Ravina, Shubham, Keshutosh,
Divya, Shweta

Page No.
Date :

Algorithm → priori Analysis (Theortical) (before implementation)

Analysis

Time → postriori Analysis (Practical) (After implementation)

Theortical Analysis in terms of time.

Design :-

1. Divide and conquer
2. Greedy method
3. Dynamic programming
4. Backtracking
5. Branch and Bound

Priori Analysis :- Mathematical function

② Asymptotic Notation :-

These are the mathematical notation which are used to do the analysis of an algorithm in terms of mathematical inequality or mathematical function or mathematical equation.

There are various kinds of mathematical notations or asymtotic notation which are as follows :-

Date :

1. Big oh (O) :- Worst case (upper bound of an algorithm)

2. Omega (Ω) :- Best case (lower bound of an algorithm)

3. Theta (Θ) :- Average case (tight bound)

① Big oh (O) :-

A function f(n) is said to be Big oh (O) of g(n) such that

$$f(n) \leq c * g(n) \quad \forall n \geq n_0$$

where c and n are positive constants and integers.

2. Omega (Ω) :-

A function f(n) is said to be Omega (Ω) of g(n) such that

$$f(n) \geq c * g(n) \quad \forall n \geq n_0$$

where c and n are positive ~~and~~ constants and integers.

3. Theta (Θ) :-

A function f(n) is said to be Theta (Θ) of g(n) such that

$$c_1 * g(n) \leq f(n) \leq c_2 * g(n)$$

where c and n are positive constants and integers.

Example :- $f(n) = n^2+2$, $g(n) = n^3$

$$f(n) \leq c * g(n)$$

suppose that $c = 1$

$$(n^2+2) \leq 1 * n^3$$
$$(n^2+2) \leq n^3$$

Suppose that $n = 1$

$$(1^2+2) \leq (1)^3$$
$$1+2 \leq 1$$
$$3 \leq 1$$ (condition is not satisfied)

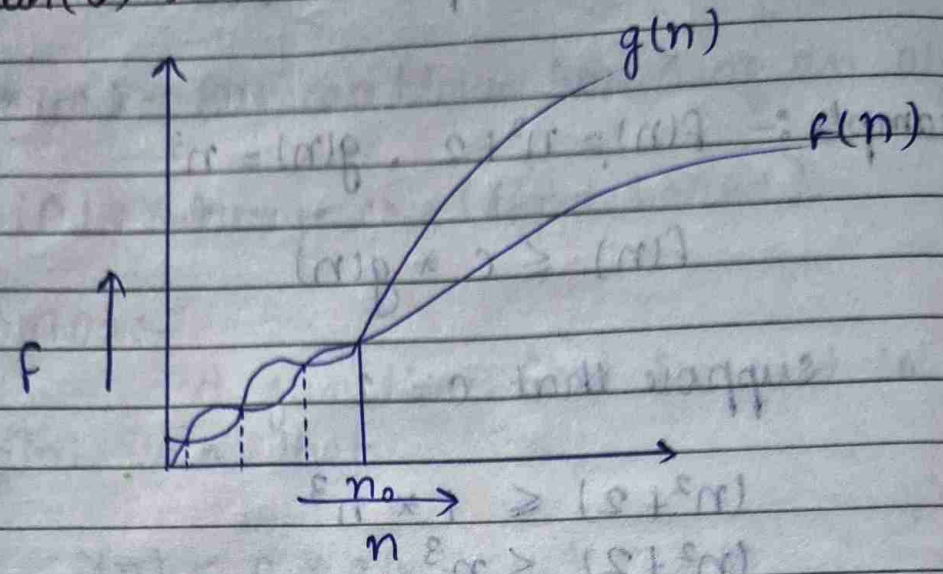
let us $n = 2$

$$(2^2+2) \leq (2)^3$$
$$4+2 \leq 8$$
$$6 \leq 8$$ (condition is satisfied)

let us we check for $n = 4, 10$ etc.

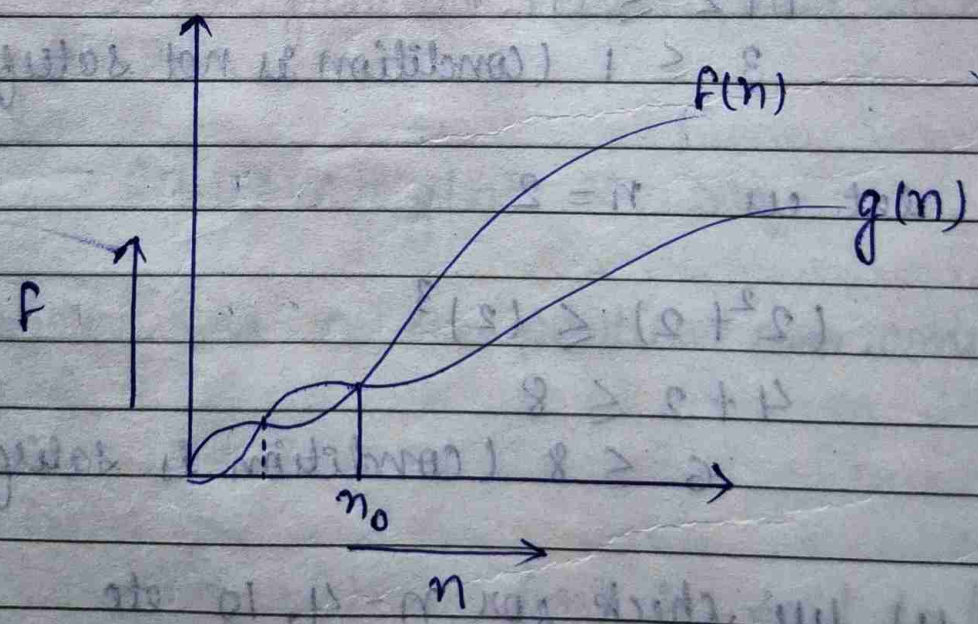
when $n = 4 \Rightarrow (4^2+2) \leq (4)^3 \Rightarrow 16+2 \leq 64 \Rightarrow 18 \leq 64$
when $n = 10 \Rightarrow (10^2+2) \leq (10)^3 \Rightarrow 102 \leq 1000$ (condition satisfied)

1. Big oh(O) :-



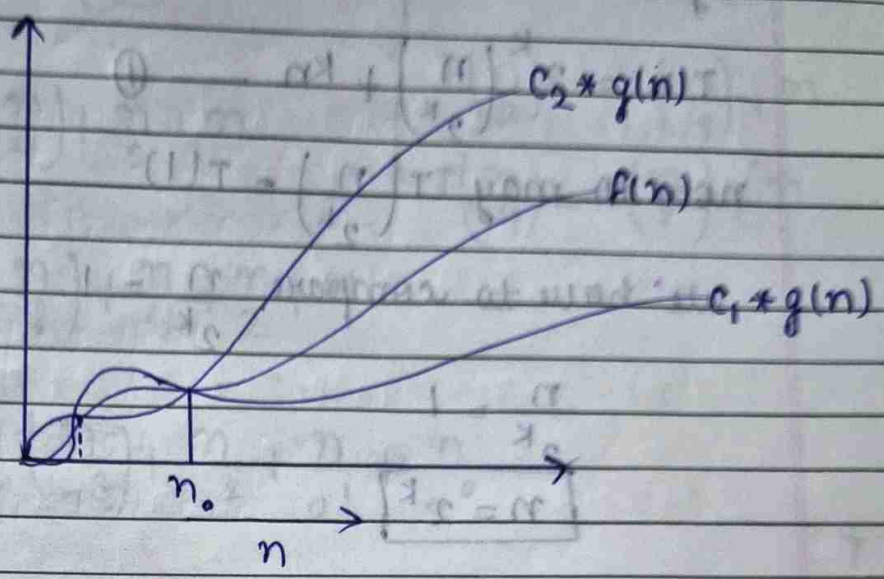
$f(n) \leq g(n) \quad \forall\, n \geq n_0$

2. Omega(Ω) :-



$f(n) \geq g(n) \quad \forall\, n \geq n_0$

3. Theta (Θ) :-



$$c_1 * g(n) \leq f(n) \leq c_2 * g(n)$$

---

* $T(n) = 2T\left(\frac{n}{2}\right) + n$ $\qquad T(1) = 0$

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

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

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

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

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

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

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

$\vdots$

K times

$$T(n) = 2^k\left(\frac{n}{2^k}\right) + kn \quad \text{——} \quad (1)$$

now to may $T\left(\frac{n}{2^k}\right) = T(1)$

we have to compare $\frac{n}{2^k} = 1$

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

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

substitute the value of $n = 2^k$ in the equation number (1)

$$\Rightarrow 2^k T\left(\frac{\cancel{2^k}}{\cancel{2^k}}\right) + kn$$

$n = 2^k$
taking $\log_2$ both sides
$\log_2 n = k$

$$\Rightarrow 2^k T(1) + kn$$

$$\Rightarrow 2^k * 0 + kn$$

$$\Rightarrow 0 + kn$$

$$\Rightarrow kn$$

$$T(n) \Rightarrow n * (\log_2 n)$$

$T(n) \leq n * \log_2 n$, $T(n) \in O\ n\log_2 n$

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

$$\begin{cases} T(1) = 0 \\ T\left(\frac{n}{2}\right) = T\left(\frac{n}{4}\right) + n \\ T\left(\frac{n}{4}\right) = T\left(\frac{n}{8}\right) + n \end{cases}$$

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

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

$$T(n) = T\left(\frac{n}{2^3}\right) + \frac{n}{2^2} + \frac{n}{2^1} + \frac{n}{2^0}$$

K times

$$T\left(\frac{n}{2^K}\right) + \left[\frac{n}{2^{K-1}} + \frac{n}{2^{K-2}} + \quad + \frac{n}{2} + n\right]$$

Date :

$T(n) = T(n-1) + n \quad , T(1) = 1$

$$\begin{cases} T(n-1) = T(n-2) + n - 1 \\ T(n-2) = T(n-3) + n - 2 \end{cases}$$

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

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

$\{1 + 2 + 3 + \ldots + (n-2) + (n-1) + n\}$

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

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

$T(n) \leq c * \frac{n^2}{2}$

$T(n) \in O(n^2)$

Ans.

Date :

Divide And Conquer

Merge sort (Algorithm)

```
Procedure Merge_sort (low, high)
Array A(n)
if low < high then
    mid = [(low+high)/2];
call Merge_sort (low, mid);
call merge_sort (mid+1, high);
call merge (low, mid, high);
endif
end Merge_Sort
```

```
Procedure Merge (low, mid, high)

Array B(n), A(n)
h = high; i = low; j = mid+1;

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;
endif
        i = i+1;
```

Page No.
Date :

```
end while
    if h > mid then

        for K = j to high Do

            B(i) = A(K);
            i = i + 1;
        end for
        else
            for K = h to mid Do
            B(i) = A(K);
            i = i + 1;
        endfor
        endif

        for K = low to high Do
            A(K) = B(K);
        end for
        end merge
```

Page No.
Date :

Recurrence of Merge Sort

$$T(n) = 2T\left(\frac{n}{2}\right) + cn \qquad n \geq 2 \qquad n(1) = 0$$
$$n = 1$$

$$\Rightarrow 2\left\{2T\left(\frac{n}{4}\right) + \frac{cn}{4}\right\} + cn \qquad \left\{\begin{array}{l} T\left(\frac{n}{2}\right) = 2T\left(\frac{n}{4}\right) + \frac{cn}{4} \\ T\left(\frac{n}{4}\right) = 2T\left(\frac{n}{8}\right) + \frac{cn}{8} \end{array}\right.$$

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

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

$$\Rightarrow 4\left\{2T\left(\frac{n}{8}\right) + \frac{cn}{8}\right\} + 2cn$$

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

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

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

$$\Rightarrow \vdots$$

K times

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

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

$$\boxed{n = 2^k} \qquad \log_2 n = \log_2 2^k$$

$$\log_2 n = k$$

$$\Rightarrow 2^k T\left(\frac{2^k}{2^k}\right) + kcn$$

$$\Rightarrow 2^k T(1) + kcn$$

$$\Rightarrow 2^k (0) + kcn$$

$$\Rightarrow kcn$$

$$\Rightarrow \log_2 n * c * n$$

$$\Rightarrow c\, n \log_2 n$$

$$T(n) \leq c * n \log_2 n$$

$$T(n) \in O(n \log_2 n)$$

MergeSort Algo :-

```
MergeSort (a, F, L)
{
  if (F<L)
  {
```

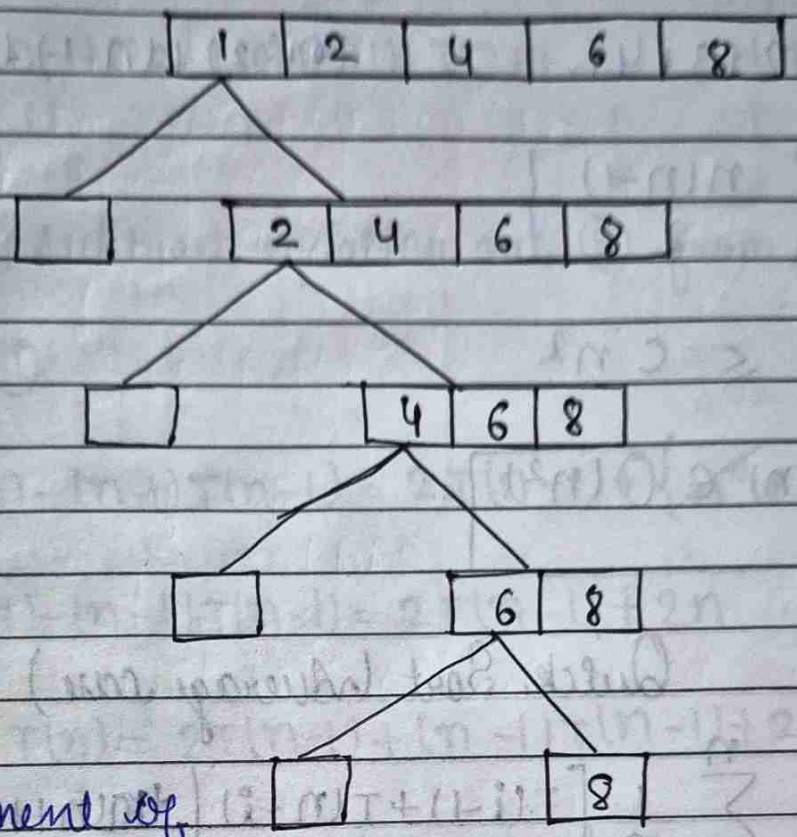
```
mid = (f+L/2)
mergeSort (a, f, mid)
mergeSort (a, mid+1, L)
}
merge (a, f, mid, L)
{
  ~~if (f<L)~~

  i = f;
  j = mid+1;
  k = 0;
  while (i ≤ mid && j≤L)
  {
    if (a[i] < a[j])
    {
      b[k] ← a[i];
      i++;
      k++;
    }
  else
  {
    b[k] ← a[j]
    j++;
    k++;
  }
}
```

Page No.
Date :

```
while (i ≤ mid)
{
    b[k] = a[i];
    i++;
    k++;
}
while (j ≤ L)
{
    b[k] = a[j];
    j++;
    k++;
}
for(i=0; i<L; i++)
{
    a[i] ← b[i]
}
```

(iii)

Quick Sort (Worst case)



first element of list is always pivot

<u>Worst case</u>:-

$$T(n) = \begin{cases} T(n-1) + cn, & n \geq 2 \\ 0, & n = 1 \end{cases}$$

$$T(n) = \left\{ T(n-1) + c(n-1) \right\} + cn$$

$$T(n) = T(n-3) + c(n-2) + c(n-1) + cn$$

$$T(1) + c*2 + c*3 + \cdots + c(n-2) + cn$$

$\Rightarrow 2c + 3c + \ldots + c(n-2) + c(n-1) + cn$

$\Rightarrow c\{2+3+4+\ldots+(n-2)+(n-1)+n\}$

$\Rightarrow c\left\{\frac{n(n-1)}{2}\right\}$

$T(n) \leq cn^2$

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

09-2017

~~T(n)~~ = Quick Sort (Average case)

$$* \quad T(n) = \sum_{i=1}^{n} \frac{1}{n}\left[T(i-1)+T(n-i)\right] + n(n+1)$$

$n \geq 2$, $n = 1$

$$nT(n) = \sum_{i=1}^{n} \{T(i-1)+T(n-i)\} + n(n+1)$$

$nT(n) = 2T(0) + 2T(1) + 2T(2) + \ldots + 2T(n-1) + n(n+1)$

$nT(n) = 2\{T(0)+T(1)+T(2)+\ldots+T(n-1)\} + n(n+1)\}$ → (1)

Replacing n with (n-1) in equation no. 1

$$(n-1)T(n-1) = 2[T(0)+T(1)+T(2)+\ldots\ldots+T(n-1-1)]+(n-1)(n-1+1)$$

$$(n-1)T(n-1) = 2[T(0)+T(1)+T(2)+\ldots\ldots+T(n-2)]+n(n-1) \rightarrow (2)$$

Now subtract equation no. (2) from equation no. (1)

(1) - (2)

$$nT(n)-(n-1)T(n-1) = 2T(n-1)+\{\cancel{n^2}+n-\cancel{n^2}+n\}$$

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

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

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

$$nT(n) = T(n-1)(n+1)+2n \quad (3)$$

Devide equation no. (3) both sides by $(n+1)$

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

$$\frac{nT(n)}{n+1} = \frac{2n}{n+1}+T(n-1) \qquad T(1)=0$$

$$= \frac{2n}{n+1}+\frac{2(n-1)}{n}+\frac{2(n-2)}{n-1}+\ldots\ldots+\frac{2}{3}+\cancel{\frac{T(1)}{2}}$$

$$= 2\left\{\frac{n}{n+1} + \frac{(n-1)}{n} + \frac{(n-2)}{n-1} + \cdots\cdots + \frac{3}{4} + \frac{2}{3}\right\}$$

$$= 2\left\{\frac{2}{3} + \frac{3}{4} + \cdots\cdots + \frac{(n-1)}{n} + \frac{n}{n+1}\right\}$$

$$\Rightarrow 2\sum_{i=3}^{n+1} \frac{1}{i} \qquad \left\{\sum_{i=3}^{n+1} \frac{1}{i} = \int_{2}^{n+1} \frac{1}{i}\, di\right\}$$

$$= 2\int_{2}^{n+1} \frac{1}{i}\, di$$

{lower bound decreases by 1 but upper bound remains unchanged}

$$\Rightarrow 2\left[\log_e i\right]_{2}^{(n+1)}$$

$$= 2\left\{\log_e(n+1) - \log_e 2\right\}$$

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

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

$$T(n) \leq \frac{2(n+1)}{n}\left\{\log_e(n+1)\right\}$$

$$T(n) \leq n\log_e(n+1)$$

$$T(n) \leq n\log_e n$$

$$\boxed{T(n) \in O(n\log_e n)}$$

method-II

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

devide by n

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

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

$$= \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \frac{T(n-3)}{n-2} \Rightarrow \frac{2}{3} + \frac{T(0)}{1}$$

$$\vdots$$

$$= \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \cdots\cdots + \frac{2}{3} + \cancel{\frac{T(0)}{1}}$$

$$= \frac{2}{3} + \frac{2}{4} + \frac{2}{5} + \cdots\cdots + \frac{2}{n-1} + \frac{2}{n} + \frac{2}{n+1}$$

$$= 2\left(\frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots\cdots + \frac{1}{n-1} + \frac{1}{n} + \frac{1}{n+1}\right)$$

