fall-24

CSE-221

fgz

DATE: 22 OCT 24
LECTURE: 01
week: 01

Quiz (best n-1)

Don't	DO
D - Disrespect/Dispute	D - Dedicated
L - late submission	L - Liability
C - copying	C - Co-operation

Selection Sort
Bubble Sort $\Big\} O(n^2)$

$O(n)$ Linear search $n=16$

$O(\log_2 n)$ Binary search

$O(\log_2 16) = 4$

$O(n \log n)$ quick sort

$O(n \log n)$ marge sort

$O(n \log_2 16)$

$\Rightarrow O(16 \log_2 16)$

$\Rightarrow O(16 \times 4)$

$\Rightarrow O(64)$

CamScanner

Chapter-03

Topic: Assymptotic Notation

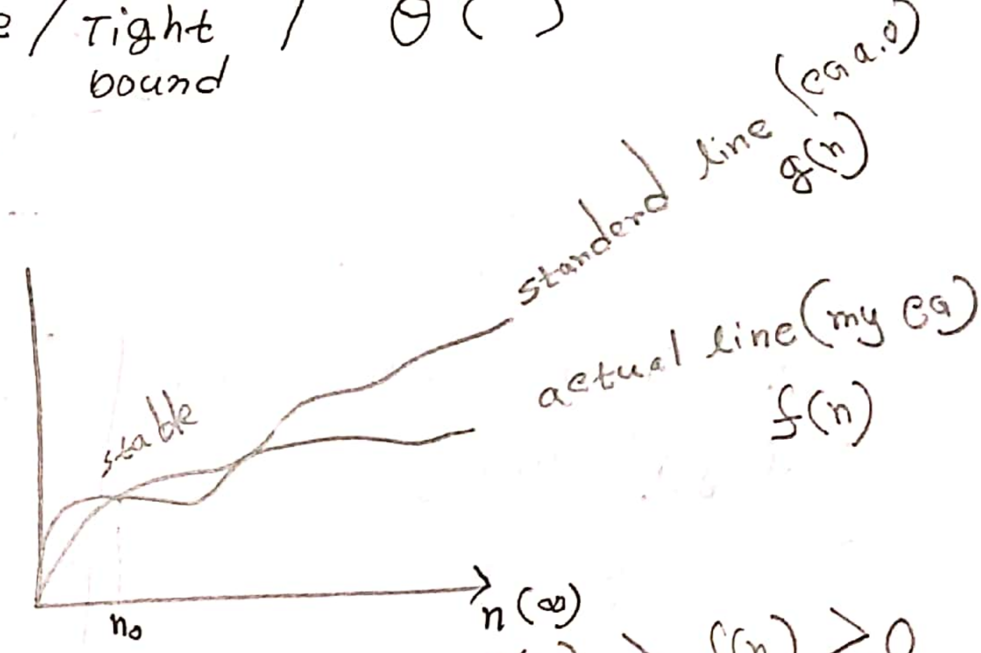
Assymptotic Notation

we are assume number will be infinity

(i) worstcase / uppercase / upper bound / $O(\ )$ Maximum Limit

(ii) Best-case / lowercase / upper bound / $\Omega(\ )$

(iii) Average / Tight bound / $\Theta(\ )$



