4) $T(n) = 7T(n/2) + 18n^2$

$a = 7, b = 2, f(n) = 18n^2$

Sol $\boxed{T(n) = n^{\log_b a}[T(1) + U(n)]}$

$= n^{\log_2 7}[T(1) + U(n)]$

$\boxed{h(n) = \frac{f(n)}{n^{\log_b a}}}$

$= \frac{18n^2}{n^{\log_2 7}} \Rightarrow \frac{18n^2}{n^{2.897}} \Rightarrow 18n^{2-2.897}$

$18n^{-ve}$

$U(n) = O(1)$ $\because \boxed{i<0, O(n^0) = O(1)}$

$T(n) = n^{\log_2 7}[T(1) + O(1)]$

$= n^{\log_2 7}$

$= n^{2.897}$

→ Once ~~the~~ an algorithm is devised, it is necessary to show that it Computes the Correct answer for all possible i/ps. The process is known as algorithm validation.

→

Q u/a Binary Search:- (l, h, mid), $mid = \frac{f+l}{2}$

→ Given value & sorted array[ ], find index i such that a[i] = value, or report that no such index exists.

REDMI NOTE 5 PRO
MI DUAL CAMERA

Example :- binary Search for 33.

6 13 14 25 33 43 51 53 64 72 81 83 95 96 97

→ Algorithm binarysearch (a, f, l, x) (Recursive B.S.A)

→ Given an array a[f...l] of elements in increasing order

→ x is the element to be searched & return the position where element is found

```
{ if (l==f) then   // f small(p)
  {
    if (x==a[f]) then
        return f;
    else
        return 0;
  }
  else     //reduce p into a smaller sub problem
  {
    mid := [(f+l)/2];
    if (x==a[mid]) then
        return mid;
    else if (x<a[mid]) then
        return binsearch (a, f, mid-1, x);
    else
        return Binsearch (a, mid+1, l, x);
  }
}
```

→ Algorithm Binsearch (iteration)

```
{
  low := 1;
  high := n;
  while (low ≤ high) do
  {
    mid := [(l+h)/2];
    if (x<a[mid]) then
    high := mid-1;
```

REDMI NOTE 5 PRO
MI DUAL CAMERA

else if (x>a[mid]) then
    low:= mid+1;
else
    return mid;
}
return 0;
}

→ worst Case Analysis:

$T(n) = T(n/2)+1$ for $n>1$

if $n = 2^k$, $T(2^k) = T(2^k/2)+1 \Rightarrow T(2^{k-1})+1$

→ if number of elements on which we need to search is 'n', then no. of times array can be splited as below

for	n=3	no. of splits	=1
"	4	"	2
	5		2
"	6	"	2
"	7		2
	8	"	3

for any values of 'n', array split is continued till size of array becomes 1;

$\therefore 2^x = n$, 'n' is the size of array, and 'x' is the no. of splits

$$\boxed{x = \log_2 n}$$

Time Complexity

1) worst case analysis

$$T(2^k) = T(2^{k-1})+1$$
$$= T(2^{k-2})+1)+1$$
$$= T(2^{k-2})+2$$
$$= T(2^{k-3})+3 \ldots$$
$$T(2^k) = T(2^{k-k})+k$$
$$= T(2^0)+k$$
$$= T(1)+k$$

$\therefore T(n) = T(1) + k$
$= 1 + \log_2 n$

$$\left[\begin{array}{c} n = 2^k \\ k = \log_2 n \\ T(n) = n \end{array}\right]$$

$$\boxed{T(n) \approx \log_2 n}$$

2) **Average Case Analysis :**

→ Average no of keys comparision made by binary search is slightly smaller than in worst case.

$\therefore T(n) \approx \log_2 n$

3) **Best Case Analysis :**

→ Element to be searched is present in mid position of given array of 'n' elements

$T(n) = 1$ ∴ [no. of comparison is 1]

```
→ for (i=1; i<=n; i*2)
  {
      for (j= i; j>0; j--)
      {
          printf ("%d", j);
      }
  }
```

	s/e	freq	s×e
for (i=1; i<=n; i*2)	1	n	n
for (j= i; j>0; j--)	1	$\frac{n(n+1)}{2}$	$\frac{n(n+1)}{2}$
printf ("%d", j);	1		

$n * \frac{(n+1)}{2} \Rightarrow \frac{1}{2}[n^2+n]$

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

$O(n^2)$

→ Auxiliary space is extra space or temporary space used by the algorithms during its execution.

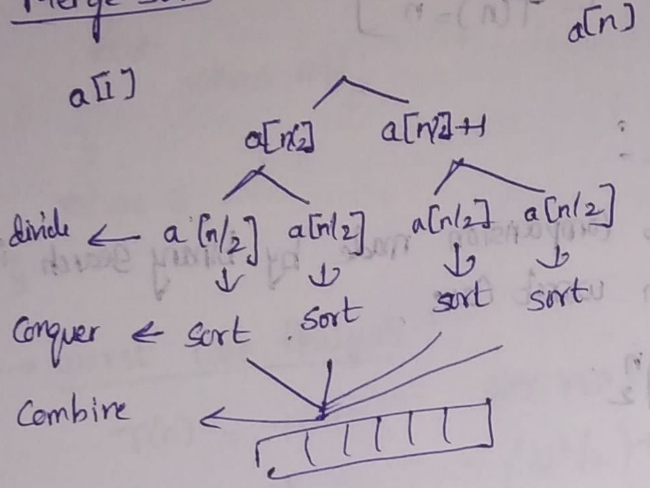
→ instruction space is used to save compiled instruction in the memory (space for code)

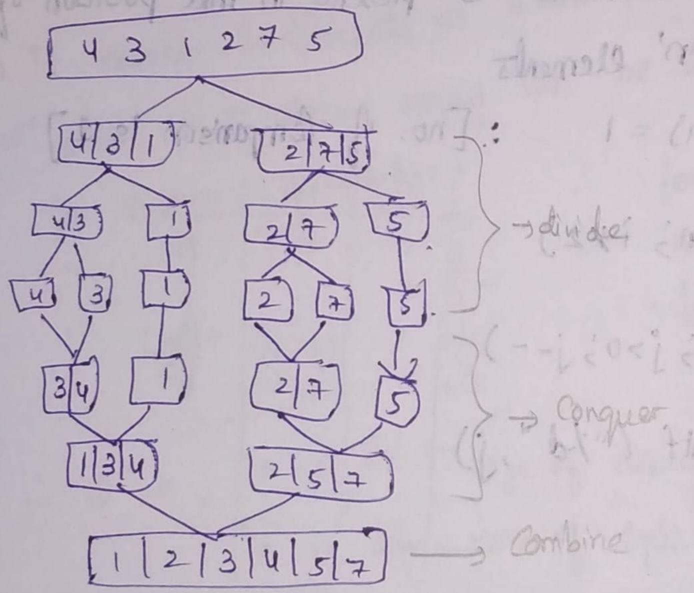
→ data space is used to store data, variables, and constants which are stored by the program and it is updated during execution

→ environmental stack space is used to storing the addresses of functions while a module calls another module or functions during execution.

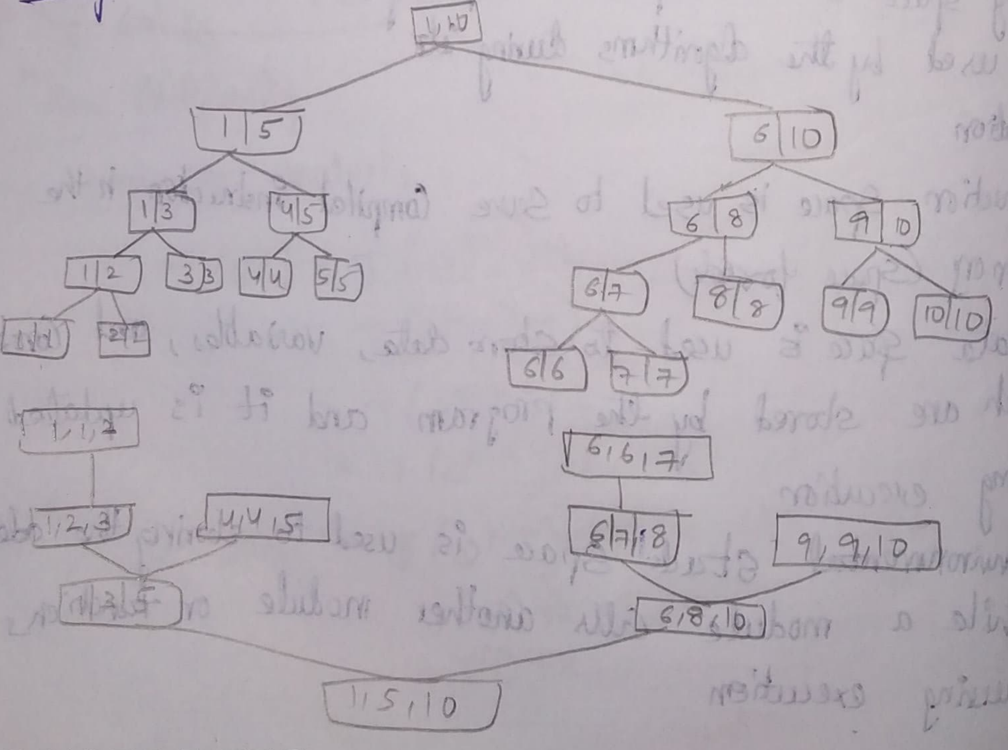
29/9/21

Merge Sort [divide & Conquer]





→ Binary decision tree that represents the tree calls of Merge sort.



Algorithm for Merge sort

```
Algorithm mergesort (low, high)
{
    if (low < high) then
    {
        mid = ((low+high)/2);
        merge sort (low, mid);
        merge sort (mid+1, high);
        merge (low, mid, high);
    }
}
```

```
Algorithm merge (low, mid, high)
{
    i = low;
    j = mid+1;
    h = low;
    while ( h <= mid) and (j <= high) do
    {
        if ( a[h] ≤ a[j]) then
        {
            b[i] = a[h];
            h = h+1;
        }
        else
        {
            b[i] = a[j];
            j = j+1;
        }
        i = i+1;
    }
    if (h > mid) then
        for (k := j to high) do
        {
            b[i] = a[k];
            i = i+1;
        }
    else
    {
        for (k := h to mid) do
        {
            b[i] = a[k];
            i = i+1;
        }
    }
    for k := low to high do
        a[k] := b[k];
}
```

Analysis of Merge Sort :-

$$T(n) = \begin{cases} a & n=1 \text{ (small)} \\ T(n/2) + T(n/2) + cn \end{cases}$$

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

$$\boxed{T(n/2) = 2T(\tfrac{n/2}{2}) + Cn/2}$$

$$\Rightarrow 2\left[2T(n/4) + Cn/2\right] \qquad \cancel{2}\,Cn/\cancel{2} \Rightarrow Cn$$

$$\Rightarrow 4T(n/4) + Cn$$

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

$\vdots$

$= 2^i T(n/2^i) + icn$

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

$\Rightarrow \cancel{n}\, T(n/\cancel{n}) + \log_2 n \; cn$

$\Rightarrow nT(1) + \log_2 n \; cn$

$\Rightarrow na + \log_2 n \; cn$

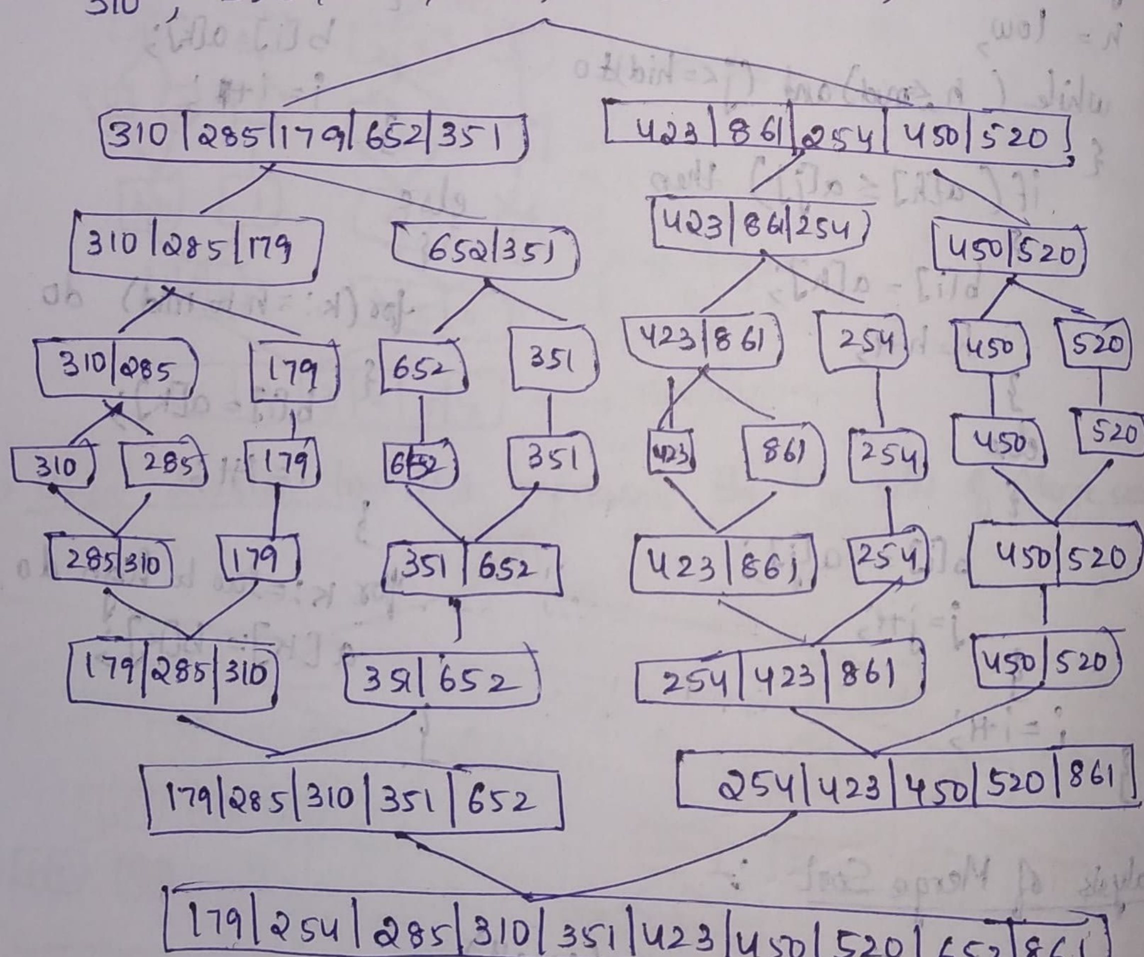
$T(n) \Rightarrow a \cdot n + cn \cdot \log n$

$T(n) = O(n \log n)$

$\boxed{\text{Average Case} \quad T(n) \approx O(n \log_2 n)}$

Q) Sort the following using merge sort

310, 285, 179, 652, 351, 423, 861, 254, 450, 520



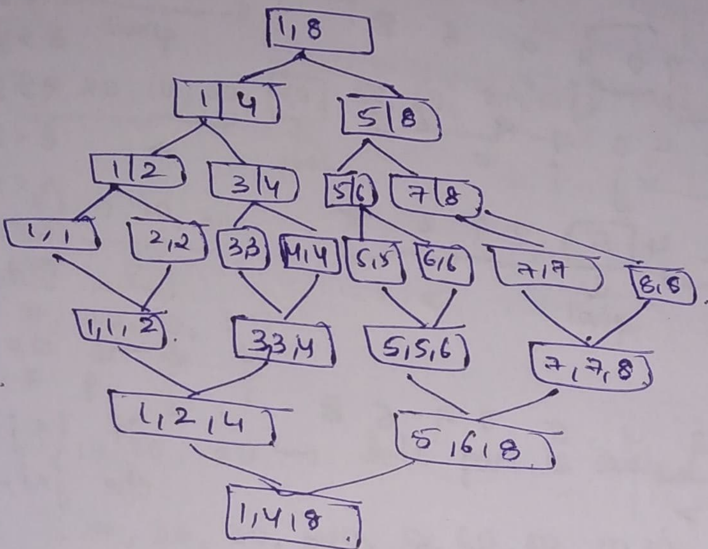
l= 0, k= 0, j=

low = 0, high => 8

mid = (0+8)/2

mid = 4.

