3/1/15 UNIT - 1

INTRODUCTION :

→ ALGORITHM : The word algorithm comes from the Persian author Abdullah Jaffar in 9th century who has given the definition of algorithm as follows.

- An algorithm is set of rules for carrying out calculations by the hand or on machine.
- An algorithm is a well defined competitional procedure that takes input and produces output
- An algorithm is a sequence of instructions or steps i.e., inputs to acheive some particular output.
- Any algorithm must follow the criteria / properties.

Input : It generally requires finite number of inputs.

Output : It must produce atleast one output.

Uniqueness : Each instruction should be clear and unambiguess (more clarity of statements)

Finiteness : It must terminate after a finite number of steps

→ ANALYSIS ISSUES OF ALGORITHM

1. What data structures to use ? (list, queue, stacks, trees etc)
2. Is it correct ? (or. all, only, most of the time)
3. How efficient is it ? (asymptotically, fixed or does it depend on the inputs)
4. If there any efficient algorithm ? (P = Np or not)

There are four different areas in order to identify the algorithm

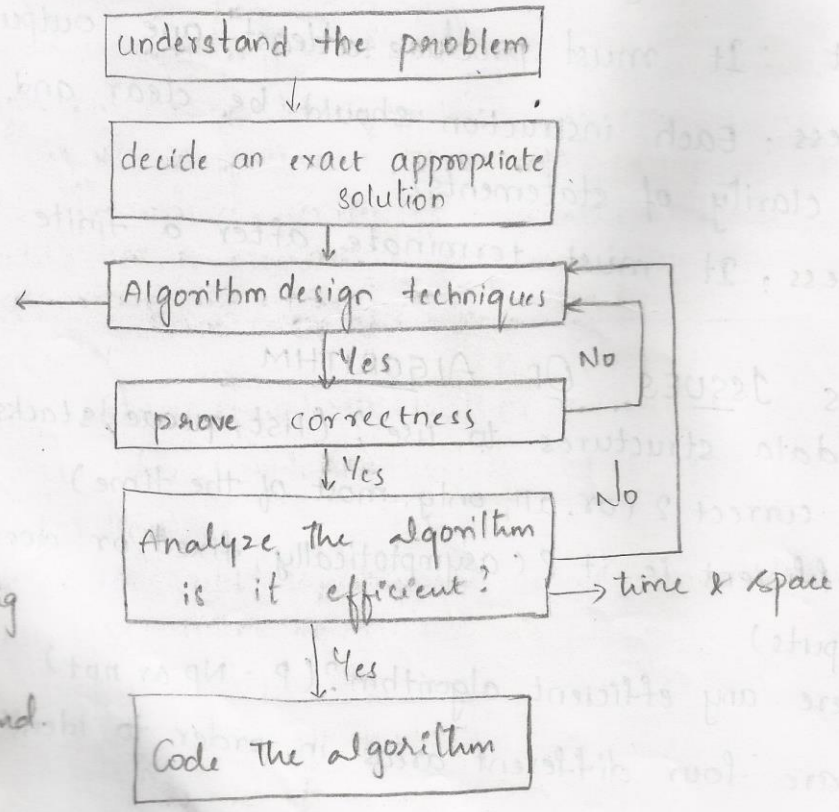
1. How to devise algorithm
2. How to validate algorithm
3. How to analyze an algorithm.
4. How to test the program

Test a program is nothing but performing the debugging and profiling an algorithm.

Validating an algorithm is nothing but cross checking every input it is producing exact output.

Analysis of algorithm is nothing but performance analysis which is done by time & space complexity.

→ PROCESS OF DESIGN ANALYSIS OF ALGORITHM



- divide & conquer
- Dynamic
- Greedy
- Back tracking
- Branch & Bound.

PERFORMANCE ANALYSIS

The evaluation can be done in two ways.

- Priori Estimates / Performance Analysis.
- Posteriori Testing / Performance Measurement

Priori	Posteriori
→ The time taken for executing the algorithm is analyzed prior to the execution of algorithm.	→ The execution time taken by an algorithm is evaluated while the algorithm is being executed.
→ It is also known as performance analysis that evaluates whether the code is readable or it performs the desired functions.	→ It is also known as Performance measurement that measures the accuracy of algorithm.
→ It focusses on determining the oder of execution of statement.	→ It focusses on determing the time & space complexity of particular algorithm.
→ It provides approximate values	→ It provides accurate values.
→ It is very expensive (depends upon the system which has been used for execution)	→ It is very less expensive (manually calculated)

The performance of any analysis of an algorithm is calculated using two types of complexities

- Space Complexity
- Time complexity.