$g(n) \geq f(n) \geq 0$

$\boxed{0 \leq f(n) \leq cg(n)}$; $n \geq n_0$

$0 \sim 9.00$

$$0 \leq f(n) \leq c \cdot g(n)$$

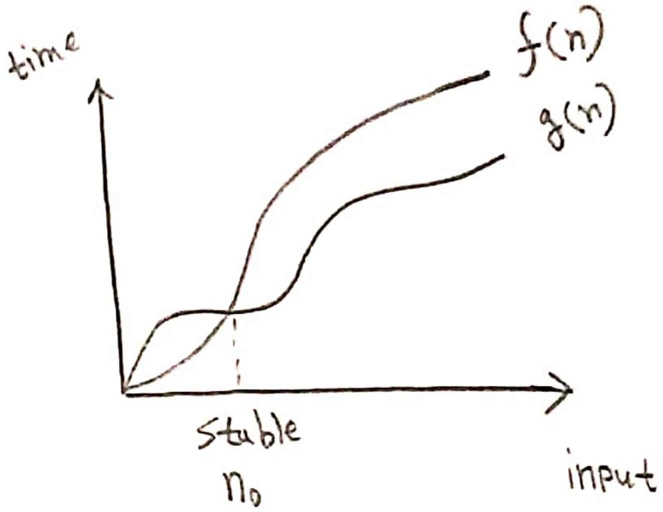
$$f(n) = O(g(n))$$

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

$$f(n) \leq n^2$$

(ii) Lower bound / Best case Scenario / Lower-case Limit

(minimum Limit)



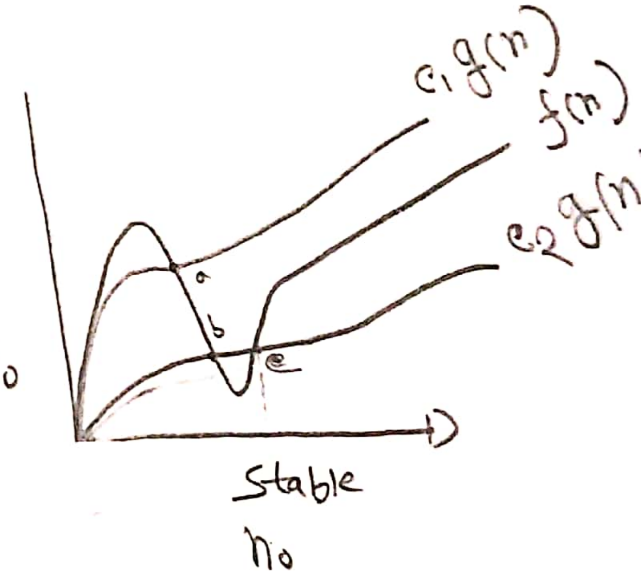
$$f(n) \geq g(n) \geq 0 \quad n \geq n_0$$

$$0 \leq c\,g(n) \leq f(n)$$

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

(iii) Average ~~boun~~ case

$$f(n),\ c_1 g(n),\ c_2 g(n),\ 0$$



Ω

ig O

$$c_1 g(n) \geq f(n) \geq c_2 g(n) \geq 0 \; ; \; n \geq n_0$$

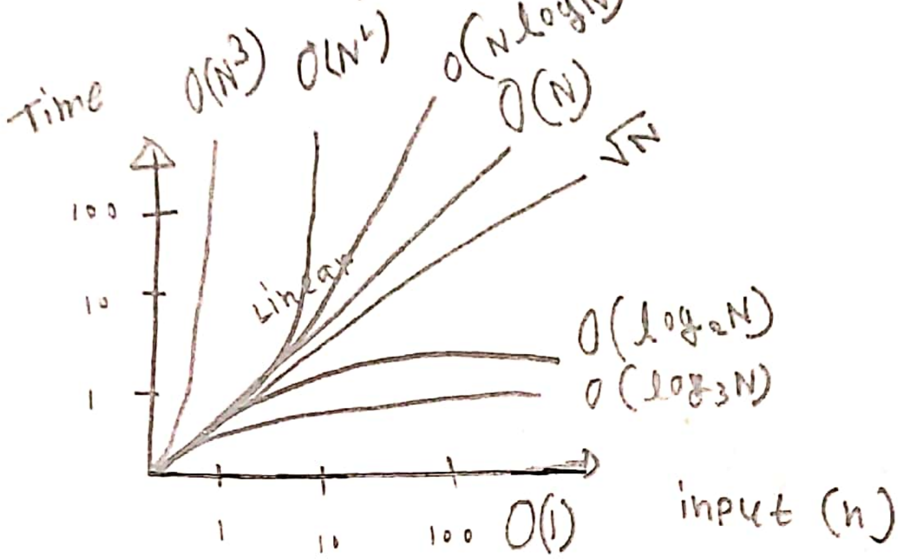
$$f(n) = \Theta(g(n))$$

অর্থাৎ theta বোঝার জন্য তাকে অবশ্যই big O ও সাথে big Omega থাকতে must be.