Page No.
Date :

$$\frac{T(n)}{n+1} = 2\sum_{i=3}^{n+1} \frac{1}{k}$$

$$\frac{T(n)}{n+1} = 2\int_{2}^{n+1} \frac{1}{k}\,dk$$

$$= 2\left[\log_e k\right]_2^{n+1}$$

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

$$T(n) = 2(n+1)\left\{\log_e k(n+1) - \log_e 2\right\}$$

$$T(n) \leq cn\log_e n$$

$$T(n) \in O(n\log_e n)$$

* Multiplication of 2 n bit numbers using Devide and conquer

$$X = \boxed{A2^{n/2} \mid B}$$
$$Y = \boxed{C2^{n/2} \mid D}$$

$$Z = X * Y$$

$$Z = (A * 2^{n/2} + B) * (C * 2^{n/2} + D)$$

$$Z = AC(2)^{n} + AD\,2^{n/2} + BC\,2^{n/2} + BD$$

$$\boxed{Z = AC(2)^{n} + 2^{n/2}(AD + BC) + B.D}$$

$$V = AC$$
$$W = BD$$

$$AD + BC = U - V - W$$

$$U = (A + B) * (C + D)$$

$$U = AC + AD + BC + BD$$

$$AD + BC = U - V - W$$

$$AD + BC = (AC + AD + BC + BD) - AC - BD$$

$$AD + BC = \cancel{AC} + AD + BC + \cancel{BD} - \cancel{AC} - \cancel{BD}$$

$$AD + BC = AD + BC$$

$$Z = AC * 2^n + (AD + BC) * 2^{n/2} + BD$$

$$Z = V * 2^n + (U - V - W) * 2^{n/2} + W$$

addition, subtraction $= kn$

$3T\left(\frac{n}{2}\right) = U, V, W$

$$T(n) = \begin{cases} 3T\left(\frac{n}{2}\right) + kn & ; n \geq 2 \\ k & , n = 1 \end{cases}$$

$$T(n) = 3\left\{3T\left(\frac{n}{4}\right) + \frac{kn}{2}\right\} + kn$$

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

$$T(n) = 3^2\left\{3T\left(\frac{n}{8}\right) + \frac{kn}{4}\right\} + \frac{3}{2}kn + kn$$

$$T(n) = 3^3 T\left(\frac{n}{8}\right) + \left(\frac{3}{2}\right)^2 kn + \frac{3}{2}kn + kn$$

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

$\vdots$

p times

$$T(n) = 3^p T\left(\frac{n}{2^p}\right) + \left(\frac{3}{2}\right)^{p-1} kn + \cdots + \left(\frac{3}{2}\right)^2 kn + \frac{3}{2}kn +$$

Page No.
Date :

$$T(n) = 3^p T\left(\frac{n}{2^p}\right) + kn\left[\left(\frac{3}{2}\right)^{p-1} + \cdots\cdots + \left(\frac{3}{2}\right)^2 + \left(\frac{3}{2}\right) + 1\right]$$

$$\left\{\text{GP series } \begin{matrix} a = 1 \\ r = \frac{3}{2} \end{matrix}, n = p \quad \text{GP} = \frac{a(r^n - 1)}{r - 1}\right\}$$

$$T(n) = 3^p T\left(\frac{n}{2^p}\right) + kn\left\{\frac{\left(\frac{3}{2}\right)^p - 1}{\frac{3}{2} - 1}\right\}$$

$$T(n) = 3^p T\left(\frac{n}{2^p}\right) + kn\left\{\frac{\left(\frac{3}{2}\right)^p - 1}{\frac{1}{2}}\right\}$$

$$T(n) = 3^p T\left(\frac{n}{2^p}\right) + 2kn\left[\left(\frac{3}{2}\right)^p - 1\right] \qquad \left\{\begin{matrix} n = 2^p \\ p = \log_2 n \end{matrix}\right\}$$

$$T(n) = 3^p T\left(\frac{\cancel{2^p}}{\cancel{2^p}}\right) + 2kn\left[\left(\frac{3}{2}\right)^p - 1\right] \qquad T(1) = k$$

$$T(n) = 3^p T(1) + 2kn\left[\left(\frac{3}{2}\right)^p - 1\right]$$

$$T(n) = 3^p k + 2kn\left[\left(\frac{3}{2}\right)^p - 1\right]$$

$$T(n) = 3^p k + 2kn\left[\frac{3^p}{2^p} - 1\right]$$

$$= 3^{\log_2 n} k + 2kn \left[\frac{3^{\log_2 n}}{2^{\log_2 n}} - 1 \right]$$

$$= n^{\log_2 3} * k + 2kn \left\{ \frac{n^{\log_2 3}}{n^{\log_2 2}} - 1 \right\}$$

$$= n^{\log_2 3} * k + 2kn \left\{ \frac{n^{\log_2 3}}{n(1)} - 1 \right\}$$

$$= n^{\log_2 3} * k + 2kn \left\{ \frac{n^{\log_2 3}}{n} - 1 \right\}$$

$$= n^{\log_2 3} * k + 2k\cancel{n} * \frac{n^{\log_2 3}}{\cancel{n}} - 2kn$$

$$= n^{\log_2 3} * k + 2kn^{\log_2 3} - 2kn$$

$$= 3kn^{\log_2 3} - 2kn$$

$$T(n) \leq c\, n^{\log_2 3}$$

$$T(n) \in O(n^{\log_2 3}) \qquad \{\log_2 3 = 1.59\}$$

$$T(n) \in O(n^{1.59})$$

$X = 1011$ (4 bits, $n=4$)
$Y = 1100$

X	10 (A)	11 (B)
Y	11 (C)	00 (D)

$$Z = AC * 2^n + (AD + BC) 2^{n/2} + BD$$

$$Z = V * 2^n + (U - V - W) 2^{n/2} + W$$

$V = AC = 10 * 11 \Rightarrow 110$
$W = BD = 11 * 00 \Rightarrow 000$

$U = (A+B) * (C+D)$

$U = (10+11) * (11+00) \Rightarrow (101) * (11) \Rightarrow 1111$

$Z = 110 * 2^4 (1111 - 110 - 000) 2^2 + 000$

$Z = 110 * 10000 (1111 - 110 - 000) 100 + 000$

$Z = 110 * 10000 + (1001) * 100 + 000$

$Z = 1100000 + 100100 + 000$

$Z = 1000100$ Ans.

Strassen's Matrix Multiplication :-

Formula:

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

$$E = (A_{21} + A_{22}) B_{11}$$

$$F = [A_{11}(B_{12} - B_{22})]$$

$$G = A_{22}(B_{21} - B_{11})$$

$$H = (A_{11} + A_{12}) B_{22}$$

$$m = (A_{21} - A_{11})(B_{11} + B_{12})$$

$$P = (A_{12} - A_{22})(B_{21} + B_{22})$$

$$C_{11} = D + G - H + P$$

$$C_{12} = F + H$$

$$C_{21} = E + G$$

$$C_{22} = D + F - E + m$$

Page No.
Date :

$$A = \begin{array}{c} \begin{array}{cc} A_{11} & A_{12} \end{array} \\ \left[\begin{array}{cc|cc} 4 & 2 & 0 & 1 \\ 3 & 1 & 2 & 5 \\ \hline 3 & 2 & 1 & 4 \\ 5 & 2 & 6 & 7 \end{array}\right] \\ \begin{array}{cc} A_{21} & A_{22} \end{array} \end{array} \qquad B = \begin{array}{c} \begin{array}{cc} B_{11} & B_{12} \end{array} \\ \left[\begin{array}{cc|cc} 2 & 1 & 3 & 2 \\ 5 & 4 & 2 & 3 \\ \hline 1 & 4 & 0 & 2 \\ 8 & 2 & 4 & 1 \end{array}\right] \\ \begin{array}{cc} B_{21} & B_{22} \end{array} \end{array}$$

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

$$\Rightarrow \left(\begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 4 \\ 6 & 7 \end{bmatrix} \right) \left(\begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix} + \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix} \right)$$

$$\Rightarrow \begin{bmatrix} 5 & 6 \\ 9 & 8 \end{bmatrix} \times \begin{bmatrix} 2 & 3 \\ 9 & 5 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 10+54 & 15+30 \\ 18+72 & 27+40 \end{bmatrix}$$

$$D = \begin{bmatrix} 64 & 45 \\ 90 & 67 \end{bmatrix}$$

$$E = \left(\begin{bmatrix} 3 & 2 \\ 5 & 2 \end{bmatrix} + \begin{bmatrix} 1 & 4 \\ 6 & 7 \end{bmatrix} \right) \times \begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix}$$

$$E = \begin{bmatrix} 4 & 6 \\ 11 & 9 \end{bmatrix} \times \begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix}$$

$$E = \begin{bmatrix} 8+30 & 4+24 \\ 22+45 & 11+36 \end{bmatrix} \Rightarrow \begin{bmatrix} 38 & 2 \\ 67 & 4 \end{bmatrix}$$

$$F = \begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix} * \left(\begin{bmatrix} 8 & 2 \\ 2 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix} \right)$$

$$F = \begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix} * \begin{bmatrix} 3 & 0 \\ -2 & 2 \end{bmatrix}$$

$$F = \begin{bmatrix} 12-4 & 0+4 \\ 9-2 & 0+2 \end{bmatrix}$$

$$F = \begin{bmatrix} 8 & 4 \\ 7 & 2 \end{bmatrix}$$

$$G = \begin{bmatrix} 1 & 4 \\ 6 & 7 \end{bmatrix} \left(\begin{bmatrix} 1 & 4 \\ 3 & 2 \end{bmatrix} - \begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix} \right)$$

$$G = \begin{bmatrix} 1 & 4 \\ 6 & 7 \end{bmatrix} * \left(\begin{bmatrix} -1 & 3 \\ -2 & -2 \end{bmatrix} \right)$$

$$G = \begin{bmatrix} -1-8 & 3-8 \\ -6-14 & 18-14 \end{bmatrix}$$

$$G = \begin{bmatrix} -9 & -5 \\ -20 & 4 \end{bmatrix}$$

$$H = (A_{11} + A_{12})\, B_{22}$$

$$H = \left(\begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 2 & 5 \end{bmatrix} \right) * \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 4 & 3 \\ 5 & 6 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 0+12 & 8+3 \\ 0+24 & 10+6 \end{bmatrix} \Rightarrow \begin{bmatrix} 12 & 11 \\ 24 & 16 \end{bmatrix}$$

$$m = (A_{21} - A_{11})(B_{11} + B_{12})$$

$$m = \left(\begin{bmatrix} 3 & 2 \\ 5 & 2 \end{bmatrix} - \begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix} \right) \left(\begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix} + \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} \right)$$

$$m = \begin{bmatrix} -1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 5 & 3 \\ 7 & 7 \end{bmatrix} \Rightarrow \begin{bmatrix} -5+0 & -3+0 \\ 10+7 & 6+7 \end{bmatrix} \Rightarrow \begin{bmatrix} -5 & -3 \\ 17 & 13 \end{bmatrix}$$

$$P = (A_{12} - A_{22})(B_{21} + B_{22})$$

$$P = \left(\begin{bmatrix} 0 & 1 \\ 2 & 5 \end{bmatrix} - \begin{bmatrix} 1 & 4 \\ 6 & 7 \end{bmatrix} \right) \left(\begin{bmatrix} 1 & 4 \\ 3 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 2 \\ 4 & 1 \end{bmatrix} \right)$$

$$P = \begin{bmatrix} -1 & -3 \\ -4 & -2 \end{bmatrix} * \begin{bmatrix} 1 & 6 \\ 7 & 3 \end{bmatrix} \Rightarrow \begin{bmatrix} -1-21 & -6-9 \\ -4-14 & -24-6 \end{bmatrix} \Rightarrow \begin{bmatrix} -22 & -15 \\ -18 & -30 \end{bmatrix}$$

$$C_{11} = D + M - H + P$$

$$C_{11} = \begin{bmatrix} 64 & 45 \\ 90 & 67 \end{bmatrix} + \begin{bmatrix} -9 & -5 \\ -20 & 4 \end{bmatrix} - \begin{bmatrix} 12 & 11 \\ 24 & 16 \end{bmatrix} + \begin{bmatrix} -22 & -15 \\ -18 & -30 \end{bmatrix}$$

$$C_{11} = \begin{bmatrix} 64 & 45 \\ 90 & 67 \end{bmatrix} + \begin{bmatrix} -43 & -31 \\ -62 & -42 \end{bmatrix}$$

$$C_{11} = \begin{bmatrix} 21 & 14 \\ 28 & 25 \end{bmatrix}$$

$C_{12} = F + H$

$$C_{12} = \begin{bmatrix} 8 & 4 \\ 7 & 2 \end{bmatrix} + \begin{bmatrix} 12 & 11 \\ 24 & 16 \end{bmatrix} \Rightarrow \begin{bmatrix} 20 & 15 \\ 31 & 18 \end{bmatrix}$$

$C_{21} = E + G$

$$C_{21} = \begin{bmatrix} 38 & 28 \\ 67 & 47 \end{bmatrix} + \begin{bmatrix} -9 & -5 \\ -20 & 4 \end{bmatrix} \Rightarrow \begin{bmatrix} 29 & 23 \\ 47 & 51 \end{bmatrix}$$

$C_{22} = D + F - E + M$

$$C_{22} = \begin{bmatrix} 64 & 45 \\ 90 & 67 \end{bmatrix} + \begin{bmatrix} 8 & 4 \\ 7 & 2 \end{bmatrix} - \begin{bmatrix} 38 & 28 \\ 67 & 47 \end{bmatrix} + \begin{bmatrix} -5 & -7 \\ 17 & 13 \end{bmatrix}$$

$$C_{22} = \begin{bmatrix} 72 & 49 \\ 97 & 69 \end{bmatrix} - \begin{bmatrix} +43 & +35 \\ +50 & +34 \end{bmatrix}$$

$$C_{22} = \begin{bmatrix} 29 & 18 \\ 47 & 35 \end{bmatrix}$$

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} 21 & 14 & 20 & 15 \\ 28 & 25 & 31 & 18 \\ 29 & 23 & 29 & 18 \\ 47 & 51 & 47 & 35 \end{bmatrix}$$

Ans,

3-10-2017

8

Recurrence Relation for strassen's Multiplication.

$$T(n) = 7T\left(\frac{n}{2}\right) + bn^2, \quad n \geq 2$$

$$a \text{ ; otherwise}$$

$$T\left(\frac{n}{2}\right) \Rightarrow 7T\left(\frac{n}{4}\right) + b\left(\frac{n}{2}\right)^2$$

$$T\left(\frac{n}{4}\right) \Rightarrow 7T\left(\frac{n}{8}\right) + b\left(\frac{n}{4}\right)^2$$

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

$$T(n) = 7\left[7T\left(\frac{n}{4}\right) + b\left(\frac{n}{2}\right)^2 + bn^2\right]$$

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

$$T(n) = 7^2\left[7T\left(\frac{n}{8}\right) + b\left(\frac{n}{4}\right)^2\right] + \frac{7bn^2}{4} + bn^2$$

$$T(n) = 7^3 T\left(\frac{n}{8}\right) + \left(\frac{7}{4}\right)^2 bn^2 + \frac{7bn^2}{4} + bn^2$$

$$= 7^3 T\left(\frac{n}{8}\right) + \left(\frac{7}{4}\right)^2 bn^2 + \frac{7}{4} bn^2 + bn^2$$

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

$$\vdots$$

k times

$$= 7^k T\left(\frac{n}{2^k}\right) + \left(\frac{7}{4}\right)^{k-1} bn^2 + \left(\frac{7}{4}\right)^{k-2} bn^2 + \cdots + \left(\frac{7}{4}\right)^2 bn^2 + \frac{7}{4} bn^2 + bn^2$$

$$= 7^k T\left(\frac{n}{2^k}\right) + bn^2 \left[\left(\frac{7}{4}\right)^{k-1} + \left(\frac{7}{4}\right)^{k-2} + \cdots + \left(\frac{7}{4}\right)^2 + \frac{7}{4} + 1\right]$$

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

$$\Rightarrow 7^k T\left(\frac{n}{2^k}\right) + bn^2 \left[\frac{(7/4)^k - 1}{3/4}\right]$$

$$= 7^k T\left(\frac{n}{2^k}\right) + \frac{4}{3} bn^2 \left[\left(\frac{7}{4}\right)^k - 1\right]$$

$$= 7^k T\left(\frac{\cancel{2^k}^{\,1}}{\cancel{2^k}}\right) + \frac{4}{3} bn^2 \left[\left(\frac{7}{4}\right)^k - 1\right]$$

$$= 7^k T(1) + \frac{4}{3} bn^2 \left[\left(\frac{7}{4}\right)^k - 1\right]$$

$$= 7^k a + \frac{4}{3} bn^2 \left[\left(\frac{7}{4}\right)^k - 1\right]$$

$$\Rightarrow a\, 7^{\log_2 n} + \frac{4}{3} bn^2 \left[\left(\frac{7}{4}\right)^{\log_2 n} - 1\right]$$

Date :

$$= a\,7^{\log_2 n} + \frac{4}{3} bn^2 \left[\frac{7^{\log_2 n}}{4^{\log_2 n}} - 1 \right]$$

$$= na^{\log_2 7} + \frac{4}{3} bn^2 \left[\frac{n^{\log_2 7}}{n^{\log_2 4}} - 1 \right]$$

$$= na^{\log_2 7} + \frac{4}{3} bn^2 \cdot \frac{n^{\log_2 7}}{n^{\log_2 4}} - \frac{4}{3} bn^2$$

$$= na^{\log_2 7} + \frac{4}{3} b\cancel{n^2} \cdot \frac{n^{\log_2 7}}{\cancel{n^2}} - \frac{4}{3} bn^2$$

$$= na^{\log_2 7} + \frac{4}{3} b\, n^{\log_2 7} - \frac{4}{3} bn^2$$

$$= n^{\log_2 7} \left(a + \frac{4}{3} b \right) - \frac{4}{3} bn^2$$

$$T(n) \le n^{\log_2 7} \left(a + \frac{4}{3} b \right)$$

$$T(n) \le C\, n^{\log_2 7}$$

$$T(n) \in O(n^{\log_2 7})$$

$$\boxed{T(n) \in O(n^{2.81})}$$

Date :

4-10-2017

The Greedy Method

1. Knapsack problem
2. minimum cost spanning tree
3. Travelling salesman problem

Greedy Method :-

```
procedure greedy (C)
begin
    S ← φ
while not solution (S) and C ≠ φ
begin
    select x ∈ C according to criteria function
    C ← C - {x}
if feasible (S ∪ {x}) then
    S ← S ∪ {x}
end if
end
    if solution (S) found then
    return(S);
else
```

return solution not found
endif
end

KnapSack Problem :-

We are given n number of objects with there associated profit and weight. We have algo given knapsack which is having a capacity M. Our objective is put the items (objects) in the knapsack ~~th~~ such that the sum of the weights of items should not exceed the capacity of the knapsack (m) such that by maximizing the profit.

To Maximize $\sum_{i=1}^{n} P_i x_i$

Subject to the condition $\sum_{i=1}^{n} w_i x_i \leq M$

and $0 \leq x_i \leq 1$ or $x_i = \{0, 1\}$

Problem 1 :- Consider the following knapsack problem with

$n = 3$, $M = 25$, $P = \{25, 24, 17\}$, $W = \{16, 14, 9\}$

Case I :- ↑ of W

$\frac{25}{16} = 1.5$

$\frac{24}{14} = 1.2$

$\frac{17}{9} = 1.9$

$3 \rightarrow 2 \rightarrow 1$

Case II :- Origional sequence

$1 \rightarrow 2 \rightarrow 3$

Case III :- ↓ of P

$1 \rightarrow 2 \rightarrow 3$

Case IV :- ↓ of P/W_t

$3 \rightarrow 2 \rightarrow 1$

Case I → $3 \rightarrow 2 \rightarrow 1$

item to be considered	x_i	$W_i x_i$	$P_i x_i$	RC	$W_i x_i \leq RC$
3	1	9	17	25	$9 \leq 25$ T
2	1	$9 + 14 = 23$	$17 + 24 = 41$	16	$14 \leq 16$ T
1	1/8	$23 + 16 \times \frac{1}{8} = 25$	$41 + \cancel{25} \times \frac{1}{8}$ 3.12 $= 44.12$	2	$16 \nleq 2$ F

$\therefore$ solution vector $= (x_1, x_2, x_3)$

$= \left(\frac{1}{8}, 1, 1\right)$ $\bar{c}$ profit (44.12)

Case II :- $1 \to 2 \to 3$

item to be considered	x_i	$w_i x_i$	$p_i x_i$	RC	$w_i x_i \leq RC$
1	1	16	25	25	$16 \leq 25$ T
2	$\frac{9}{14}$	$16 + \cancel{14}^{\cancel{14}} \times \frac{9}{\cancel{14}}$ $= 25$	$25 + \cancel{24}^{12} \times \frac{9}{\cancel{14}_{7}}$ $25 + \cancel{\frac{108}{7}}$ 15.42 $= 40.42$	9	$14 \nleq 9$ F
3	0	$25 + 0 = 25$	40.42	0	$9 \nleq 0$

$\therefore$ solution vector $= (x_1, x_2, x_3)$

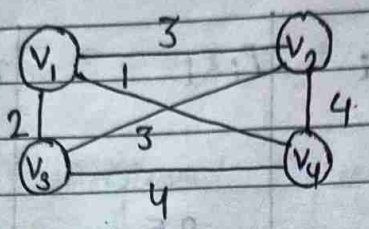
$= \left(1, \frac{9}{14}, 0\right)$ $\bar{c}$ profit $\boxed{40.42}$

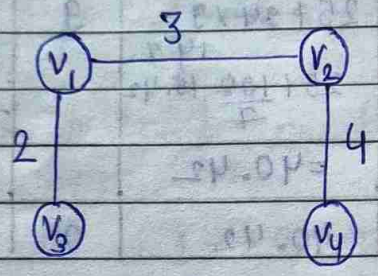
Final optimal solution of given knapsack problem is 44.12 $(3 \to 2 \to 1)$

9-10-2017

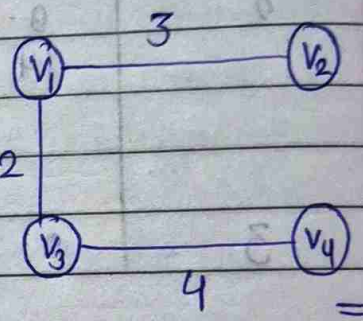
Minimum Cost Spanning Tree :-

1. Kruskal's Algorithm
2. Prim's Algorithm

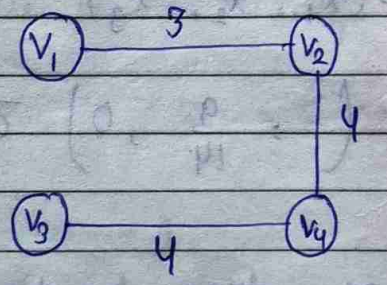


1.


