24
31.1.202

TM

Introduction of Algorithm :-

- Step-by-step procedure to solve any Problem
- a set of instruction

Advantages :-

- simple language
- check and fix complexity

Criteria :-

- Input
- Output
- ~~Finiteness~~ Difiniteness
- Finiteness
- Effectiveness

Methods :- (for analyzing algorithm)

- Correctness
- Time complexity
- Space complexity

- pseudocode

HW – দুটো Number এর মধ্যে maximum বের করো, এবং display করো

- **Computational Complexity :-**
  - program time কত নিচ্ছে, space কত নিচ্ছে
  - শুরু থেকে শেষ পর্যন্ত (depends on number of inputs and steps)

- Sum(1, 3)

$= 5+6+4$

$= 15$

#	Value
1	5
2	6
3	4
4	7
5	2
6	5
7	3
8	9
9	8
10	1

Steps:

$5 + (0) = 5$ ---- (i)

$5 + 6 = 11$ ---- (ii)

$11 + 4 = 15$ --- (iii)

Time = 3s (1+1+1)

Task = Sum(start, end)

Task	Steps ~~Details~~	~~Steps~~ Details	time	time complexity
Sum(1,3)	5+(0)=5 5+6=11 11+4=15	5+6+4	3s	10s
Sum(4,4)	7+(0)=7	7	1s	10s
Sum(1,10)	5+(0)=5 5+6=11 11+4=15 15+7=22 22+2=24 24+5=29 29+3=32 32+9=41 41+8=49 49+1=50	5+6+4+7 +2+5+3 +9+8+1	10s	10s

Time in sec = number of data

Number of task = T

Number of data = N

time = N sec

time complexity = N

task (i) = sum('Start, end)

(i) Total = 10+10+10 = 30s

(ii) T=3, N=10

$T = 10^3$

$N = 10^4$

$TC = 10^7$

(iii)

Time complexity of all tasks = 30×10

= T*N

Task, T = Sum_data [i] = sum(1,i)

#	Value	Sum_data
1	5	5
2	6	11
3	4	15
4	7	22
5	2	24
6	5	29
7	3	32
8	9	41
9	8	49
10	1	50

- Sum (Start, end)

= Sum_data[end] – Sum_data[Start-1]

Sum (3, 6) = sum_data[6] – sum_data[2]

= 29 – 11 = 18

Task	Steps	Details	Time	Time Complexity
Sum(1,3)	=Sum_data[3] – Sum_data[0] = 15 – 0 = 15	5+6+4	1s	3s
Sum(4,4)	Sum_data[4] – Sum_data[3] = 22 – 15 = 7	7	1s	3s
Sum(1,10)	Sum_data[10] – Sum_data[0] = 50 – 0 = 50	5+6+4 +7+2+5 +3+9+8 +1	1s	3s

Max = 1s

Number of data = N

time = 1s

T = 3, N = 10

time complexity

= 3*1

= T*1

= T

HW

Step 1: Start

Step 2: Declare two integer a and b, max.

Step 3: Input a and b

Step 4: If a<b then max then max = b

else max = a

Step 5: Display max

Step 6: End

02-04-2024

CSE 246

Find min :-

Start 7 3 5 1 8

$1 \times 5 = 5s$

min = 7 7 3 3 1 1

Sorting :

7 3 5 _ 8	5s
7 _ 5 _ 8	5s
7 _ _ _ 8	5s
7 _ _ _ 8	5s
_ _ _ _ _	5s

$5 \times 5 = 25s$

⇒ times × element

→ 1, 3, 5, 7, 8

01 - code

```
function code(n)
{
  return n;
}
```

n	# steps
1	1
5	1
1000	1

∴ time complexity = 1s

2 - code

```
function compute(n)
{
  sum = n + 5;
  return n;
}
```

n	steps
1	2
5	2

2s = complexity

Scanned with CamScanner

03 - code

```
function compute(n)
{  if(n<0) {
        return 1;
   else {
        return 0;
   }
}
```

n	#steps
1	2
5	2
-1	2

time complexity = 2s

04-code

```
function compute(n)
{
   if (n<0) {
     n = n*-1
   }
   return n;
}
```

n	steps
1	2
5	2
-1	3

time complexity = 3s

05-code

```
function compute(n)
{
    sum=0 ;          → 1
    for(i=0; i<=100; i++) {
         sum = sum+i;        → 1×100
    }
   return sum;        → 1
}
```

n	step
1	102
5	102
-1	102

time complexity = 102s

06-code

previous 5 code
but, for(i=0 ; i<=n, i++) {

n	step
1	n+2=3
5	7
100	102

time complexity = n+2

07 - code

n	Step
1	3
5	7
100	102

$n+2$
= complexity

```
function compute(n)
{
   key = (n*3) % n;  ——> 1
   for (i = 1 ; i<=n ; i++) {  ——> n*1
      if (~~key~~i = = key) {
         return true;
      }
   }
   return false;  ——> 1
```

08 - code

Time complexity = $\sqrt{n}$

n	steps
1	2
5	$\sqrt{n}$
2	1

```
function compute (n)
{
   if (n<2) {
      return false;
   }
   for (i = 2; i*i <= n; i++) {
      if (n % i == 0) {
         return true;
   } }
   return false;
}
```

satisfy করলে execution এর Time count হবে,

satisfy না করলে, execution না হলে count ও হবে না,
example : 2 = n

09 - code

```
function compute (n)
{  sum = 0;  ——> 1
   for (i = 1; i<=n; i++)  ——> 1 x n
{     for (i = 1; j<=n; j++) {  ——> 1 x n
         sum = (sum + i + j);
      }
}  return sum;  ——> 1
}
```