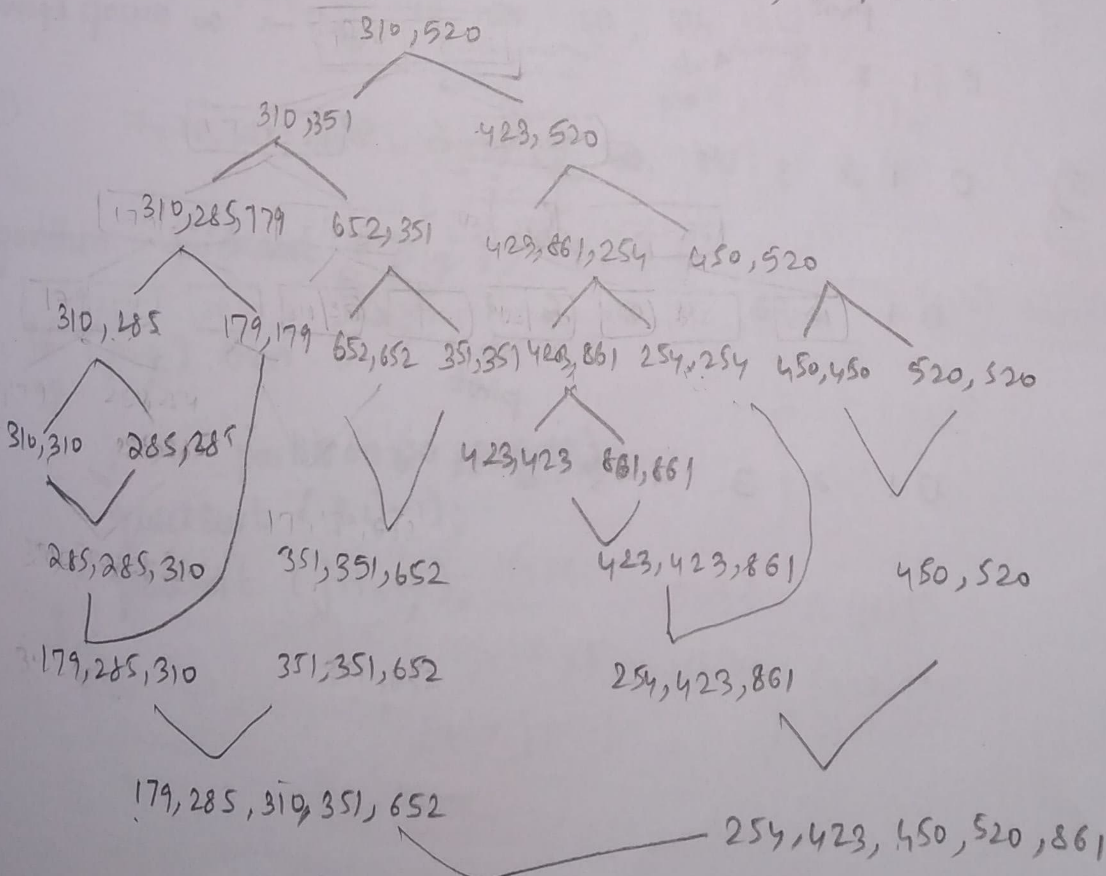
2) 1,8.



1/10/21

Quick Sort

310, 285, 179, 652, 351, 423, 861, 254, 450, 520



179, 254, 285, 310, 351, 423, 450, 520, 652, 861

179, 351, 861

Quick Sort

Ex:-

1) 5 1 3 9 7 0 4 2 6 8

pivot = 5, i = 9, j = 2

$a[i] > pivot$
$a[j] < pivot$

1 > 5 ×
3 > 5 ×
9 > 5 ✓ i = 9
8 < 5 ×
6 < 5 ×
2 < 5 ✓ j = 2
Swap (i, j)

5 1 3 2 7 0 4 9 6 8

2 > 5 ×
7 > 5 ✓

5 1 3 2 4 0 7 9 6 8

if $i < j \rightarrow swap(i,j)$
else $j < i \rightarrow swap(p, j)$

0 1 3 2 4 $\boxed{5}$ 7 9 6 8

$\underbrace{0\ 1\ 3\ 2\ 4}_{<}$ 5 (pivot) $\underbrace{7\ 9\ 6\ 8}_{>}$

2) 0 1 3 2 4 5 7 9 6 8

pivot = 0, $\underbrace{1\ 3\ 2\ 4}_{>}$

$\boxed{\ }$ <

3) 0 1 3 2 4 5 7 9 6 8

0 <, pivot = 1, $\underbrace{3\ 2\ 4}_{>}$

4) 0 1 3 2 4

pivot = 3, j, i

i ⇒ 2 > 3 ×
4 > 3 ✓
i ⇒ 4

j ⇒ 4 < 3 ×
2 < 3 ✓
j ⇒ 2

j < i → so swap (pivot, j)

0 1 2 3 4

5) 0 1 2 3 4 5 7 9 6 8

pivot = 7, i, j

0 1 2 3 4 5 6 7 9 8

pivot = 7, i, j

0 1 2 3 4 5 6 7 8 9

2) i) 40, 20, 10, 80, 60, 50, 7, 30, 100

pivot i j swap

40, 20, 10, 30, 60, 50, 7, 80, 100

i j

40, 20, 10, 30, 7, 50, 60, 80, 100

p j i

swap

$\underbrace{7, 20, 10, 30,}_{<}$ $\boxed{40,}$ $\underbrace{50, 60, 80, 100}_{>}$

pivot

ii) 7 | $\underbrace{20, 10, 30}_{>}$

pivot

iii) 7, 20, 10, 30

p j i

7, 10, 20, 30 → left part is sorted

iv) 7, 10, 20, 30, $\underbrace{40,}_{<}$ 50, $\underbrace{60, 80, 100}_{>}$

pivot

v) 7, 10, 20, 30, 40, $\underbrace{50,}_{<}$ 60, $\underbrace{80, 100}_{>}$

pivot

vi) 7, 10, 20, 30, 40, $\underbrace{50, 60,}_{<}$ 80, $\underbrace{100}_{>}$

pivot

vii) 7, 10, 20, 30, 40, 50, 60, 80, 100

Algorithm quicksort (p, q)

{

if (p, q) then

{

j := partition (a, p, q+1);

quicksort (p, j-1);

quicksort (j+1, q);

}

}

REDMI NOTE 5 PRO
MI DUAL CAMERA

```
Algorithm partition(a, m, p)
{
    v := a[m]; i := m, j := p;
    repeat
    {
        repeat
            i := i+1;
        until (a[i] ≥ v);
        repeat
            j := j-1
        until (a[j] ≤ v);
        if (i<j) then interchange (a, i, j);
    }
    until (i ≥ j);
    a[m] := a[j];
    a[j] := v;
    return j;
}
Algorithm interchange (a, i, j) // for swapping
{
    p := a[i];
    a[i] := a[j];
    a[j] := p;
}
```

Quick Sort Analysis [Best Case]

$$T(n) = \begin{cases} a & \text{if } n=1 \\ cn + T(n/2) + T(n/2) & \text{otherwise} \end{cases}$$

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

$$T(n/2) = \frac{cn}{2} + T\left(\frac{n/2}{2}\right) + T\left(\frac{n/2}{2}\right)$$

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

$$\Rightarrow cn/2 + 2T(n/4)$$

$\Rightarrow$

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

$\Rightarrow 2T(n/2) + cn$

$T(n/2) \Rightarrow 2T(\frac{n/2}{2}) + cn/2$

$\Rightarrow 2T(n/4) + cn/2$

$n = 2^k \Rightarrow 2T(n^{k-1}/2) + c \cdot 2^k$

$T(n) = 2^k \cdot T(2^0) + k \cdot c \cdot 2^k$

$T(2^k) = 2^k \cdot T(1) + k \cdot c 2^k$

$T(n) = n \cdot T(1) + cn \log_2 n$

$T(n) \Rightarrow an + cn \log_2 n$

$\boxed{T(n) \approx n \log_2 n}$

$cn + T(n-1)$

$T(n-1) = c(n-1) + T(n-1-1)$

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

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

$= c(n-3) + T(n-4)$

worst Case Analysis

$T(n) = cn + T(a) + T(n-1)$ $\boxed{\because T(a) = 0}$

$= cn + T(n-1)$

$= cn + [c(n-1) + T(n-1-1)]$

$= cn + [c(n-1) + T(n-2)]$

$= cn + c(n-1) + c(n-2) + T(n-1)$

$= cn + c(n-1) + c(n-2) + c(n-3) + T(n-4) \ldots\ldots + c(1) + T(0)$

$\vdots$

$= c[n + (n-1) + (n-2) + \ldots + 1]$

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

$= c\left(\frac{n^2+n}{2}\right) \Rightarrow \frac{cn^2}{2} + c(n/2)$

$\boxed{T(n) \approx n^2}$

REDMI NOTE 5 PRO
MI DUAL CAMERA

5/10/21

Strassen's matrix multiplication [volker strassen's]

7 mul's
18 add's & sub's

matrix multiplication A, b

$$\boxed{C_{ij} \Rightarrow C_{ij} + A_{ik} * B_{kj}}$$

$$a = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad b = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

$$C_{11} = a_{11} b_{11} + a_{12} b_{21}$$

$$c = \begin{bmatrix} \underset{1}{a_{11} b_{11}} + \underset{2}{a_{12} b_{21}} & \underset{3}{a_{11} b_{12}} + \underset{4}{a_{12} b_{22}} \\ \underset{5}{a_{21} b_{11}} + \underset{6}{a_{22} b_{21}} & \underset{7}{a_{21} b_{12}} + \underset{8}{a_{22} b_{22}} \end{bmatrix}$$

no. of multiplies $\Rightarrow$ 8
no. of add's $\Rightarrow$ 4

$$2 \log_2^8 \Rightarrow 2 \log_1^3$$
$$2^3 \log_2 2 \Rightarrow \boxed{2^3}$$

Algorithm

i := 1 to n
j := 1 to n
k := 1 to n
$c_{ij} := 0$
$c_{ij} += A_{ik} + B_{kj}$

time Complexity

$O(n^3)$
$\downarrow$ size of matrix

$$T(N) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} c = aN^3 = O(N^3)$$

→ fastest algorithm [fewer mul's & more add's]

$P = (A_{11} + A_{22})(B_{11} + B_{22})$
$Q = (A_{21} + A_{22}) B_{11}$
$R = A_{11}(B_{12} - B_{22})$
$S = A_{22}(B_{21} - B_{11})$
$T = (A_{11} + A_{12}) B_{22}$
$U = (A_{21} - A_{11})(B_{11} + B_{12})$
$V = (A_{12} - A_{22})(B_{21} + B_{22})$

Using divide & Conquer

$$T(n) = \begin{cases} b \\ 8T(n/2) + cn^2 \end{cases}$$

$$\rightarrow T(n) = \begin{cases} b \\ 7T(n/2) + an^2 \end{cases}$$

strassen

$C_{11} = P + S - T + V$
$C_{12} = R + T$
$C_{21} = Q + S$
$C_{22} = P + R - Q + U$

Example

$a = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times b = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$

Compute matrix mul using strassen's matrix multiplication

$P = (A_{11} + A_{22})(B_{11} + B_{22})$

$= (1+4)(5+8)$

$= (5)(13) \Rightarrow 65$

$q = (A_{21} + A_{22}) B_{11} \Rightarrow (3+4)5 \Rightarrow 7(5) \Rightarrow 35$

$R = A_{11}(B_{12} - B_{22}) \Rightarrow 1(6-8) \Rightarrow -2$

$S = A_{22}(B_{21} - B_{11}) \Rightarrow 4(7-5) \Rightarrow 4(2) \Rightarrow 8$

$T = (A_{11} + A_{12}) B_{22} \Rightarrow (1+2)8 \Rightarrow 3(8) = 24$

$U = (A_{21} - A_{11})(B_{11} + B_{12}) \Rightarrow (3-1)(5+6) \Rightarrow 2(11) = 22$

$V = (A_{12} - A_{22})(B_{21} + B_{22}) \Rightarrow (2-4)(7+8) \Rightarrow (-2)(15)$

$= -30$

$C_{11} \Rightarrow P + S - T + V = 65 + 8 - 24 + (-30) \Rightarrow 19$

$C_{12} \Rightarrow R + T \Rightarrow -2 + 24 \Rightarrow 22$

$C_{21} \Rightarrow Q + S \Rightarrow 35 + 8 \Rightarrow 43$

$C_{22} \Rightarrow P + R - Q + U \Rightarrow 65 + (-2) - 35 + 22$

$\Rightarrow 65 - 2 - 35 + 22$

$\Rightarrow 65 - 37 + 22 \Rightarrow 87 - 37 \Rightarrow 50$

$C_{ij} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$

2) find the ~~matrix~~ strassen's matrix multiplication for 4*4 matrix

$A = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix} \quad B = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$

$a_{11} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad a_{12} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad B_{11} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad b_{12} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$a_{21} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad a_{22} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad B_{21} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b_{22} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$P \Rightarrow (A_{11}+A_{22})(B_{11}+B_{22})$

$\Rightarrow \left(\begin{bmatrix}1&1\\1&1\end{bmatrix}+\begin{bmatrix}1&1\\1&1\end{bmatrix}\right)\left(\begin{bmatrix}1&0\\0&1\end{bmatrix}+\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)$

$\Rightarrow \begin{bmatrix}2&2\\2&2\end{bmatrix}\begin{bmatrix}2&0\\0&2\end{bmatrix} \Rightarrow \begin{bmatrix}4&4\\4&4\end{bmatrix}$

$q \Rightarrow (A_{21}+A_{22})B_{11}$

$\Rightarrow \left(\begin{bmatrix}1&1\\1&1\end{bmatrix}+\begin{bmatrix}1&1\\1&1\end{bmatrix}\right)\begin{bmatrix}1&0\\0&1\end{bmatrix} \Rightarrow \begin{bmatrix}2&2\\2&2\end{bmatrix}\begin{bmatrix}1&0\\0&1\end{bmatrix} \Rightarrow \begin{bmatrix}2&2\\2&2\end{bmatrix}$

$r \Rightarrow A_{11}(B_{12}-B_{22})$

$\Rightarrow \begin{bmatrix}1&1\\1&1\end{bmatrix}\left(\begin{bmatrix}0&0\\0&0\end{bmatrix}-\begin{bmatrix}1&0\\0&1\end{bmatrix}\right) \Rightarrow \begin{bmatrix}1&1\\1&1\end{bmatrix}\begin{bmatrix}-1&0\\0&-1\end{bmatrix}$

$\Rightarrow \begin{bmatrix}-1+0 & 0+(-1)\\-1+0 & 0+(-1)\end{bmatrix} \Rightarrow \begin{bmatrix}-1&-1\\-1&-1\end{bmatrix}$

$s \Rightarrow A_{22}(B_{21}-B_{11})$

$\Rightarrow \begin{bmatrix}1&1\\1&1\end{bmatrix}\begin{pmatrix}0&0 & - & 1&0\\0&0 & & 0&1\end{pmatrix} \Rightarrow \begin{bmatrix}1&1\\1&1\end{bmatrix}\begin{bmatrix}-1&0\\0&-1\end{bmatrix} \Rightarrow \begin{bmatrix}-1&-1\\-1&-1\end{bmatrix}$

$A = \begin{bmatrix}2&2\\2&2\end{bmatrix}\begin{bmatrix}2&0\\0&2\end{bmatrix}$

$P \Rightarrow (A_{11}+A_{22})(B_{11}+B_{22}) \Rightarrow (2+2)(2+2) \Rightarrow 16$

$q = (A_{21}+A_{22})B_{11} \Rightarrow (2+2)(2) \Rightarrow 8$

$T \Rightarrow (A_{11}+A_{12})B_{22} \Rightarrow \begin{bmatrix}1&1 & + & 1&1\\1&1 & & 1&1\end{bmatrix}\begin{bmatrix}1&0\\0&1\end{bmatrix}$

$\Rightarrow \begin{bmatrix}2&2\\2&2\end{bmatrix}\begin{bmatrix}1&0\\0&1\end{bmatrix} \Rightarrow \begin{bmatrix}2&2\\2&2\end{bmatrix}$

$U \Rightarrow (A_{21}-A_{11})(B_{11}+B_{12})$

$\begin{bmatrix}1&1 & - & 1&1\\1&1 & & 1&1\end{bmatrix}\begin{bmatrix}1&0 & + & 0&0\\0&1 & & 0&0\end{bmatrix} \Rightarrow \begin{bmatrix}0&0\\0&0\end{bmatrix}\begin{bmatrix}1&0\\0&1\end{bmatrix}$

$\Rightarrow \begin{bmatrix}0&0\\0&0\end{bmatrix}$

$V \Rightarrow (A_{12}-A_{22})(B_{21}+B_{22})$

$\begin{bmatrix}1&1 & - & 1&1\\1&1 & & 1&1\end{bmatrix}\begin{bmatrix}0&0 & + & 1&0\\0&0 & & 0&1\end{bmatrix} \Rightarrow \begin{bmatrix}0&0\\0&0\end{bmatrix}$

REDMI NOTE 5 PRO
MI DUAL CAMERA

$r \rightarrow A_{11}(B_{12} - B_{22})$

$\rightarrow 2(0-2) \Rightarrow 2(-2) \Rightarrow -4$

$s \rightarrow A_{22}(B_{21} - B_{11})$.