Space- Time- Tradeoff

constant running Time :

$O(1)$ = constant



$O(N) = 9$

$O(\log_2 N) = 2$

$O(N \log_2 N) = 8$

More Big-O

$f(n) = O(g(n)) \rightarrow f(n) \le c\,g(n)$

· Prove that $\underbrace{20n^2+2n+5}_{f(n)} = O(\underbrace{n^2}_{g(n)})$

{degree থেকে 1 বেশি নিবো

$\Rightarrow 20n^2+2n+5 \le c \cdot n^2$

$c = 21$
$n_0 = 4$

$\Rightarrow 20n^2+2n+5 \le 21n^2$

$\Rightarrow 2n+5 \le n^2$

$\Rightarrow$

Try-error

$n=1$ ×
$7 \le 1$ false

$n=2$
$9 \le 4$ false

$n=3$
$11 \le 9$ false

$n=4$
$13 \le 16$ true

$n=5$
$15 \le 25$ true

$n=10$
$25 \le 100$ true

Homework → $20n^2+2n+5 = O(n^2)$ prove that big Omega

- $n = O(n^2) \neq \Theta(n^2)$ $\qquad n \le n^2 \quad n = O(n^2)$
- $200n^2 = O(n^2) = \Theta(n^2)$ $\qquad n = \Omega(n^2)$

Assymptotic fuctions (O, Ω, Θ) ignores $\rightarrow$

(i) Any constant
(ii) Insignificant value

$$O(\cancel{20}\underbrace{n^2}+\cancel{8n}+\cancel{2})$$

$$O(\cancel{300}n^2)$$

$$O(n^2) \qquad \Omega(\cancel{200}n^2)$$

$$n^{2.5} \neq O(n^2) \neq \Theta(n^2)$$

$$\Rightarrow n^{2.5} \not\le n^2$$

- $20n^3 + 7n + 1000 = \Theta(n^3)$

Show that ~~$2^n + n^2$~~ $2^n + 2^n = O(2^n)$

$c = 2, n_0 = 4$

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

$\Rightarrow n^2 \leq 2 \cdot 2^n - 2^n$

$\Rightarrow n^2 \leq 2^n (2-1)$

$\Rightarrow n^2 \leq 2^n$

$n = 1$ ✓

$n = 2$ ✓

$n = 3$ ✗

$\boxed{n = 4}$ ✓

$n = 5$ ✓

$n = 6$ ✓

$n = 7$ ✓

$n = 10$ ✓

CS CamScanner

```
for (j=0; j<n; j++){
    //3 atomics
    if (conditions) break;
}
```

$O(n)$

$\Omega(1)$

complexity . $O(n)$

$O(1) + O(1 \cdot O(n))$

$O(1) + O(O(n))$

$O(1) + O(n)$

$O(n)$

independent nested loop

```
sum1= 0;
for (i=1; i<=n; i++) {
    for (j=1; j<=n; j++){
        sum++;
    }
}
```

$O(n^2)$

step	outer lop(i)	inner loop (j)
0	1	n
1	2	n
2	3	n
⋮		
n	n+1	n

$n+n+n+\dots+n$

n^2

$O(n^2)$

CS CamScanner

```
Sum2 = 0;
for (i = 1; i<=n; i++)
  for(j=1; j<=i; j++)
    Sum2 ++;
```

Step	outloop(i)	j
0	1	1
1	2	2
2	3	3
⋮	⋮	

$1+2+3+\ldots+n$

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

$O\left(\frac{n^2+n}{2}\right) \qquad = \frac{n^2+n}{2}$

$O(n^2)$

$=$

```
sum1 = 0;
for (k=1; k<=n; k*2)      -> independence
    for (j=1; j<=n; j++)      O(T.C)
        sum1++;               -> O(n) *
```

```
for(i=1); i≤n; i*=2)
{            }
```

n=16

i	
i=1	0
i=2	1
4	2
8	3
16	4

$\log_2(16) = 4$

step	loop variable value (i)
0th	$1 = 2^0$
1st	$2 = 2^1$
2nd	$4 = 2^2$
3rd	$8 = 2^3$
4th	$16 = 2^4$
⋮	
kth	2^k

