Problem Set 1

a) $f_1(n) = n^{0.999999} \log n$.

$f_2(n) = 10000000n$

$f_3(n) = 1.000001^n$

$f_4(n) = n^2$

definately, in terms of complexity.

$O(f_1), O(f_2) < O(f_4) < O(f_3)$

OK, comparing f_1 & f_2.

$n^{0.999999} \log n$, $10000000n$.

proportion

$1000000n > n^{0.999999} \log n$.

$\Rightarrow 1000000 n^{0.000001} > \log n$.

this is correct as

$f'(n^{0.000001}) > f'(\log n)$

$\| \qquad \|$

$n^{-0.999999} \qquad n^{-1}$

$\rightarrow$ grows faster compared to $\nearrow$

∴ the order is

$O(f_1) < O(f_2) < O(f_4) < O(f_3)$

(b). $f_1 = 2^{2^{1000000}}$, $f_2 = 2^{1000000n}$

$f_3 = {}^{n}C_2$ $f_4 = n\sqrt{n}$.

f_1 is a constant ∴ is $O(1)$.

f_2 is exp ∴ is $O(2^n)$ has the worst complexity.

blw f_3 & f_4.

$f_3 = O(n^2)$ & $f_4 = O(n^{3/2})$.

∴ the correct order is

$O(f_1) < O(f_4) < O(f_3) < O(f_2)$.

(c) $f_1 = n^{\sqrt{n}}$, $f_2 = 2^n$, $f_3 = n^{10} \cdot 2^{n/2}$

$f_4 = \sum_{i=1}^{n} (i+1) = \frac{n(n+1)}{2} + n$ ..

clearly $f_3 < f_2$
(exp of $\frac{n}{2}$) (exp of n)

& $f_4 < f_3, f_2$

as f_4 is $O(n^2)$

Analysing f_1,

$y = n^{\sqrt{n}}$

$\ln y = \sqrt{n} \ln n$

$\frac{dy}{y} = \sqrt{n} \times \frac{1}{n} + \ln n \times \frac{+1}{2\sqrt{n}}$

$= \frac{1}{\sqrt{n}} \left[1 + \ln\sqrt{n}\right]$

$\Rightarrow dy = n^{\sqrt{n}-\frac{1}{2}} \left[1 + \ln\sqrt{n}\right]$

NEXT PAGE

Comparing $n^{\sqrt{n}} \& k2^{n/2}$.

proposition $n^{\sqrt{n}} < k2^{n/2}$.

apply log. on both sides. & n is large

$$\sqrt{n}\ln n < \frac{n}{2}\ln kn.$$

$$\Rightarrow \frac{1}{\sqrt{n}}\ln n < \frac{1}{2}\ln kn$$

$$\Rightarrow n^{1/\sqrt{n}} < \sqrt{kn}$$

This is true as for $n \to \infty$.

$$\lim_{n\to\infty} n^{1/\sqrt{n}} = \lim_{n\to\infty} e^{\log n^{1/\sqrt{n}}}$$

$$= e^{\lim_{n\to\infty} \frac{\log n}{\sqrt{n}}}.$$

$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} \Rightarrow \frac{\infty}{\infty}$ ∴ we can apply L'hopital rule.

$$\Rightarrow \lim_{n\to\infty} \frac{\frac{1}{n}}{\frac{1}{2\sqrt{n}}} = \lim_{n\to\infty} \frac{1}{2\sqrt{n}} = 0.$$

$$\therefore \lim_{n\to\infty} n^{1/\sqrt{n}} = e^0 = 1.$$

But R.H.S.

$$\lim_{n\to\infty} \sqrt{kn} \longrightarrow \infty.$$

∴ for large values of n.

$$n^{\sqrt{n}} < k2^{n/2}$$

∴ our proposition is correct.

$$\therefore \boxed{O(f_4) < O(f_1) < O(f_3) < O(f_2)}$$

1.2) (a)

$T(n,$

(b) $T(n$

$T($

$T(n,$

$T(n$

1.2) (a) $T(x, c) = \Theta(x)$ for $c \leq 2$.