$1 + n^2 + 1$

$= n^2 + 2$

$\therefore$ time complexity n^2+2

10 - Code:

```
function complete(n)
{
 count = 0;                          → 1
 for (i=1, i<=n, i++){               → n
   count = count+1;
 }
 for (i=1; i<=n; i++){               → n^2
   for(j=1; j<=n; j++){
     count = count+1;
 } }
 return 0;  → 1
}
```

$\boxed{n^2+n+2}$ → Time complexity

C.W
07-02-2024

CSE 216

$(1+3+6+10+ \ldots\ldots + S_n)$

$n=1$ series $=1$
$n=2$ series $=3$
$n=3$ series $=6$

$S_n = 1+2+3+\ldots+n$
$S_1 = 1$
$S_2 = 1+2 = 3$
$S_3 = 3+3 = 6$

Soln - 2 TC:

$\cancel{i * 2n - (i \times 2n) + 2} = \frac{n^2-n}{2} + 2n + 3$

$= \frac{n^2-n+4n+6}{2} = \frac{n^2+3n+6}{2}$ = Time complexity (slide এর ② নং)

$6n^2+10n+217$

Steps for $6n^2+10n+217$

n	$6n^2+10n+217$	$6n^2$	$10n$	217
1	233	6 (2.5%)	10 (4%)	217 (93%)
10	917	100 (65%)	100 (10%)	217 (23%)
100	61217	60000	1000	217
1000		6000000	10000	217

$6n^2+10n+217$

$6n^2$ → Fastest growing Term

	FGT	TC
$2n^3+3n^2+4n+5$	$2n^3$	n^3
$3n^2+4n+5$	$3n^2$	n^2
$4n+5$	$4n$	n
5	5	1

1. Sum = d + 5
2. Sum = Sum * 3

2 steps

$\therefore$ Sum = (d+5) * 3 — 1 step

Assymptetic Notation growing:

① ② ③ ④ ⑤

⑥ ⑦ ⑧ ⑨ ⑩

Best case : 1

Average case : (4+3+~~6~~8)/4

Worst case : 10

(I) Ω : Lower Bound / Best case → Useless

(II) Θ : Both Upper Bound and lower Bound / Average case

(III) O : Best way upper case / Best case (Big-O)

$3n^2+2n+4 \rightarrow 3n^2 \rightarrow n^2 \rightarrow$ FGT

$f(n) = 3n^2+4n+5$

$g(n) = n^2$

Formula: $O(g(n))$:=

$\{f(n)$: there exist positive constants c and n_0 such that $0 <= f(n) < c \cdot g(n)$ for all $n >= n_0\}$

□ $f(n) = 4n+5$

$0 <= f(n) < \underbrace{c \cdot g(n)}$

$g(n) = n$

$\therefore c \cdot g(n) \geq f(n)$

~~g(n = c~~ $\Rightarrow c \cdot n >= 4n+5$ $[g(n) = n]$

$\Rightarrow 5n >= 4n+5$ $[c = 4+]$

$\Rightarrow 5n - 4n >= 5$

$\Rightarrow n \geq 5$

Ques: Why we take $g(n) = n^2$ from n, n^2? or $g(n) = n^3$ from n, n^2, n^3?

□ $f(n) = 2n^2+3n+5$

$g(n) = n2$

$\therefore c \cdot g(n) \geq f(n)$

$\Rightarrow c \cdot n^2 \geq 2n^2+3n+5$

$\Rightarrow 3n^2 \geq 2n^2+3n+5$

$\Rightarrow 3n^2 - 2n^2 \geq 3n+5$

$\Rightarrow n^2 \geq 3n+5$

$\Rightarrow$

* Worst case

12-02-24

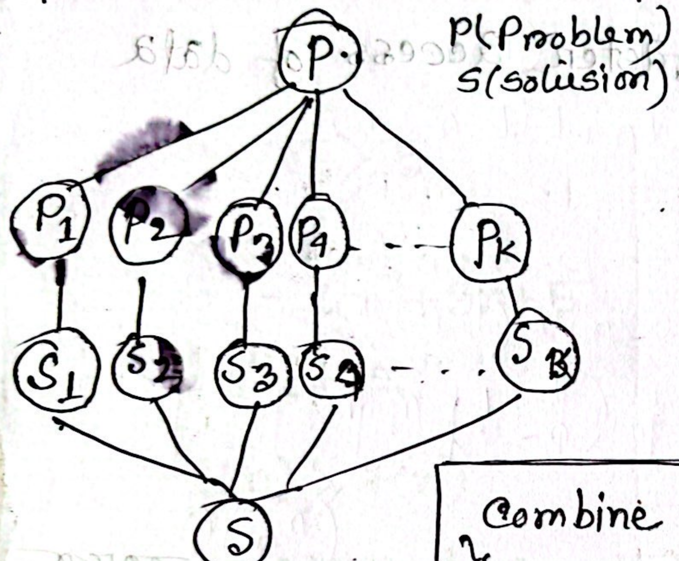
CSE 246

logN → Big problem কে subset এ solve করি

Divide and Conquer Algorithm :-

Recurance relation ; This a way of approach to design algorithm.