The space complexity of an algorithm is a amount of memory it needs to run for the completions. It consists of two components → fixed part / static (independent of size of I/o, space for code, constant → variables, simple variables.
→ variable part / dynamic (dependent on the size of variables declared for a problem to be solved i.e., space for referrenced variables, recursion stack space. It is denoted as

$$S(p) = C + S_p$$

- $S(p)$ → problem
- C → static / fixed
- S_p → dynamic variable → instance characteristics

Note: We concentrate only on measuring the/estimating the space required for dynamic path.

→ Space Complexity refers to the worst case and denoted as an asymptotic expression in size of input.

O(n): Space algorithm requires a constant amount of memory for input

O(1): Space algorithm requires a constant amount of space independent of size of input.

Ex: Algorithm sum(a,n)

```
{
    s : 0.0;
   for i := 1 to n do
    s := s+a[i];
    return s;
}
```

First we must identify the number of variables declared in the algorithm. After identifying, allocate one space for each variable. In above algorithm s, i, n will occupy one space each. Since a is declared as array it requires n words of space that 'a' must hold for n elements to be sumed. The total space occupied by above algorithm is n+3.

→ TIME COMPLEXITY

The time complexity is the amount of the compute time it needs to run for completion i.e., sum of compile time and run time (execution)

Compile time does not depends on instant (variable) characteristics. So we concentrate more on runtime to perform.

The types of time complexity are i) Count method.
ii) frequency method iii) Asymptotic Notations.

$$tp(n) = CaADD(n) + CsSUB(n) + CmMUL(n) + \cdots$$

tp(n): problem; instant characteristic; time complexity for addition

→ COUNT METHOD :

We introduce a new variable count into the program, it is a global variable with initial value 0.

Each time a statement in the original program is executed count is incremented by the step count of that statement.

Ex : Algorithm sum(a, n)

```
{
    s := 0.0   // count := count + 1;  ①
    for i := 1 to n do
    {  // count := count + 1
2n      s := s + a[i];  // count := count + 1
    }  // count := count + 1
    return s;  // count := count + 1
}
```

⇒ 2n + 3

2) Algorithm Add(a, b, c, m, n)

```
{
    for i := 1 to m do
    {  count := count + 1
       for j := 1 to n do
       {  count := count + 1
2m     2m    c[i,j] = a[i,j] + b[i,j];  count := count + 1
       }  count := count + 1
    }  count := count + 1
}  count := count + 1
```

⇒ 2mn + 2m + 1

3) Algorithm MUL(a, b, c, m, p, n)

```
{
    for(i := 1 to m do
    {  count := count + 1;
       for j := 1 to p do
       {  count := count + 1;
          c[i,j] := 0;  count := count + 1
          for k := 1 to n do
          {  count := count + 1
2m        3mp   2mnp   c[i,j] := c[i,j] + a[i,k] * b[k,j];  count := count + 1
          }  count := count + 1
       }  count := count + 1
    }  count := count + 1
}  count := count + 1
```

⇒ 2mnp + 3mp + 2m + 1

→ FREQUENCY METHOD :

The 2nd method is said to be frequency method which is to be determine the step count of an algorithm is to build a table in which we list the total no. of steps contributed by each statement.

1st column → Statement in which create the algorithm of a given problem.

2nd column → s/e, which indicate steps for execution of the statement

3rd column → is frequency which indicates the total no. of (frequency) times each statement is executed.

4th column → total steps that is "s/e × frequency".

Ex :

1)

Statement	s/e	frequency	total steps
Algorithm sum(a,n)	0	0	0
{	0	0	0
s := 0.0 ;	1	1	1
for i := 1 to n do	1	n+1	n+1
s := s+a[i];	1	n	n
return s;	1	1	1
}	0	0	0

total : 2n+3

2)

Statement	s/e	frequency	total steps
Algorithm add(a,b,c,m,n)	0	0	0
{	0	0	0
for i := 1 to m do	1	m+1	m+1
{	0	0	0
for j := 1 to n do	1	m(n+1)	m(n+1)
{	0	0	0
c[i,j] := a[i,j] + b[i,j]	1	mn	mn
}	0	0	0
}	0	0	0
}	0	0	0

total : 2mn + 2m + 1

Statement	s/e	frequency	total steps
Algorithm mul(a,b,c,m,p,n)	0		0
{	0		0
for i:=1 to m do	1	m+1	m+1
for j:=1 to p do	1	m(p+1)	m(p+1)
c[i,j]:=0;	1	mp	mp
for k:=1 to n do	1	mp(n+1)	mp(n+1)
c[i,j]:=c[i,j]+a[i,k]*b[k,j];	1	mnp	mnp
}	0		0

total steps: $2mnp+3mp+2m+1$

→ ASYMPTOTIC NOTATIONS

There are used for classification of functions according to the rate of growth of functions. i.e., we try to find the order of growth running time of an algorithm but not the exact running time.