$T(c, y) = \Theta(y)$ for $c \leq 2$, and

$T(x, y) = \Theta(x+y) + T\left(\frac{x}{2}, \frac{y}{2}\right)$ ..

$= \Theta(x+y) + \Theta\left(\frac{x+y}{2}\right) + T\left(\frac{x}{4}, \frac{y}{4}\right)$

$= \Theta(x+y) + \Theta\left(\frac{x+y}{2}\right) + \Theta\left(\frac{x+y}{4}\right) + T\left(\frac{x}{8}, \frac{y}{8}\right)$

$T(n, n)$

$= \underbrace{\Theta(2n) + \Theta(n) + \Theta\left(\frac{n}{2}\right) + \Theta\left(\frac{n}{4}\right) + \ldots \Theta\left(\frac{n}{2^{\log n}}\right)}_{\log n \text{ times}}$

$= \underbrace{K\,2n + Kn + K\frac{n}{2} + \ldots + K1}_{\log n \text{ times}}$

$= k'n = \underline{\Theta(n)}$

(b) $T(x, c) = \Theta(x)$ for $c \leq 2$

$T(c, y) = \Theta(y)$ for $c \leq 2$.

$T(x, y) = \Theta(x) + T\left(x, \frac{y}{2}\right)$

$T(n, n) = \Theta(n) + T\left(n, \frac{n}{2}\right)$

$= \Theta(n) + \Theta(n) + T\left(n, \frac{n}{4}\right)$

$= \Theta(n) + \Theta(n) + \Theta(n) + T\left(n, \frac{n}{8}\right)$ ..

$= \underbrace{\Theta(n) + \Theta(n) + \Theta(n) + \ldots + T(n, 2)}_{\log n \text{ times}}$

$= \underline{\Theta(n \log n)}$

(c). $T(x, c) = \Theta(x)$ for $c \leq 2$.

$T(x, y) = \Theta(x) + S(x, \frac{y}{2})$

$S(c, y) = \Theta(y)$ for $c \leq 2$.

$S(x, y) = \Theta(y) + T(\frac{x}{2}, y)$.

$T(n, n) = \Theta(n) + S(n, \frac{n}{2})$

$= \Theta(n) + \Theta(\frac{n}{2}) + T(\frac{n}{2}, \frac{n}{2})$

$= \Theta(n) + T(\frac{n}{2}, \frac{n}{2})$

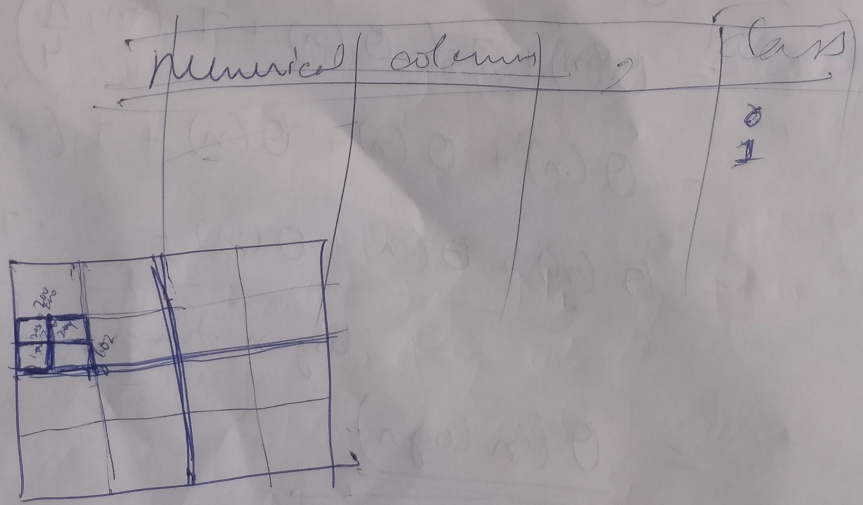
$= \Theta(n) + \Theta(\frac{n}{2}) + T(\frac{n}{4}, \frac{n}{4})$

$= \Theta(n) + \Theta(\frac{n}{2}) + \Theta(\frac{n}{4}) + \ldots + \Theta(1)$

log n times.