~~Lecture 6 (Quicksort)~~




```
DAC (P) {
  if (Small (P)
     S(P)
  else {
     divide P into
        P1, P2, P3, P4 .... Pk
  Apply DAC(P1), DAC(P2)... DAC(Pk)
  Combine DAC(P1), DAC(P2) ......... DAC(Pk).
}
```

(i) Binary Search
(ii) Quick Sort
(iii) Multiple Sort etc.

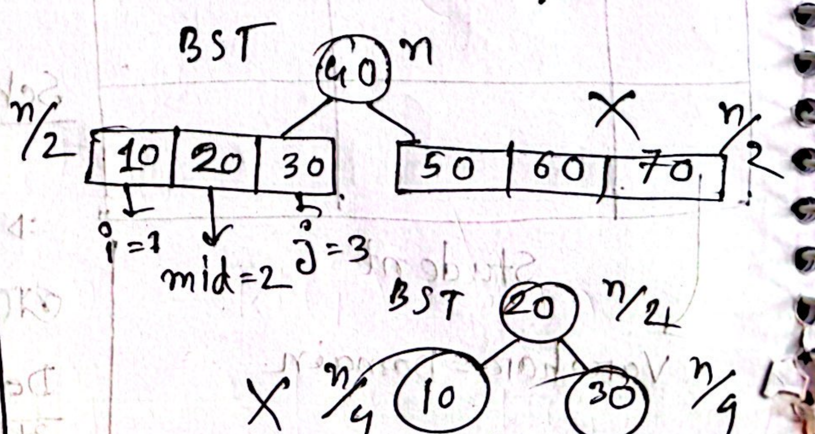
```
BS(a, i, j, x)   → searching element
{
     mid = (i+j)/2
     if (a[mid] == x)
          return mid;
else if (a[mid] > x)
          BS(a, i, mid-1, x)
  else (a[mid] < x)
          BS(a, mid+1, j, x)
```

10 20 30 (40) 50 60 70

$i = 1$ $j = 7$

$mid = \frac{1+7}{2} = \frac{8}{2} = 4$

∴ middle element = 40



n
$\downarrow$
$n/2$
$\downarrow$
$n/4$
$\downarrow$
$n/8$

$\boxed{T(n) = T\left(\frac{n}{2}\right) + c}$ This time complexity is called recursiance relation

Time when, mid = x checking ~~time~~ Best case

মনে করি, small data set এর ক্ষেত্রে $n = 1$

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

$\therefore T(n) = T(n/2) + c$ ---- (i)

Substitution Method: $T(n/2) = T(n/4) + c$ ---- (ii)

$T(n/4) = T(n/8) + c$ ---- (iii)

(ii) কে (i) নং এ বসিয়ে পাই, (substitution)

$T(n) = T(n/4) + c + c = T(n/4) + 2c$

$= T\left(\frac{n}{2^2}\right) + 2c$

$= T(n/8) + 3c$

$= T\left(\frac{n}{2^3}\right) + 3c$

$= T\left(\frac{n}{2^4}\right) + 4c$

K Times ---------------- $\therefore T\left(\frac{n}{2^k}\right) + kc$

Let, $n = 2^k$,

$T\left(\frac{n}{n}\right) + kc = T(1) + kc = 1 + kc$

$= 1 + \log n^c$

$= O(\log_2 n)$

$n = 2^k$

$\Rightarrow \log n = k \log 2$ $[\log 2 = 1]$ কারণ base 2

$\Rightarrow \log n = k \times 1$ $\therefore k = \log n$

Scanned with CamScanner

$$T(n) = \begin{cases} 1 & n=1 \\ n*T(n-1) & n>1 \end{cases}$$

$T(n) = n*T(n-1)$ ...... (i)

$T(n-1) = (n-1)*T(n-2)$ ...... (ii)

$T(n-2) = (n-2)*T(n-3)$ ...... (iii)

$T(n) = n*(n-1)*T(n-2)$

$= n*(n-1)*(n*2)*T(n-3)$

$n-1$

$= n*(n-1)*(n-2)*(n-3)\ldots 3*2*1$

$= n*n\left(1-\frac{1}{n}\right)*n\left(1-\frac{2}{n}\right)*\ldots n\left(\frac{1}{n}\right)$

$O(n^n)$.

$$T(n) = \begin{cases} 1 & n=1 \\ 2T\left(\frac{n}{2}\right)+n & n>1 \end{cases}$$

$T(n) = 2T\left(\frac{n}{2}\right)+n$ ...... (i)

$\cancel{T(n-1) = 2T\left(\frac{n-1}{2}\right)+n = T(n-1)+(n-1)}$

$\cancel{T(n-2) = 2T\left(\frac{n-2}{2}\right)+n = T(n-2)+(n-2)}$

$T\left(\frac{n}{2}\right) = 2T\left(\frac{n}{4}\right)+\frac{n}{2}$ ...... (ii)

$T\left(\frac{n}{4}\right) = 2T\left(\frac{n}{8}\right)+\frac{n}{4}$ ...... (iii)

$T(n) = 2*\left[2T\left(\frac{n}{4}\right)+\frac{n}{2}\right]+n = 4T\left(\frac{n}{4}\right)+\frac{n}{2}+n$