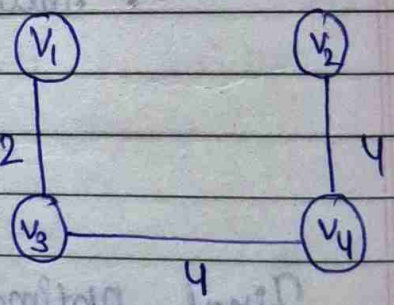
= 9

2.


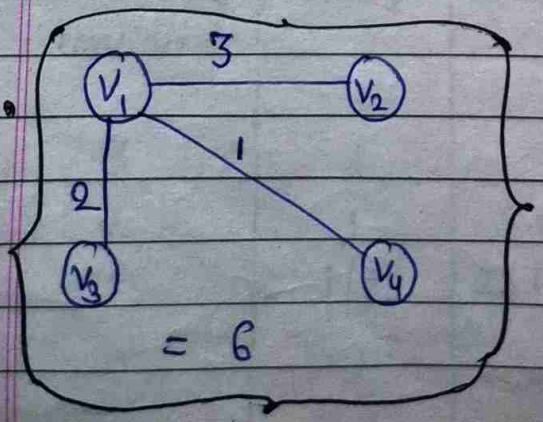
= 9

3.


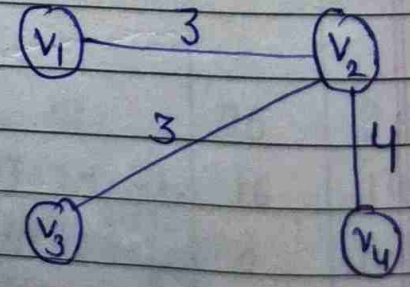
= 11

4.


= 10

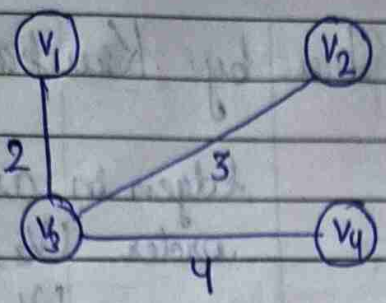
5.


= 6

6.


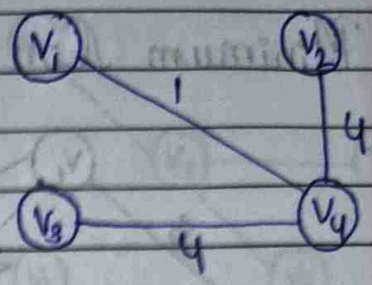
= 10

7.



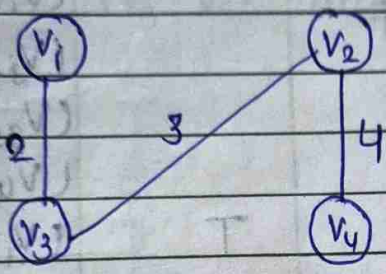
$= 9$

8.



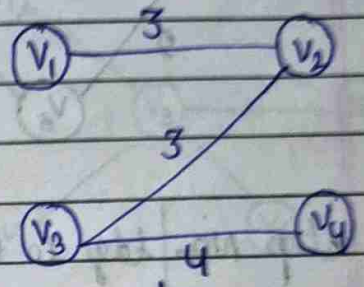
$= 9$

9.



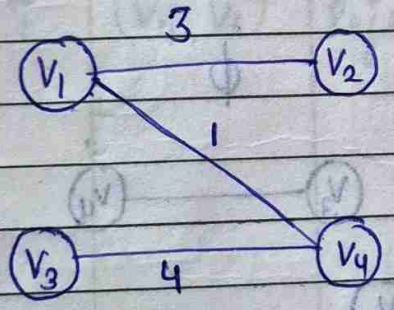
$= 9$

10.



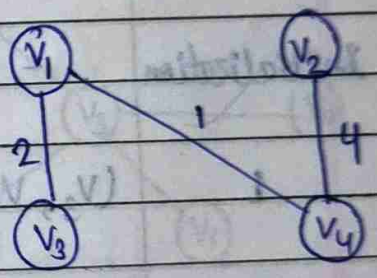
$= 10$

11.



$= 8$

12.

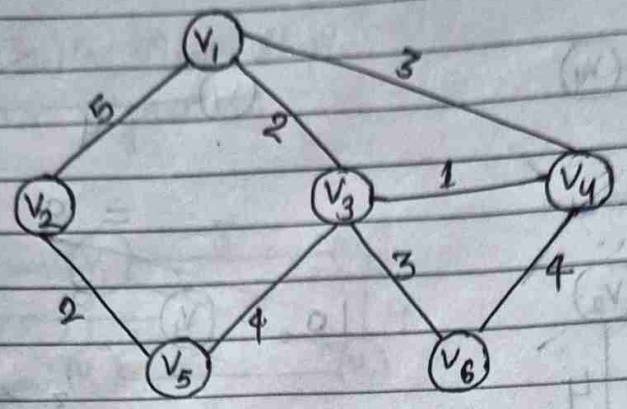


$= 7$

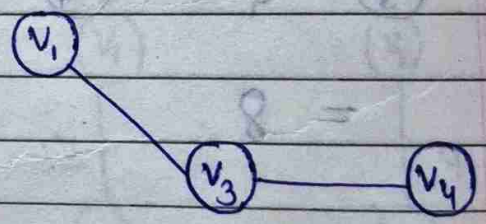
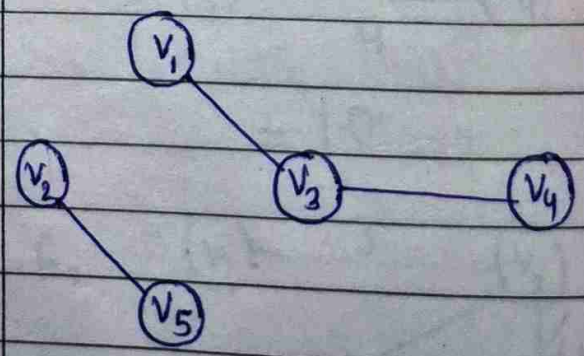
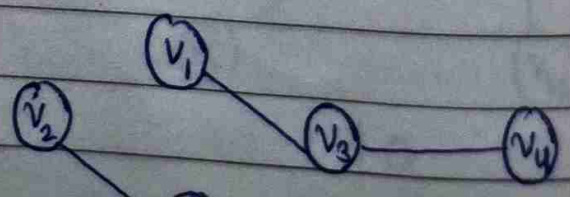
Date :

11-10-2017

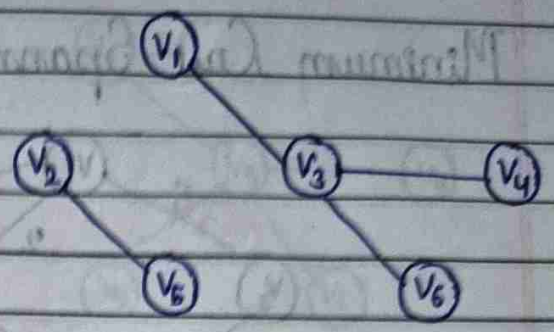
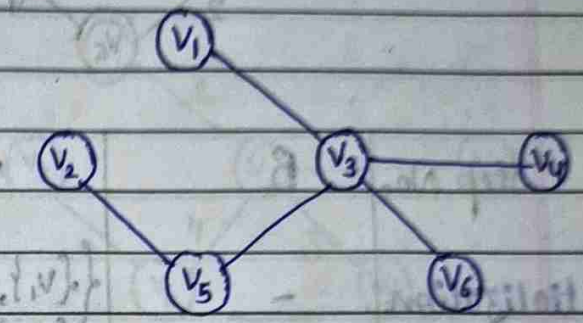
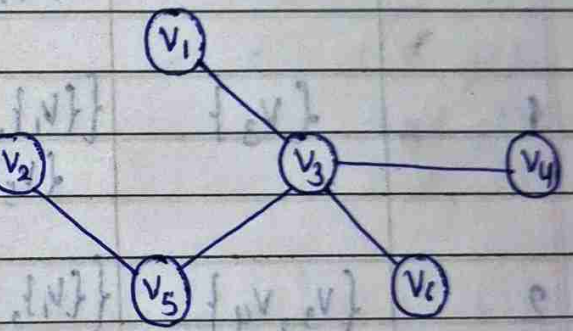
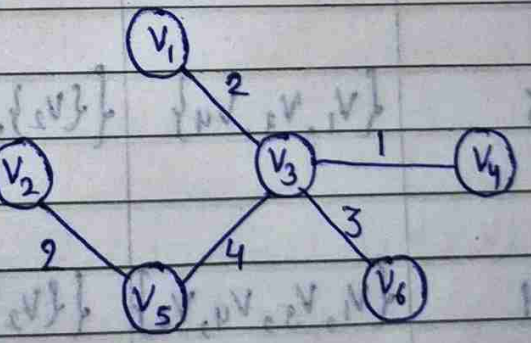
Minimum Cost Spanning Tree by Kruskal's Algo-



Edges in ascending order
(V_3, V_4)
(V_1, V_3)
(V_2, V_5)
(V_1, V_4)
(V_3, V_6)
(V_3, V_5)
(V_4, V_6)
(V_1, V_2)

step No.	Edge to be considered	comment	T
Initialization	-	-	ϕ
1	(V_3, V_4)	Accept	
2	(V_1, V_3)	accept	
3	(V_2, V_5)	accept	
4	(V_1, V_4)	Reject	

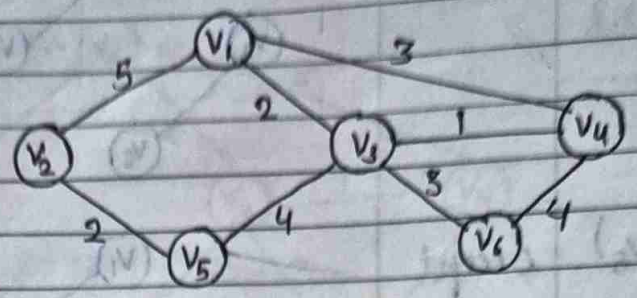
Date :

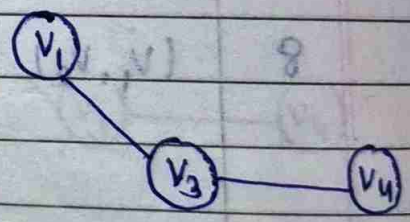
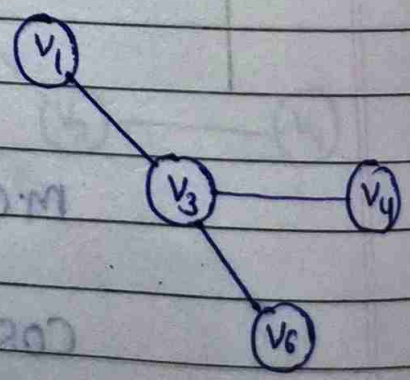
5	(V_3, V_6)	accept	
6	(V_3, V_5)	accept	
7	(V_4, V_6)	Reject	
8	(V_1, V_2)	Reject	

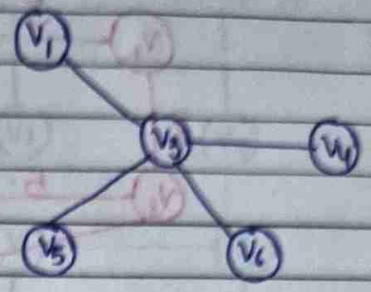
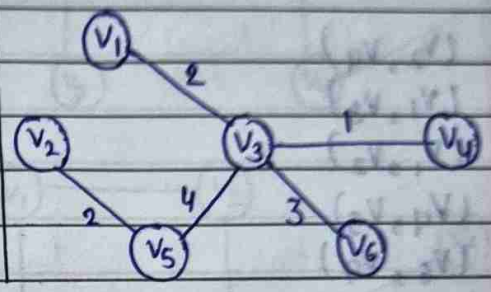
M.CST = 12

COST = 12

Minimum Cost Spanning Tree By Prim's Algo →

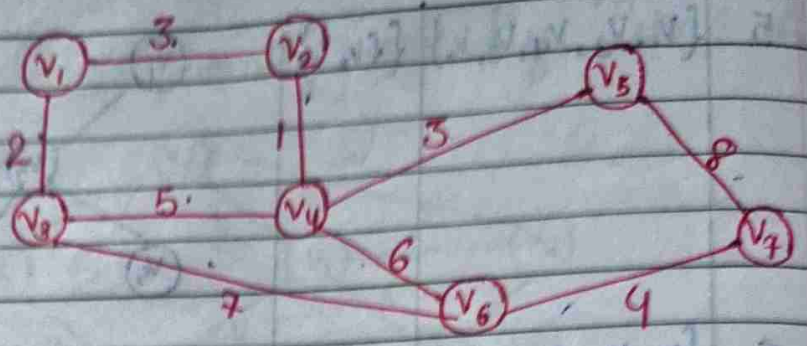


Step No.	B	N	T
initialization	–	$\{\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}\}$	ϕ
1	$\{v_3\}$	$\{\{v_1\}, \{v_2\}, \{v_4\}, \{v_5\}, \{v_6\}\}$	
2	$\{v_3, v_4\}$	$\{\{v_1\}, \{v_2\}, \{v_5\}, \{v_6\}\}$	
3	$\{v_1, v_3, v_4\}$	$\{\{v_2\}, \{v_5\}, \{v_6\}\}$	
4	$\{v_1, v_3, v_4, v_6\}$	$\{\{v_2\}, \{v_5\}\}$	

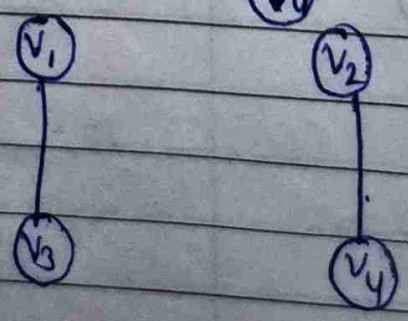
5	$\{v_1, v_3, v_4, v_5, v_6\}$	$\{\{v_1\}\}$	
6	$\{v_1, v_2, v_3, v_4, v_5, v_6\}$	ϕ	

cost = 12.

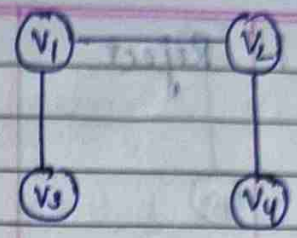
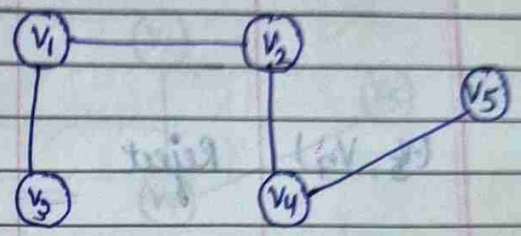
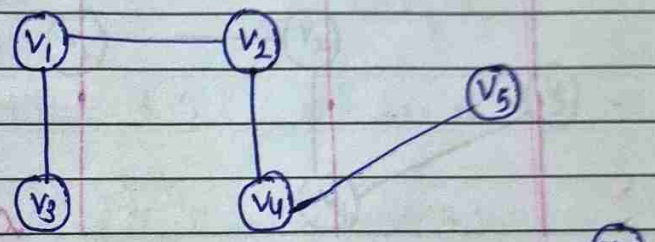
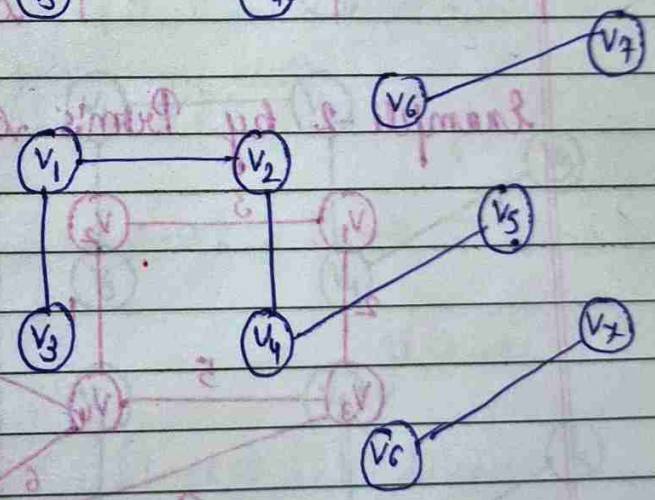
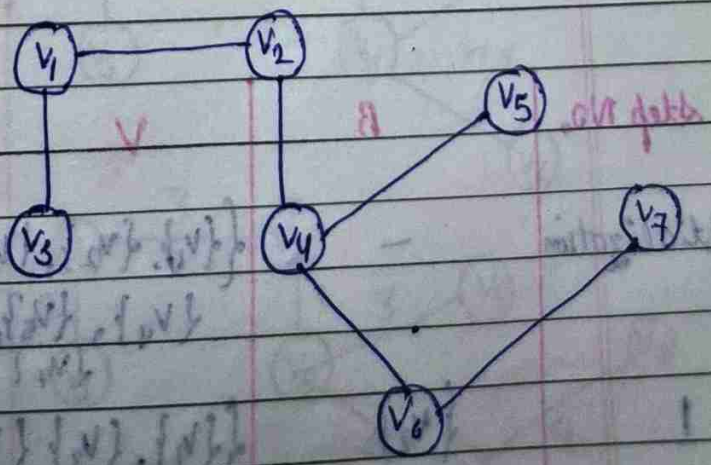
Example-2



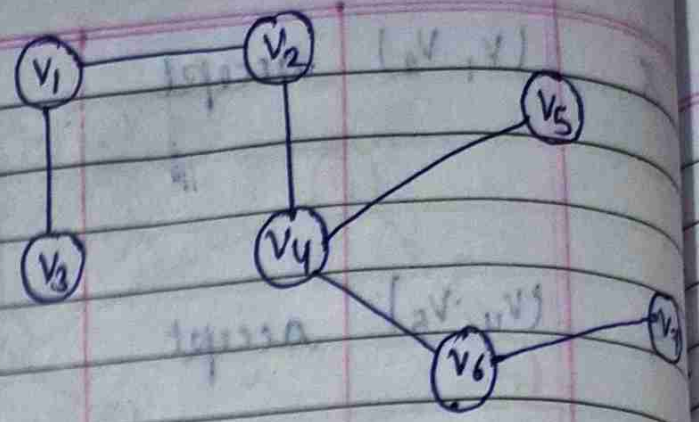
(v_2, v_4)
(v_1, v_3)
(v_1, v_2)
(v_4, v_5)
(v_6, v_7)
(v_3, v_4)
(v_3, v_6)
(v_3, v_7)
(v_5, v_7)