$\boxed{2^k = n}$

$k = \log_2(n)$

$O(\log_2 n . n)$

$O(n \cdot \log_2 n)$

$\boxed{O(n \log n)}$

0	1	n
1	2	n
3	4	n
⋮		
i	2^i	

$n + n + n + \dots + n$

$n * \log_2 n$

$n \log n$

**

```
Sum = 0;
for( k = 1; k <= n; k *= 2)
    for( j = 1; j <= k; j++)
        sum++;
```

$2^0 + 2^1 + 2^2 + 2^3 + 2^4 + \dots 2^i$

$r = 2$

step	outer loop	inner loop
0	$1 = 2^0$	1
1	$2 = 2^1$	2
2	$4 = 2^2$	4
3	$8 = 2^3$	8
4	$16 = 2^4$	16
⋮	⋮	
i	2^i	

$1 + 2 + 4 + 16 + \dots + 2^i$

$= 2^0 + 2^1 + 2^2 + 2^3 + 2^4 + \dots 2^i$

geometric sum formula for summig upto m, where $r > 1$;

$$r^0 + r^1 + r^2 + r^3 + \dots + r^k = \sum_{k=0}^{m} r^k = \boxed{\frac{r^{m+1} - 1}{r - 1}}$$

$r > 1$

∞

$2^0 + 2^1 + 2^2 + 2^3 + \dots + 2^i$ $\quad r = 2$

$$\Rightarrow \boxed{\sum_{k=0}^{i} r^k = \frac{r^{i+1} - 1}{r - 1}}$$

$$\sum_{k=0}^{i} r^k = \frac{r^{i+1} - 1}{r - 1}$$

$$= \frac{2^{i+1} - 1}{2 - 1}$$

$\Rightarrow 2^{i+1} - 1$

$\Rightarrow 2^i . 2 - 1$

$\Rightarrow n . 2 - 1$

$= 2n - 1$

$$\sum_{k=0}^{i} 2^k = 2n - 1$$

$2^i = n$

$\Rightarrow \log_2 2^i = \log_2 n$

$\Rightarrow \boxed{i = \log_2 n}$

$O(2n - 1)$

$O(n)$

CamScanner

for ( ) +1 K
  for( ) ++ K

$\sum 1+2+3+4+5+ \ldots +n = \frac{n(n+1)}{2}$

$= \frac{n^2+n}{2}$

$O\left(\frac{n^2+n}{2}\right)$

$O(n^2)$

for ( ) ~~*~~=2
  for( ) ++

1	n
2	n
2	n

K=1

$\sum n+n+n+n \ldots n$

$2^0+2^1+2^2+2^4+ \ldots +2^K \longrightarrow$

```
for(i=0; i≤10; i++){
}
```

$O(1)$

- Searchig
- Sorting

```
for(i=1, i<=n; i++){
  for(j=1; j<=i; j*=2)
    op()
```

Step	outer loop (i)	inner loop (j)
0	1	$\log_2 1$
1	2	$\log_2 2$
2	3	$\log_2 3$
4	4	$\log_2 4$
⋮	⋮	⋮
n	n	$\log_2 n$

$\log_2 4 + \log 2 + \dots = \log n$

$\log(ab) = \log a + \log b$

$\Rightarrow \log(1 \times 2 \times 4 \dots \times n) = \log 1 + \log 2 + \log + \dots \log n$

$\Rightarrow \log(n!)$ $\qquad O(\log(n!))$

$O(n \log n)$

$\log 1 + \log 2 + \log 3 + \log 4 + \ldots + \log n \leq \log n + \log n + \log n \ldots$
$\ldots + \log n$