$\cancel{= 2*\left[2*\left\{2T\left(\frac{n}{8}\right)+\frac{n}{4}\right\}+\frac{n}{2}\right]+n} = 4T\left(\frac{n}{4}\right)+2n$

Scanned with CamScanner

$$= \cancel{8T}\ 4\left[2T\left(\frac{n}{8}\right)+\frac{n}{4}\right]+2n$$

$$\Rightarrow 8T\left(\frac{n}{8}\right)+n+2n$$

$$\Rightarrow 8T\left(\frac{n}{8}\right)+3n$$

$$\Rightarrow 2^3 T\left(\frac{n}{2^3}\right)+3n$$

$$\Rightarrow 2^4 T\left(\frac{n}{2^4}\right)+4n$$

⋮ K times

$$\Rightarrow 2^K T\left(\frac{n}{2^K}\right)+Kn.$$

Let, $n = 2^K$; $\log n = \log_2 2^K$

$$\Rightarrow K = \log n$$

$$\therefore 2^K T\left(\frac{n}{2^K}\right)+Kn = 2^K T(1)+Kn$$

$$= n.1 + n\log n$$

Time Complexity: $O(n\log n)$.

LAB

24, 9, 29, 14, 19, 27

```
Quick Sort (A,s,e){
  if(s<e)
  {
    Partition(A,s,e) => p कौ?
    QuickSort(A, s, p-1)
    QuickSort(A, p+1, e)
  }
}
```

p 24 9 29 14 19 27

s e

24 9 29 14 19 27

19 9 29 14 $\boxed{24}$, 27

19 9 29 14 $\boxed{24}$ 27

19 9 $\boxed{24}$ 14 29 e 27

$\boxed{19\ 9\ 14}$ $\boxed{24}$ $\boxed{29\ 27}$

P/2 p P/2

9 14 19 24 27 29

```
Partition (A, s, e) {
    Pivot : A[s]
    i = s - 1
    for (j = s to e-1)
  { if ( A(j) < PIVOT)
       i = i+1
    swap P{A[i] with A[j] }
     swap + [i+1] with A[e]
     return i+1;
}
```

35	50	15	25	80	20	90	45

P, S, e

(2)

20	50	15	25	80	35	90	45

S, P,e

20	50	15	25	80	35	90	45

S, P,e

(3)

20	35	15	25	80	50	90	45

P,S e e

(4)

20	25	15	35	80	50	90	45

S P,e

P

35 50 15 25 80 20 90 45

S e e e

(2)