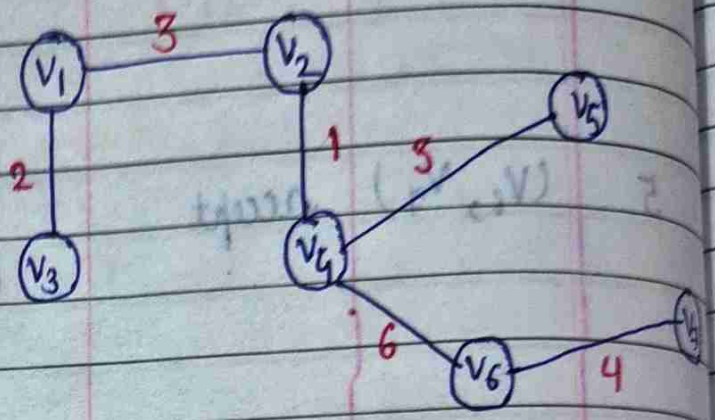
Step No.	Edge to be considered	comment	T
initialization	-	-	ϕ
1	(v_2, v_4)	accept	
2	(v_1, v_3)	accept	

Page No.
Date :

3	(v_1, v_2)	accept	
4	(v_4, v_5)	accept	
5	(v_6, v_7)	accept	
6	(v_3, v_4)	Reject	
7	(v_4, v_6)	accept	

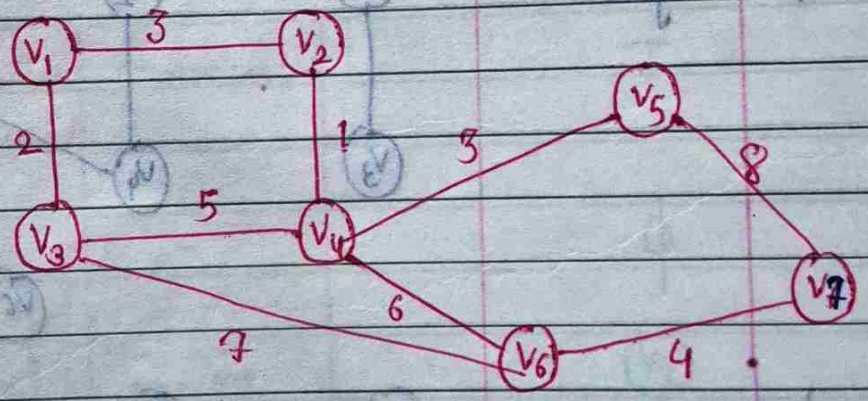
8	(V_3, V_6)	Reject
9	(V_5, V_7)	Reject



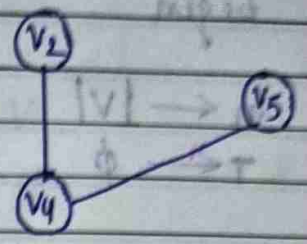
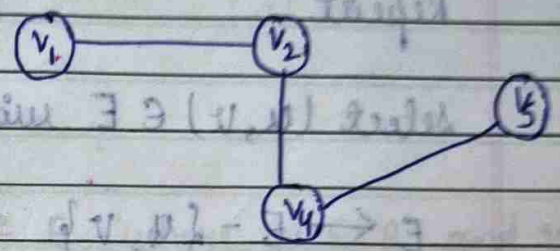
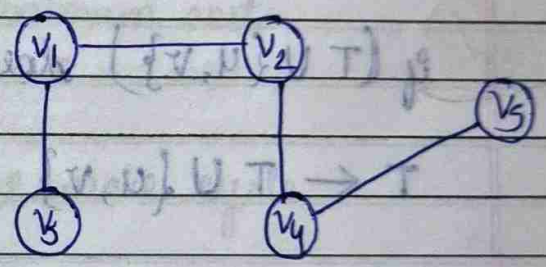
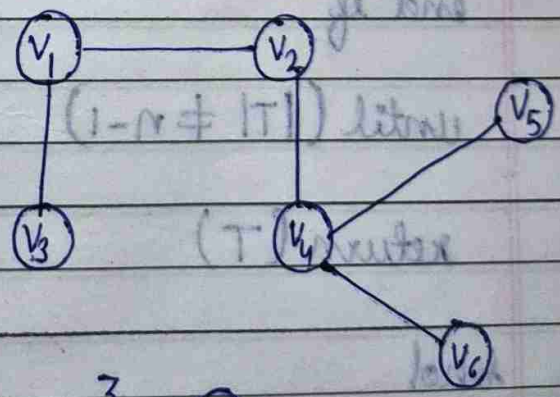
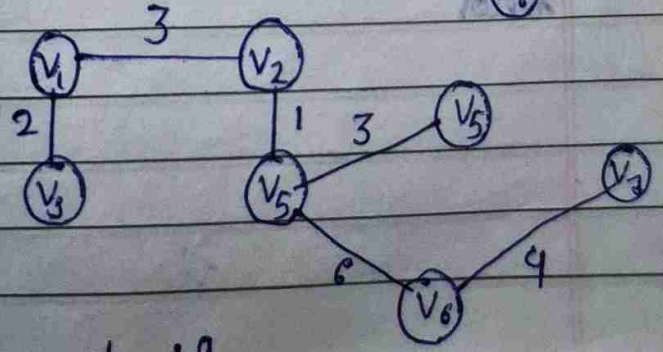


cost = 19

Example-2 by Prim's Algo



Step No.	B	V	T
initialization	–	$\{\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7\}\}$	ϕ
1	$\{v_2\}$	$\{\{v_1\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7\}\}$	v_2

2	$\{v_2, v_4\}$	$\{\{v_1\},\{v_3\},\{v_5\},\{v_6\}\}$ $\{v_7\}$	
3	$\{v_2, v_4, v_5\}$	$\{\{v_1\},\{v_3\},\{v_6\}\}$ $\{v_7\}$	
4	$\{v_2, v_3, v_4, v_5\}$	$\{\{v_1\},\{v_6\}\}$ $\{v_7\}$	
5.	$\{v_1, v_2, v_3, v_4, v_5\}$	$\{\{v_6\},\{v_7\}\}$	
6.	$\{v_1, v_2, v_3, v_4, v_5, v_6\}$	$\{v_7\}$	
7.	$\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$	ϕ	 cost = 19

23-10-2017

Page No.
Date :

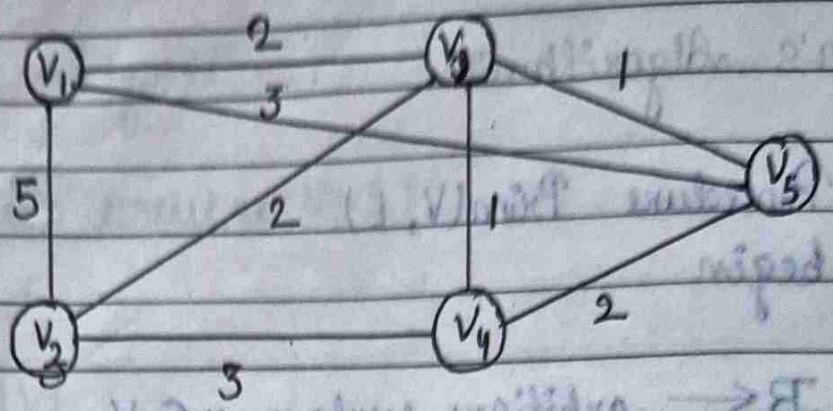
Kruskal's Algorithm :-

```
procedure kruskal(V, E)
begin
n ← |V|
T ← ϕ

Repeat
select (u, v) ∈ E with minimum cost
  E ← E - {u, v}
  if (T ∪ {u, v}) does not make cycle
    T ← T ∪ {u, v}
  end if
until (|T| ≠ n-1)
return (T)
end
```

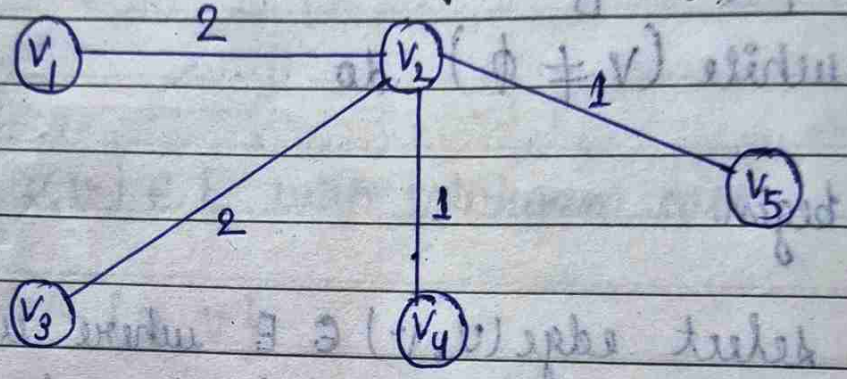
Page No.
Date :

Prim's Algorithm:-

```
Procedure Prim(V, E)
begin

B ← arbitiary vertex v ∈ V

T ← B
while (V ≠ ϕ) do

begin

select edge (u, v) ∈ E where u ∈ B and v ∈ V
                with minimum cost
  E ← E - {u, v}

if (T U v) does not make cycle

  T ← T U {v}
  B ← B U {v}
else V ← V - {v}
   Reject (u, v)
endif

end
   return (T)

end
```



Minimum cost spanning tree by kruskals algo:-



cost = 6

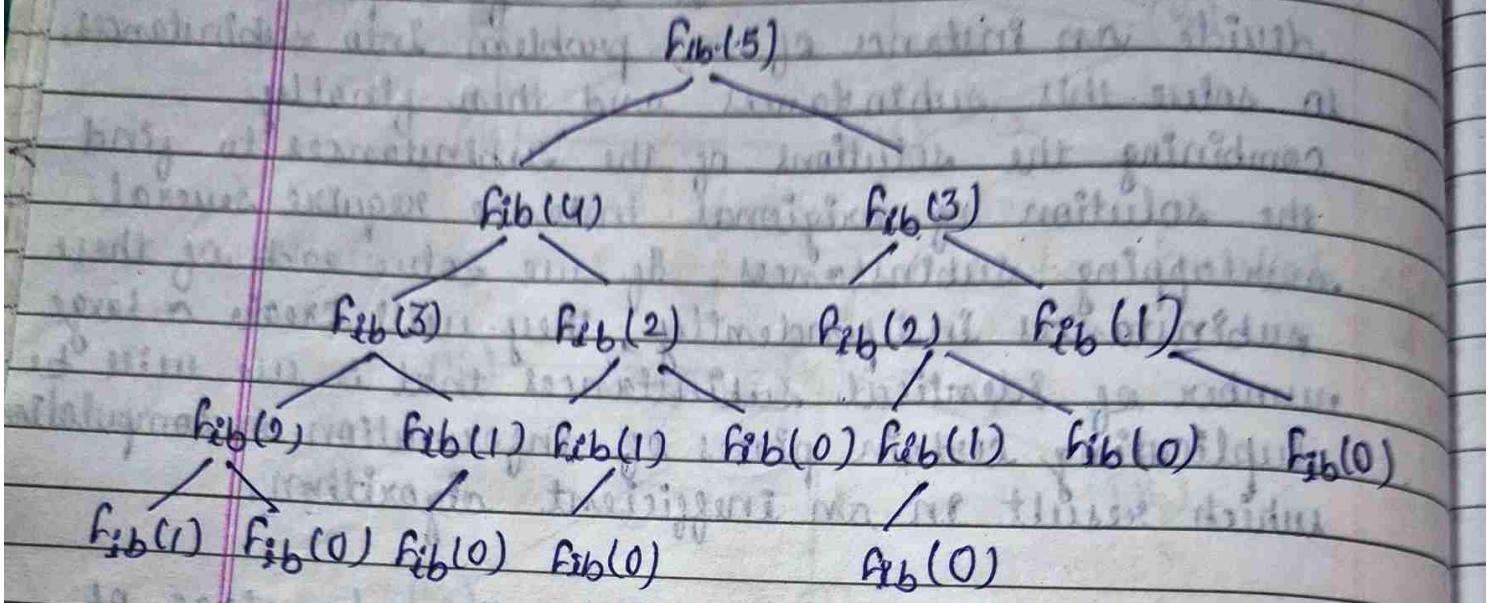
Dynamic Programming :-

It is often possible to divide an instance of a problem into subinstan to solve this subinstances and then finally combining the solutions of the subinstances to fin the solution of origional instance require several overlapping subinstances. If we solve each of the subinstances independently, they will create a lar number of identical subinstances that is we will duplicating certain pieces of information or compute which result in an inefficient algorithm.

On the other hand, if we take the advantage of duplication and solve each subinstance only once by saving the solution for later use, then a more efficient algorithm will result.

The underline idea of dynamic programming is t avoid calculating the same thing twice, usually by keping a table of already known results, which we fill up as the subinstances are solved.

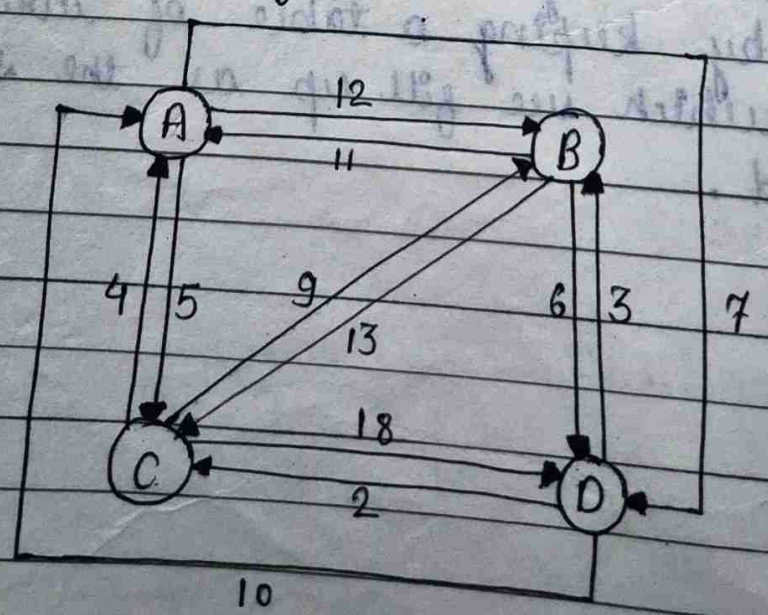
Example :- Fibonacci series



(iv) Principle of Optimality :-

This states that in a optimal sequence of decisions or choices, each subsequence must also be optimal.

TSP (Travelling Salesman Problem) :-



Page No.
Date :

(13)

$$\begin{array}{c|cccc} & A & B & C & D \\ \hline A & \infty & 12 & 5 & 7 \\ B & 11 & \infty & 13 & 6 \\ C & 4 & 9 & \infty & 18 \\ D & 10 & 3 & 2 & \infty \end{array}$$

* Formula *

$$\boxed{Sp(i, S) = \min_{j \in S} \{ W_{ij} + Sp(j, S - \{j\}) \}}$$

Sp = Shortest path

i = current vertex

S = vertex other than i

Start the solution with by assuming the size o
Set S empty (that is size is 0)

$S = \phi$

$sp(i, S) =$

$Sp(B, \phi) = W_{BA} = 11$

$Sp(C, \phi) = W_{CA} = 4$

$sp(D, \phi) = W_{DA} = 10$

S = Single element

$Sp(i, S)$

$Sp(A, \{B\}) = min\{W_{AB} + Sp(B, \phi)\}$

$= min\{12+11\} = 23$

$Sp(C, \{B\}) = min\{W_{CB} + Sp(B, \phi)\} =$

$= min(9+11) = 20$

$Sp(D, \{B\}) = min\{W_{DB} + Sp(B, \phi)\}$

$= \cancel{min\{5+4\}} = \cancel{9}\ min(3+11) = 14$

$Sp(A, \{C\}) = min\{W_{AC} + Sp(C, \phi)\}$

$= min(5+4) = 9$

$Sp(B, \{C\}) = min\{W_{BC} + Sp(C, \phi)\}$

$= min(13+4) = 17$

$Sp(D, \{C\}) = min\{W_{DC} + Sp(C, \phi)\}$

$= min(2+4) = 6$

$Sp(A, \{D\}) = min\{W_{AD} + Sp(D, \phi)\}$

$= min(7+10) = 17$

$Sp(B, \{D\}) = min\{W_{BD} + Sp(D, \phi)\}$

$= min(6+10) = 16$

Page No.
Date :

$Sp(C, \{D\}) = \min\{W_{CD} + Sp(D, \phi)\}$
$= \min(18+10) = 28$

$S = 2$ elements

$Sp(i, S)$

$Sp(A, \{B, C\}) = \min\{W_{AB} + Sp(B, \{C\})\}, \min\{W_{AC} + Sp(C, \{B\})\}$
$= \min(12+17), \min(5+20)$
$= \min(29, 25) = 25$

$Sp(D, \{B, C\}) = \min\{W_{DB} + Sp(B, \{C\})\}, \min\{W_{DC} + Sp(C, \{B\})\}$
$= \min(3+17), \min(2+20)$
$= \min(20, 22) = 20$

$Sp(A, \{C, D\}) = \min\{W_{AC} + Sp(C, \{D\})\}, \min\{W_{AD} + Sp(D, \{C\})\}$
$= \min(5+28), \min(7+6)$
$= \min(33, 13) = 13$

$Sp(B, \{C, D\}) = \min\{W_{BC} + Sp(C, \{D\})\}, \min\{W_{BD} + Sp(D, \{C\})\}$
$= \min(13+28), \min(6+6)$
$= \min(41, 12) = 12$

$Sp(A, \{B, D\}) = \min\{W_{AB} + Sp(B, \{D\})\}, \min\{W_{AD} + Sp\{D, \{B\}\}\}$
$= \min(12+16), \min\{7+14\}$
$= \min(28, 21) = 21$

$S = 3$ elements

$Sp(i, S)$

$Sp(C, \{B, D\}) = \min\{W_{CB} + Sp(B, \{D\})\}, \min\{W_{CD} + Sp\{D, \{B\}\}\}$
$Sp = \min(9+16), \min(18, 14)$
$= \min(25, 32) = 25$

$$Sp(A, \{B, C, D\}) = \min\{W_{AB} + Sp(B, \{C, D\})\},$$
$$\min\{W_{AC} + Sp(C, \{B, D\})\},$$
$$\min\{W_{AD} + Sp(D, \{B, C\})\}$$

$$= \min\{12 + 12\}, \min\{5 + 25\}, \min\{7 + 20\}$$
$$= \min(24, 30, 27) \Rightarrow 24$$

$$A \rightarrow B \rightarrow D \rightarrow C \rightarrow A$$

0/1 Knapsack problem using Dynamic Programming

The problem is to select objects out of n number of objects such that

$\sum_{1 \le i \le n} p_i x_i$ is maximum subject to condition

$\sum_{1 \le i \le n} w_i x_i \le m$ where $[P1, P2, \ldots, Pn]$ is the profit vector and $[W1, W2, \ldots, Wn]$ is the weight vector.

Here x_i is either zero or one for all i in $1 \le i \le n$ and m is the capacity of the knapsack.