$= \underline{\Theta(n)}$

1.3) peak: values of N, E, S, T are ≤ value in the block.



(C). $T(x, c) = \Theta(x)$ for $c \leq 2$.

$T(x, y) = \Theta(x) + S(x, \frac{y}{2})$

$S(c, y) = \Theta(y)$ for $c \leq 2$.

$S(x, y) = \Theta(y) + T(\frac{x}{2}, y)$.

$T(n, n) = \Theta(n) + S(n, \frac{n}{2})$

$= \Theta(n) + \Theta(\frac{n}{2}) + T(\frac{n}{2}, \frac{n}{2})$

$= \Theta(n) + T(\frac{n}{2}, \frac{n}{2})$

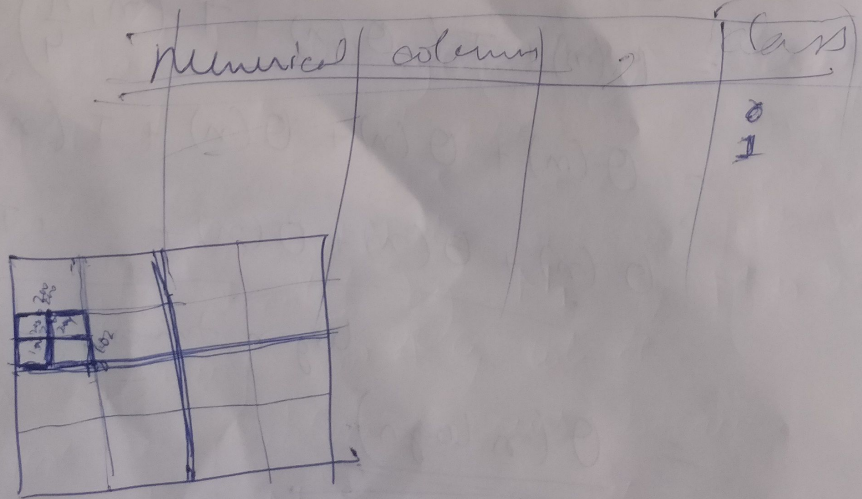
$= \Theta(n) + \Theta(\frac{n}{2}) + T(\frac{n}{4}, \frac{n}{4})$

$= \Theta(n) + \Theta(\frac{n}{2}) + \Theta(\frac{n}{4}) + \ldots + \Theta(1)$

Log n times.

$= \underline{\Theta(n)}$

1.3) peak: values of N, E, S, T are ≤ value of the block.



1.3) and 1.4) **algorithm 1 :-**

Recursion problem involves half the no. of columns.

$$\Rightarrow T(x, y) = \Theta\left(\frac{x}{y}\right) + T\left(x, \frac{y}{2}\right)$$

Solving for $x = n$ & $y = n$.

$$T(n, n) = \Theta(n) + T\left(n, \frac{n}{2}\right)$$

$$= \Theta(n) + \Theta\left(\frac{n}{2}\right) + \Theta(n) + T\left(n, \frac{n}{4}\right)$$

$$= \underbrace{\Theta(n) + \Theta(n) + \ldots + T(n, 1)}_{\log n \text{ times}}$$

$$= \underline{\Theta(n \log n)}$$

Yes the algorithm is correct with a worst-case run time of $\Theta(n \log n)$.

(b) this algorithm searches every cell in the layout thus the complexity is $\underline{\Theta(n^2)}$ or $\Theta(m \cdot n)$

This algorithm guarantees a ~~response~~ result as in the entire layout, there is at least 1 global maximum.

This global maxima is definately greater than all its surrounding cells. Thus ensuring a peak.