20 50 15 25 80 35 90 45

S e P

20 $\boxed{35}$ 15 25 80 50 90 45

P swap e

20 25 15 $\boxed{35}$ P 80 50 90 45

e

CSE246

*** • Divide and conquer
• x^y

$\Rightarrow 2^3 = 2\times2\times2$

• Pivot element

Quick sort

□ Partition code: →

35 50 15 25 80 20 90 45

i / Pivot

left/i বড় খুঁজবে
right/j ছোট খুঁজবে

20 50 15 25 80 35 90 45

20 35 15 25 80 50 90 45

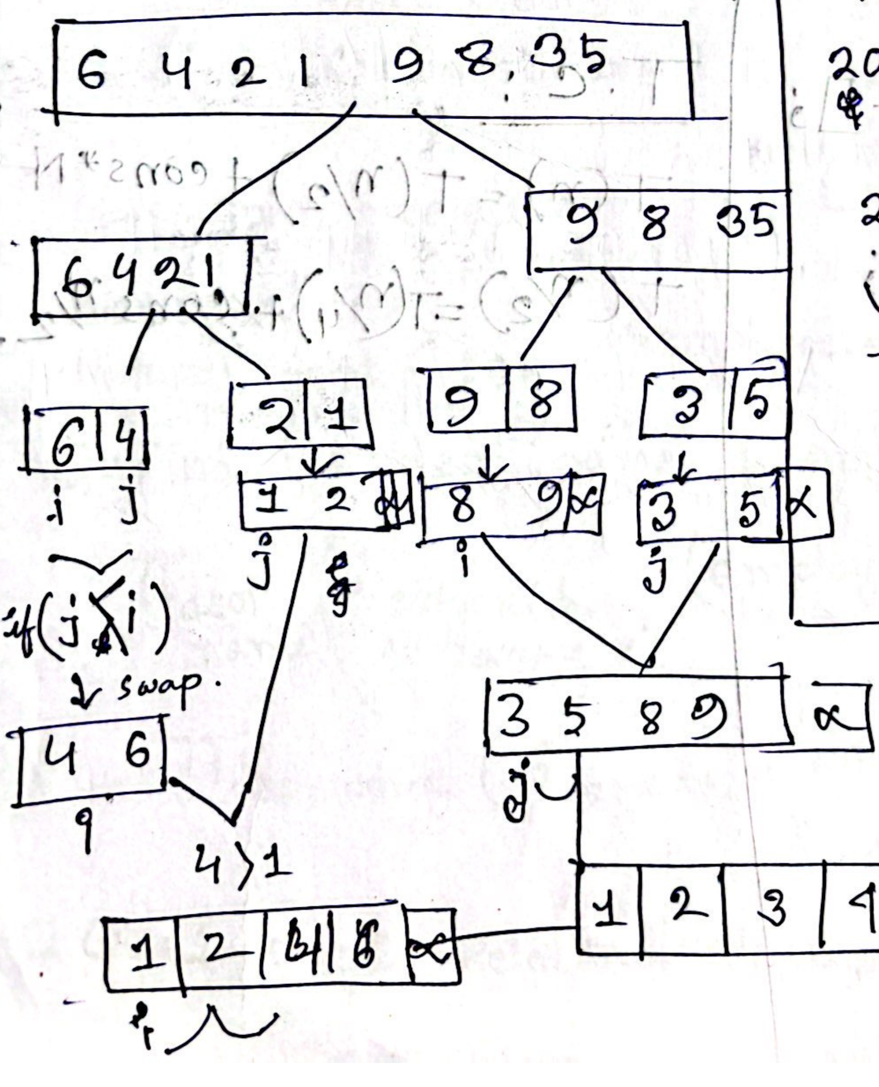
20 25 15 35 80 50 90 45

if (low<high)

• Quicksort → Programiz

• Time complexity
• Quick sort (only for lab)
• Divide & Conquer

Merge sort



27 - lab submission
6 March - Quiz on TC

TC : n logn

first index, last index

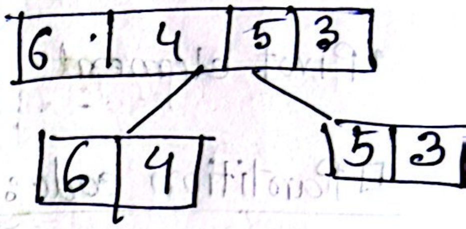
merge-sort (A, p, r) {   (A → array)

if $(p<r)$

{ $q = \lfloor p+r/2 \rfloor$

~~Partition~~ / Merge Sort (A, p, q)

~~Partition~~ / Merge Sort $(A, q+1, r)$



Merge(A, p, q, r)

{

$n_1 = q-p+1$

$n_2 = r-q$

Create array $L[1, \ldots \ldots n1+1]$

" " $R[1, \ldots n2+1]$

for $(i=1$ to $n1)$

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

for $(j=1$ to $n2)$

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

$L[n1+1] = \infty$

$R[n2+1] = \infty$

$i=j=1$

for $(k=p$ to $r)$

if $(L[i] \leq R[j])$

$A[k] = L[i]$,

$i{+}{+}$;

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

$j{+}{+}$;

T.C :

$T(n) = T(n/2) + cons*N$

$T(n/2) = T(n/4) + cons\ n/2$

04-03-2024

CSE 246

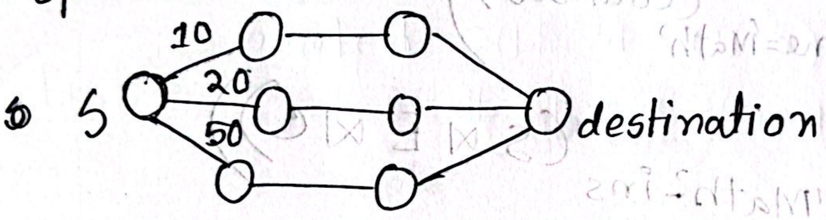
Greedy Algorithm

- to find best choice
- local optimal choice বের করবো global optimal choice ~~বের ক~~ কে পাওয়ার জন্য

- Time Complexity
- Divide & Conquer
- Recurrence

Local optimal choice = each stage
- feasible solution
- optimal solution

Next Wednesday
Quiz - 06-03-24



Application :
- Scheduling
- Job sequencing
- Prims } MST
- krusk } MST
- Digkstra } Shortest Path
- Huffman encoding

Coin change problem : Minimum coin change problem.

Coins [ ] = {5, 10, 20, 25}

Val = 50

Fractional Knapsack Problem

	ob1	ob2	ob3
Profit	25	20	15
Weight	18	15	20

$M = 20$

(i) Greedy about profit:-

highest profit রেখে রাখলাম → obj1 } 18, } 2

$M = 20$

$\therefore 25 + \frac{2}{15} \times 24 = 25 + \frac{48}{15} = 25 + 3.2 = 28.2 =$ Max profit

(ii) Greedy about weight:

minimum weight → obj3 } ~~20~~ 15, } 10

$15 + \frac{10}{15} \times 24$

$= 31$

(iii) Greedy about Proportion P/ω

	obj1	obj2	obj3
Profit	25	24	15
Weight	18	15	10
P/ω	1.3	1.6	1.5

$24 + \frac{5}{10} \times 15$

$= 31.5$

obj2] 15, } 5

Pseudocode for knapsack:

1. for (i=1 to n)
2. calculate profit / weight → O(n)
2. sort objects in decreasing P/ω → O(nlogn)
3. for(i=1 to n) if M>0 && ω[i] <M

Scanned with CamScanner

M = M − ω[i];
P = P + P [i];

else
break;

for i=1 to {
if (M>0) {
$P = \cancel{P} + P[i] * \frac{M}{\omega[i]}$
}

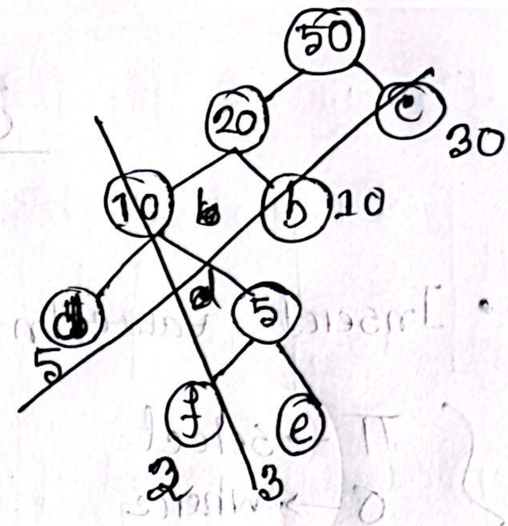
$T.C = O(n) + O(n\log n) + O(n)$
$= O(n\log n)$