$\Rightarrow \log(1*2*3*4*5)..n) \leq n * \log n$

$\Rightarrow \underbrace{\log(n!)}_{f(n)} \leq \underbrace{n * \log n}_{g(n)}$

$f(n) = O(g(n))$

$\cancel{O(\log(n!))}$

$$\boxed{\log n! = O(n \log n)}$$

$n = n$

for $n > 2$

$(n-1) * n < n * n$

$\Rightarrow (n-2) * (n-1) < n * n * n$

$\Rightarrow (n-3) * (n-2) * (n-1) * n < n * n * n * n$

continue till 1

$1 * 2 * 3 \ldots (n-2) * (n-1) * < n < n^n$

CamScanner

$n! < n^n$

$\log(n!) < \log(n^n)$

$\log(n!) < n * \log(n)$

$\boxed{\text{Example : 15}}$

inner loop এ n j increment হবে।

n

```
for (i=0; i<n; i=i+3)
  for (j=n; j>1; j=j/5)     log5 n
    for (k 1; k<=n; k=k*5)  log5 n
```

$n * \log_5 n * \log_5 n$

$$\boxed{n(\log_5 n)^2}$$

```
for i in range(1,n)
    j = 1
    while j < i*i
        j = j+1
```

```
for i in range(1,n)
    j = 1
    while j*j < i
        j = j+1
```

$j * j < i$

$\Rightarrow j^2 < i$

$\Rightarrow \boxed{j < \sqrt{i}}$

$$\boxed{\sqrt{1} + \sqrt{2} + \sqrt{3} + \sqrt{4} \ldots + \sqrt{n}}$$

Homework

CamScanner

④ quaternary search is not popular (why)?

$\boxed{\text{insertion Sort}}$ 3 Page

best case	worst case	Avg case	why? How?
~~$O(n)$~~ $\Omega(n)$	$O(n^2)$	theta (n^2)	
- Array already Sorted in desired sort	- reverse order		

যত no iteration হবে ততগুলো number sorted হবে।

Spring 23

week-03

Lecture-06

$\boxed{\text{CSE221}}$

Date: 5thNOV24

Page: 12

- The divide-and-conquer approach:
- DNC
- DNC Time Complexity

Divide and conquer Algo $\boxed{\text{51 page}}$

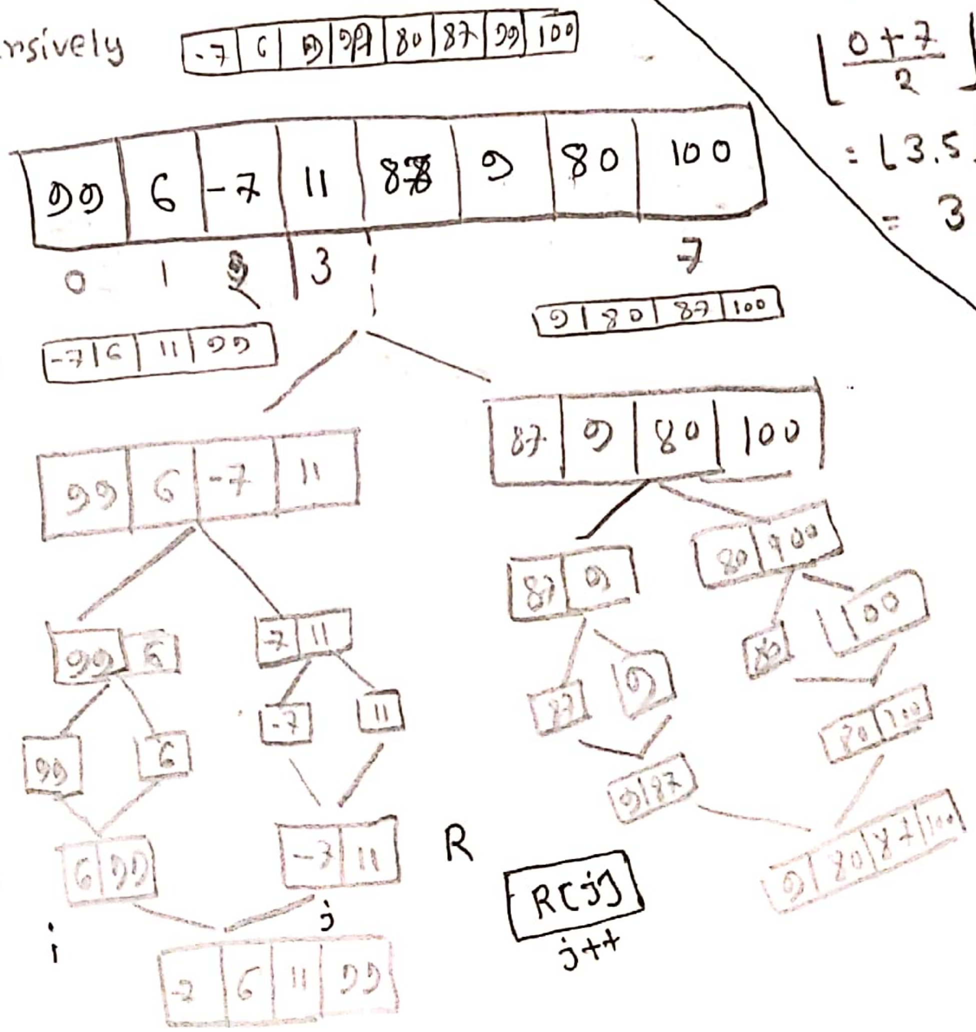
① Divide: recursively dives the given input into a unit size. subproblem