Date :

Formula :-

$$\boxed{f_i(y) = \max\{f_{i-1}(y),\ f_{i-1}(y - w_i) + p_i\}}$$

i = item to be considered

Q.1 $n = 3$

$(W_1, W_2, W_3) = (2, 3, 3)$

$(P_1, P_2, P_3) = (1, 2, 4)$, $m = 6$

$$f_i(y) = \max\{f_{i-1}(y),\ f_{i-1}(y - w_i) + p_i\}$$

assumptions :- 1. $f_0(y) = f_1(y) = f_2(y) = f_3(y) = -\infty \ \forall y <$

2. $f_0(y) = 0 \ \forall \ y \geq 0$

3. $\underline{f_1(0) = f_2(0) = f_3(0) = 0}$

$i = 1$, $y = 1$ to 6

$$f_1(1) = \max\{f_{1-1}(1),\ f_{1-1}(1-2) + 1\}$$

$$= \max\{f_0(1),\ f_0(-1) + 1\}$$

$$= \max\{0,\ -\infty + 1\} = \max\{0,\ -\infty\} = 0$$

$$f_1(2) = \max\{f_0(2),\ f_0(0) + 1\}$$

$$= \max\{0,\ 0 + 1\} = \max\{0,\ 1\} = 1$$

$f_1(3) = \max\{f_0(3), f_0(1)+1\}$
$= \max\{0, 0+1\} = \max\{0, 1\} = 1$

$f_1(4) = \max\{f_0(4), f_0(2)+1\}$
$= \max\{0, 0+1\} = \max\{0, 1\} = 1$

$f_1(5) = \max\{f_0(5), f_0(3)+1\}$
$= \max\{0, 0+1\} = \max\{0, 1\} = 1$

$f_1(6) = \max\{f_0(6), f_0(4)+1\}$
$= \max\{0, 0+1\} = \max\{0, 1\} = 1$

$i = 2, \; y = 1$ to 6

$f_2(1) = \max\{f_1(1), f_1(-2)+2\}$
$= \max\{0, -\infty+2\} = \max\{0, -\infty\} = 0$

$f_2(2) = \max\{f_1(2), f_1(-1)+2\}$
$= \max\{1, -\infty+2\} = \max\{1, -\infty\} = 1$

$f_2(3) = \max\{f_1(3), f_1(0)+2\}$
$= \max\{1, 0+2\} = \max\{1, 2\} = 2$

$f_2(4) = \max\{f_1(4), f_1(1)+2\}$
$= \max\{1, 0+2\} = \max\{1, 2\} = 2$

$f_2(5) = \max\{f_1(5), f_1(2)+2\}$
$= \max\{1, 1+2\} = \max\{1, 3\} = 3$

Page No.
Date :

$$f_2(6) = \max\{f_1(6) + f_1(3) + 2\}$$
$$= \max\{1, 1+2\} = \max\{1, 3\} = 3$$

$i = 3$, $y = 1$ to 6

$$f_3(1) = \max\{f_2(1), f_2(-2) + 4\}$$
$$= \max\{0, -\infty + 4\} = \max\{0, -\infty\} = 0$$

$$f_3(2) = \max\{f_2(2), f_2(-1) + 4\}$$
$$= \max\{1, -\infty + 4\} = \max\{1, -\infty\} = 1$$

$$f_3(3) = \max\{f_2(3), f_2(0) + 4\}$$
$$= \max\{2, 0+4\} = \max\{2, 4\} = 4$$

$$f_3(4) = \max\{f_2(4), f_2(1) + 4\}$$
$$= \max\{2, 0+4\} = \max\{2, 4\} = 4$$

$$f_3(5) = \max\{f_2(5), f_2(2) + 4\}$$
$$= \max\{3, 1+4\} = \max\{3, 5\} = 5$$

$$\boxed{\begin{aligned} f_3(6) &= \max\{f_2(6), \underline{f_2(3) + 4}\} \\ &= \max\{3, 2+4\} \Rightarrow \max\{3, 6\} = 6 \end{aligned}}$$

$$(x_1, x_2, x_3) = (0, 1, 1)$$

with maximum profit value 6

Q.2 $n = 4$, $m = 26$

$P = (11, 21, 31, 33)$, $W(2, 11, 22, 15)$

$$\boxed{f_i(y) = \max\{f_{i-1}(y), f_{i-1}(y - w_i) + p_i\}}$$

$i = 1$, $y = 1$ to 26

Assumptions :- $f_0(y) = f_1(y) = f_2(y) = f_3(y) = f_4(y) = \infty \quad \forall\, y < 0$

2. $f_0(y) = 0 \quad \forall\, y \geq 0$

3. $f_1(0) = f_2(0) = f_3(0) = 0$

$i = 1$, $y = 1$ to 26

$f_1(1) = \max\{f_0(1), f_0(1-2) + 11\}$
$= \max\{0, f_0(-1) + 11\} = \max\{0, -\infty + 11\} \Rightarrow 0$

$f_1(2) = \max\{f_0(2), f_0(0) + 11\}$
$= \max\{0, 0 + 11\} = \max\{0, 11\} \Rightarrow 11$

$f_1(3) = \max\{f_0(3), f_0(1) + 11\}$
$= \max\{0, 0 + 11\} = \max\{0, 11\} \Rightarrow 11$

$f_1(4) = \max\{f_0(4), f_0(2) + 11\}$
$= \max\{0, 0 + 11\} = \max\{0, 11\} = 11$

Date :

$F_1(5) = \max\{F_0(5), F_0(3)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(6) = \max\{F_0(6), F_0(4)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(7) = \max\{F_0(7), F_0(5)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(8) = \max\{F_0(8), F_0(6)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(9) = \max\{F_0(9), F_0(7)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(10) = \max\{F_0(10), F_0(8)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(11) = \max\{F_0(11), F_0(9)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(12) = \max\{F_0(12), F_0(10)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(13) = \max\{F_0(13), F_0(11)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(14) = \max\{F_0(14), F_0(12)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(15) = \max\{F_0(15), F_0(13)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(16) = \max\{F_0(16), F_0(14)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(17) = \max\{F_0(17), F_0(15)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(18) = \max\{F_0(18), F_0(16)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(19) = \max\{F_0(19), F_0(17)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(20) = \max\{F_0(20), F_0(18)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(21) = \max\{F_0(21), F_0(19)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(22) = \max\{F_0(22), F_0(20)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(23) = \max\{F_0(23), F_0(21)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$F_1(24) = \max\{F_0(24), F_0(22)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

Page No.
Date :

$f_1(25) = \max\{f_0(25), f_0(23)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$f_1(26) = \max\{f_0(26), f_0(24)+11\}$
$= \max\{0, 0+11\} = \max\{0, 11\} = 11$

$i = 2$, $y = 1$ to 26

$f_2(1) = \max\{f_1(1), f_1(1-11)+21\}$
$= \max\{0, f_1(-10)+21\} = \max\{0, -\infty+21\} = 0$

$f_2(2) = \max\{f_1(2), f_1(-9)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(3) = \max\{f_1(3), f_1(-8)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(4) = \max\{f_1(4), f_1(-7)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(5) = \max\{f_1(5), f_1(-6)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(6) = \max\{f_1(6), f_1(-5)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(7) = \max\{f_1(7), f_1(-4)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(8) = \max\{f_1(8), f_1(-3)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(9) = \max\{f_1(9), f_1(-2)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

$f_2(10) = \max\{f_1(10), f_1(-1)+21\}$
$= \max\{11, -\infty+21\} = \max\{11, -\infty\} = 11$

★ $f_2(11) = \max\{f_1(11), f_1(0)+21\}$
$= \max\{11, 0+21\} = \max\{11, 21\} = 21$

$f_2(12) = \max\{f_1(12), f_1(1)+21\}$
$= \max\{11, 0+21\} = \max\{11, 21\} = 21$

$f_2(13) = \max\{f_1(13), f_1(2)+21\}$
$= \max\{11, 10+21\} = \max\{11, 31\} = 31$

$f_2(14) = \max\{f_1(14), f_1(3)+21\}$
$= \max\{11, 10+21\} = \max\{11, 31\} = 31$

$f_2(15) = \max\{f_1(15), f_1(4)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(16) = \max\{f_1(16), f_1(5)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(17) = \max\{f_1(17), f_1(6)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(18) = \max\{f_1(18), f_1(7)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(19) = \max\{f_1(19), f_1(8)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(20) = \max\{f_1(20), f_1(9)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(21) = \max\{f_1(21), f_1(10)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(22) = \max\{f_1(22), f_1(11)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(23) = \max\{f_1(23), f_1(12)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(24) = \max\{f_1(24), f_1(13)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(25) = \max\{f_1(25), f_1(14)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$f_2(26) = \max\{f_1(26), f_1(15)+21\}$
$= \max\{11, 11+21\} = \max\{11, 32\} = 32$

$\ell = 3$, $y = 1$ to 20

$f_3(1) = \max\{f_2(1), f_2(-21)+31\}$
$= \max\{0, -\infty+31\} = \max\{0, -\infty\} = 0$

$f_3(2) = \max\{f_2(2), f_2(-20)+31\}$
$= \max\{10, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(3) = \max\{f_2(3), f_2(-19)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(4) = \max\{f_2(4), f_2(-18)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(5) = \max\{f_2(5), f_2(-17)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(6) = \max\{f_2(6), f_2(-16)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(7) = \max\{f_2(7), f_2(-15)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(8) = \max\{f_2(8), f_2(-14)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(9) = \max\{f_2(9), f_2(-13)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(10) = \max\{f_2(10), f_2(-12)+31\}$
$= \max\{11, -\infty+31\} = \max\{11, -\infty\} = 11$

$f_3(11) = \max\{f_2(11), f_2(-11)+31\}$
$= \max\{21, -\infty+31\} = \max\{21, -\infty\} = 21$

$f_3(12) = \max\{f_2(12), f_2(-10)+31\}$
$= \max\{21, -\infty+31\} = \max\{21, -\infty\} = 21$

$f_3(13) = \max\{f_2(13), f_2(-9)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(14) = \max\{f_2(14), f_2(-8)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(15) = \max\{f_2(15), f_2(-7)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(16) = \max\{f_2(16), f_2(-6)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(17) = \max\{f_2(17), f_2(-5)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(18) = \max\{f_2(18), f_2(-4)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(19) = \max\{f_2(19), f_2(-3)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

Date :

$f_3(20) = \max\{f_2(20), f_2(-2)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(21) = \max\{f_2(21), f_2(-1)+31\}$
$= \max\{32, -\infty+31\} = \max\{32, -\infty\} = 32$

$f_3(22) = \max\{f_2(22), f_2(0)+31\}$
$= \max\{32, 0+31\} = \max\{32, 31\} = 32$

$f_3(23) = \max\{f_2(23), f_2(1)+31\}$
$= \max\{32, 0+31\} = \max\{32, 31\} = 32$

$f_3(24) = \max\{f_2(24), f_2(2)+31\}$
$= \max\{32, 11+31\} = \max\{32, 42\} = 42$

$f_3(25) = \max\{f_2(25), f_2(3)+31\}$
$= \max\{32, 11+31\} = \max\{32, 42\} = 42$

$f_3(26) = \max\{f_2(26), f_2(4)+31\}$
$= \max\{32, 11+31\} = \max\{32, 42\} = 42$

$i = 4$, $y = 1$ to 26

$f_4(1) = \max\{f_3(1), f_3(-14)+33\}$
$= \max\{0, -\infty+33\} = \max\{0, -\infty\} = 0$

$f_4(2) = \max\{f_3(2), f_3(-13)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

Page No.
Date :

$f_4(3) = \max\{f_3(3), f_3(-12)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(4) = \max\{f_3(4), f_3(-11)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(5) = \max\{f_3(5), f_3(-10)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(6) = \max\{f_3(6), f_3(-9)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(7) = \max\{f_3(7), f_3(-8)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(8) = \max\{f_3(8), f_3(-7)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(9) = \max\{f_3(9), f_3(-6)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(10) = \max\{f_3(10), f_3(-5)+33\}$
$= \max\{11, -\infty+33\} = \max\{11, -\infty\} = 11$

$f_4(11) = \max\{f_3(11), f_3(-4)+33\}$
$= \max\{21, -\infty+33\} = \max\{21, -\infty\} = 21$

$f_4(12) = \max\{f_3(12), f_3(-3)+33\}$
$= \max\{21, -\infty+33\} = \max\{21, -\infty\} = 21$

$f_4(13) = \max\{f_3(13), f_3(-2)+33\}$
$= \max\{32, -\infty+33\} = \max\{32, -\infty\} = 32$

$f_4(14) = \max\{f_3(14), f_3(-1)+33\}$
$= \max\{32, -\infty+33\} = \max\{32, -\infty\} = 32$

$f_4(15) = \max\{f_3(15), f_3(0)+33\}$
$= \max\{32, 0+33\} = \max\{32, 33\} = 33$

$f_4(16) = \max\{f_3(16), f_3(1)+33\}$
$= \max\{32, 0+33\} = \max\{32, 33\} = 33$

$f_4(17) = \max\{f_3(17), f_3(2)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(18) = \max\{f_3(18), f_3(3)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(19) = \max\{f_3(19), f_3(4)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(20) = \max\{f_3(20), f_3(5)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(21) = \max\{f_3(21), f_3(6)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(22) = \max\{f_3(22), f_3(7)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(23) = \max\{f_3(23), f_3(8)+33\}$
$= \max\{32, 11+33\} = \max\{32, 44\} = 44$

$f_4(24) = \max\{f_3(24), f_3(9)+33\}$
$= \max\{42, 11+33\} = \max\{42, 44\} = 44$

$f_4(25) = \max\{f_3(25), f_3(10)+33\}$
$= \max\{42, 11+33\} = \max\{42, 44\} = 44$

$$\boxed{\begin{aligned} f_4(26) &= \max\{f_3(26), f_3(11)+33\} \\ &= \max\{42, 21+33\} = \max\{42, 54\} = 54 \end{aligned}}$$

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

Page No.
Date :

* BackTracking *

1. State space tree.
2. Gigarranted solution
3. Exhaustive Search
4. Depth first search

Probles which we will be sort out using backtrack

1. N- Queen Problem
2. Sum of subsets
3. 0/1 knapsack

Backtracking :-

A Backtracking is a method of building feasible solution of a combinatorial optimization problem through enumaration. A backtracking is a kind of an exhaustive search i.e. all feasible solutions are considered and therefore it will always find the optimal solution.

The solution to the problem for which backtracking is useful, are in the form of a vector

$$X = (X_1, X_2, \text{------} X_N)$$

The vector component X_K is to be selected from a finite set S. There is always a criteria functi F of the feasible solution. The method calls

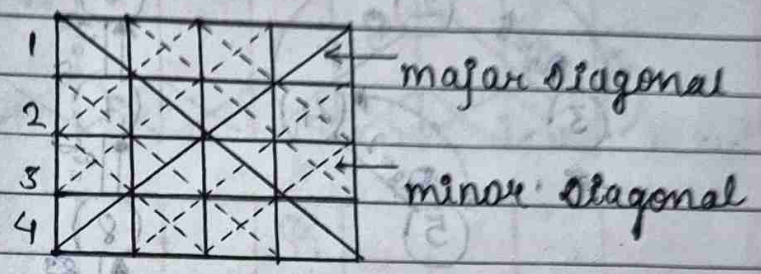
for either maximization or minimization of F(X).

⑦ N - Queen Problem :-

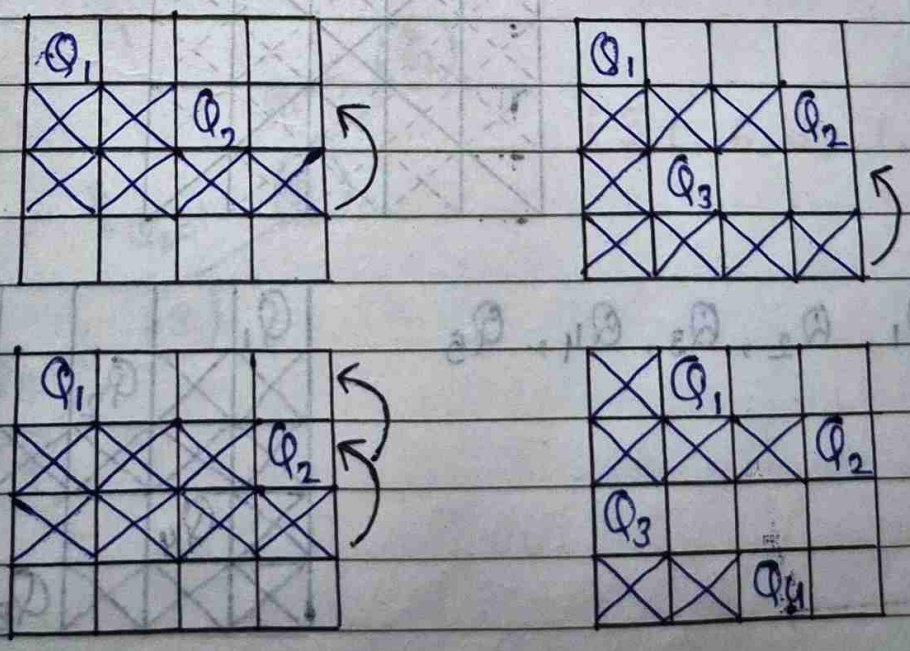
4 - Queen Problem :-

Q1, Q2, Q3, Q4 , N = 4

4×4

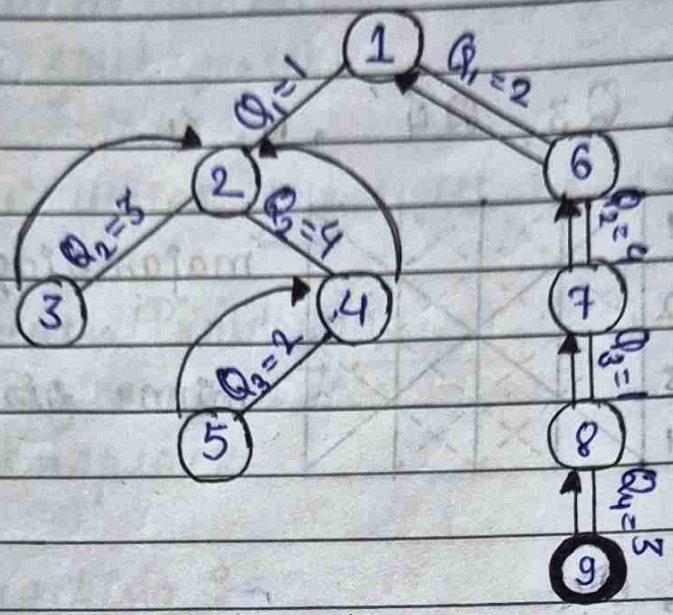


1. Q1, Q2, Q3, Q4
2. No two queen can be placed on same column.
3. No two queen can be placed on major or minor diagonal.



$(Q_1, Q_2, Q_3, Q_4) = (2, 4, 1, 3)$
$(Q_1, Q_2, Q_3, Q_4) = (3, 1, 4, 2)$

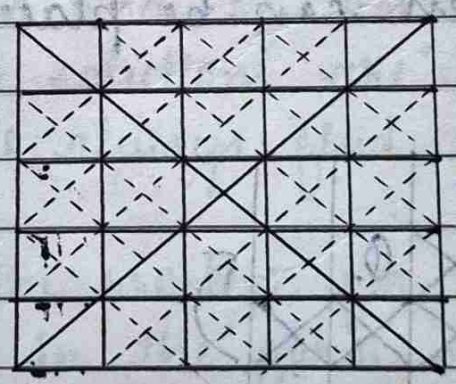
(2 unique solutions)



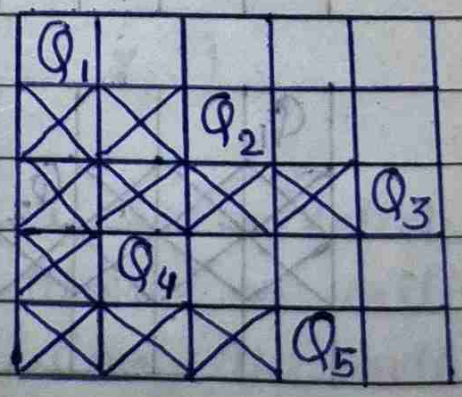
$(Q_1, Q_2, Q_3, Q_4) = (2, 4, 1, 3)$

5 Queen Problem :-

N = 5

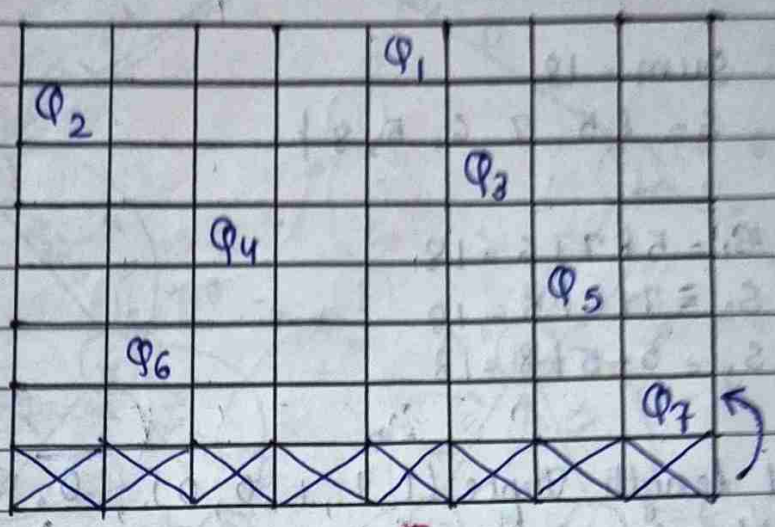


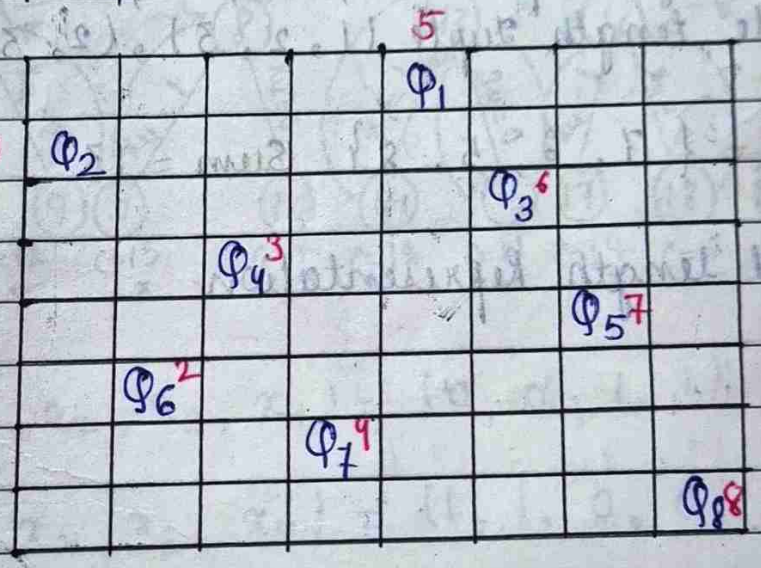
Q_1, Q_2, Q_3, Q_4, Q_5

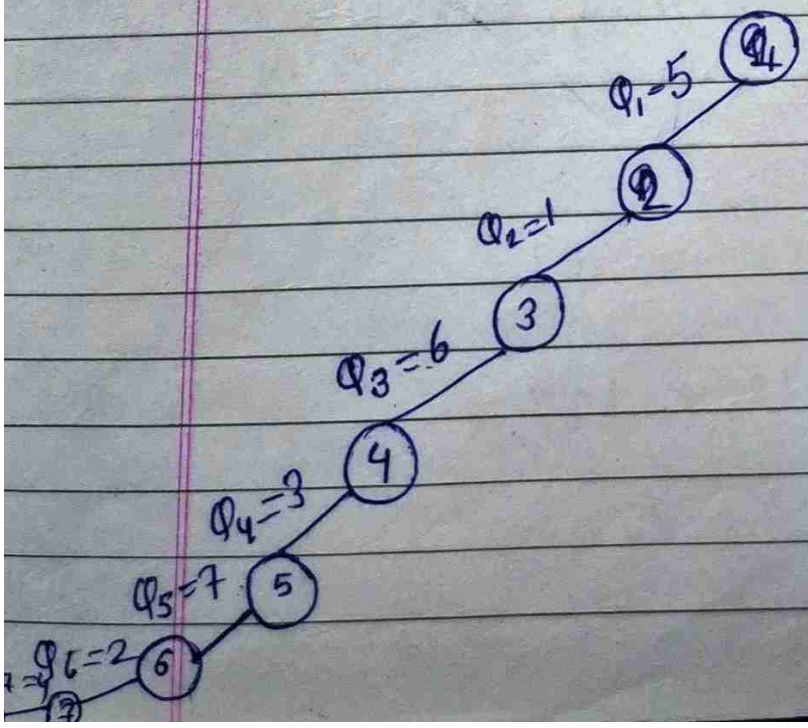


Date :