Activity Selection Problem:

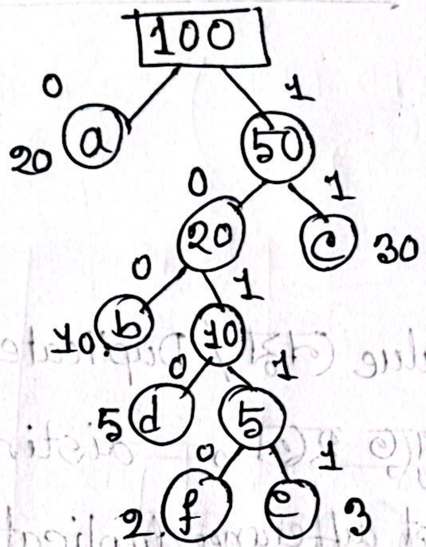
Sorted থাকলে $O(n)$
Sorted না থাকলে $O(n) + O(n\log n)$.

CSE246

Huffman Codes



a = 50
b = 10
c = 30
d = 5
e = 3
f = 2



$$\text{Transmit} = \frac{M}{B}$$

$$= \frac{700}{B}$$

a = 0 (50×1)
b = 100 (10×3)
c = 11 (30×2)
d = 1010 (5×4)
e = 10111 (3×5)
f = 10110 (2×5)

Total 185.

* DDL, DML (Theory)
* SQL
* Relational Algebra

$\boxed{2^n = \log n}$

11/03/2024

CSE246

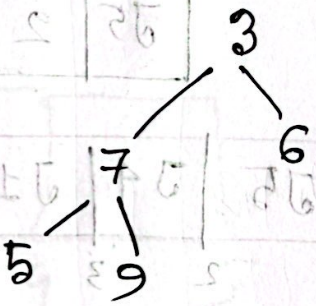
Huffman encoding

$M = 30$

aa bbbb a bbb ccc ddd eee ccc eee dd ee

30×7
$= 210$ bit

$a = 3$ $d = 5$
$b = 7$ $e = 9$
$c = 6$



3	7	6	7	9	

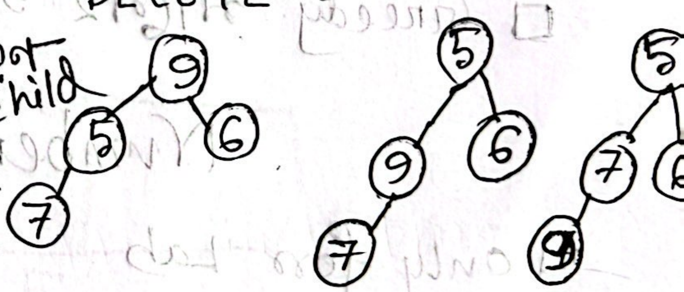
1

$LC = 2i+1$
$RC = 2i+2$

Child to parent $\left[\frac{i-1}{2}\right]$

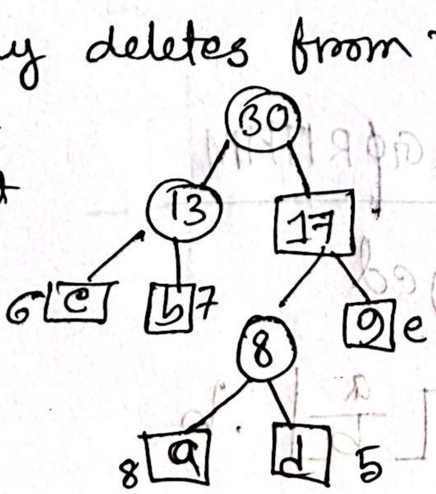
DELETE 3

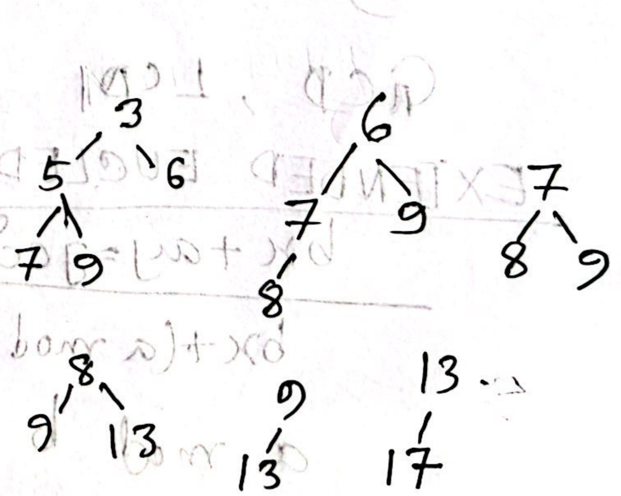
আগে root 9 এর child নাই, last index থেকে



Heap only deletes from root

1 → left
2 → Right





Scanned with CamScanner

Job Scheduling Algorithm

কোন job কী order এ করলে loss কম হবে।

Single Processor এ Applicable.