$2(0-2) \Rightarrow 2(-2) \Rightarrow -4$

$t = (A_{11} + A_{12}) B_{22} \Rightarrow (2+2)2 \Rightarrow 8$

$u \Rightarrow (A_{21} - A_{11})(B_{11} + B_{12}) \Rightarrow (2-2)(2+0)$

$= 0(2) \Rightarrow 0$

$v = (A_{12} - A_{22})(B_{21} + B_{22}) \Rightarrow (2-2)(0+2) \Rightarrow 0$

$P \Rightarrow \quad C_{11} \Rightarrow P+S-T+V \Rightarrow 16 + (-4) - 8 + 0 \Rightarrow 4$

$C_{12} \Rightarrow R+T \Rightarrow -4+8 \Rightarrow 4$

$C_{21} \Rightarrow Q+S \Rightarrow 8-4 \Rightarrow 4$

$C_{22} \Rightarrow P+R-Q+U \Rightarrow 16+4-8+0$

$= 4$

$P \Rightarrow \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$

$C_{11} \Rightarrow P+S-T+V$

$\Rightarrow \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix} + \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$\begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$C_{12} \Rightarrow R+T \Rightarrow \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} + \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$C_{21} \Rightarrow Q+S \Rightarrow \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} + \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$C_{22} \Rightarrow P+R-Q+U \Rightarrow \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix} + \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$= \begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$

Example ⇒ $\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$

$$A = \begin{bmatrix}1&1&1&0\\1&1&1&0\\1&1&1&0\\0&0&0&0\end{bmatrix} \qquad B = \begin{bmatrix}1&1&1&0\\1&1&1&0\\1&1&1&0\\0&0&0&0\end{bmatrix}$$

$A_{11} = \begin{bmatrix}1&1\\1&1\end{bmatrix}$ $A_{12} = \begin{bmatrix}1&0\\1&0\end{bmatrix}$ $B_{11} = \begin{bmatrix}1&1\\1&1\end{bmatrix}$ $B_{12} = \begin{bmatrix}1&0\\1&0\end{bmatrix}$

$A_{21} = \begin{bmatrix}1&1\\0&0\end{bmatrix}$ $A_{22} = \begin{bmatrix}1&0\\0&0\end{bmatrix}$ $B_{21} = \begin{bmatrix}1&1\\0&0\end{bmatrix}$ $B_{22} = \begin{bmatrix}1&0\\0&0\end{bmatrix}$

$$p = (A_{11}+A_{22})(B_{11}+B_{22})$$
$$= \left(\begin{bmatrix}1&1\\1&1\end{bmatrix}+\begin{bmatrix}1&0\\0&0\end{bmatrix}\right)\left(\begin{bmatrix}1&1\\1&1\end{bmatrix}+\begin{bmatrix}1&0\\0&0\end{bmatrix}\right)$$
$$= \begin{pmatrix}2&1\\1&1\end{pmatrix}\begin{pmatrix}2&1\\1&1\end{pmatrix} \Rightarrow \begin{bmatrix}2\times2+1\times1 & 2\times1+1\times1\\1\times2+1\times1 & 1\times1+1\times1\end{bmatrix}$$
$$= \begin{bmatrix}5&3\\3&2\end{bmatrix}$$

$$q \Rightarrow (A_{21}+A_{22})B_{11}$$
$$\Rightarrow \left(\begin{matrix}1&1\\0&0\end{matrix}+\begin{matrix}1&0\\0&0\end{matrix}\right)\begin{bmatrix}1&1\\1&1\end{bmatrix} \Rightarrow \begin{bmatrix}2&1\\0&0\end{bmatrix}\begin{bmatrix}1&1\\1&1\end{bmatrix}$$
$$\Rightarrow \begin{pmatrix}2\times1+1\times1 & 2\times1+1\times1\\0\times1+0\times1 & 0\times1+0\times1\end{pmatrix} \Rightarrow \begin{bmatrix}3&3\\0&0\end{bmatrix}$$

$$r \Rightarrow A_{11}(B_{12}-B_{22})$$
$$\begin{bmatrix}1&1\\1&1\end{bmatrix}\left[\begin{matrix}1&0\\1&0\end{matrix}-\begin{matrix}1&0\\0&0\end{matrix}\right] \Rightarrow \begin{bmatrix}1&1\\1&1\end{bmatrix}\begin{bmatrix}0&0\\1&0\end{bmatrix}$$
$$= \begin{bmatrix}1\times0+1\times1 & 1\times0+1\times0\\1\times0+1\times1 & 1\times0+1\times0\end{bmatrix} = \begin{bmatrix}1&0\\1&0\end{bmatrix}$$

$$s \Rightarrow A_{22}(B_{21}-B_{11})$$
$$\begin{bmatrix}1&0\\0&0\end{bmatrix}\left(\begin{matrix}1&1\\0&0\end{matrix}-\begin{matrix}1&1\\1&1\end{matrix}\right) \Rightarrow \begin{bmatrix}1&0\\0&0\end{bmatrix}\begin{bmatrix}0&0\\-1&-1\end{bmatrix} =$$
$$\begin{bmatrix}1\times0+0\times-1 & 1\times0+0\times-1\\0\times0+0\times-1 & 0\times-1+0\times-1\end{bmatrix} = \begin{bmatrix}0&0\\0&0\end{bmatrix}$$

$T = (A_{11} + A_{12}) B_{22} \Rightarrow \left[\begin{smallmatrix}1 & 1\\ 1 & 1\end{smallmatrix}\right] + \left[\begin{smallmatrix}1 & 0\\ 1 & 0\end{smallmatrix}\right] \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right]$

$= \begin{bmatrix}2 & 1\\ 2 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} = \begin{bmatrix}2\times1 + 1\times0 & 2\times0 + 1\times0\\ 2\times1 + 1\times0 & 2\times0 + 1\times0\end{bmatrix} = \begin{bmatrix}2 & 0\\ 2 & 0\end{bmatrix}$

$U = (A_{21} - A_{11})(B_{11} + B_{12}) \Rightarrow \left[\begin{smallmatrix}1 & 1\\ 0 & 0\end{smallmatrix} - \begin{smallmatrix}1 & 1\\ 1 & 1\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & 1\\ 1 & 1\end{smallmatrix} + \begin{smallmatrix}1 & 0\\ 1 & 0\end{smallmatrix}\right]$

$= \begin{bmatrix}0 & 0\\ -1 & -1\end{bmatrix}\begin{bmatrix}2 & 1\\ 2 & 1\end{bmatrix} \Rightarrow \begin{bmatrix}0\times2 + 0\times2 & 0\times1 + 0\times1\\ -1\times2 + -1\times2 & -1\times1 + -1\times1\end{bmatrix} = \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$

$V = [A_{12} - A_{22}][B_{21} + B_{22}] = \left[\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix} - \begin{smallmatrix}1 & 0\\ 1 & 0\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & 1\\ 0 & 0\end{smallmatrix} + \begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right]$

$= \begin{bmatrix}0 & 0\\ -1 & 0\end{bmatrix}\begin{bmatrix}2 & 1\\ 0 & 0\end{bmatrix} \Rightarrow \begin{bmatrix}0\times2 + 0\times0 & 0\times1 + 0\times0\\ -1\times2 + 0\times0 & -1\times1 + 0\times0\end{bmatrix} = \begin{bmatrix}0 & 0\\ -2 & -1\end{bmatrix}$

$C_{11} \Rightarrow P + S - T + V$

Strassen's Matrix multiplication

Analysis

$T(n) = 7 \cdot T(n/2) + an^2$

$= 7^2 \cdot T\left(\frac{n}{2^2}\right) + \left(\frac{7}{4}\right)an^2 + an^2$

$= 7^3 \cdot T\left(\frac{n}{2^3}\right) + \left(\frac{7}{4}\right)^2 \cdot an^2 + \left(\frac{7}{4}\right)an^2 + an^2$

Assume $n = 2^k$ for some integers k

$\Rightarrow 7^{k-1} \cdot 7\left(\frac{n}{2^{k-1}}\right) + an^2\left[\left(\frac{7}{4}\right)^{k-2} + \cdots + 1\right]$

$\Rightarrow 7^{k-1} \cdot b + an^2 \left[\frac{\left(\frac{7}{4}\right)^{k-1} - 1}{\frac{7}{4} - 1}\right]$

$\leq b \cdot 7^k + c \cdot n^2 \cdot \left(\frac{7}{4}\right)^k$

$$= b \cdot 7^{\log n} + cn^2 \cdot \left(\frac{7}{4}\right)^{\log n}$$

$$\Rightarrow b \cdot 7^{\log n} + cn^2 (n)^{\log \frac{7}{4}}$$

$$= b \cdot n^{\log 7} + cn^{\log 7}$$

$$= (b+c)\, n^{\log 7}$$

$$= O(n^{\log 7})$$

$$= O(n^{2.81})$$

Unit-2

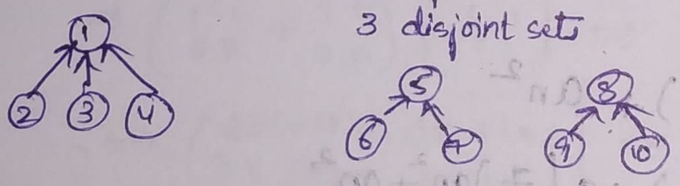
disjoint Sets

$S_1 = \{1, 2, 3\}$, $S_2 = \{a, b, c\}$

$S_1 \cup S_2 = \{1, 2, 3, a, b, c\}$

$S_1 \cap S_2 = \{\phi\}$

Ex: $n = 10$, elements can be partioned into 3 disjoint sets

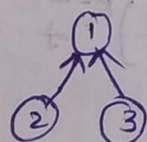
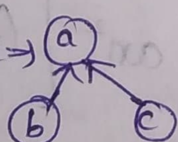
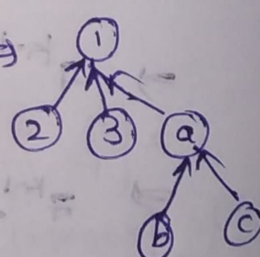


Operations on sets

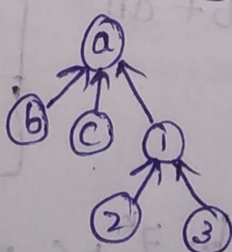
1) Disjoint set Union :-

Ex:- $S_1 = \{1, 2, 3\}$, $S_2 = \{a, b, c\}$

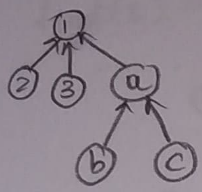
$S_1 \cup S_2 = \{1, 2, 3, a, b, c\}$

$S_1 \Rightarrow$  $S_2 \Rightarrow$  $S_1 \cup S_2 \Rightarrow$ 

(or)



2) find(i) :

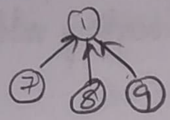


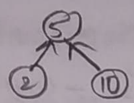
find(1) ⇒ -1

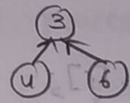
find(b) ⇒ a

find(a) = -1

Ex:







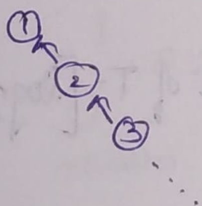
i	1	2	3	4	5	6	7	8	9	10
p	-1	5	-1	3	-1	3	1	1	1	5

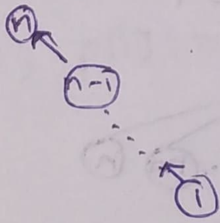
Algorithm SimpleUnion (i, j)
{
  p[i] := j;
}

Algorithm Simplefind (i)
{
  while (p[i] ≥ 0) do
    i := p[i];
  return i;
}

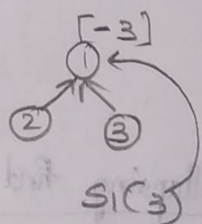
degenerate tree

Ex: 1 2 3 4 5 6

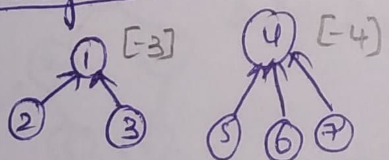




→ weighted Union ( )
→ collapsing find (i)
→ count ( )



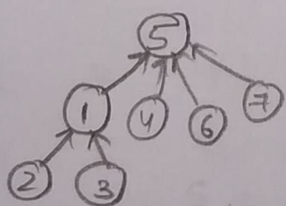
→ weighted Union :



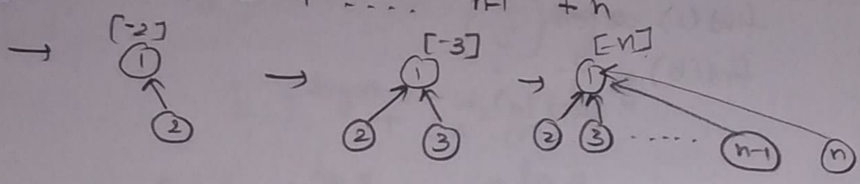
→ $S_1(3)$ $S_2(4)$

represent no. of elements.

$S_1 \cup S_2$ ⇒



Ex: $\begin{array}{cccccc} -1 & -1 & -1 & -1 & & \\ 1 & 2 & 3 & 4 & \cdots & n-1 \quad +n \end{array}$



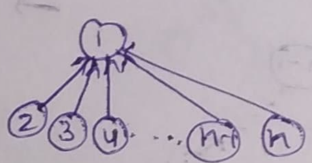
Algorithm weightedUnion (i, j)
{

$p[i] = -\text{Count}[i]$
$p[j] = -\text{Count}[j]$ } weighing rule

temp := p[i] + p[j];
if (p[i] > p[j]) then
{
p[i] := j // few nodes
p[j] = temp;
}
else // few or equal nodes
{
p[j] := i;
p[i] = temp;
}
}

18/10/21

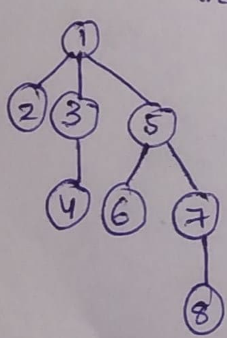
Lemma 2.3 : Tree - T



then,
height of $T = \left[\log_2^m\right] + 1$

Collapsing find ():

Ex: Consider tree,



Simple find ( ):

find(8) = 7 (not the root node)
↳ find (7) = 5
↳ find(5) = 1

So here we need 3 Operations.

→ if the question is asked to process 8 finds, then

8×3 = 24 moves [∵ each find requires 3 moves]

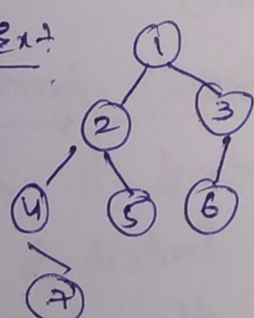
∴ Total moves = 24

→ collapsing find( ) will checking each node's parent only for the first time, for multiple processing of find it collapses all the iteration and gives the result directly

Algorithm for Collapsing find :

```
Algorithm collapsingfind (i)
{
    r := i;
    while (p(r)>0) do r := p(r);
    while (i≠r) do
    {
        s := p[i]; p[i] := r; i = s;
    }
    return r;
}
```

Ex: ~~find~~ find (7)



parent nodes of each nodes :

:	1	2	3	4	5	6	7
p[i]	-1	1	1	2	2	3	4

$i = 7$

$r := i \Rightarrow r = 7$

while $p(r) > 0$,

(I1) $p(7) > 0$

$4 > 0$

$r = 4$

(I2) $P[4] > 0$

$2 > 0$

$r = 2$

(I3) $P[2] > 0$

$1 > 0$

$r = 1$

(I4) $P[1] > 0$

$-1 > 0$ ✗

$r : p[r] \Rightarrow r = p[r] = 1$

while $(i \neq r)$

(I1) $i = 7, r = 1 \Rightarrow 7 \neq 1$

$s := p[i]$

$p[i] := r$

$i := s$

$s = 4$

$p[i] = 4$

$i = 4$

(I2) $i = 4, r = 1 \Rightarrow 4 \neq 1$

$s := 2$

$p[4] := 1$

$i = 2$

(I3) $i = 2, r = 1 \Rightarrow 2 \neq 1$

$s := 1$

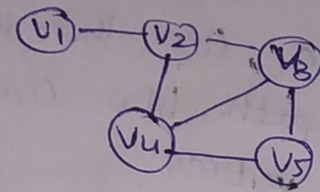
$p[2] = 1$

$i = 1$

(I4) $i = 1, r = 1 \Rightarrow 1 \neq 1$ ✗

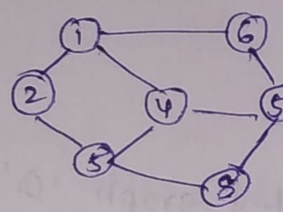
Connected graph :

A graph is Connected if there exists path b/w every pair of distinct nodes, otherwise, it is disconnected

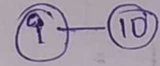
Eg :- 

Connected.