8- Queen Problem :







(5, 1, 6, 3, 7, 2, 4, 8)

(14) Sum of Subsets :-

e.g. sum = 18

$S = \{5, 7, 6, 5, 8\}$

$S_1 = 5+7+6 = 18$

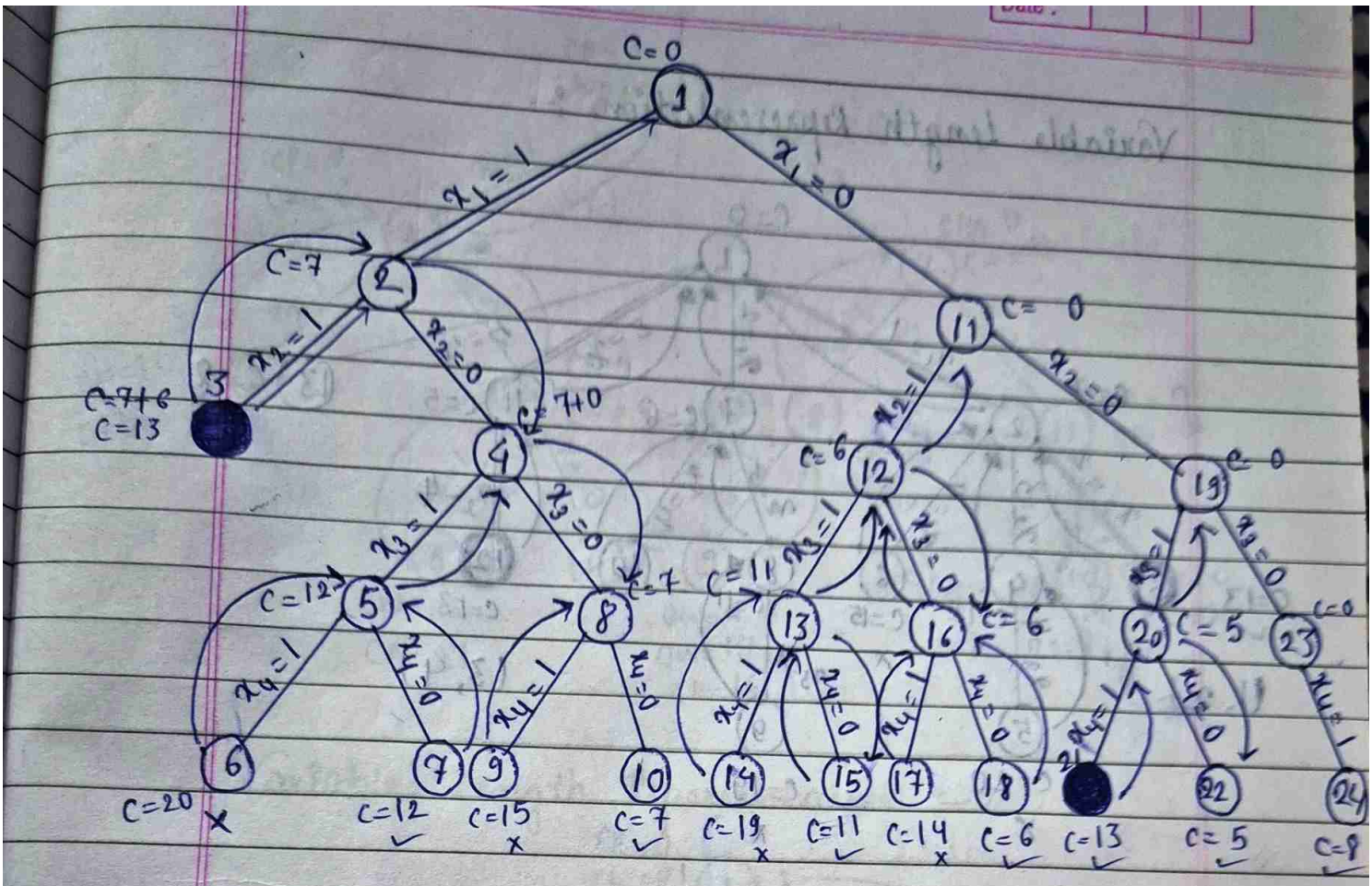
$S_2 = 7+6+5 = 18$

$S_3 = 5+5+8 = 18$

1. Fixed length Tuple (1, 1, 1, 0, 0), (0, 1, 1, 1, 0), (1, 0, 0)
2. Variable length Tuple (1, 2, 3), (2, 3, 4), (1, 4, 5)

$S = \{7, 6, 5, 8\}$ sum = 13

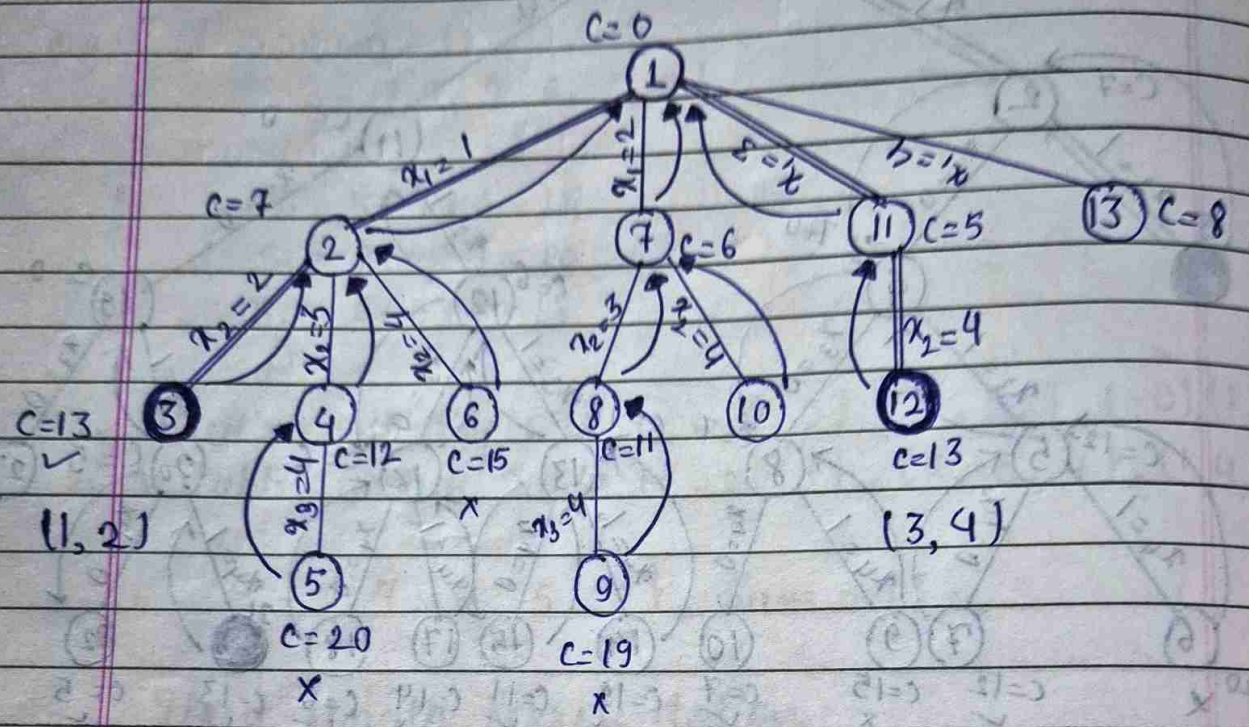
Fixed length Representation



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

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

Variable Length Representation :-



$(x_1, x_2) = (1, 2)$

$(x_3, x_4) = (3, 4)$

---

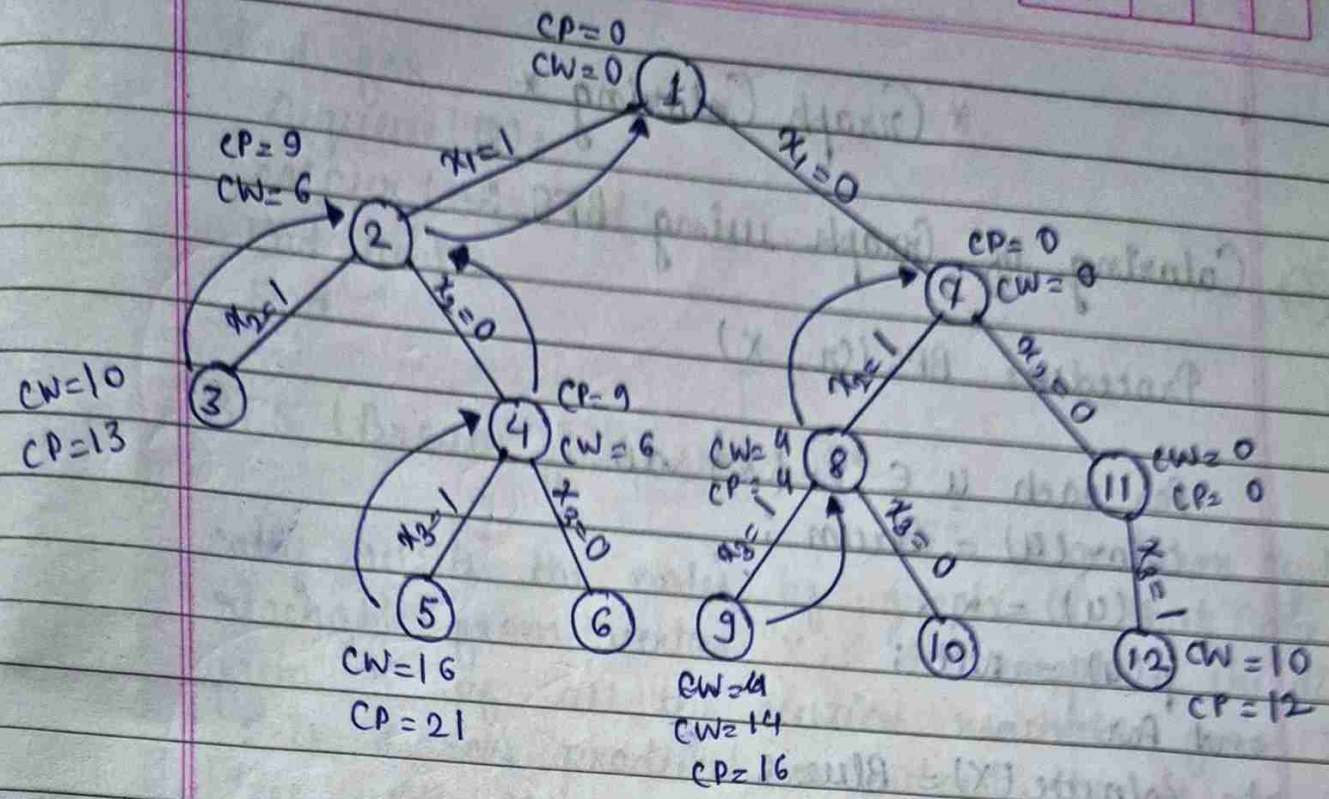
* 0/1 knapsack problem using Backtracking :-

$n = 3$

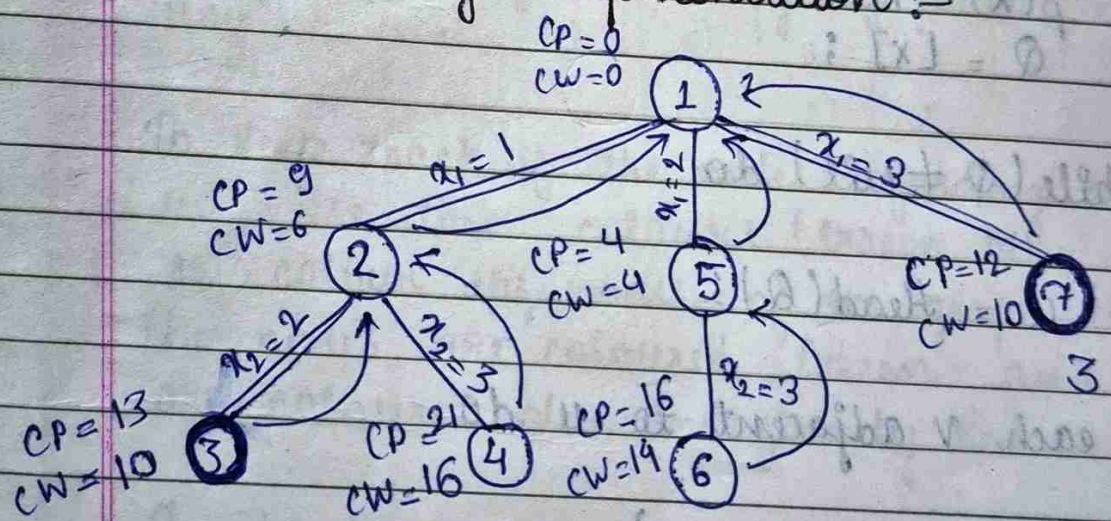
$(W_1, W_2, W_3) = (6, 4, 10)$

$(P_1, P_2, P_3) = (9, 4, 12)$

$C = 10$



Variable Length Representation :-



(1, 2)

* Graph Coloring *

Coloring of Graph using BFS :-

```
Procedure BFS (G, X)

  For each u ∈ V-{x} do
    color (u) = Green ;
    d (u) = ∞ ;
    p(u) = Nil ;
end for
    colour (X) = Blue ;
      d(x) = 0 ;
      p(x) = Nil ;
       Q = [x] ;

while (Q ≠ Nil) do

      u = Head(Q);

for each v adjacent to u do

    if (colour(V) = Green)

    colour(V) = Blue ;
      d(V) = d(u) + 1;
      p(V) = u ;
      Q = {Q, V}
end if
```

Date :

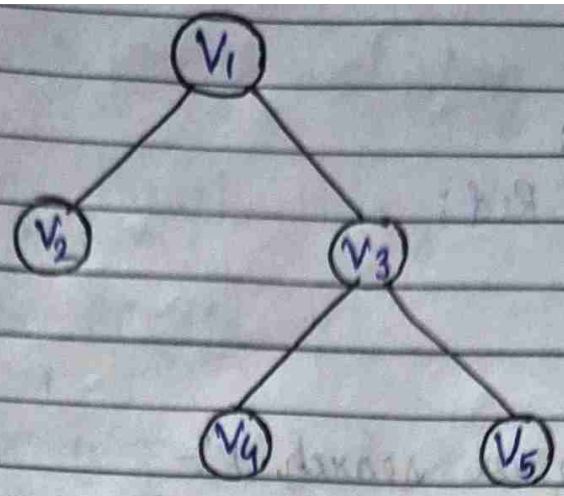
```
End for
 Dequeue(Q);
colour(u) = Red;
End while
End
```

BFS (Breath first search):-

BFS algorithm systamatically visits the nodes of graph G that are reachable from vertex X. It completes the distance from X of all the vertices reachable from X. This search produces as output the breath first tree with X as the root node. This search expands the frontier of discovery of nodes in a systamatic way.

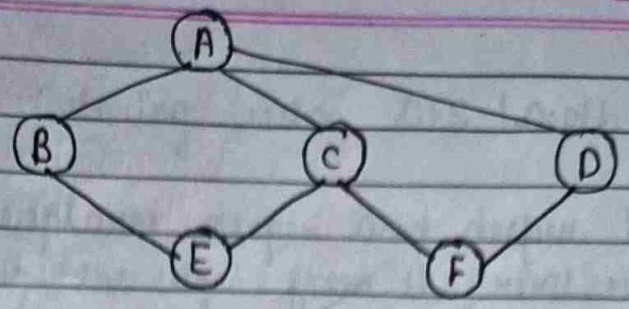
To keep track of the nodes regarding there discovery, we use three colours (Green, Blue and Red) to colour the nodes of the graph. Initially all the nodes are coloured Green and later these are coloured blue and finally Red.

To do so we construct the following matrix to show the working of the algorithm.