- Job Scheduling Loss Minimization Problem

	T	L
J1	2	1
J2	4	2
J3	1	3
J4	3	5
J5	2	6

$\frac{L}{T}$

$\frac{2}{4} =$

$\frac{L}{T}$
0.5
0.5
3
1.6
3

J3	J5	J4	J1	J2
0 – 1	1 – 2	2 – 3	3 – 4	4 – 5

- Greedy মানেই sort and ($n\log n$)

Number Theory

- only for Lab

GCD, LCM

EXTENDED EUCLEDIAN ALGORITHM

$$\frac{bx + ay = gcd}{bx + (a \bmod b)\, y} = gcd$$

$$a \bmod b = a - \left\lfloor \frac{a}{b} \right\rfloor . b$$

Scanned with CamScanner

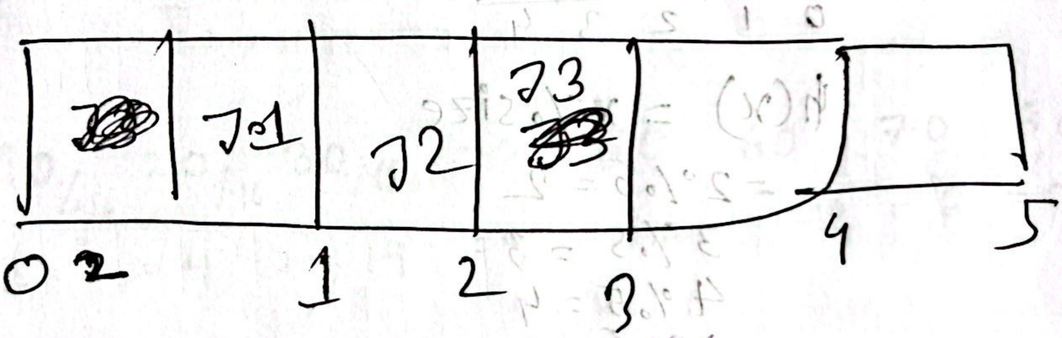
$$bx + \left(a - \lfloor \tfrac{a}{b} \rfloor . b\right) . y$$

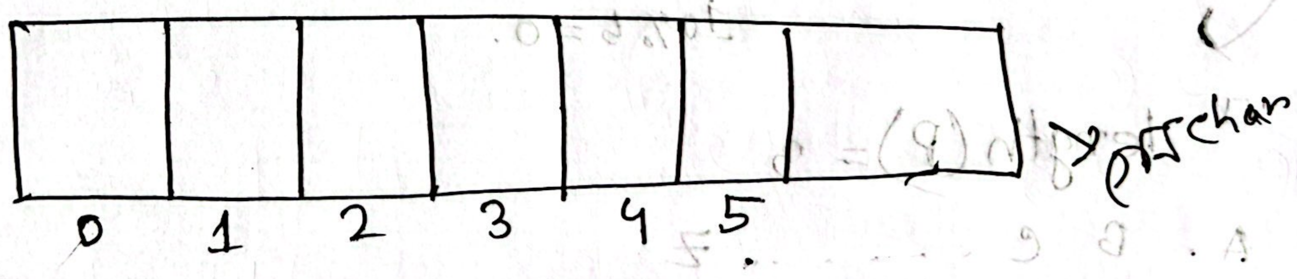
$$= bx + ay - y \lfloor \tfrac{a}{b} \rfloor . b$$

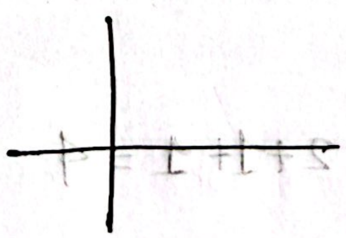
$$= ay + b\left(x - y \lfloor \tfrac{a}{b} \rfloor\right)$$

□ Sieve Method

jid →	J3	J5	J1	J2	J4
P →	20	15	10	5	1
D →	2	2	1	3	3







1/03/2...

***

String Pattern

Text :	1 2 3 4 5 6 7 8 9 10 AABAACABAA
Pattern :	BAA

3, 8

$T \cdot C = O(nm)$

① Bruteforce

② efficient :

ⓐ Rabin Karp

ⓑ KMP

☐ Rabin Karp :-

☐ Hash Method:

Data: 2 3 4 5 10

5	10	2	3	4
0	1	2	3	4

$h(x) = x \% size$

$= 2 \% 5 = 2$

$3 \% 5 = 3$

$4 \% 5 = 4$

$5 \% 5 = 0$

$10 \% 5 = 0.$

☐ length (P) = 3

A . B C . . . . Z

1 2 3 ---------- 26

Pattern Value = B A A = 2+1+1 = 4

A A B = 1+1+2 = 4

4-1=3 , A B A = 3+1 = 4

4-1=3, B A A = 3+1 = 4

Scanned with CamScanner

Hash Function Technique :-

$\boxed{T.C = m+n}$

Text : AABAACABAA

Pattern: ~~AB~~B AA

Rolling Hash Technique:-

$\overset{2}{B}\overset{1}{A}\overset{0}{A}$

Assume, $P = 5381 \longrightarrow (1\times 26^0)\%P$

$$[(2\times 26^2)\%P + (1\times 26^1)\%P + (1\times 26^0)\%P]\%P = 1379$$

$$AAB = [(1\times 26^2)\%5381 + (1\times 26^1)\%5381 + (2\times 26^0)\%5381]$$

$$= 676 + 26 + 2 = 704$$

10%16	20%16	30%16	40	50	60	70	80	90	100
= 2	= 4	= 14	= 8	= 2	= 12	= 6	= 0	= 10	= 4