(2)



G_1



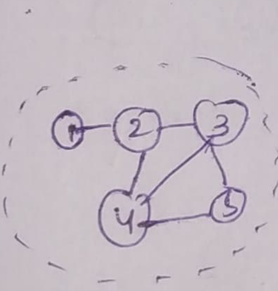
G_2

disconnected graphs

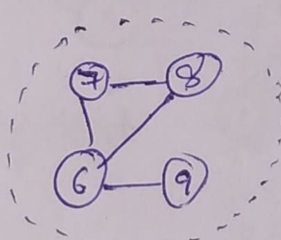
Connected Component :-

→ if a graph is disconnected, it can be partitioned into a no. of graphs such that each of them is Connected. Each such graph is called a "Connected Component".

Eg :-



Connected component 1



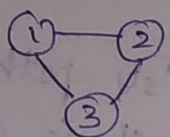
Connected component 2

Strongly Connected graph :-

→ A directed graph h is s.c.g. iff and every pair of of distinct vertices u and v in V(G), there is directed path from U to V and also from V to u.

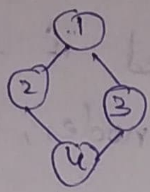
→ A strongly Connected Component is a maximal subgraph that is strongly Connected.

Eg :



Isolated graph

S.C.G



Connected graph.

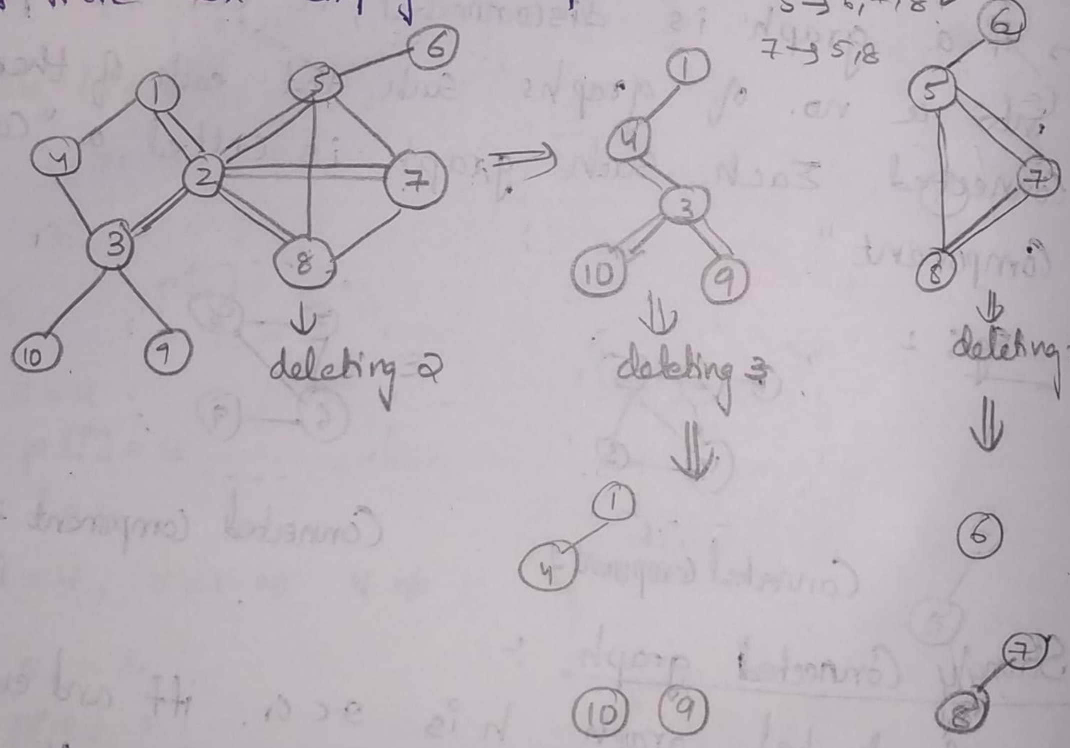
not SCG ∵, coz there is no direct path b/w (1,4) & (2,3) vertices.

20/10/21

Articulation point :

'A' vertex 'v' in a Connected graph 'G' is an articulation point iff the deletion of vertex 'v', ∴ with all edges incident to 'v' disconnects the graph into 2 or more non empty components.

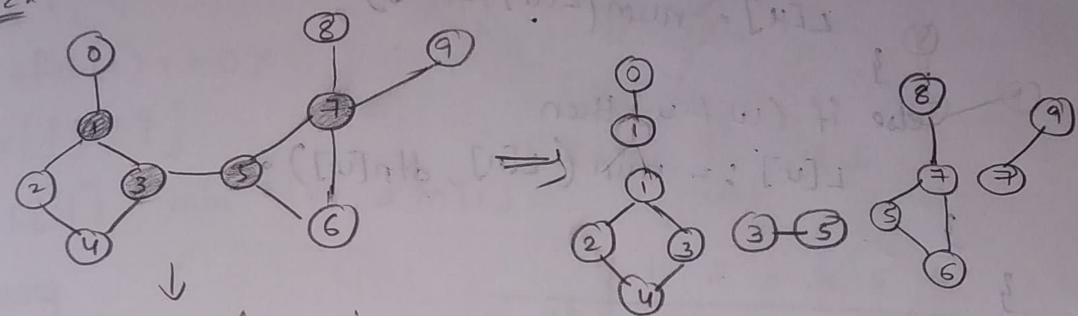
5 → 6, 7, 8 ✓
7 → 5, 8



Biconnected graph : A B.G is a Connected graph. If it contains an articulation points.

Biconnected Component :- biconnected → B. Graph ~~and~~ of a Connected graph G is a maximal subgraph H of G.

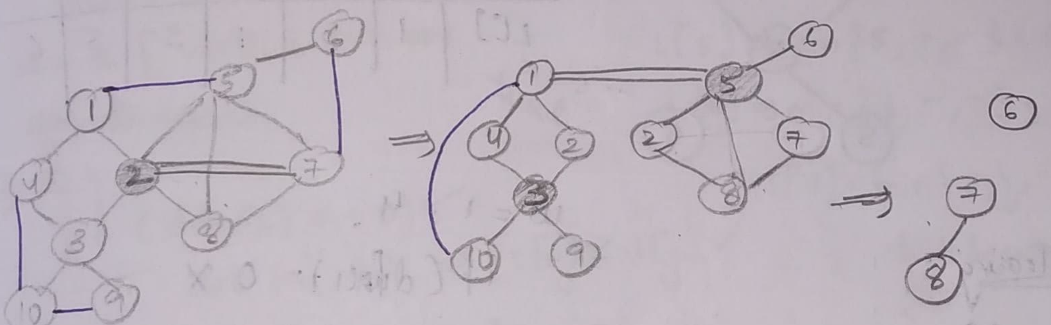
Ex :-

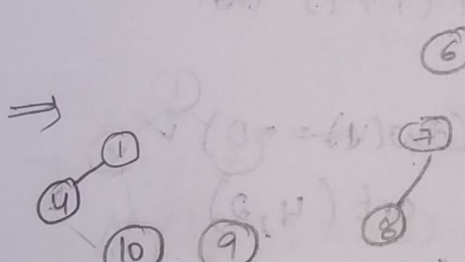


Connected graph
1, 3, 5, 7 → articulation point

Biconnected Components

→ Construction of biconnected Component





In BFS, edges that are not included - Cross edges

In DFS → edges that are not included → ~~spanning trees~~ back edges.

22/10/21

Algorithm art (u, v)
{

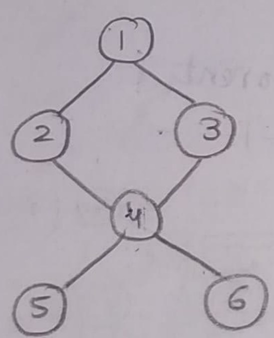
u → start vertex
v → parent vertex.

dfn[u] := num ;
L[u] := num ;
num := num+1 ;
for each vertex w adjacent from u do
{

$L[w] \geq dfn[u]$

EDMI NOTE 5 PRO
DUAL CAMERA

if (dfn(w) == 0) then
{
art(w, u);
L[u] = num(L(u), L(w]);
}
else if (w ≠ u) then
L[u] := min(L[n], dfn[w]);
}
}



Depth first number

	1	2	3	4	5	6
dfn	1	2	4	3	5	6
L[]	1	2	1	3	5	6

lowest

Tracing :

dfn = 0
num = 1
num = 6

→ Art(1, 2)
dfn[1] = 1
l[1] = 1
num := 2

→ art(2, 3)
w = 2
{
dfn[w] = 0
dfn[2] = 0
art(2, 1)
dfn[2] = 2
L[2] = 2
num = 3

w = 1, 4.
if (dfn(1)) = 0 ×
else if (1 ≠ 1) then.

W = 4.
if (dw(u) == 0) ✓
art(4, 2)
→ dfn(4] := 3.
L[u] := 3
num = 4

w = 2, 3, 5, 6
dfn(2) == 0 ×
2 ≠ 2 ×

w = 3
dfn(3) = 0 ✓
art(3, 4)

$dfn(3) = 4$

$L(3) = 4$

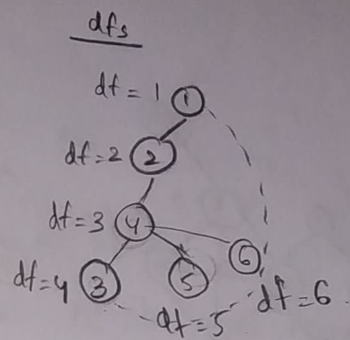
num = 5

$w = 1, 4.$

$dfn(1) == 0$ ×

else $[1 \neq 4]$

$L[4] = \min L[4], dfn[1]$



	1	2	3	4	5	6
df	1	2	4	3	5	6
L	1	1	1	1	5	6

Tracing

dfn = 0

num = 0

n = 6

→ $dfn[1] := 1$

$L[1] := 1$

num = 2

u = 1 [vertices that are

~~adj~~ w = ~~2~~, 3

1) w = 2

if $(dfn(2) == 0)$ ✓

art $(\overset{u}{2}, \overset{v}{1}) \Rightarrow \underset{w}{L[2]} \geq \underset{v}{L[1]}$ ✓

↳ $dfn[2] := 2$

$L[2] := 2$

num := 3

u = 2

→ w = 1, 4.

if $(dfn(1) == 0)$ ×

art $(1,2) \Rightarrow L[1] \geq L($

$L[3] \Rightarrow \min\{4, -, 1\} = 1$

$L[5] \Rightarrow \min\{5, -, -\} = 5$

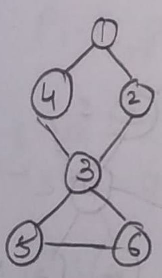
$L[6] \Rightarrow \min\{6, -, -\} = 6$

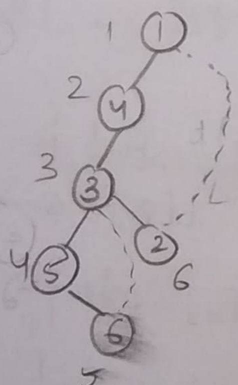
$L[4] = \min\{3, 1, -\} = 1$

childedges ⇒ 3 = 1
5 = 5
6 = 6

$L[2] \Rightarrow \min\{2, 1, -\} = 1$

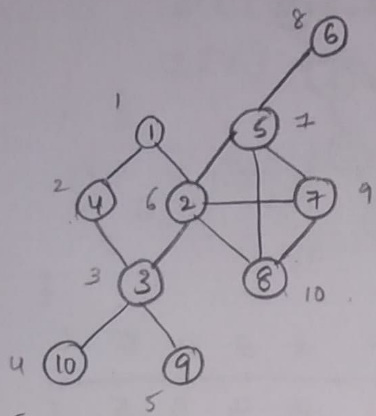
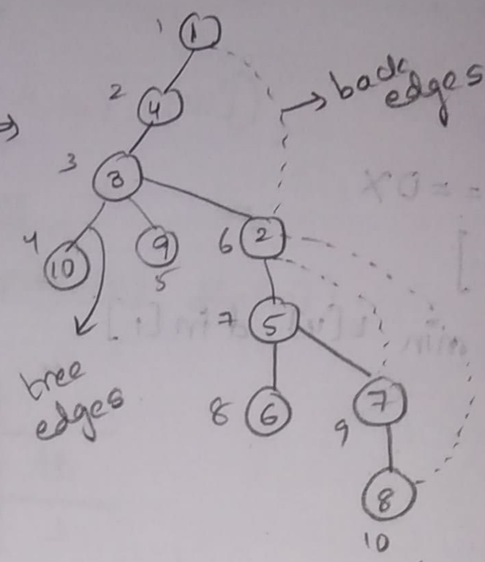
$L[1] \Rightarrow \min\{1,$

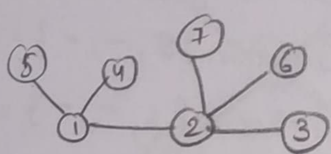




1	2	3	4	5	6
1	6	3	2	4	5
1	1	1	1	3	3

dfs


$\Rightarrow$




adjacent vertices

bfs : 1, 2, 4, 5, 7, 3, 6

dfs : 1, 2, 3, 6, 7, 4, 5

	1	2	3	4	5	6	7	8	9	10
dfn	1	6	3	2	7	8	9	10	5	4
L	1	1	1	1	6	8	6	6	5	4

$L(u) = \min\{ dfn[u], \min\{ L(w) / w \text{ is a child of } u \},$
$\min\{ dfn[w] / (u,w) \text{ is a back edge} \} \}$

$L[10] = \min\{4, -, -\} = 4$

$L[9] = \min\{5, -, -\} = 5$

$L[6] = \min\{8, -, -\} = 8$

$L[8] = \min\{10, -, 6\} = 6 \rightarrow dfn[2] = 6$

$L[7] = \min\{9, 6, 6\} = 6 \rightarrow$ d $L[7]$ descendent is 8

$L[8] = 6$

$L[5] = \min\{7, 6, -\} = 6$

$5 \rightarrow 6, 7$

$L[6] = 8, L[7] = 6$ ✓

$L[4] = \min\{2, 1, -\}$
$= 1$

$L[2] = \min\{6, 6, 1\} = 1$

$L[1] = \min\{1, 1, 1\} = 1$

$L[3] = \min\{3, 1, -\} = 1$

$3 \rightarrow 10, 9, 2$
$\downarrow\ \downarrow\ \downarrow$
$4\ \ 5\ \ 1$

26/10/21

Greedy Method

i/p
↓ → satisfies
Constraints
↓ → called
feasible solution
↓ ⟶ Objective fn (either minimum or maximum)
optimal solution.

→ **Control abstraction of greedy method:**

```
Algorithm greedy (a,n)
{
  solution := φ;
  for i:=1 to n do
  {
     x:= select (a);
     if (feasible(solution, x) then
           solution = Union(solution, x);
  }
  return solution;
}
```

Knap Sack problem ✓ imp **

bag
↓
x_i - object → 1, 0, 1/2, 2/3, 1/4 ...
w_i - weight → kept in a knapsack or bag then profit is earned.
p_i - profit. → $p_i x_i$

→ $w_i > 0$
→ $p_i > 0$ then $p_i x_i$

$w_i x_i < M$ → then feasible solution
↓
optimal solution.