V_1	V_2	V_3	V_4	V_5	d	P	Q
	Gr	Gr	Gr	Gr			
B	Gr	Gr	Gr	Gr	0	Nil	$[V_1]$
B	B	Gr	Gr	Gr	1	V_1	$[V_1, V_2]$
B	B	B	Gr	Gr	1	V_1	$[V_1, V_2, V_3]$
R	B	B	Gr	Gr	1	–	$[V_2, V_3]$
R	R	B	Gr	Gr	–	V_1	$[V_3]$
R	R	B	B	Gr	2	V_3	$[V_3, V_4]$
R	R	B	B	B	2	V_3	$[V_3, V_4, V_5]$
R	R	R	B	B	–	–	$[V_4, V_5]$
R	R	R	R	B	–	–	$[V_5]$
R	R	R	R	R	–	–	[ ]

Date :



A	B	C	D	E	F	d	P	Q
	Gr	Gr	Gr	Gr	Gr			
B	Gr	Gr	Gr	Gr	Gr	0	Nil	[A]
B	B	Gr	Gr	Gr	Gr	1	A	[A, B]
B	B	B	Gr	Gr	Gr	1	A	[A, B, C]
B	B	B	B	Gr	Gr	1	A	[A, B, C, D]
R	B	B	B	Gr	Gr			[B, C, D]

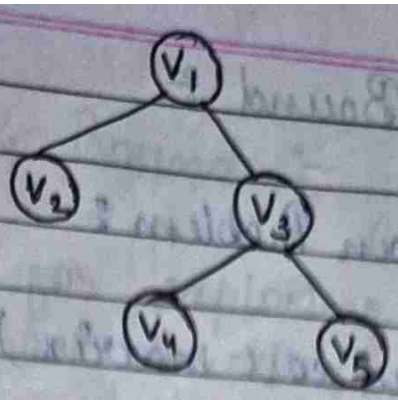
(6) Graph Coloring using DFS (Depth First Search):-

DFS explores deeper and deeper in the graph by following the edges from the root node. It explores deeper from the most recently explored node to which another unexplored node is connected by an edge. This process continuous till all the reachable nodes are covered. If some more nodes are still left, the search continues from one such node in the depth first manner this whole process can be illustrated by using adjacency list for the depth first search.

```
Procedure Main_DFS (G)

For each vertex u ∈ V do

    colour (u) = green;

    p(u) = Nil;

end for
        T = 0;

For each vertex u ∈ V do

    if (colour(u) = green) then

    call DFS (u, T);
```

```
end if
end for
end
    Procedure DFS (u, T)
        colour(u) = Blue ;
        T = T+1 ;
        t1(u) = T ;

    For each v ∈ adjecancy list (u) do

        if colour(V) = green

            p(V) = u ;

            call DFS (V, T) ;

end if
end for
        colour(u) = Red ;
        T = T+1 ;
        t2(u) = T ;
End
```

Date :



$V_1 \rightarrow [V_2 \mid V_3]$

$V_2 \rightarrow [V_1]$

$V_3 \rightarrow [V_1 \mid V_4 \mid V_5]$

$V_4 \rightarrow [V_3]$

$V_5 \rightarrow [V_3]$

V_1	V_2	V_3	V_4	V_5
G	G	G	G	G
B	G	G	G	G
B	B	G	G	G
B	R	G	G	G
B	R	B	G	G
B	R	B	B	G
B	R	B	R	G
B	R	B	R	B
B	R	B	R	R
B	R	R	R	R
R	R	R	R	R

Branch and Bound :

Travelling Salesman Problem :

(reduced cost matrix)

$$\begin{bmatrix} 2 & 3 & 1 \\ 0 & 2 & 1 \\ 1 & 3 & 2 \end{bmatrix}$$

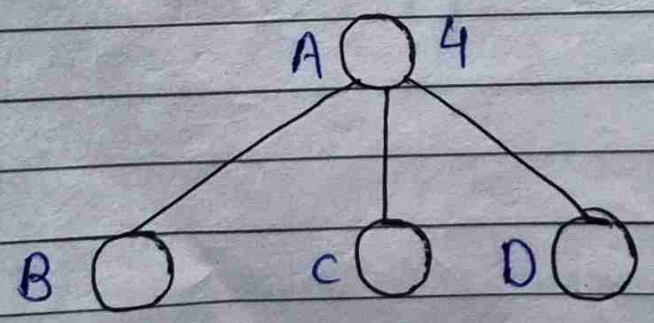
$$\begin{bmatrix} 1 & 2 & 0 \\ 0 & 2 & 1 \\ 0 & 2 & 1 \end{bmatrix} \begin{matrix} 1 \\ 0 \\ 1 \end{matrix}$$

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

$$\begin{matrix} 0 & 2 & 0 \end{matrix}$$

$$\boxed{T} = 4$$

value of Bound



V

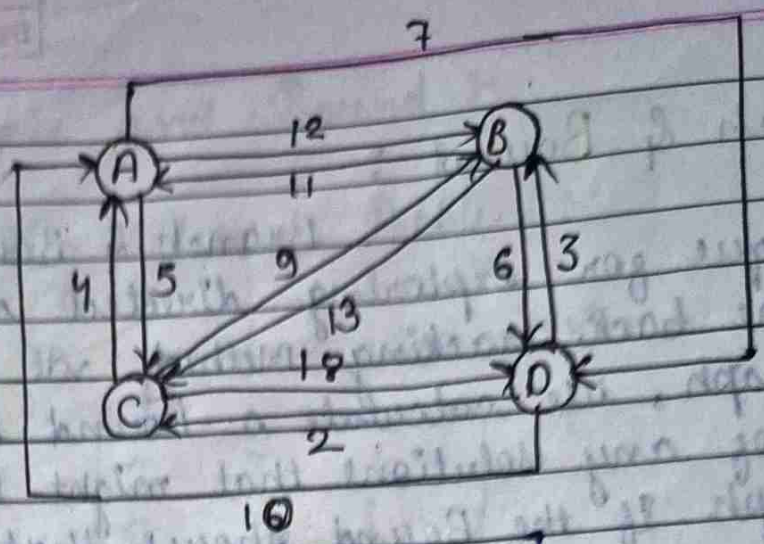
Branch & Bound :-

Branch & Bound is a technique for exploring directed acyclic graph like the back tracking method. At each node of the graph, we calculate a Bound on the possible value of any solutions that might lie further in the graph. If the Bound shows that any such solution must necessarily be worse than the best solution found so far then we do not need to go on exploring this part of the solution of the graph.

There are few differences between backtracking & Branch & Bound.

A Backtracking algorithm does a pre order Depth first search of the state space tree whereas branch & Bound is not restricted to DFS of the state space tree. A Branch & Bound technique must completely search the state space tree for an optimal value.

12+6+2+4

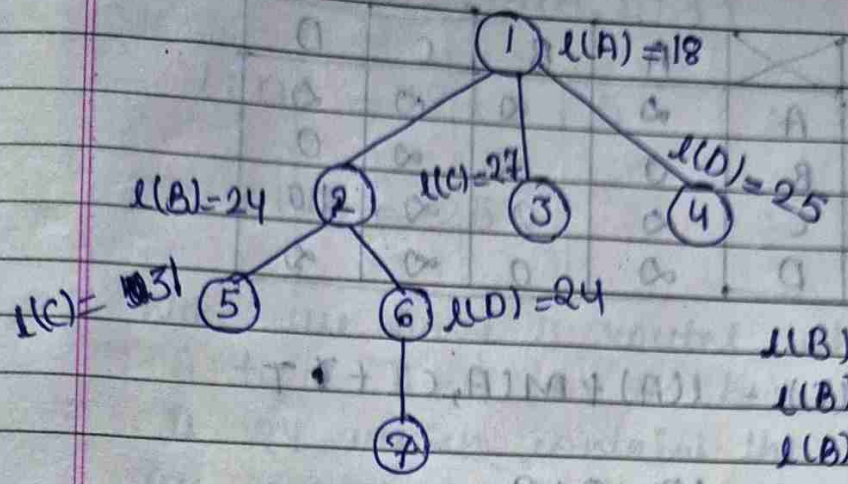


	A	B	C	D	
A	∞	12	5	7	5
B	11	∞	13	6	6
C	4	9	∞	18	4
D	10	3	2	∞	2
					17

(b)

	A	B	C	D	
A	∞	7	0	2	5 0
B	5	∞	7	0	6 0
C	0	5	∞	14	4 0
D	8	1	0	∞	2 0
	0	1	0	0	18

	A	B	C	D	
A	∞	6	0	2	5
B	5	∞	7	0	6
C	0	4	∞	14	4
D	8	0	0	∞	2
	0	1	0	0	18



$l(B) = l(A) + M(A,B) + T$

$l(B) = 18 + 6 + 0$

$l(B) = 24$

$M(A,B) = 6$

(A, B)

(1)

	A	B	C	D	
A	∞	∞	∞	∞	
B	∞	∞	7	0	0
C	0	∞	∞	14	0
D	8	∞	0	∞	0
	0	0	0	0	0

(A, C)

(2)

	A	B	C	D
A	∞	6	0	2
B	5	∞	7	0
C	0	4	∞	14
D	8	0	0	∞

$M(A,C) = 0$

	A	B	C	D	
A	∞	∞	∞	∞	0
B	5	∞	∞	0	0
C	∞	4	∞	14	4
D	8	0	∞	∞	0
	5	0	0	0	9

	A	B	C	D
A	∞	∞	∞	∞
B	0	∞	∞	0
C	∞	0	∞	10
D	∞	0	∞	∞

$l(C) = l(A) + M(A, C) + T$

$l(C) = 18 + 0 + 9$

$l(C) = 27$

③ (A, D)

	A	B	C	D
A	∞	6	0	2
B	5	∞	7	0
C	0	4	∞	14
D	8	0	0	∞

$M(A, D) = 2$

	A	B	C	D	
A	∞	∞	∞	∞	0
B	5	∞	7	∞	5
C	0	4	∞	∞	0
D	∞	0	0	∞	0
	0	0	0	0	5

	A	B	C	D
A	∞	∞	∞	∞
B	0	∞	2	∞
C	0	4	∞	∞
D	∞	0	0	∞

$T = 5$

$l(D) = l(A) + M(A, D) + T$

$l(D) = 18 + 2 + 5$

$l(D) = 25$

Now we have 3 routes which are A to B, A to C & A to D but the minimum cost is 24 which contains the route A to B so we explore only A to B node instead of C & D.

④ (B, C)

$M(B, C) = 7$

	A	B	C	D	
A	∞	∞	∞	∞	–
B	∞	∞	7	0	0
C	0	∞	∞	14	0
D	8	∞	0	∞	0
	0	–	0	0	0

~~$l(\blacksquare) = l(B) + M(B, C) + T$~~

~~$l(C) = 24 + 7 + 0$~~

~~$l(C) = 24$~~ 31

⑤ (B, D)

	A	B	C	D	
A	∞	∞	∞	∞	–
B	∞	∞	∞	∞	–
C	∞	∞	∞	14	14
D	8	∞	∞	∞	8
					22

	A	B	C	D
A	∞	∞	∞	∞
B	∞	∞	∞	∞
C	∞	∞	∞	0
D	0	∞	∞	∞

$l(C) = l(B) + M(B, C) + T$

$l(C) = 24 + 7 + 22$

$l(C) = 53$

⑤ (B, D)

$M(B, D) = 0$

	A	B	C	D
A	∞	∞	∞	∞
B	∞	∞	7	0
C	0	∞	∞	14
D	8	∞	0	∞

	A	B	C	D
A	∞	∞	∞	∞
B	∞	∞	∞	∞
C	0	∞	∞	∞
D	8	∞	0	∞

	A	B	C	D	
A	∞	∞	∞	∞	–
B	∞	∞	∞	∞	–
C	0	∞	∞	∞	0
D	∞	∞	0	∞	0
	–	–	0	–	0

$l(D) = l(B) + M(B, D) + T$

$l(D) = 24 + 0 + 0$

$l(D) = 24$

⑥ (D, C)

$M(D, C) = 0$

	A	B	C	D
A	∞	∞	∞	∞
B	∞	∞	∞	∞
C	0	∞	∞	∞
D	∞	∞	0	∞

	A	B	C	D	
A	∞	∞	∞	∞	–
B	∞	∞	∞	∞	–
C	0	∞	∞	∞	0
D	∞	∞	∞	∞	–
	–	–	–	–	0

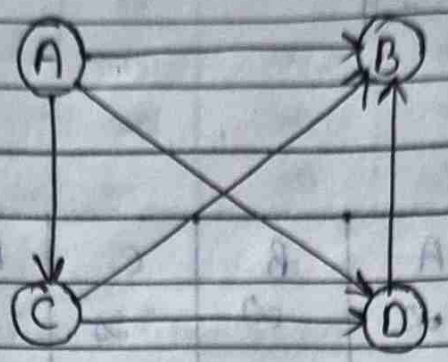
$l(C) = l(D) + M(D, C) + T$

$l(C) = 24 + 0 + 0$

$l(C) = 24$

$A \xrightarrow[12]{} B \xrightarrow[6]{} D \xrightarrow[2]{} C \xrightarrow[4]{} A \Rightarrow 24$

Transitive closure of the Graph :-



$$P_{(0)} = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 \\ C & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 \end{array}$$

$$P_{(1)} = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 1 & 1 \\ B & 0 & 0 & 0 & 0 \\ C & 0 & 1 & 0 & 1 \\ D & 0 & 1 & 0 & 0 \end{array}$$

$$P_{(2)} = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 0 & 1 \\ B & 0 & 0 & 0 & 0 \\ C & 0 & 1 & 0 & 0 \\ D & 0 & 0 & 0 & 0 \end{array}$$

$$P_{(3)} = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 0 & 0 \\ B & 0 & 0 & 0 & 0 \\ C & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 \end{array}$$

$$P_{(4)} = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$P = P_{(1)} + P_{(2)} + P_{(3)} + P_{(4)}$$

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

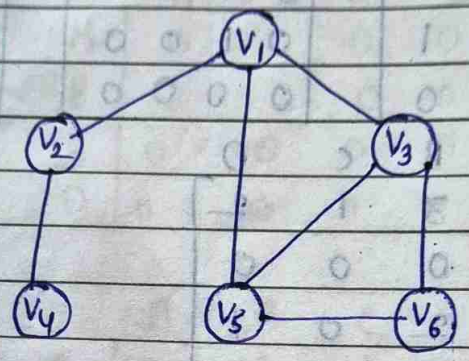
$$\Rightarrow \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 3 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

There are 3 paths from A to B, 1 path from A to C, 2 path from A to D, 2 path from C to B, 1 path from C to D and 1 path from D to B.

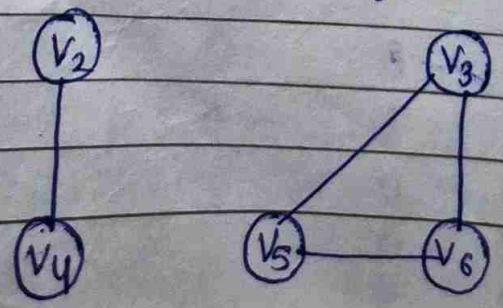
(vi) Articulation Point :-

In a graph, a vertex is called an articulation point if removing it and all the edges associated with it results in the increase of the number of connected components in the graph.

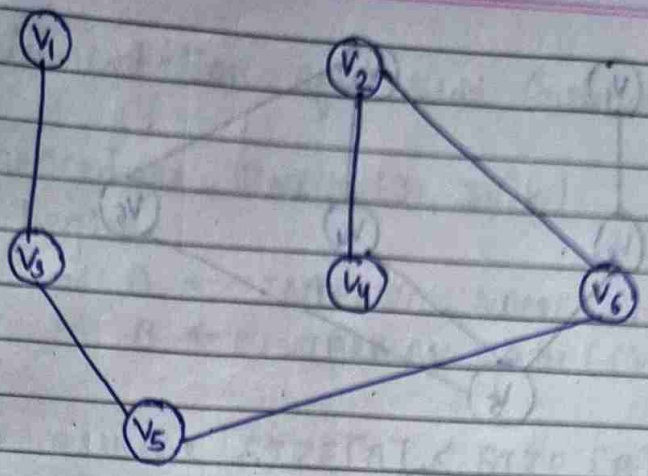
For example consider the following graph :-



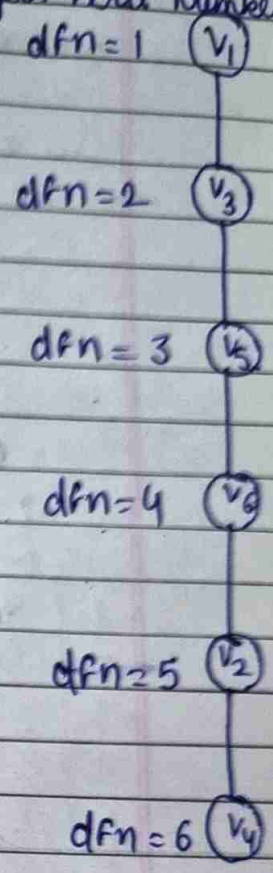
If the above graph, vertex V_1 and all the edges associated with i.e. the edges $V_1 - V_2$, $V_1 - V_5$ & $V_1 - V_3$ are removed, there will be no path to reach any of the vertices V_3, V_5 & V_6 from the vertices V_2, V_4. that means the graph will split into two separate components, one consisting of the vertices V_2 and V_4 and another one consisting of the vertices V_3, V_5, V_6 as shown in the following figure.

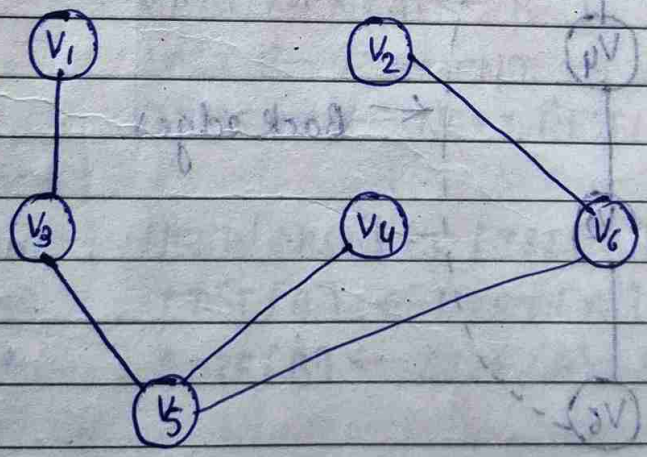


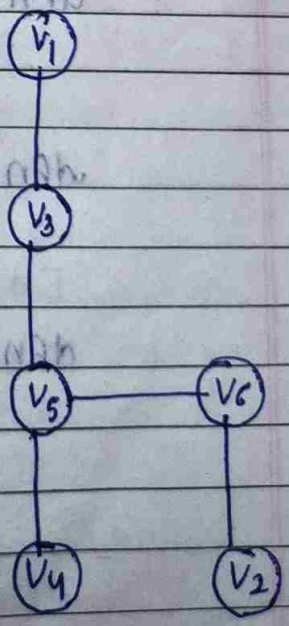
Page No.
Date :

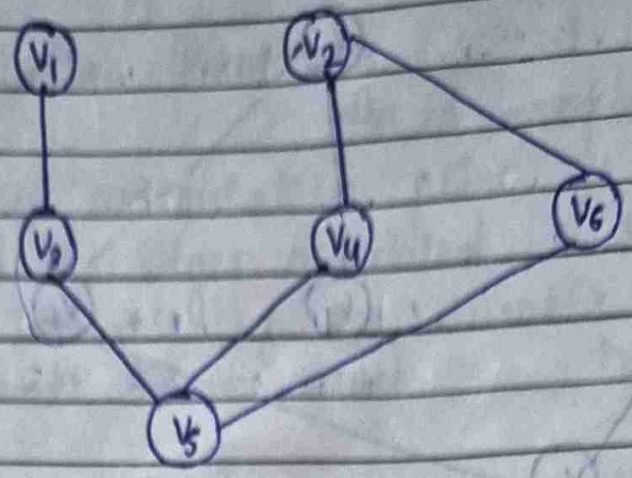


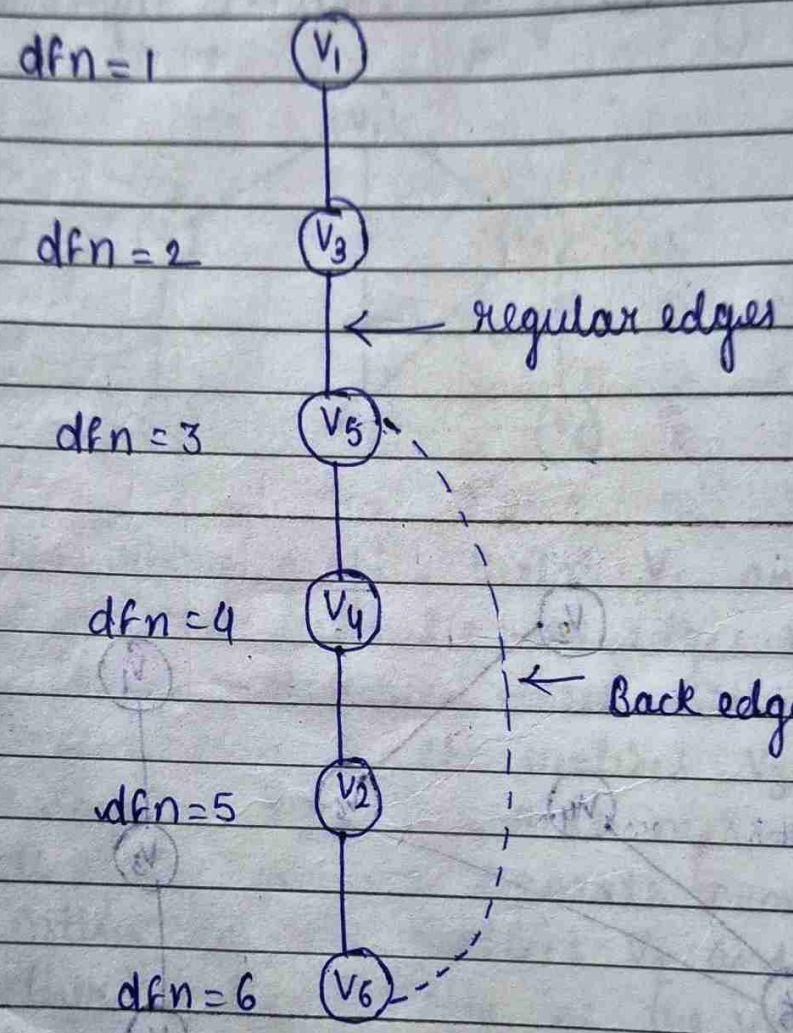
Depth First Number











$G = (V, E)$

$G_T = (V, E_T)$

Set of Back edges $= E - E_T$

Page
Date :

(9) Implementation of UNION Instruction

```
procedure UNION(I, J, K)
  begin
      A ← INTERNAL_NAME[i]
      B ← INTERNAL_NAME[j]

  wlg assume SIZE[A] ≤ SIZE [B]

otherwise interchange roles of A and B in

  begin
      ELEMENT ← LIST[A]
  while
      ELEMENT ≠ 0 do
  begin
      R[ELEMENT] ← B
      LAST ← ELEMENT
      ELEMENT ← NEXT[ELEMENT]
  end;
      NEXT[LAST] ← LIST[B]
      LIST[B] ← LIST[A]
      SIZE[B] ← SIZE[A]+SIZE[B]

      INTERNAL_NAME[K] ← B
      EXTERNAL_NAME[B] ← K
  end
end.
```

Date :

Q. Suppose $n = 8$ and we have the collection of three sets $\{1, 3, 5, 7\}$, $\{2, 4, 8\}$ and $\{6\}$ with real names 1, 2 and 3 respectively

real names & external names are same

$1 = \{1, 3, 5, 7\}$ $\quad = 2 \quad = 3\ (2,4,8)$

$2 = \{2, 4, 8\}$ $\quad = 3 \quad = 2\ (1,3,5,7)$

$3 = \{6\}$

$A = 3$
$B = 2$

Data structure before UNION Instruction

	R	NEXT
1	2	3
2	3 (2)	4
3	2	5
4	3 (2)	8
5	2	7
6	1	0
7	2	0
8	3 (2)	0 (1)

	LIST	SIZE	EXTERNAL_NAME
1	6	1	3
2	1 (2)	4 (7)	1 4
3	2	3	2
4			4

(Sets with external name)

$1 = \{1, 3, 5, 7\}$

$2 = \{2, 4, 8\}$

$3 = \{6\}$

	INTERNAL_NAME
1	2
2	3
3	1
4	2

external name $\neq$ Internal name

plot or pdf

(LUP decomposition & solution of linear eqn & matrix inverse)

Page No.
Date :

Problem Taxonomy :-

The universe of problems may broadly divided into catagories of undecidable problems, tractable problems and intractable problems.

