log n}) / (7^{\log n})$

$T(n) = T(1) + n.\ n^{\log 5} / n^{\log 7}$

(as $n^{\log 5} > 5^{\log n}$ & $n^{\log 7} > 7^{\log n}$)

$T(n) = T(1) + n * n^{\log_7 5}$

$T(n) = T(1) + n * n^{0.82}$

$T(n) = k + n^{1.82}$

$T(n) = O(n^{1.82}) \cong O(n^{2})$

Question 3:

a) If (A[i] > A[m])
m = i

indent -1

b) RecurSort (A, i, ~~n~~)
If i < 2 then return
j= FindMax(A, i)
Exchange(A[i],A[j])
RecurSort (A, i-1, ~~n~~)

-4

Question 3 (c):

Time taken for finding max (FindMax) is **O(n)** and RecurSort is called **(n-1)** times.
So, recurrence relation is $T(n) = T(n-1) + n * d$

By ignoring Constants, where "d" is constant

$T(n) = T(n-1) + n * d$

$T(n) = T(n-1) + d*n$ – **Step 1;** $d>0$

$T(n-1) = T(n-2) + d(n-1)$ -- **Step 2**

Substituting Step 2 in Step 1

$T(n) = T(n-2) + d(n+ (n-1))$

$T(n) = T(n-3) + d[n + (n-1) + (n-2)]$

.

.

.

$T(n) = T(n-k) + d[n + (n-1) + (n-2)+ \ldots\ldots. + n-(k-1)]$

If $n-k=1 \Rightarrow k=n$
$T(n) = T(1) + d[n + (n-1) + (n-2)+ \ldots\ldots. + n-(k-1)]$

$T(n) = T(1) + d[n + (n-1) + (n-2)+ \ldots\ldots. + 2]$

$T(n) = c + d[n(n+1)/2 – 1]$

$T(n) = c + d/2 (n^2+n-2)$

$T(n) = O(n^2)$

Question 4:

Min = - $-\infty$ (Minimum initial value)

DcMin(low,high, ~~min~~)

If low == high then

min = a[i]

else if low == high-1

~~if a [i] > a[j]~~

min = a[j]

else

min = a[i]

else -10

mid = (low + high) / 2

minL= DcMin (low, mid, ~~min~~)

minR= DcMin(mid+1, high, ~~min1~~)

if min1 < min

min = min1

? DcMin (1,n, min)

Recurrence Relation

T(n) = T(n/2) + T (n/2) + 2c

T(n) = 2 T(n/2) + c

As per master theorem, if

f(n) is O ($n^{\log_b a - \varepsilon}$) for some $\varepsilon > 0$

then T(n) = θ ($n^{\log_b a}$)

here a and b are 2,2 and f(n) is constant

f(n) = O ($n^{\log_2 2}$)$-\varepsilon$, $\varepsilon = 1$ case 1 applies

➔ O(n) ~~which~~ is true.

So, T(n) = θ (n)