EDMI NOTE 5 PRO

problem

1) $P_i = 4$, $W_i = 4$ → $1/2 = 2$
$P_j = 3$, $W_j = 2$ → $1 = 3$
5

$n = 3$, $m = 20$.
↓ no. of objects, ↓ mass

$(p_1, p_2, p_3) = (25, 24, 15)$
$(w_1, w_2, w_3) = (18, 15, 10)$

The four feasible solutions are:

1) $(1, \frac{2}{15}, 0) \Rightarrow 1 \times 25 + \frac{2}{15} \times 24 + 0 \times 15 \Rightarrow 28.2$
2) $(0, 1, 1/2) \Rightarrow 0 \times 25 + 1 \times 24 + \frac{1}{2} \times 15 \Rightarrow 31.5$
3) $(1, 0, 1/5) \Rightarrow 1 \times 25 + 0 \times 24 + \frac{1}{5} \times \overset{3}{\cancel{15}} \Rightarrow 28$

$\boxed{P_i/W_i}$

Case 1: Increasing order

weight → 10, 15, 18
increorder

M = 20

object 3 2 1
profit 15, 24, 25

Remaining capacity	Object selected	weight	fraction of Object selected
initial = 20	3	10	1 ⟹
20 − 10 = 10	2	15	$\frac{10}{15}$th ⟹ 2/3
0	–	–	

Solution vector (x_1, x_2, x_3) $(0, 2/3, 1)$
profit earned $= 1 * 25 + \frac{2}{3} * 24 + 0 \times 15 = 31$

Case 2: decreasing order

weight → 18, 15, 10
profit → 25, 24, 15
object → 1 2 3

case 1: increasing ↑ weights
case 2: decreasing ↓ profits
case 3: $\frac{P_i}{W_i}$ → decreasing order of answer (ratio)

$m = 20$

Remaining-Capacity	Object selected	weight	
20	1	18	1
$20-18=2$	2	15	$2/15 \Rightarrow$
0	–	–	–

solution vector (x_1, x_2, x_3) $(1, 2/15, 0)$

profit earned $= 1\times 25 + \frac{2}{15}\times 24 + 0\times 15 \Rightarrow 28.2$

Case 3 : decreasing order of $\frac{P_i}{W_i}$

$$\frac{P_i}{W_i} = \left(\frac{P_1}{W_1}, \frac{P_2}{W_2}, \frac{P_3}{W_3}\right)$$

$$= \left(\frac{25}{18}, \frac{24}{15}, \frac{5}{10}\right)$$

$$= (\underset{1}{1.3}, \underset{2}{1.6}, \underset{3}{1.5})$$

decreasing order $(\underset{2}{1.6}, \underset{3}{1.5}, \underset{1}{1.3})$

R.C.	O.S.	W	f
20	2	~~18~~ 15	1
$20-15=5$	3	10	$5/10 \Rightarrow$
0	–	–	

solution vector $\Rightarrow (x_1, x_2, x_3) = (1, 5/10, 0)$

profit earned $\Rightarrow 1\times 25 + \frac{5}{10}\times 24 + 0\times 15 \Rightarrow 31.5$

2) Let $n = 4$, $m = 40$, $p_i\ (20, 30, 40, 5)$ $\Rightarrow \frac{20}{20}, \frac{30}{5}, \frac{40}{25}, \frac{5}{10}$

$w_i\ (\underset{1}{20}, \underset{2}{15}, \underset{3}{25}, \underset{4}{10})$

Case 1 : ~~increasing~~ decreasing order $\Rightarrow \underset{3}{25}, \underset{1}{20}, \underset{2}{15}, \underset{4}{10}$ $\Rightarrow (1, 6, 1.6, 0.5)$

profit $\Rightarrow 40, 20, 30, 5$

R.C	O.S.	W.	f
40	3	25	1
40-25 ⇒ 15	1	20	$\frac{15}{30}$ $\frac{3}{4}$
0	-	-	-
0	-	-	-

solution vector ⇒ $(x_1, x_2, x_3, x_4) = (1, \frac{3}{4}, 0, 0)$

profit earned = $1 \times 40 + \frac{3}{\cancel{4}_1} \times \overset{5}{\cancel{20}} + 0 \times 30 + 0 \times 5$

55

⇒ 65//

Case 2 : ~~decreasing order~~ increasing order of weights

w_i ⇒ (10, 15, 20, 25)
4 2 1 3

p_i ⇒ (5, 30, 20, 40).

RC	O.S	W	f
40	4	10	1
40 - 10 = 30	2	15	1
30 - 15 = 15	1	20..	$\frac{15}{20}$ ⇒ $3/4$
0	-	-	

Solution vector ⇒ $(1, 1, 3/4, 0)$

profit earned ⇒ $1 \times 5 + 30 \times 1 + \frac{3}{\cancel{4}} \times \overset{5}{\cancel{20}} + 0 \times 40$

⇒ ~~50~~ 5 + 30 + 15

⇒ 50

Case 3 : decreasing order of $\frac{P_i}{W_i}$

20, 30, 40, 5

20, 15, 25, 10

$$\frac{P_i}{W_i} = \left(\frac{P_1}{W_1}, \frac{P_2}{W_2}, \frac{P_3}{W_3}, \frac{P_4}{W_4}\right)$$

$$= \left(\frac{40}{25}, \frac{30}{20}, \frac{20}{15}, \frac{5}{10}\right)$$

$$\Rightarrow \left(\frac{20}{20}, \frac{30}{15}, \frac{40}{25}, \frac{5}{10}\right)$$

$C_1 = 50$

$\Rightarrow (1, 2, 1.6, 0.5)$

$C_2 = 70$

$C_3 = 70$

decreasing order $\Rightarrow$ (2; 1.6; 1; 0.5)
② ③ ① ④

Optimal = 70 (C-3)

RC	OS	W	F
40	2	15	1
40 - 15 = 25	3	25	1
25 - 25 = 0	4	—	—
0	—	—	—

Soln vector = $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$

Profit earned = $1 \times 30 + 1 \times 40 + 0 + 0 = 70$

case 2 : decreasing of profits :

P (20, 30, 40, 5) w (20, 15, 25, 10)
③ ② ① ④

RC	OS	W	F
40	3	25	1
40 - 25 = 15	2	15	1
15 - 15 = 0	—	—	—
—	—	—	—

Soln vector = $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$

Profit earned = $1 \times 40 + 1 \times 30 + 0 + 0 = 70$

REDMI NOTE 5 PRO
MI DUAL CAMERA

Job Sequence with deadlines

optimal solution ⇒ maximum value of feasible soln

J_i = Job

d_i = deadline

P_i = profit

Ex: $n = 4$ $(p_1, p_2, p_3, p_4) = (100, 10, 15, 27)$,

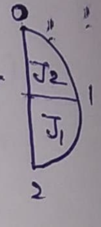
$(d_1, d_2, d_3, d_4) = (2, 1, 2, 1)$

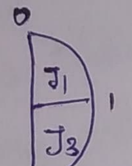
$(J_1, J_2, J_3, J_4) = (1, 2, 3, 4)$

i) (J_2, J_1)

(100, 10)

Profit = 110



ii) 

⇒ 100, 15 ⇒ 115 //

feasible sol	processing seq	profit value
iii) (J_1, J_4) ⇒	(4, 1) ⇒	(27+100) = 127 ✓ (optimal sol)
iv) (2, 3) ⇒	(2, 3)	= 25
v) (3, 4) ⇒	(4, 3) ⇒	15 + 27 = 42
vi) (1) ⇒	(1) ⇒	100
vii) (2) ⇒	(2) ⇒	10
viii) (3) ⇒	(3) ⇒	15
ix) (4) ⇒	(4) ⇒	27 //

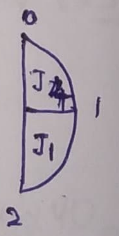
2) $n = 4$, $(p_1, p_2, p_3, p_4) = (100, 10, 15, 27)$ (1, 2, 3, 4)

$(d_1, d_2, d_3, d_4) \Rightarrow (2, 1, 2, 1)$ Using greedy method

Sol

decreasing P_i ⇒ 100, 27, 15, 10 (1, 4, 3, 2)

d_i ⇒ 2, 1, 2, 1



$(J_4, J_1) \Rightarrow (100 + 27)$ (27+100)

= 127 //

J = {1} → feasible seq ⇒ (4, 1)

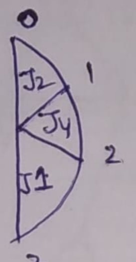
J = {1, 4} → feasible

J = {1, 3, 4} → not feasible.

Ex:2

Job	1 ✓	2 ✓	3	4	5	6	7
deadline	3	1	1	3	1	3	2
profit	40	35	30	25	20	15	10

i)



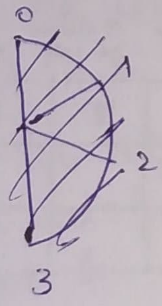
J3 ×
J5 ×
J6 ×
J7 ×

$J_4 \rightarrow$ dead $\rightarrow$ 3
0-2 ✓

processing seq $\Rightarrow \{J_2, J_4, J_1\} \Rightarrow \{35, 25, 40\}$

$P_i = 100$ //

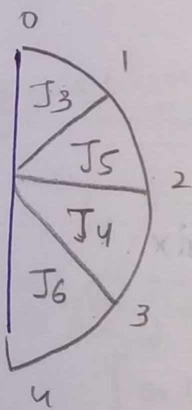
ii)



Job	1	2	3	4	5	6
d	2	1	2	3	2	4
p	20	25	30	40	35	20

decreasing order :-

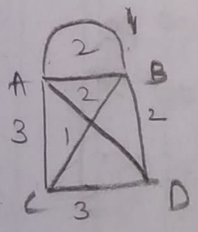
P_i	40	35	30	25	20	20 ✓
J_i	4 ✓	5 ✓	3 ✓	2	1	6
d_i	3	2	2	1	2	4



3, 5, 4, 6

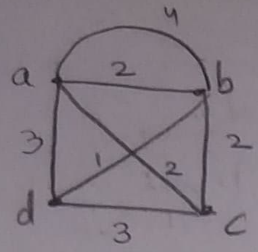
** Minimum Cost Spanning Tree

n-1 $\Rightarrow$ edges.
n $\Rightarrow$ no. of vertices

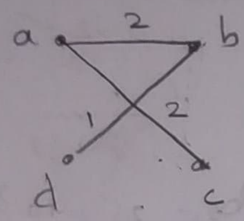


2 methods

1) Prism's algorithm
2) Krushal's "



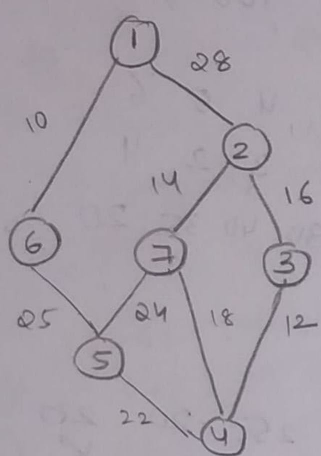
✓	✓	×	✓			
1	2	2	2	3	3	4
bd	ab	bc	ac	ad	dc	ab



$1+2+2 \Rightarrow 5$

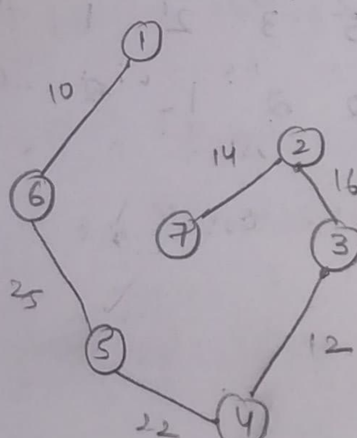
$n-1 \Rightarrow$ edges.

2)



$7-1 \Rightarrow 6$ edges

✓	✓	✓	✓	× closed loop	✓	×	✓	×
10	12	14	16	18	22	24	25	28
1-6								



$10+14+16+12+22+25$

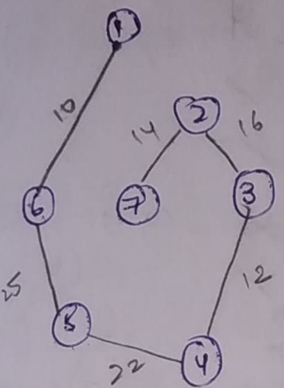
adjaceny matrix

$$
\begin{array}{c|c}
 & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\
\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} &
\begin{bmatrix}
\infty & 10 & \infty & 30 & \infty & \infty \\
10 & \infty & 50 & \infty & \infty & 40 \\
\infty & 50 & \infty & \infty & 35 & 15 \\
30 & \infty & \infty & \infty & \infty & 20 \\
\infty & \infty & 35 & \infty & \infty & 25 \\
\infty & 40 & 15 & 20 & 25 & \infty
\end{bmatrix}
\end{array}
$$

prism algorithms $T.C \Rightarrow O(n^2)$

steps

1)



Treeedge

i	K,L
1	1,6
2	5,6
3	4,5
4	3,4
5	2,3
6	7,2

Near → vertices

j	Near(j)
1	~~6~~ 0
2	~~1~~ ~~3~~ 0
3	~~1~~ ~~4~~ 0
4	~~1~~,~~5~~ 0
5	~~6~~ 0
6	~~1~~ 0
7	~~1~~,~~5~~ ~~4~~ ~~2~~ 0

① find the adjacency matrix →
② Take min cost edge.
③ write it in the tree edge [1,6]
④ take the adjacent edges, if it is undefined then put 1

$$\begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & \infty & 28 & - & - & - & 10 & - \\ 2 & 28 & - & 16 & - & - & - & 14 \\ 3 & - & 16 & - & 12 & - & - & - \\ 4 & - & - & 12 & - & 22 & - & 18 \\ 5 & - & - & - & 22 & - & 25 & 24 \\ 6 & 10 & - & - & - & 25 & - & - \\ 7 & - & 14 & - & 18 & 24 & - & - \end{array}$$

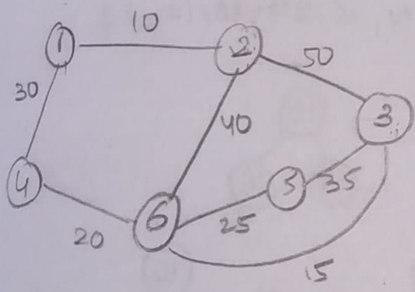
$- \rightarrow \infty$

⑤

mincost (28, ∞, ∞, ∞, 25, ∞)
mincost (∞, ∞, 22, ∞, 24)
mincost (28, 12, 18)
mincost (16, 18)
mincost (14)
mincost (T) ⇒ 99 //

57
12
30
99

Ex: ②



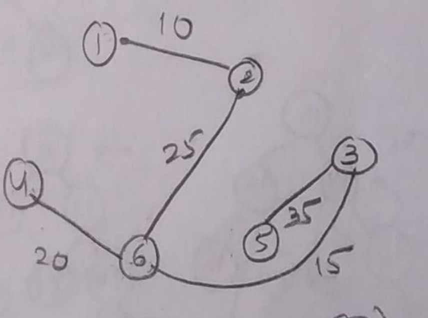
⇒ 105

Treeedge

i	k,l
1	(1,2)
2	(6,2)
3	(3,6)
4	(4,6)
5	(5,3)

near (j)

j	near(j)
1	~~2~~ 0
2	~~1~~ 0
3	~~2~~ ~~6~~ 0
4	~~1~~ ~~6~~ 0
5	~~2~~ ~~6~~ ~~3~~ 0
6	~~2~~ 0



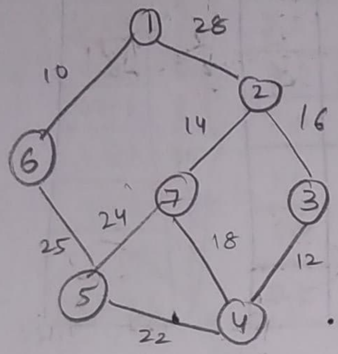
mincost (50, 40, 25)
mincost (15, 20, 55)
mincost (20, 35)
mincost (35)

35
10 + 25 + 20 + 35 + 15
⇒ 105

EDMI NOTE 5 PRO

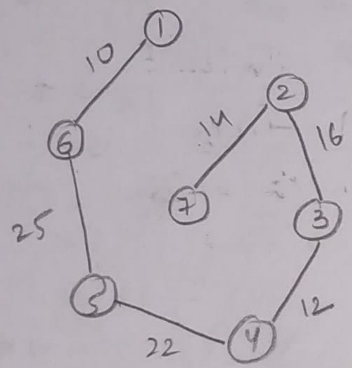
Kruskal's algorithm

T.C → $O(|E| \log|E|)$, edges



Tree Edge	Cost
1, 6	10 ✓
3, 4	12 ✓
2, 7	14 ✓
2, 3	16 ✓
4, 7	18 ✗
5, 4	22 ✓
5, 7	24 ✗
5, 6	25 ✓
2, 1	28 ✗