② conquer: apply the goal on the subproblems subSolutions recursively

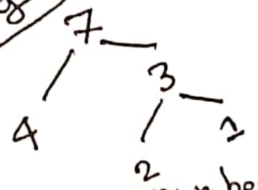
③ combine: combine the subsolutions recursively

$$mid = \left\lfloor \frac{l+h}{2} \right\rfloor$$

$$\left\lfloor \frac{0+7}{2} \right\rfloor = \lfloor 3.5 \rfloor = 3$$



$\boxed{\text{assending}}$



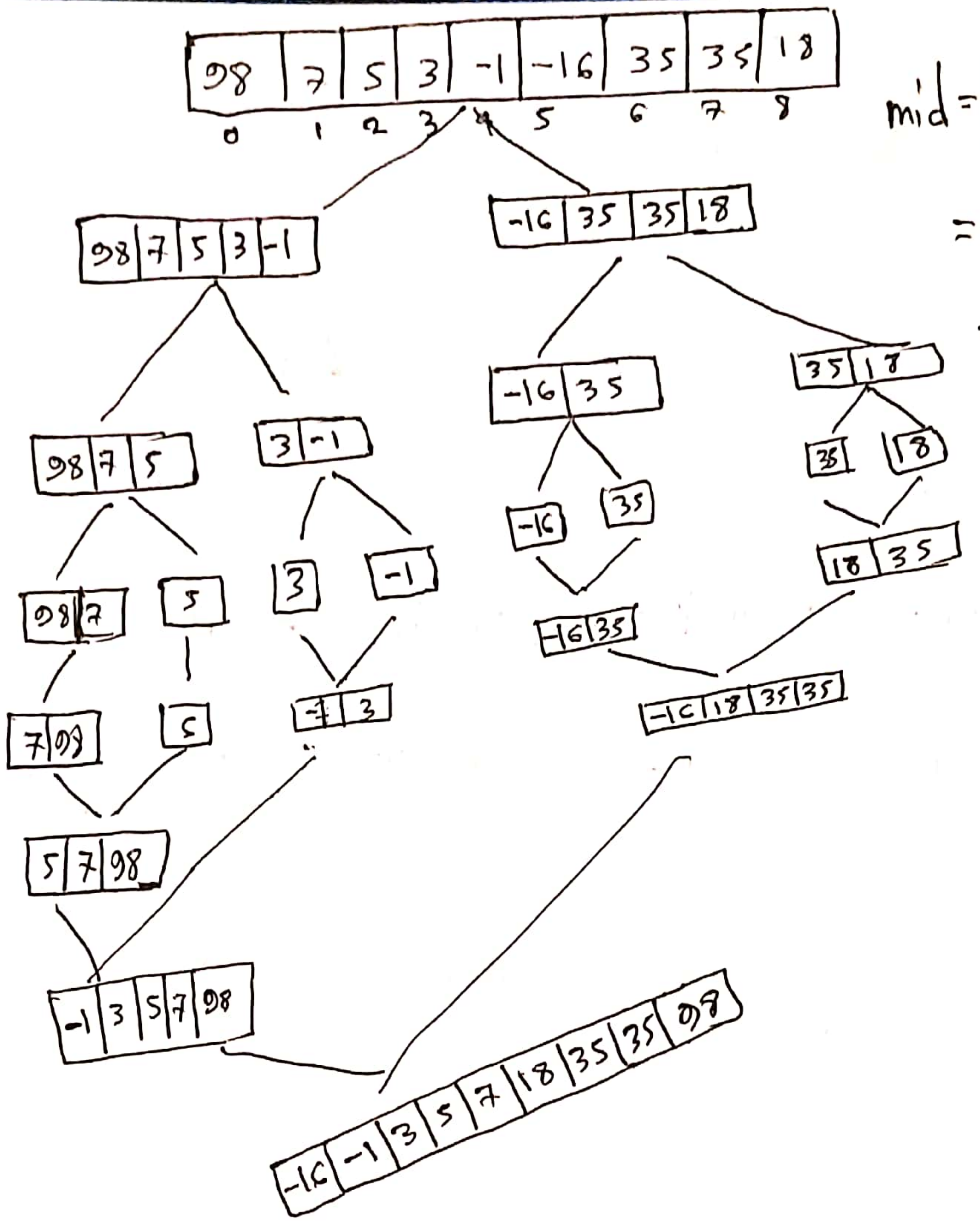
বড় number left এ থাকবে always

optimal substructure property

$L[i] \le R[j]$

$\boxed{L[i]}$

i++



$$mid = \left\lfloor \frac{low + high}{2} \right\rfloor$$

$$= \left\lfloor \frac{0+8}{2} \right\rfloor$$

$$= 4$$

if $p < r$

$q = \lfloor (p+r)/2 \rfloor$

MERGE-SORT(A, p, q) $T(n/2)$

MERGE-SORT$(A, q+1, r)$ $T(n/2)$

MERGE(A, p, q, r) $O(n)$

(যদি same ভ্যালুগুলো হয় তবে এইটার Avoid করার জন্য)

$q - p + 1$
$3 - 0 + 1$
$= 4$

$r - q$
$7 - 3$
4

8
4 4

p = left.index
R = left
q = mid

$\frac{0+7}{2} = 3$

MERGE

$n_1 = q - p + 1$

$n_2 = r - q$

let $L[1 \ldots n_1+1]$ and $R[1 \ldots n_2+1]$ be new array