(c). it follows the ~~same~~ recurrence

$$T(x,y) = \Theta(x+y) + T\left(\frac{x}{2}, \frac{y}{2}\right)$$

@ $x=n$, $y=n$.

$$T(n,n) = \Theta(n) + T\left(\frac{n}{2}, \frac{n}{2}\right)$$

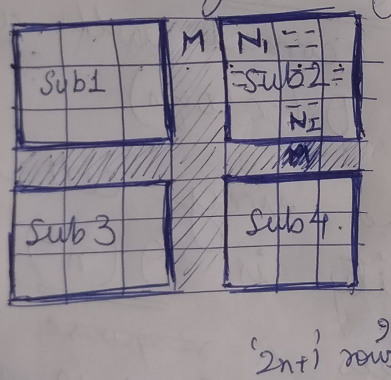
$$= \Theta(n) + \Theta\left(\frac{n}{2}\right) + T\left(\frac{n}{4}, \frac{n}{4}\right)$$

$$= \underbrace{\Theta(n) + \Theta\left(\frac{n}{2}\right) + \ldots + T(1,2)}_{\log n \text{ times}}$$

$$= \underline{\underline{\Theta(kn)}}, \text{ where } k = \text{constant}$$

$$= \Theta(n) \text{ Complexity.}$$

However, this algorithm fails in certain cases.



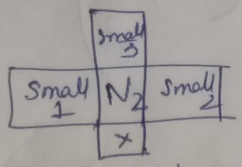
→ initially, it searches for maximum along the shaded row & column.

→ Next it selects 4 ~~quad~~ regions as above, in our example, there are '2n+1' row & cols.

→ assuming the getBetter neighbour falls in the sub2, denoted by N_1 & it is the maximum along row 4 & col 4.

→ Here the problem occurs because we don't follow along 'N₁'s row & col. instead we follow along the center row & col of Sub2 denoted in the fig by [=]

Suppose the max along [=] is @ N_2.
& say N_2 is > its secondly neighbours in 'sub2' block.. ~~Hor~~



However, we don't have access to below element [x] as get better Neighbour fn doesn't have access to this block, it may be that this block [x] is > [N_2], but the algorithm incorrectly returns N_2 as a peak.

(d) ~~This algo follows~~
Can make this work if we add the following line.

```
if neighbor == crossLoc and
problem.get(crossLoc) >= problem.get
                          (bestseen):

    return crossLoc
```

instead of

```
if neighbor == crossLoc:
    return crossLoc
```

(d) This algorithm works even though it's the ~~same~~ kinda similar to algorithm 3 because it has the condition.

```
if neighbor == bestLoc and problem.get(bestLoc)
                          >= problem.get(bestseen):

    return bestLoc
```

$T(n) = O(n) + O($

$$T(x,y) = \Theta(x) + S\left(\frac{x}{2}, y\right)$$

and

$$S(x,y) = \Theta(y) + T\left(x, \frac{y}{2}\right)$$

for $x = n, y = n$

$$T(n,n) = \Theta(n) + S\left(\frac{n}{2}, n\right)$$

$$= \Theta(n) + \Theta(n) + T\left(\frac{n}{2}, \frac{n}{2}\right)$$

$$= \Theta(n) + T\left(\frac{n}{2}, \frac{n}{2}\right)$$

$$= \Theta(n) + \Theta\left(\frac{n}{2}\right) + T\left(\frac{n}{4}, \frac{n}{4}\right)$$

$$= \Theta(n) + \Theta\left(\frac{n}{2}\right) + \Theta\left(\frac{n}{4}\right) + \ldots + \Theta(2)$$

Log n times

$$= \Theta(kn) \text{ or } \underline{\Theta(n)} \text{ times}$$

Complexity

1.6)

0	0	0	10	11	2	2
1	1	1	1	5	3	1
2	2	2	2	2	5	4
3	3	3	1	5	6	4
4	4	4	2	1	1	1
5	5	5	3	2	2	2
6	6	6	4	3	3	3

This is the counter-example that fails for Algorithm 3.