→ ascending order of cost

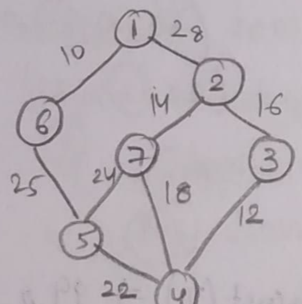


min cost(T) ⇒ 10+25+22+14+12+16

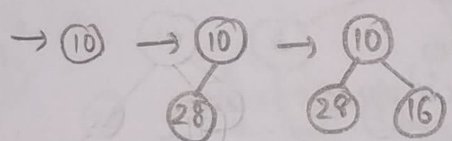
→ 99

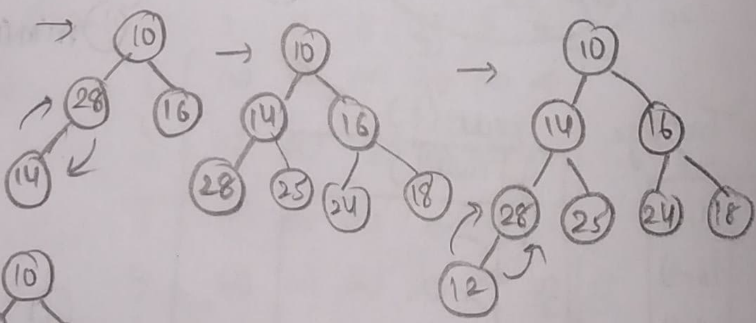
29/10/21

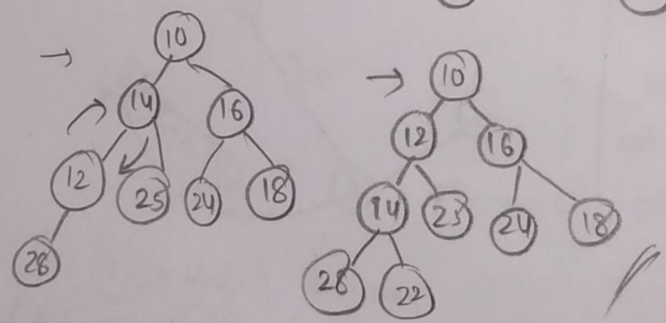
Min ~~Max~~ heap:



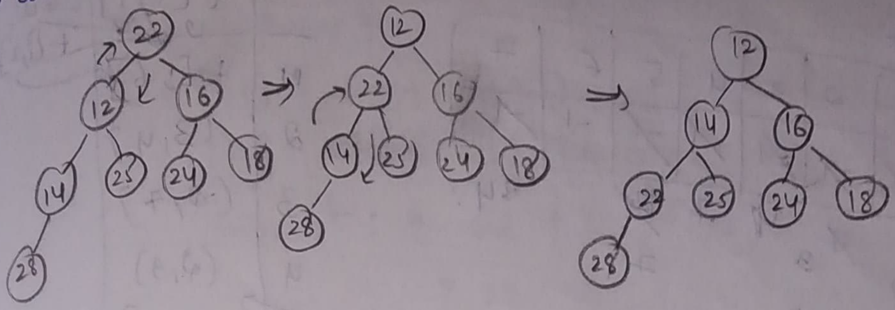
10, 28, 16, 14, 25, 24, 18, 12, 22



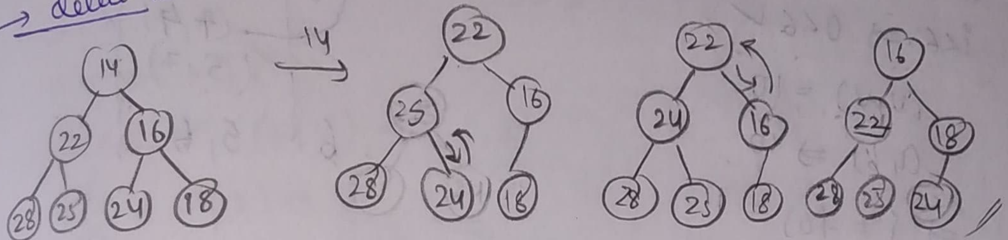




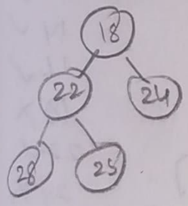
→ delete 10



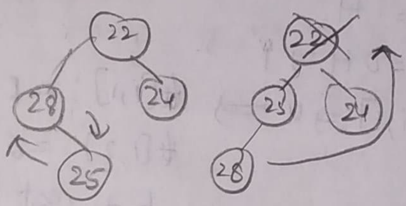
→ delete 12



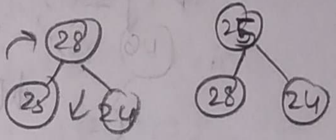
→ delete 16



delete 18



~~delete~~ 22



→ Algorithm Kruskals (E, cost, n,

Tracing

i	1	2	3	4	5	6	7
p	~~1~~	~~1~~	~~1~~ -1	~~1~~	~~1~~	-1	~~1~~
	6	~~7~~ 3	4	7. ~~4~~ 7.			4.

	$t[i,1] := u$, $t[i,2] := v$
1	$t[1,6]$
2	$(3,4)$
3	$(2,7)$
4	$(2,3)$
5	$[5,4]$
	$(4,4)$ ✗
	$(5,7)$ ✗
6	$5,6$

$i = 0$

mincost $= 0$

$i < 6 \Rightarrow 0 < 6$ ✓

$(u,v) = 10$

$(1,6) \Rightarrow 10$.

$\Rightarrow j = f(1) = 1$

$k = f(6) = 6$

if $(j \neq k)$ $i = i+1$

$i = 0+1 \Rightarrow 1$

$t[i,1] :\Rightarrow u \Rightarrow t[1,1] := 1$

$t[1,2] := 6$

mincost := mincost + cost $[u,v]$;

$= 0 + 10$

$= 10$

10 ✓
12 ✓
14 ✓
16 ✓
18 ✗
22 ✓
24 ✗
25 ✓
28

at here
$5 < 6$ ✓

$\Rightarrow$ $i = 1$

mincost $= 10$

$1 < 6 \Rightarrow (u,v) \Rightarrow (3,4) \Rightarrow 12$

$j = f(3) = 3$

$k = f(4) \Rightarrow 4$

if $(j \neq k)$ $i = i+1$

$\Rightarrow 1+1 \Rightarrow 2$

$t[i,1] :\Rightarrow u \Rightarrow t[2,1] :\Rightarrow 3$

$t[2,2] :\Rightarrow 4$.

mincost := $10 +$ cost $[3,4]$

$= 10 + 12$

$= 22$ //

$\rightarrow$ $i = 2$

mincost $= 22$

$2 < 6 \Rightarrow (u,v) \Rightarrow (2,7) \Rightarrow 14$.

$j = f(2) = 2$

$k = f(7) = 7$

if $(j \neq k) \Rightarrow i = i+1$

$i = 2+1 \Rightarrow 3$

$t[i,1] \Rightarrow u \rightarrow [t[3,1] \Rightarrow 2$

$t[i,2] \Rightarrow v \Rightarrow t[3,2] \Rightarrow 7$

mincost := 22 + cost[2,7]

= 22 + 14

= 36 //

$i = 3$

mincost = 36

$3 < 6$ ✓ $(u,v) \Rightarrow (2,3)$

$j = f(2) \Rightarrow 7$

$f(3) \Rightarrow 4$

if $(j \neq k) \Rightarrow i = i+1 \Rightarrow 3+1 \Rightarrow 4$

$t[i,1] \Rightarrow u = t[4,1] \Rightarrow 2$

$t[4,2] \Rightarrow 3$

mincost := mincost + cost[2,3]

= 36 + 16

= 52

$i = 4$

mincost = 52 cost = 18 ✗

$4 < 6$ (7,4)

$j = f(7) = 4$

$k \Rightarrow f(4) \Rightarrow 4$

if $(j \neq k) \Rightarrow$ ✗

$i = 4$

mincost = 52

$4 < 6$ [5,4]

$j = f(5) = 4$

$k = f(4) \Rightarrow 7$

$(j \neq k) \Rightarrow i = i+1 \Rightarrow 5$

$t[i,1] \Rightarrow u = t[5,1] \Rightarrow 5$

$[5,2] \Rightarrow 4$

mincost := mincost + cost[5,4]

= 52 + 22

= 74 //

$i = 5$ //

mc = 74 [cost : 24]

$5 < 6$ [5,7]

$j = f(5) = 4$

$k = f(7) = 4$

$(j \neq k) \rightarrow$ ✗

$i = 5$

mc = 74.

$5 < 6$ [5,6]

$j = f(5) \Rightarrow 7$

$k = f(7) = -1$

$(j \neq k)$ ✓ $i = i+5 \Rightarrow 6$

$t[i,1] \Rightarrow u \Rightarrow t[6,1] \Rightarrow 5$

$t[6,2] \Rightarrow 6$

mincost = 74 + 25

$\boxed{= 99 //}$

final answer.

Single Source shortest path

adjacency matrix

~~if it is~~ ~~undi~~ → no edge = ∞

→ if $i==j$ = -1

30/10/21

Unit 3

dynamic programming

→ divide and conquer.

★ principle of optionality : In d/p It states that an optimal sequence of decisions had the property that whatever the initial state & decision are, remaining decisions must constitute an optimal decision sequence with regard to the state resulting from the first decision.

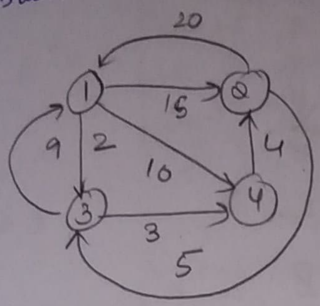
All pairs shortest path [floyd's algorithm]

→ if $i==j$ [A(i,j) then 0

→ if edge doesn't exists, then ∞

```
Algorithm AllPairs (cost, A, n)
{
    for i:=1 to n do
        for j:=1 to n do
            A[i,j] = cost [i,j];
    for k:=1 to n do
        for i:=1 to n do
            for j:=1 to n do
                A[i,j] = min (A[i,j], A[i,k]+A[k,j]);
```

→ Obtain shortest distance from all nodes to all other nodes



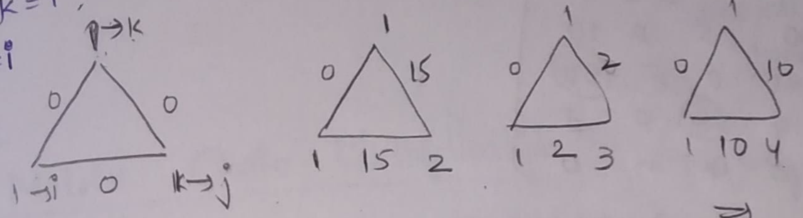
1) Cost adjacency matrix:

	1	2	3	4
1	0	15	2	10
2	20	0	5	∞
3	9	∞	0	3
4	∞	4	∞	0

2) Considering the intermediate node when k = 1.

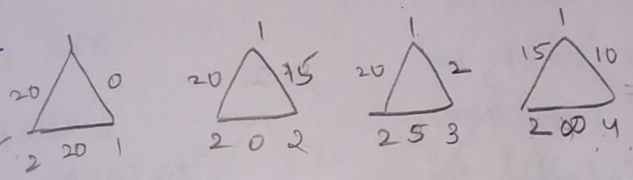
k = 1,

i=i



⇒ (0, 15, 2, 10)

i=2



⇒ (20, 0, 5, 30)

$A(i,j) = \min(A[i,j],\ A[i,k] + A[k,j])$

1) ⇒ (0, 0+0) ⇒ 0

2) ⇒ (15, 0+15) ⇒ 15

3) ⇒ (2, 0+2) ⇒ 2

4) ⇒ (10, 0+10) ⇒ 10

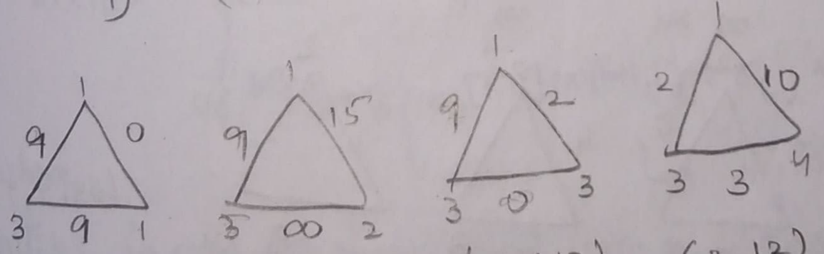
1) (20, 0+20) ⇒ 20

2) (0, 20+15) ⇒ 0

3) (5, 20+2) ⇒ 5

4) (∞, 15+10) ⇒ 30

i=3



(9, 0+9) ⇒ 9

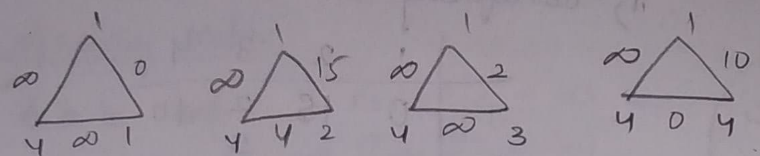
(∞, 9+15) ⇒ 24

(0, 9+2) ⇒ 0

(3, 12) ⇒ 3.

(9, 24, 0, 3)

$i = 4$



$(\infty, 0+\infty)$ $\quad$ $4, \infty+15$ $\quad$ $\infty, \infty+2$ $\quad$ $0, \infty+10$

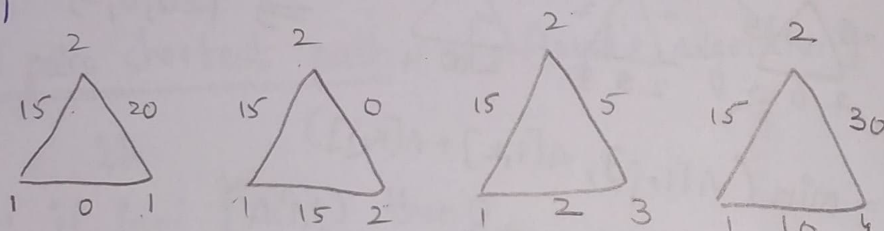
$= \infty$ $\quad$ 4 $\quad$ ∞ $\quad$ $= 0$

$= (\infty, 4, \infty, 0)$

$$A^1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 15 & 2 & 10 \\ 2 & 20 & 0 & 5 & 30 \\ 3 & 9 & 24 & 0 & 3 \\ 4 & \infty & 4 & \infty & 0 \end{array}$$

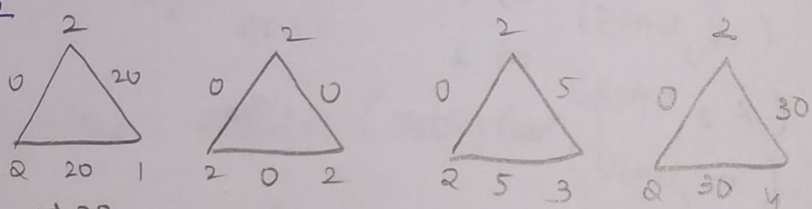
$k = 2$

$i = 1$



$\Rightarrow 0$ $\quad$ $\Rightarrow 15$ $\quad$ $\Rightarrow 2$ $\quad$ $\Rightarrow 10$ $\quad$ $(0, 15, 2, 10)$

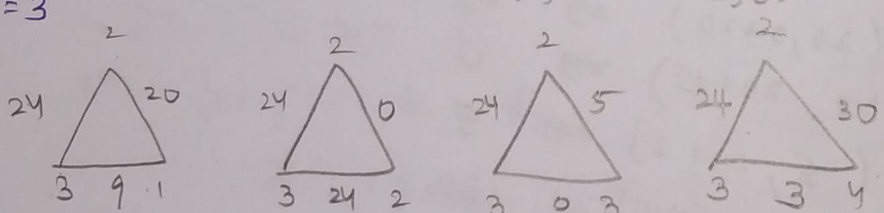
$i = 2$



$(20, 0, 5, 30)$

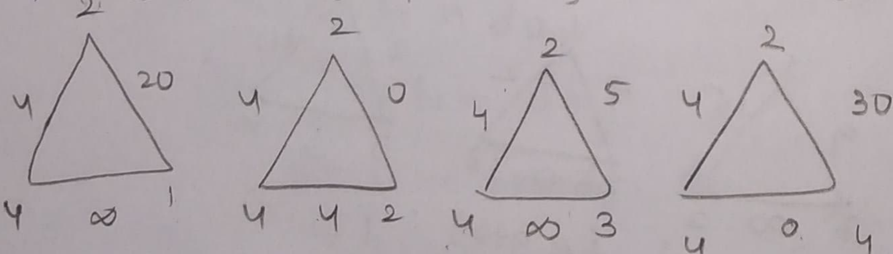
$\Rightarrow 20$ $\quad$ $\Rightarrow 0$ $\quad$ $\Rightarrow 5$ $\quad$ $\Rightarrow 30$

$i = 3$



$(9, 24, 0, 3)$

$i = 4$ $\quad$ $\Rightarrow 9$ $\quad$ $\Rightarrow 24$ $\quad$ $\Rightarrow 0$ $\quad$ $\Rightarrow 3$



$(24, 4, 9, 0)$

$\Rightarrow 24$ $\quad$ $\Rightarrow 4$ $\quad$ $\Rightarrow 9$ $\quad$ $\Rightarrow 0$

$$A^2 = \begin{bmatrix} 0 & 15 & 2 & 10 \\ 20 & 0 & 5 & 30 \\ 9 & 24 & 0 & 3 \\ 24 & 4 & 9 & 0 \end{bmatrix}$$

$K = 3$.

$$A^3 = \begin{bmatrix} 0 & 15 & 2 & 5 \\ 14 & 0 & 5 & 8 \\ 9 & 24 & 0 & 3 \\ 18 & 4 & 9 & 0 \end{bmatrix}; \quad A^4 = \begin{bmatrix} 0 & 9 & 2 & 5 \\ 14 & 0 & 5 & 8 \\ 9 & 7 & 0 & 3 \\ 18 & 4 & 9 & 0 \end{bmatrix}$$

⇓ shortest distance of all nodes

11/12/21

Matrix Chain Multiplication

No. of scalar multiplication

1) → $A \times B$

2) $A \times B \times C$

$(A \times B) \times C$

$A(B \times C$.