Order of growth:

If algorithms are faster for values of n and slower when n is large then we cannot say these algorithms are good.

This is what we call order of growth.

S.no	Function	Name
1.	1	constant
2.	log n	logarithmic
3.	n	linear
4.	n log n	n logn
5.	n^2	quadratic
6.	n^3	cubic
7.	2^n	exponential
8.	n!	factorial

1) BIG 'O' NOTATION :

The function $f(n) = O\,g(n)$ if and only if there exists positive constant c, n_0 such that $f(n) \leq c * g(n)$ for all values of 'n', $n \geq n_0$.

2) 'Ω' NOTATION :

The function $f(n) = \Omega\, g(n)$ if and only if there exists positive constant c, n_0 such that $f(n) \geq g(n) * c$ for all values of 'n', $n \geq n_0$.

3) 'Θ' NOTATION :

The function $f(n) = \Theta\, g(n)$ if and only if there exists positive constants n, c_1, c_2 such that $c_1 * g(n) \leq f(n) \leq c_2 * g(n)$ for all 'n', $n \geq n_0$.

4) LITTLE 'Oh' NOTATION :

The function $f(n) = o(g(n))$ if and only if

$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ $g(n) \rightarrow$ upperbond.

5) LITTLE 'ω' NOTATION

The function $f(n) = \omega\, g(n)$ if and only if

$\lim_{n \to \infty} \frac{g(n)}{f(n)} = 0$ $f(n) \rightarrow$ upperbond.

→ Problems :-

1) $3n + 3 = O(n)$

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

$f(n) < c * g(n)$

$3n + 3 \leq c * n$

$3n + 3 < 4n$

$n = 1$ $\quad 6 \leq 4$ ✗ $\qquad n = 3$ $\quad 12 \leq 12$ ✓

$n = 2$ $\quad 9 \leq 8$ ✗ $\qquad n \geq 3$

2) $10n^2+4n+2 = O(n^2)$

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

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

$10n^2+4n+2 \le c*n^2$

$10n^2+4n+2 \le 11n^2 \quad (n \ge 5)$

$n=1$ $\quad 16 \le 11(1)$ ×

$n=2$ $\quad 50 \le 22(2)$ ×

$n=3$ $\quad 104 \le 33(3)$ ×

$n=4$ $\quad 160+16+2 \le 44$

$178 \le 44.4$ ×

$n=5$ $\quad 250+20+2 \le 275$

$275 \le 275$ ✓

$n \ge 5$

3) $6*2^n + n^2 = O(2^n)$

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

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

$6\times2^n + n^2 \le c\times2^n$

$6\times2^n + n^2 \le 7*2^n$ $\quad n \ge 1$

$n=1$ $\quad 12+1 \le 7\times2$

$13 \le 14$ ✓

4) $100n+6 = O(n)$

$f(n) \le g(n)\times c$

$100n+6 \le 101n$

$n=1$ $\quad 100+6 \le 101$

$106 \le 101$ ×

$n=2$ $\quad 206 \le 202$ ×

$n=3$ $\quad 306 \le 303$ ×

$n=4$ $\quad 406 \le 404$ ×

$n=5$ $\quad 506 \le 505$ ×

$n=6$ $\quad 606 \le 606$ ✓

$n = 6$

7) $3n+2 = \Omega(n)$

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

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

$3n+2 \ge 3n$

$n=1$ $\quad 5 \ge 3$

$n=2$

$n \ge 1$

8) $3n+3 = \Omega(n)$

$3n+3 \ge 3n$

$n=1$ $\quad 6 \ge 3$.