for $i = 1$ to n_1

$L[i] = A[p+i-1]$

for $j = 1$ to n_2

$R[j] = A[q+j]$

$L[n_1+1] = \infty$

$R[n_2+1] = \infty$

$i = 1$

$j = 1$

for $k = p$ to r

if $L[i] \le R[j]$

$A[k] = L[i]$

$i = i+1$

else $A[k] = R[j]$

$j = j+1$

L $\underbrace{[\,.\,|\,.\,|\,.\,|\,.\,|\,\infty\,]}_{n_1}$ $\quad$ $\underbrace{[\;|\;|\;.\,|\,.\,|\,\infty\,]}_{n_2}$

$n_1 + 1$ $\quad$ $n_2 + 1$

if ∞ এর কাজ
centinel case

$$T(n) = T(n/2) + T(n/2) + O(n)$$

$T(n) = 2T(n/2) + O(n)$ } recursion Relationship

$$T(n) = \begin{cases} \Theta(1) & ; n = 1 \\ 2T(n/2) + \Theta(n) & ; \text{otherwise or } n > 1 \end{cases}$$

$$= \begin{cases} c \\ 2T(n/2) + cn \end{cases}$$

$$= \begin{cases} 1 \\ 2T(n/2) + n \end{cases}$$

CamScanner

Date :

Page :

$\{5, 2, 4, 7, 1, 3, 2, 6\}$

$\underbrace{0 \quad 1 \quad 2 \quad 3}$ $\underbrace{4 \quad 5 \quad 6 \quad 7}$

$$mid = \left\lfloor \frac{l+h}{2} \right\rfloor$$

$$= \frac{0+7}{2}$$

$$= 3$$