3) $A \times B \times C \times D$

⇒ $A(B(C \times D))$

⇒ $((AB)(CD))$

⇒ $((AB)C)D$

$(A(BC)D)$

→ we can only multiply 2 matrixes if they are Compatible

(no. of cols of A = no. of rows of B)

→ Ex:-

$A_{10 \times 100}$, B 100×5, C 5×50

$((A \times B)C) \Rightarrow D \times C \Rightarrow 10 \times 5 \times 50 \Rightarrow$

→ $m[i,j] = \begin{cases} 0 & \rightarrow \text{if } i=j \text{ ie Only 1 matrix} \\ \min_{i \le k < j} \{ m[i,k] + m[k+1,j] + r_{i-1} * r_k * r_j \} & \text{if } j > i \end{cases}$

where

→ $m[i,k]$ is min cost of evaluating $m_i * m_{i+1} * m_{i+2} \cdots m_k = m^1$.

→ $m[k+1,j]$ is the min cost evaluating $m_{k+1} * m_{k+2} * m_j = m^0$.

EDMI NOTE 5 PRO
DUAL CAMERA

→ 3rd term is the cost of multiplying m' by m''

→ m' is an $(r_{i-1} * r_k)$ matrix

→ m'' is an $(r_k * r_j)$ matrix

→ ∴ cost of $m' * m'' = r_{i-1} * r_k * r_j$

k is taken from all possible values b/w i to j-1.

Algorithm matrixchainorder (p[ ], n)

```
for i := 1 to n
{
    m[i,j] := 0
    for j := 1 to n-1 do
    {
        for k := 1 to n-1 do
        {
            j = i+1
            mij = min (m[i,k] + m[k+1,j] + r_{i-1} * r_k * r_j)
        }
        write
    }
}
```

→ Construct the matrix chain multiplication for the following

$(r_0, r_1, r_2, r_3, r_4) = (13, 5, 89, 3, 34)$

$A = (13 \times 5)$, $B = (5 \times 89)$; $C (89 \times 3)$; $D = (3 \times 34)$

$m_{11} = 0$ $m_{22} = 0$ $m_{33} = 0$ $m_{44} = 0$

(13×5) (5×89) (89×3) (3×34)

$m_{12} \Rightarrow (13 \times 5 \times 89) = 5785$

⇓

(13×89)

$m_{23} = (5 \times 89 \times 3) = 1335$

(5×3)

$m_{34} = (89 \times 3 \times 34) = 9078$

(89×34)

$m_{13} = 1530$

$m_{24} = 1645$

$m_{14} = 2856$

$m_{13} \Rightarrow m_{ij} = \min\limits_{\substack{i \le k < 3 \\ k=1}} \left(m[1,1] + m[2,3] + r_{1-1} * r_1 * r_3 \right)$

$r_0 * r_1 * r_3$

$k=2$ $\left(m[1,2] + m[3,3] + r_0 * r_2 * r_3 \right)$

$\min(0 + 1335 + 13 * 5 * 3)$

$\Rightarrow 1530$ //

$\min(5785 + 0 + 13 * 89 * 3)$

$\Rightarrow 9256$ //

$\Rightarrow \min(1530, 9256) \Rightarrow 1530$ //

$m_{24} \Rightarrow m_{ij} \Rightarrow \min\limits_{\substack{2 \le k < 4 \\ k=2}} \left(m(2,2) + m(3,4) + r_1 * r_2 * r_4 \right)$

$\Rightarrow (0 + 9078 + 5 * 89 * 34)$

$\Rightarrow 24208$

$\Rightarrow \min\limits_{k=3} \left(m(2,3) + m(4,4) + r_1 * r_3 * r_4 \right)$

$\Rightarrow (1335 + 0 + 5 * 3 * 34)$

$\Rightarrow 1845$

$\Rightarrow \min(24208, 1845)$

$\Rightarrow 1845$

$m_{14} = m_{ij} \Rightarrow \min\limits_{\substack{1 \le k < 4 \\ k=1}} \left(m(1,1) + m(2,4) + r_0 * r_1 * r_4 \right)$

$\Rightarrow (0 + 1845 + 13 * 5 * 34)$

$\Rightarrow 4055$

$\Rightarrow \min\limits_{k=2} \left(m(1,2) + m(3,4) + r_0 * r_2 * r_4 \right)$

$\Rightarrow (5785 + 9078 + 13 * 89 * 34)$

$\Rightarrow 54201$

✓ $\Rightarrow \min\limits_{k=3} \left(m(1,3) + m(4,4) + r_0 * r_3 * r_4 \right)$

$\Rightarrow (1530 + 0 + 13 * 3 * 34)$

$\Rightarrow 2856$

$\Rightarrow \min(4055, 54201, 2856)$

$m_{14} \Rightarrow 2856$

$m_{14} \Rightarrow m_{13} + m_{44} + 13 \times 3 \times 34$

$\downarrow$

$m_{11} + m_{23} + 13 \times 5 \times 3$

$\downarrow$

$m_{22} + m_{33} + 5 \times 89 \times 3$

Order of parenthesis $\Rightarrow ((A_1 (A_2 A_3)) A_4)$

H/w :-

1) $(r_0, r_1, r_2, r_3, r_4) \Rightarrow (10, 20, 50, 1, 100)$
   Calculate cost of computing $m_1 * m_2 * m_3 * m_4$.

2) A is of order 2
   R_0 to $R_4 \Rightarrow (2, 10, 3, 5, 6)$

Objectives

→ worst case of quick sort is $O(n^2)$
→ Space requirement $s(p)$ is written as $C + s(p)$
   $s(p) = C + s(p)$
→ Each instruction is clear & unambiguous is definiteness
→ Algorithm which calls itselfs is called recursion
→ strasser's mul → divide & conquer
→ Time complexity of binary search is $O(\log n)$
→ Recursion relation of normal matrix multiplication using divide & conquer strategy is $8T(\frac{n}{2}) * cn^2$
→ recu relation of strassen's matrix mul is
   $7T(\frac{n}{2}) * an^2$
→ given 3 objects with profits $(40, 30, 20)$ weights $(20, 10, 5)$ and capacity of knapsack is 20
   find total profit gain?

decreasing order → 40, 30, 20
20, 10, 5

$\frac{40}{20}, \frac{30}{10}, \frac{20}{5}$
2, 3, 4
→ (4, 3, 2)
obj → 3, 2, 1
5, 10, 20

	objects	w	+
20	3	5	1
20-5=15	2	10	1
15-10=5	1	20	$\frac{5}{20}$ 4

⇒ vector (1, 1, 1/4)

~~1×40 + 1×30 + 1/4×20~~

1× 30 + 1×20 + $\frac{1}{4}$×40 → 10

⇒ 60 //

→ Job sequencing with standard problem fits what paradigm? Subset

→ minimum cost spanning tree fit in Ordering paradigm

→ In Big O(O) asymptotic notation is $a(n) \geq f(n)$

→ ~~polynomial~~ performance analysis refers to the task of determining Time & Space complexity

→ In D&C strategy, a problem results subprograms of same

→ Give the recurrence relation computing time of merge sort ⇒ $2T(n/2) + cn$

→ if S1 & S2 are 2 sets and no common elements b/w is called disjoint set

→ T.C. of strassen's matrix mul is $n^{2.81}$

→ Graph G is said to be bidirectional if n articulation points

→ Optimal sol is called

→ if J is a node of path from i to the root & P(i) ≠ root (i) then P(j) to root of i is said to be collapsing find( ) algorithm

REDMI NOTE 5 PRO
MI DUAL CAMERA

→ In greedy method, at each stage a decision is made regarding a particular soln is optimal sol

→ A tree which includes all the vertices is called Spanning tree

→ The obj of spanning tree is it maximized total no. of profit earned

→ Node satisfies

$L(w) \geq DFN(v)$

→ The esscence of greedy algorithm is maximization / minimazation

→ In Krushal's edges are selected in increasing order

→ Single source shortest path is Greedy technique

→ All pairs path is ________

→ Complexity of adding 2 matrix n×m is nm

→ The running time of which technique is based on whether the partition q is balanced quick sort or not

→ Union & find are 2 operations r considered for disjoint sets

→ Recursion is a technique for which it determines it self

→ krushal's alg → min cost of spanning tree

→ Time required for prism's algo is n^2

→ The no. of times a statement is referred as frequency

→ worst case is max no. of steps that can be used to find max no. of steps

REDMI NOTE 5 PRO
MI DUAL CAMERA

→ Job sequencing with deadline uses greedy method
→ The edges which r passed in spanning tree but not vertices is Cross edges & back edges
→ Avg case of quick sort is nlogn
→ quick sort worst t.c is 0
'' best t.c ⇒ n logn
→ program is an expression of algo in prg lng
→ root
→ Greedy algo don't give optimal soln
→ Space Complexity is amount needed to complete.
→ "n" vertices → "n-1" edges → in s.t.
→ n-1 unions can be processed in O(n)
→ for any job & profit P(i) is earned by
→ A vertex v is an articultion point definition
→ Time req of finding no. of findsu is O(n)
→ In Control abstraction of greedy method the fn is boolean valu
fn
Questions
→ The sol for T(n) is 2T * n/2 +17+n is O(nlogn)
proof whether sol is equal to nlogn
→ give a recurrence algo two positive m & n using only sub and addition.
→ Explain briefly about mergesort write algorithm.
→ Consider n=7 (m) knapsort Capacity = 15
P_1 to P_7 = (10, 5, 15, 7, 6, 18, 3)
w_1 to w_7 = (2, 3, 5, 7, 1, 4, 1)
obtain the optimal sol for knapsack instance.
REDMI NOTE 5 PRO
MI DUAL CAMERA

→ Consider a graph to show single source shortest path

→ Construct the optimal schedule for n = 7

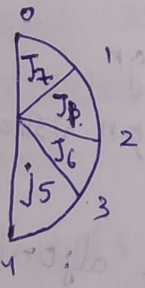
P_1 to P_7 = (100, 10, 15, 27, 120, 55, 40)

deadlines d_1 to d_7 = (2, 1, 2, 2, 1, 4, 3, 1)

feasible sol	processing seq	profit value

i) decreasing

pi ⇒	120	100	55	40	27	15	10
di ⇒	4	2	3	1	2	2	1
ji ⇒	5	1	6	7	4	3	2
					×	×	



→ J_7, J_1, J_6, J_5

⇒ (40 + 100 + 55 + 120)

⇒ 315 //

100
120
15
55
290

40
100
55
120
315

→ define space complexity

→ T(P) strassen's matrix mul

→ Recurrence relation of binary search with worst case

→ In matrix chain problem for 4 matrices in how many ways can we multiply ?

→ define alg & characteristics of algo
→ Explain Eg, graph, definitions of Big $O(n)$
→ weighted Union & collapsing find → def, Eg
→ Assume the quick sort first item as pivot item
i) give the list of elements of array 10, Considering worst case scenario
ii) " best case, worst Case
give Eg and determine t.c.

→ Explain Algorithm
→ explain prism algorithm.
→ feasible, optimal solution
→
→ Greedy sol of knap sack → 2 marks
↓
optimal sol = $\frac{p_i}{w_i}$

→ performance analysis of Alg (space & time)
→ Components of Space Complexity.
→ Time Complexity ⇄ Step Count, Operation Count.
→ define biconnected graph
→ explain strassen's matrix mul
→ find S.C. of Recursive algo
→ algo for merge sort
→ what is profiling
→ what is debugging
→ recurrence relation will be given $T(n) = T(n-1) + 1$.
where $T(n) = 1$.

$T(n) = 3T + \frac{n}{4} + n$
where $T(1) = 2$. Using substitution method / master method

→ Compare / contrast dynamic programming with greedy
→ what are diff mathematics of notations for finding T.C & S.P is asymptotic ~~notation~~.

EDMI NOTE 5 PRO
I DUAL CAMERA

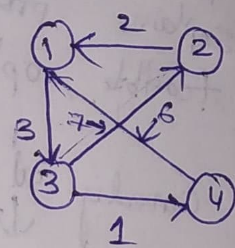
→ analyze t.c. of quicksort
→ compare avg t.c of quick sort with

3/11/21

1) Obtain the shortest distance from all nodes to all the other nodes (Floyd's Algorithm)

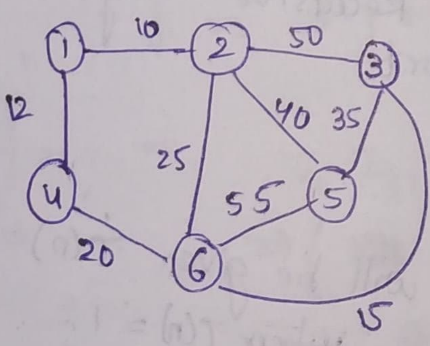
2) Obtain the matrix chain multiplication of the following
$A = (2 \times 10)$, $B = (10 \times 3)$, $C = (3 \times 5)$, $D = (5, 6)$

3) find an optimal solution to the knapsack instance
$n = 7$, $m = 15$
$(P_1, \ldots P_7) = (10, 5, 15, 7, 6, 18, 3)$
$(w_1, \ldots w_7) = (2, 3, 5, 7, 1, 4, 1)$



4) Job : 1 2 3 4 5 6
deadline : 2 1 2 3 2 4
profit : 20 25 30 40 35 20
Solve the instance using greedy method.

5) solve the following using prism's Algo to find the minimum cost spanning tree.



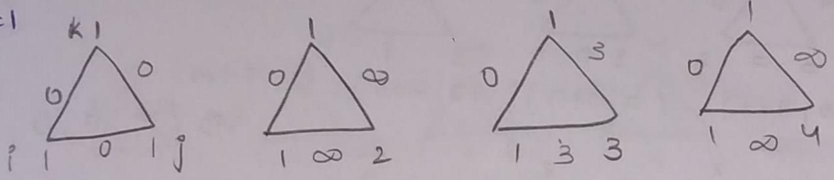
Answers

1) Cost adjacency matrix

	1	2	3	4
1	0	∞	3	∞
2	2	0	∞	∞
3	∞	7	0	1
4	6	∞	∞	0

k=1

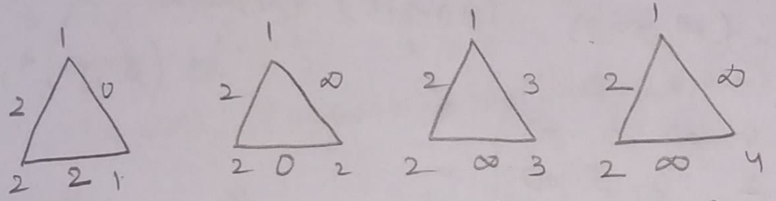
i=1



$\Rightarrow (0, 0+0), (\infty, 0+\infty), (3, 0+3), (\infty, 0+\infty)$

$\Rightarrow (0, \infty, 3, \infty)$

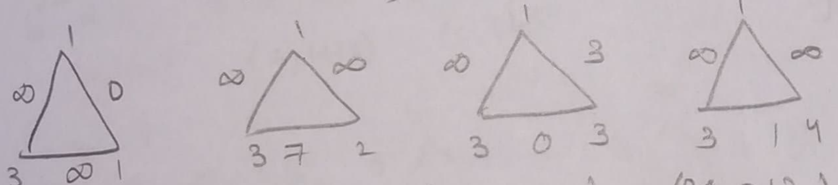
i=2



$\Rightarrow (2, 2+0), (0, 2+\infty), (\infty, 2+3), (\infty, 2+\infty)$

$\Rightarrow (2, 0, 5, \infty)$

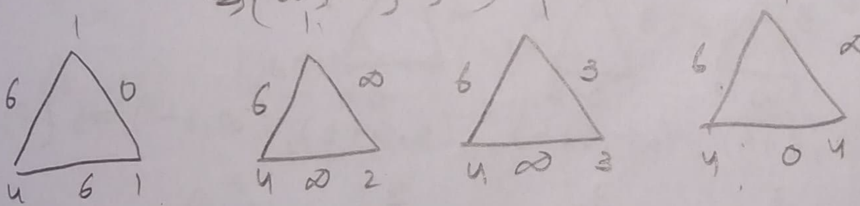
i=3



$\Rightarrow (\infty, 0+\infty), (7, \infty+\infty), (0, \infty+3), (1, \infty+\infty)$