Examples of undecidable problems are the halting problem of Turing Machine, tiling a rectangular area with given tiles with certain conditions, word correspondence problems etc.

The Intractable problems include finding Hamiltonian cycle in a given graph, finding clique of given size, set partitioning problem etc.

Tractable problems include Sorting, searching, Matrix Multiplication etc.

The Tractable problems have polynomial time complexity

The intractable problem have super polynomial time complexity where as Undecidable problems do not have algorithms of any known complexity.

The algorithms of tractable problems are good algorithm irrespective of the degree of polynomial.

There are different types of decidable problems

Date :

1. Decision Problem :-
In this class of problems, the output is either "Yes" or "NO".

For example :- whether a given number is prime i decision problem, whether a given g is having spanning tree or not.

2. Counting Problem :-
The output of this class of algorithm is always a Natural Number.

For example :- How many distinct spanning tre the given graph.

3. Optimization Problem :-
This class of algorithm optim some objective function (Maximization or Minimization

For example :- For a given weighted graph, findin a minimum spanning tree.

(x) Two important classes among the decidable problem are

1. class P
2. class NP
   → a. NP Hard
   b. NP complete

Date :

1. class P :-

This is the class of problems for wh we have deterministic polynomial time algorith This means that if we have a polynomial ti algorithm on one model, we will have a polyno time algorithm on the other models.

Examples :- Sorting, searching, Matrix multiplicat shortest path finding.

2. class NP :-

This is the class of problem solved by non deterministic algorithm in polynomial time. The class NP is a kind of diction proble that have an efficient proof system which mean that each "yes" candidate must have atleast one succession whose validity can be verified.

For example :- Hamiltonian cycle, travelling salesmen problem, independent set, three colourability of the graph.

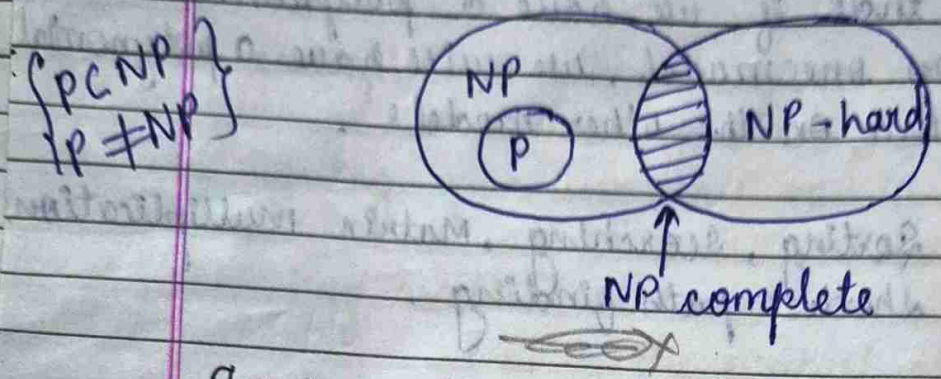
a. NP Hard :-

A problem L is NP Hard if and only if Satisfyability reduces to L or any other NP Hard problem reduces to L.

b. NP-complete :-

A problem L is NP complete if and

only if L is NP Hard and L ∈ NP



For example :- Vertex cover, clique, hamiltonian circuit, independent set, partition, integer knapsack.

String Matching Pattern Algorithm :-

Regular Expression
Regular Language (symbols)
Transition Diagram
NFA (Non deterministic Finite Automata)
DFA (Deterministic Finite Automata)
conversion from NFA to DFA

Regular expression are used to generate patterns of strings.

regular expression

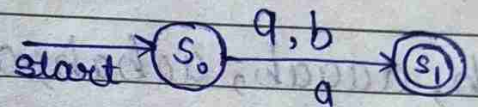
(Followed by) $a/b = \{a, b\}$ ← regular language

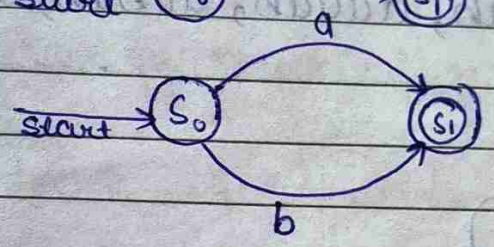
$a.b = \{ab\}$

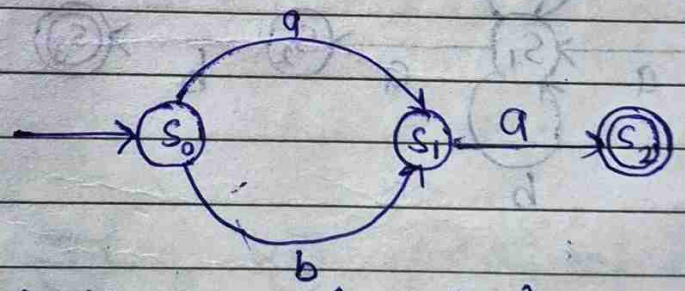
(One or More) $a^+ = \{a, aa, aaa, \ldots\}$

(zero or More) $a^* = \{\epsilon, a, aa, aaa, \ldots\}$

a/b

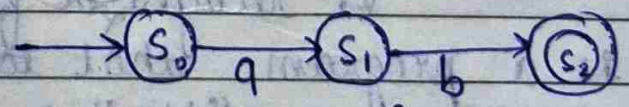




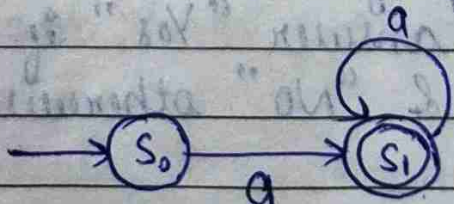


$(a/b)a \Rightarrow \{aa, ba\}$

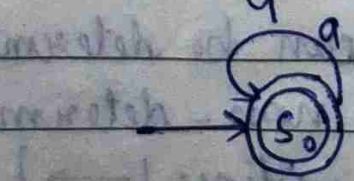
$a.b \Rightarrow$



$a^+ \Rightarrow$

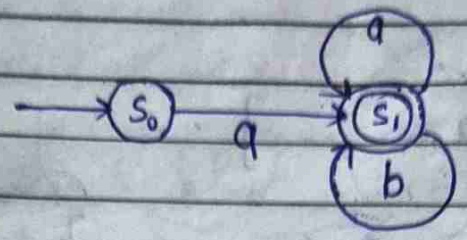


$a^* \Rightarrow$



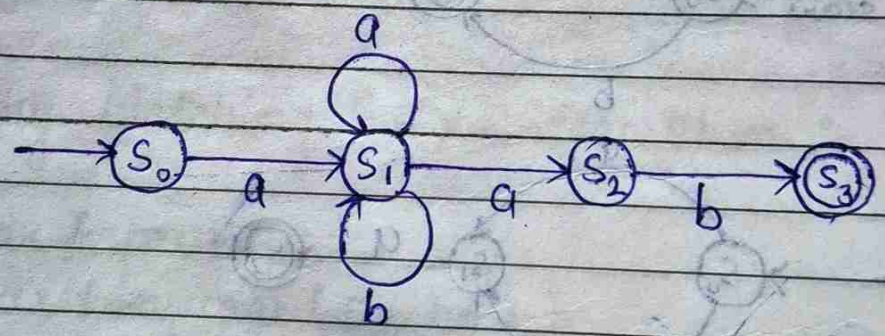
$a(a/b)^*$

{a, aa, ab, aaa, aab, aba, abb, ------}



$a(a/b)^* ab$

{aab, aaab, abab, aaaab, aabab, abaab, abbab, ------}



Finite Automata :-

A recognizer for a language (regular) is a program that takes input as a string x and answer "Yes" if x is a sentence of the language & "No" otherwise.

A Finite Automata can be deterministic or Non-deterministic where non-deterministic means that more than one transition (→) out of a state

which may be possible on the same input symbol.

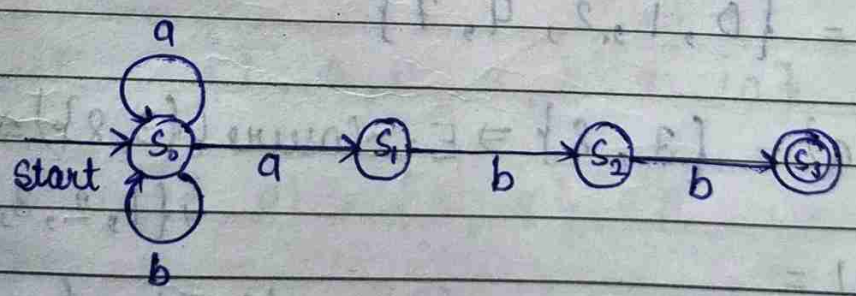
A Finite Automata is a kind of transition diagram which is constructed when an regular expression is compiled into a recognizer.

Both deterministic (DFA) or non deterministic (NFA) finite automata are capable of recognizing the regular set. Thus they both can recognize exactly what regular expressions can denote.

DFA can lead to faster recognizer than NFA. A DFA can be much bigger than an equivalent NFA.

Expression :- $(a/b)^* abb$

NFA :-

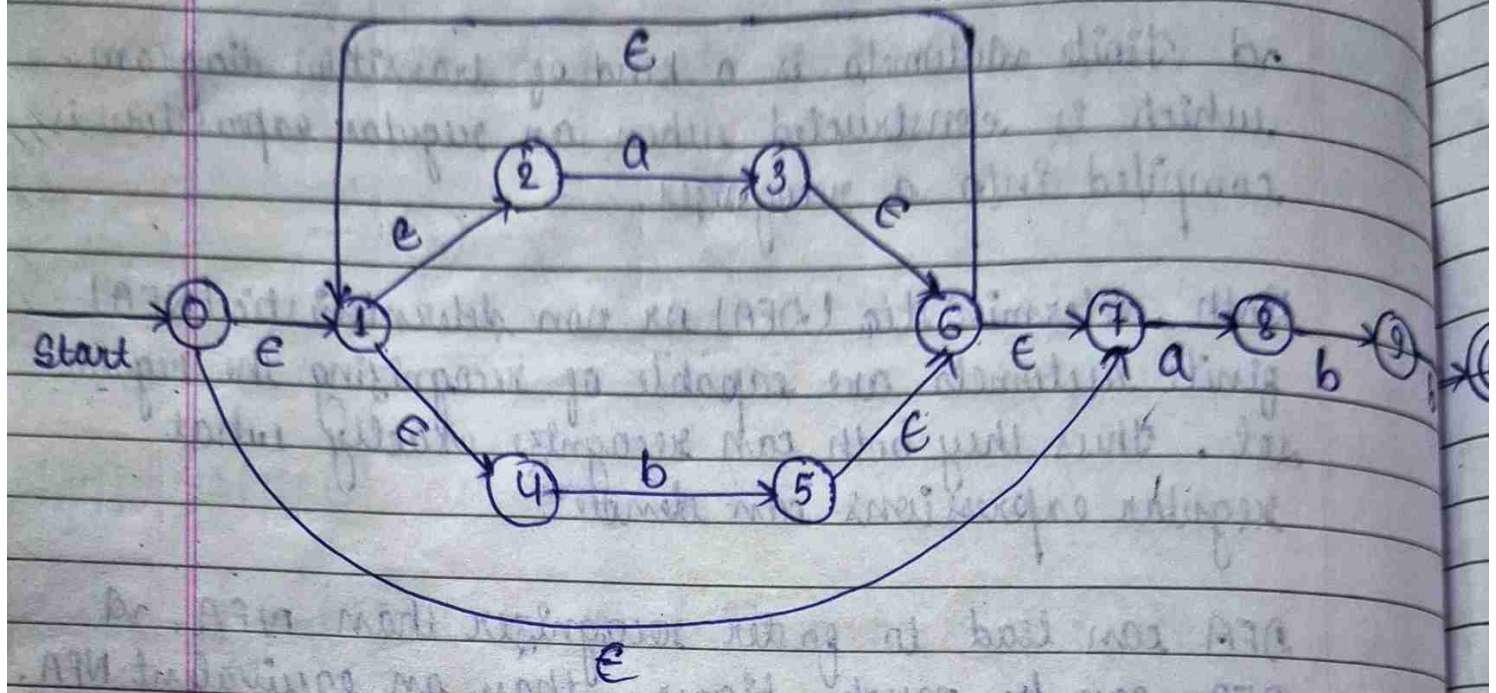


{abb, aabb, babb, aabaabb, bbaabb, ----}

$\epsilon \Rightarrow$ Null Movement

Date :

(16) NFA of $(a|b)^*abb$ using Thomson's Algo:-



NFA to DFA Conversion :-

E-closure{0} = {0, 1, 2, 4, 7}

A = {0, 1, 2, 4, 7}

move(A, a) = {3, 8} ⇒ E-closure({3, 8}) ⇒ {3, 8, 6, 7, 1, 2, 4}

⇒ {1, 2, 3, 4, 6, 7, 8}

move(A, b) =

{5} ⇒ E-closure({5}) ⇒ {5, 6, 7, 1, 2, 4}

B = {1, 2, 3, 4, 6, 7, 8}

C = {1, 2, 4, 5, 6, 7}

move{B, a} = {3, 8} = B

move{B, b} = {5, 9} ⇒

E-closure({5, 9}) ⇒ {5, 6, 7, 1, 2, 4, 9}

D = {1, 2, 4, 5, 6, 7, 9}

move{C, a} = {3, 8} = B

move{C, b} = {5} = C

move{D, a} = {3, 8} = B

move{D, b} = {5, 10} ⇒

E-closure({5, 10}) ⇒ {5, 6, 7, 1, 2, 4, 10}

E = {1, 2, 4, 5, 6, 7, 10}

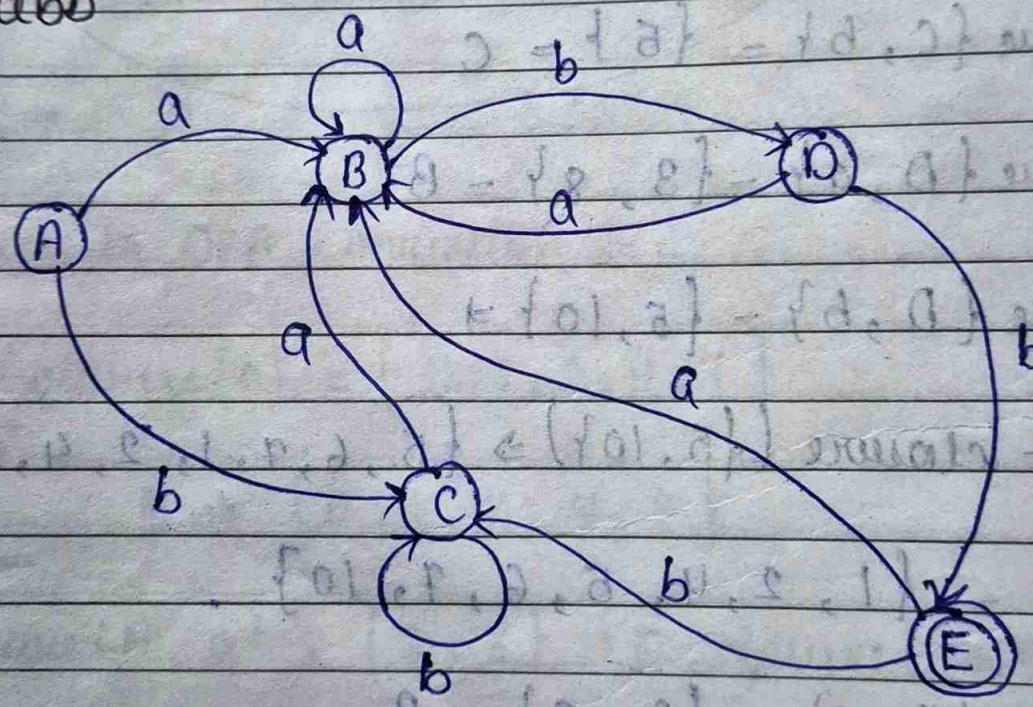
move(E, a) = {3, 8} = B

move(E, b) = {5} = C

Now we create transition table to show various states and input symbol combination.

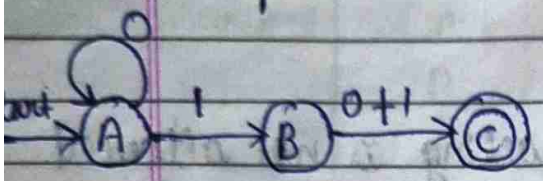
Status	Input Symbol a	b
A	B	C
B	B	D
C	B	C
D	B	E
E	B	C

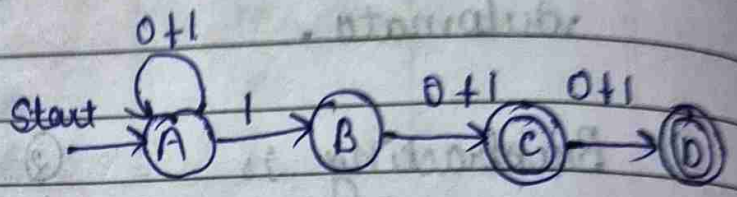
$(a|b)^*$ abb



DFA	NFA
① DFA stands for Deterministic Finite Automata.	① NFA stands for Non deterministic Finite automata.
② Backtracking is allowed.	② Backtracking is not allowed
③ can be constructed with every input and output.	③ can not be constructed for every input and output.
④ Easy to check membership.	④ Difficult to check membership.
⑤ Only one final state	⑤ Numerous possible next states.
⑥ Require more space than NFA.	⑥ Require less space.
⑦ cannot use empty string transition.	⑦ can use empty string transition
⑧ Difficult to construct.	⑧ easy to construct
⑨ DFA can be understood as one Machine.	⑨ NFA can be understood as multiple little machines computing at the same time.

(10) The time needed for executing an Input string is less as compared to NFA.	(10) The time needed for executing an Input string is more as compared to DFA.





Date :

0/1 Knapsack Problem using Branch and Bound:-

Procedure :-

```
float UBound(float cp, float cw, int k, float m)
{
    int i;
    float b, c;

    b = cp;
    c = cw;

    for(i=k+1; i<n; i++)
    {
        if (c+w[i] <= m)
        {
            c = c+w[i];

            b = b-p[i];
        }
    }
    return b;
}
```

Date :

Procedure :-

```
Float Bound (float cp, float cw, int k)
{
    int i ;
    Float b , c ;

    b = cp ;
    c = cw ;

    for (i = k+1 ; i <= n ; i++)
    {
        c = c + w[i] ;

        if (c < m)
            b = b - p[i] ;
        else
            return (b - (1 - (c - m)/w_i) * P_i)).
    }
    return b ;
}
```