□ for (i=1; i<n; i=i+2)

{ stmt; ⟶ $\frac{n}{2}$

}

$\boxed{O(n)}$

p=0;

for (i=1; p<=n; i++)

{ p=p+i; $\boxed{O(\sqrt{n})}$

}

Assume $p>n$

$\frac{K(K+1)}{2}>n$

$K^2>n$

i	p
1	0+1
2	0+1+2
3	0+1+2+3
⋮	⋮
K	0+1+....K

□ for(i=0; i<n; i++) ⟶ n

{ for(j=0; j $\underline{<i}$; j++) ⟶ n

{ stmt;

}

}

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

i	j	No. of Exc. Time
0	0 ✗	0
1	0 ✓ 1 ✗	1
2	0 ✓ 1 ✓ 2 ✗	2
3	0 ✓ 1 ✓ 2 ✓ 3 ✗	3
⋮		⋮
n		n

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

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

$= n^2$

□ for (i=1; i<n; i=i*2){

stmt; $\boxed{O(\log_2 n)}$

}

i
1
$1\times2=2^1$
$2\times2=2^2$
$4\times2=2^3$
⋮
2^K

Assume,

$i>n$

$\Rightarrow 2^K>n$

$\Rightarrow K=\log_2 n$

□ for(i=n; i>=1; i=i/2)

{ stmt;

}

$\boxed{O(\log_2 n)}$

i
n
$n/2$
$n/4 = n/2^2$
⋮
$n/2^K$

Assume,

$i<1$

$\frac{n}{2^K}<1$

$\frac{n}{2^K}=1$

$\Rightarrow n=2^K$

$\Rightarrow \log_2 n = K$

(7) for (i=0; i*i<n; i++)

{ stmt;

}

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

Assume

$i*i>n$

$\Rightarrow i^2>n$

$\Rightarrow i>\sqrt{n}$

Scanned with CamScanner

(9)
```
p = 0
for(i=1; i<n; i=i*2)
{   p++;   → logn
}
for(j=1; j<p; j=j*2)
{  stmt;  → [logP]
}
```
What is P? log(logn)

~~log log n~~

$O(\log \log n)$

---

(10)
```
for(i=0; i<n; i++){          → n
      for(j=1; j<n; j=j*2){  → logn
          stmt;
      }
}
```
$O(n\log n)$

Formula :

- for(i=0 ; i<n ; i++) → $O(n)$
- for (i=0 ; i<n ; i=i+2) → $O(n)$
- for(i=n ; i>1 ; i--) → $O(n)$
- for(i=1 ; i<n ; i=i*2) → $O(\log_2 n)$
- for (i=1 ; i<n ; i=i*3) → $O(\log_3 n)$
- for(i=n ; i>1 ; i=i/2) → $O(\log_2 n)$

```
for (i=2 ; i<=n/2 ; i++) {
    j=1
    while (j<=m){
    sum++;
    j *= 10;
    }
}
```

i	j	
2	1	$=10^0$
3	1×10	$=10^1$
4	$1\times10\times10$	$=10^2$
k times	k times	
k	10^k	

-6, -5, -3, -2, -2, 1, 5, 6

~~6 5 1 -2~~ | -2 | -3 -5 -6

03/2024

KMP Algorithm

Pattern : abc dabc

Prefix : a ab abc ~~abd~~ abcd

Suffix : c bc abc dabc

Prefix function (π)

P1 : abcd abc abf
(positions 1 2 3 4 5 6 7 8 9 10)
index : 1 2 3 1 2

0 0 0 0 1 2 0 1 2 0

P2 : abcdc abf abc
0 0 0 0 0 1 2 0 1 2 3

P3 : aa bcad aabc
0 1 0 0 1 0 1 2 3 0

P4 : aaaabaacd
0 1 2 3 0 1 2 0 0

String : ~~ababcabc~~
i
ababcabcabababd.

Pattern:

j=0	1	2	3	4	5
	a	b	a	b	d
	0	0	1	2	0

$i=1$
$j=0$

$S[i] == P[j+1]$

$i{+}{+}\ ;\ j{+}{+};$

$i=2$
$j=1$

$i=3$
$j=2$

$i=4$
$j=3$

$i=5$
$j=4$

$S[i] \neq P[j+1]$ $\boxed{j=2}$

$i=5$
$j=2$

mismatch. $\boxed{j=0}$

$i=6$
$j=0$

$i=7$
$j=1$

$i=8$
$j=2$

mismatch

String: n

$i=$ ↓
ababcabcabababd

Pattern: m

j=	0	1	2	3	4
	a	b	a	b	d
	0	0	1	2	0

$(m+n)$

$O(n)$

MID Ques → 30 Marks (5×6)

- ✓ Time complexity Big(O)
- Divide conquer, recurrence subtraction
- Set of number/scenario is it possible in divide and conquer
- String Pattern Step দেখাবে (R.K and K.MP)
- Greedy Algorithm → Code.

- String = ababcababd
- Pattern:

j=	0	1	2	3	4
	a	b	a	b	d
	0	0	1	2	0