$\Rightarrow (\infty, 7, 0, 1)$

i=4



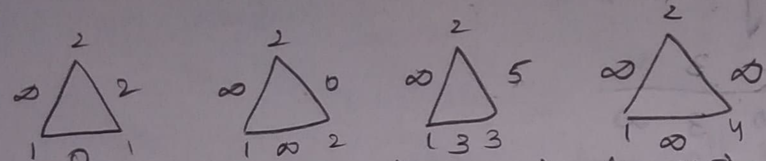
$\Rightarrow (6, 0+6), (\infty, 6+\infty), (\infty, 6+3), (0, 6+\infty)$

$\Rightarrow (6, \infty, 9, 0)$

$$A^1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & \infty & 3 & \infty \\ 2 & 2 & 0 & 5 & \infty \\ 3 & \infty & 7 & 0 & 1 \\ 4 & 6 & \infty & 9 & 0 \end{array}$$

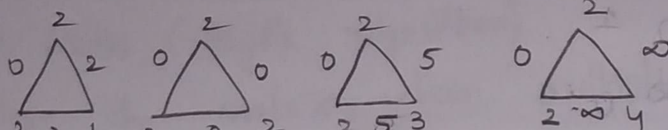
$k=2$

$i=1$



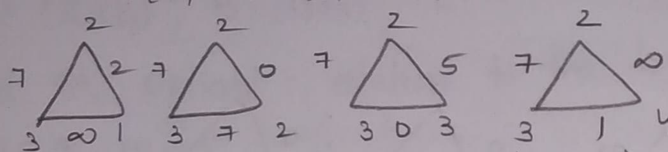
$(0, \infty+2)\ (\infty, 0+\infty),\ (3, \infty+5),\ (\infty, \infty+\infty) \Rightarrow (0, \infty, 3, \infty)$

$i=2$



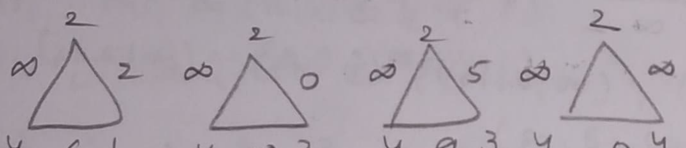
$(2, 0+2)\ (0, 0+0),\ (5, 0+5),\ (\infty, 0+\infty) \Rightarrow (2, 0, 5, \infty)$

$i=3$



$(\infty, 7+2)\ (7, 0+7),\ (0, 7+5),\ (1, 7+\infty) \Rightarrow (9, 7, 0, 1)$

$i=4$

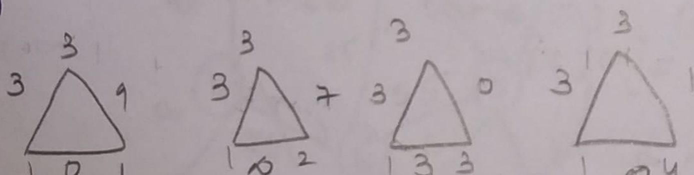


$(6, \infty+2),\ (\infty, \infty+0),\ (9, \infty+5)\ (0, \infty+\infty) \Rightarrow (6, \infty, 9, 0)$

$$A^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & \infty & 3 & \infty \\ 2 & 2 & 0 & 5 & \infty \\ 3 & 9 & 7 & 0 & 1 \\ 4 & 6 & \infty & 9 & 0 \end{array}$$

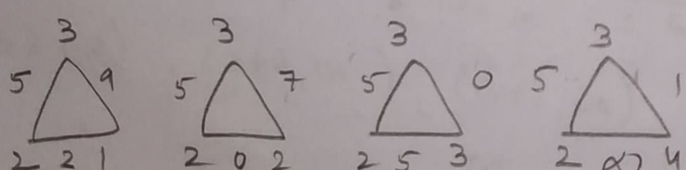
$k=3$

$i=1$



$(0, 3+9),\ (\infty, 3+7),\ (3, 0+3),\ (\infty, 3+1) \Rightarrow (0, 10, 3, 4)$

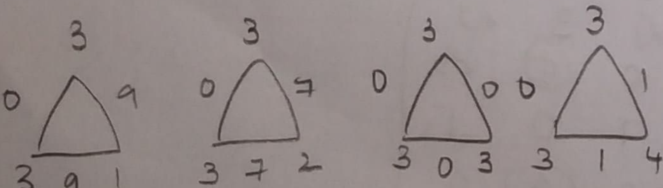
$i=2$



$\Rightarrow (2, 0, 5, 6)$

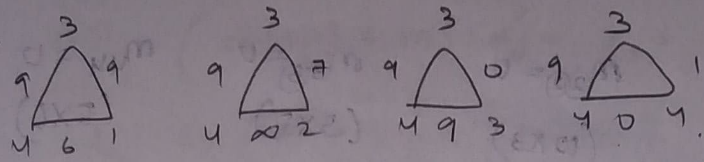
$(2, 5+9),\ (0, 5+7),\ (5, 0+5),\ (\infty, 5+1)$

$i=3$



$(9, 0+9)\ (7, 0+7)\ (0, 0+0)\ (1, 0+1) \Rightarrow (9, 7, 0, 1)$

$i=4$.



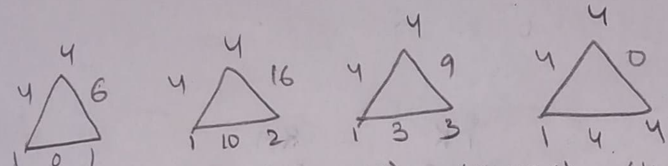
$(6, 9+9), (\infty, 9+7)\ (9, 0+9)\ \ (0, 9+1)$

$\Rightarrow (6, \infty, 9, 0)$

$$A^3 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 10 & 3 & 4 \\ 2 & 2 & 0 & 5 & 6 \\ 3 & 9 & 7 & 0 & 1 \\ 4 & 6 & 16 & 9 & 0 \end{array}$$

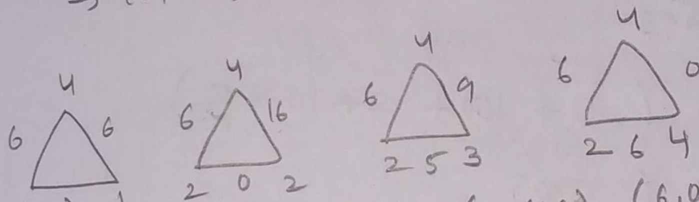
$k=4$

$i=1$



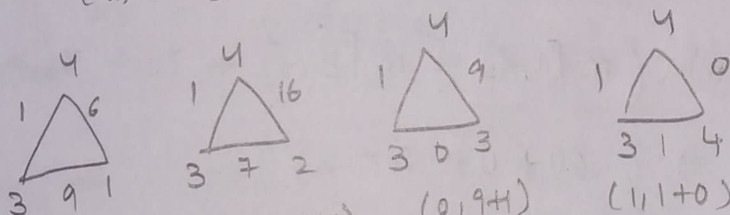
$\Rightarrow (0, 4+6)\ ;\ (10, 4+16)\ ,\ (3, 4+9)\ ,\ (4, 4+0) \Rightarrow (0, 10, 3, 4)$

$i=2$



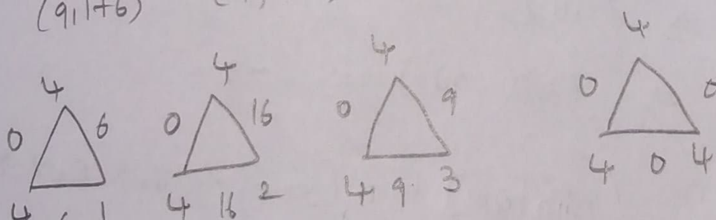
$(2, 6+6),\ (0, 6+16)\ \ (5, 6+9)\ \ (6, 0+6) \Rightarrow (2, 0, 5, 6)$

$i=3$



$(9, 1+6)\ \ (7, 1+16)\ \ (0, 9+1)\ \ (1, 1+0) \Rightarrow (9, 7, 0, 1)$

$i=4$



$(6, 0+6)\ \ (16, 0+16)\ \ (9, 0+9)\ \ (0, 0+0) \Rightarrow (6, 16, 9, 0)$

$$A^4 = \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 9 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix}$$

2)

$m_{11} = 0$ (2×10) $m_{22} = 0$ (10×3) $m_{33} = 0$ (3×5) $m_{44} = 0$ (5×6)

$r_0 = 2$
$r_1 = 10$
$r_2 = 3$
$r_3 = 5$
$r_4 = 6$

$m_{12} = (2 \times 10 \times 3) = 60$ $m_{23} = (10 \times 3 \times 5) = 150$ $m_{34} = (3 \times 5 \times 6) = 90$

$m_{13} \Rightarrow 90$ $m_{24} = 270$

$m_{14} \Rightarrow 150$

$$m_{13} \Rightarrow m_{ij} = \min_{1 \le k < 3} (m[i,k] + m[k+1,3] + r_0 * r_1 * r_3)$$

$k=1$

$$\Rightarrow (0 + 150 + 2 * 10 * 5)$$

$$\Rightarrow (250)$$

$$k=2 \Rightarrow \min (m[1,2] + m[3,3] + r_0 * r_2 * r_3)$$

$$\Rightarrow (60 + 0 + 2 * 3 * 5)$$

$$\Rightarrow (90)$$

$$m_{13} \Rightarrow (250, 90) \Rightarrow 90$$

$$m_{24} \Rightarrow m_{ij} \Rightarrow \min_{2 \le k < 4} (\min[2,2] + m[3,4] + r_1 * r_2 * r_4)$$

$$k=2 \Rightarrow (0 + 90 + 10 * 3 * 6) \Rightarrow 270$$

$$\min_{2 \le k < 4} (\min[2,3] + m[4,4] + r_1 * r_3 * r_4)$$

$$k=3 \Rightarrow (150 + 0 + 10 * 5 * 6) \Rightarrow 450$$

$m_{14} \Rightarrow \min_{1 \le k < 4} (m[1,1] + m[2,4] + r_0 * r_1 * r_4)$

$k=1 \Rightarrow (0 + 270 + 2*10*6) \Rightarrow 390$

$k=2 \quad (m[1,2] + m[3,4] + r_0 * r_2 * r_4)$

$\Rightarrow (60 + 90 + 2*3*6) \Rightarrow 186$

$k=3 \quad (m[1,3] + m[4,4] + r_0 * r_3 * r_4)$

$\Rightarrow [90 + 0 + \underline{2*5*6}] \Rightarrow 150$

$\Rightarrow (390 , 186 , 150)$

$\Rightarrow 150 //$

$\underline{m_{14}} \Rightarrow \underline{m_{13}} + \underline{m_{44}} + \underline{2*5*6}$

$\downarrow$

$m_{12} + m_{33} + \underline{2*3*5}$

$\downarrow$

$m_{11} + m_{22} + \underline{2*10*3}$

Order of parenthesis $\Rightarrow$ $\underline{A_1 A_2 A_3 A_4}$

$A_3) A_4$

$\underline{m_{14}} \Rightarrow m_{13} + m_{44}$

$(\underline{m_{12} + m_{33}}) \underline{A_4}$

$m_{11} \; m_{22}$

$(((A_1 A_2) A_3) A_4) //$

$m_{14} \Rightarrow \underline{m_{13}} + \underline{m_{44}}$

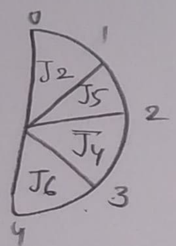
$((m_{12} * A_3) A_4$

$(((A_1 A_2) A_3) A_4)$

REDMI NOTE 5 PRO
MI DUAL CAMERA

4) Job : 1 2 3 4 5 6
dead : 2 1 2 3 2 4
profit 20 25 30 40 35 20

profit	40	35	30	25	20	20
deadline	3	2	2	1	2	4
Jobs	4	5	3	2	1	6
	✓	✓	×	✓		✓



(J2, J5, J4, J6)

⇒ (25 + 35 + 40 + 20)

⇒ 120 //

3) n = 7, m = 15

(P1, P7) ⇒ (10, 5, 15, 7, 6, 18, 3)

(w1, w7) ⇒ (2, 3, 5, 7, 1, 4, 1)

Case 1 : increasing order :

weight : 1, 1, 2, 3, 4, 5, 7

profit : 6, 3, 10, 5, 18, 15, 7

3 object 5, 7, 1, 2, 6, 3, 4

✓ ✓ ✓ ✓ ✓ × ×

value

Cap = 15

Remai	O.S	weight	fracti
15	5	1	1
15-1 = 14	7	1	1
14-1 = 13	1	2	1
13-2 = 11	2	3	1
11-3 = 8	6	4	1
8-4 = 4	3	5	4/5 = 0.8
	-	-	0
		-	0

solution vector ÷ 1, 1, 1, 1, 1, 0.8, 0,

$\Rightarrow$ profit = $(1 \times 6 + 1 \times 3 + 1 \times 10 + 1 \times 5 + 1 \times 18 + 0.8 \times 15)$

$\Rightarrow$ 54 //.

Case 2 :

P = 18, 15, 10, 7, 6, 5, 3

W = 4, 5, 2, 7, 1, 3, 1.

O = 6, 3, 1, 4, 5, 2, 7

Rem	O.S	we	fr
15	6	4	1
15 - 4 = 11	3	5	1
11 - 5 = 6	1	2	1
6 - 2 = 4	4	7	$\frac{4}{7} \Rightarrow 0.57$
0	-	-	=
0	-		

vector = 1, 1, 1, 0.57

profit earned $\rightarrow$ $(18 \times 1, 1 \times 15 + 1 \times 10 + 0.57 \times 7$

= ~~47.9~~ // , 46.99

Case 3 :

1	2	3	4	5	6	7
5,	$\frac{5}{3}$ $\rightarrow$ 1.6,	3,	1,	6,	$\frac{18}{4}$ $\rightarrow$ 4.5	3

$\Rightarrow$ 6, 5, 4.5, 3, 3, 1.6, 1 $\rightarrow$ v

object $\Rightarrow$ 5, 1, 6, 3, 7, 2, 4.

	ob	w	
15	5	1	1
15-1 = 14	1	2	1
14-2 = 12	6	4	1
12-4 = 8	3	5	1
8-5 = 3	7	1	1
3-1 = 2	2	3	2/3 $\Rightarrow$ 0.6

profit earned = ~~1× 1× 1× 1× 1× 1×~~ 0.6

$\Rightarrow 1\times 1 + 1\times 2 + 1\times 4 + 1\times 5 + 1\times 1 + 0.6\times 3$

5, 1, 6, 3, 7, 2
1
0.6

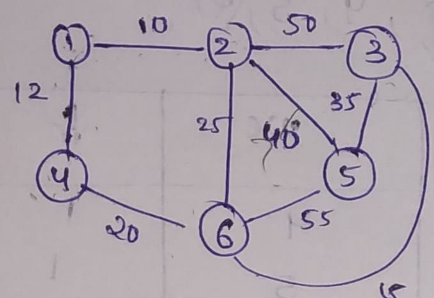
$\Rightarrow 14.8$ //

$= 1\times 6 + 1\times 10 + 1\times 18 + 1\times 15 + 1\times 3 + 0.6\times 5$

$= 55$

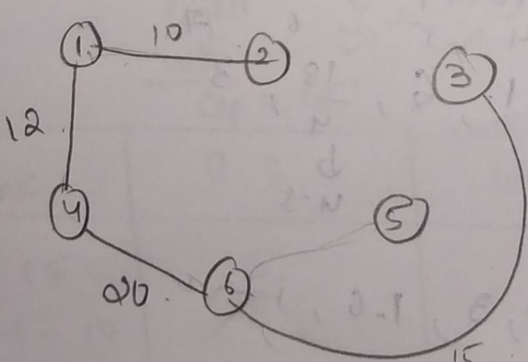
Optimal solution = 55
(case 3)

5)



$$
\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 10 & \infty & 12 & \infty & \infty \\
2 & 10 & 0 & 50 & \infty & 40 & 25 \\
3 & \infty & 50 & 0 & \infty & 35 & 15 \\
4 & 12 & \infty & \infty & 0 & \infty & 20 \\
5 & \infty & 40 & 35 & \infty & 0 & 55 \\
6 & \infty & 25 & 15 & 20 & 55 & 0
\end{array}
$$

Tree edge		near	
i	k,L	j	Near[j]
1	1,2	1	~~2~~ 0
2	4,1	2	~~1~~ 0
3	6,3	3	~~2~~ ~~3~~ 6 0
4	6,4	4	~~1~~ 0
5		5	~~2~~ ~~3~~ 6
		6	~~2~~ ~~3~~ ~~4~~ 0



⇒ 50, 12, 40, 25 ✓

~~50 40~~ 25
35

⇒ ~~35~~, 35, 15 ✓

⇒ [illegible], 55, 20 ✓