$n \ge 1$

9) $100n + 6 = \Omega(n)$

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

$100n + 6 \geq 100n$

$n = 1$

$106 \geq 100$ ✓

$\boxed{n \geq 1}$

According to definition of Ω, the no value should be greater than zero always

10) $3n + 3 = \Theta(n)$

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

$n * c_1 \leq 3n + 3 \leq c_2 * n$

$3n \leq 3n + 3 \leq 4n$

$n = 1$

$3 \leq 6 \leq 4$

$n = 2$

$6 \leq 9 \leq 8$

$n = 3$

$9 \leq 12 \leq 12$ ✓

$\boxed{n \geq 3}$

11) $10n^2 + 4n + 2 = \Theta(n^2)$

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

$n^2 * c_1 \leq 10n^2 + 4n + 2 \leq n^2 * c_2$

$10n^2 \leq 10n^2 + 4n + 2 \leq 11n^2$

$n = 1$ $10 \leq 16 \leq 11$ ×

$n = 2$ $40 \leq 50 \leq 44$ ×

$n = 3$ $90 \leq 104 \leq 99$ ×

$n = 4$ $160 \leq 178 \leq 176$ ×

$n = 5$ $250 \leq 272 \leq 275$ ✓

$\boxed{n \geq 5}$

→ FINDING TIME COMPLEXITY USING NOTATIONS.

Statement	S/e	Frequency	Total steps
Algorithm Sum(a,n)	0	0	0
{	0	0	0
s := 0.0;	1	1	1
for i := 1 to n do	1	n+1	n+1
s = s + a[i];	1	n	n
return s;	1	1	1
}	0	0	0

$\Theta(1)+\Theta(n+1)+\Theta(n)+\Theta(1)$

$\Rightarrow \Theta(n+1)+\Theta(n)$ ($\because \Theta(1)$ = constant according to asymptotic notations constants are neglected)

$\Rightarrow \Theta(n)+\Theta(1)+\Theta(n)$

$\Rightarrow 2\Theta(n)$

$\Rightarrow \Theta(n)$

2)

Statement	s/e	Frequency	total steps
Algorithm Add(a,b,c,m,n)	0	0	0
{	0	0	0
for i:=1 to m do	1	m+1	m+1
for j:=1 to n do	1	m(n+1)	mn+m
c[i,j]:=a[i,j]+b[i,j];	1	mn	mn
}			

$\Theta(m+1)+\Theta(mn+m)+\Theta(mn)$

$\Rightarrow \Theta(m)+\Theta(1)+\Theta(mn)+\Theta(m)+\Theta(mn)$

$\Rightarrow 2\Theta(m)+2\Theta(mn)$

$\Rightarrow \Theta(m)+\Theta(mn)$

$\Rightarrow \Theta(mn)$ $\because n < mn$

3)

Statement	s/e	Frequency	total steps
Algorithm mul(a,b,c,m,n)	0	0	0
{	0	0	0
for i:=1 to m do	1	m+1	m+1
for j:=1 to p do	1	m(p+1)	m(p+1)
c[i,j]:=0.0;	1	m*p	mp
for k:=1 to n do	1	mp(n+1)	mp(n+1)
c[i,j]:=c[i,j]+a[i,k]*b[k,j];	1	mnp	mnp
}	0	0	

$\theta(m+1)+\theta(mp+m)+\theta(mnp+mp)+\theta(mnp)+\theta(1)$

$\Rightarrow \theta(m)+\theta(1)+\theta(mp)+\theta(m)+\theta(mnp)+\theta(mp)+\theta(mnp)$ [$\because \theta(1)$ = constant]

$\Rightarrow 2\theta(m)+2\theta(mp)+2\theta(mnp)$

$\Rightarrow \theta(m)+\theta(mp)+\theta(mnp)$

$\Rightarrow \theta(mnp)$

4)

Statement	s/e	frequency	total steps
Algorithm mul(a,b,c,m,n)	0	0	0
{	0	0	0
for i:=1 to n do	1	n+1	n+1
for j:=1 to n do	1	n(n+1)	n^2+n
c[i,j]:=0.0;	1	1	1
for k:=1 to n do	1	$n^2(n+1)$	n^3+n^2
c[i,j]:=c[i,j]+a[i,k]*b[k,j]	1	1	1
}			

$\theta(n+1)+\theta(n^2+n)+\theta(n^3+n^2)+\theta(1)+\theta(1)$

$\Rightarrow \theta(n+1)+\theta(n^2+n)+\theta(n^3+n^2)$ [$\because \theta(1) \rightarrow$ constant]

$\Rightarrow \theta(n)+\theta(n^2)+\theta(n)+\theta(n^3)+\theta(n^2)+\theta(1)$

$\Rightarrow 2\theta(n)+2\theta(n^2)+\theta(n^3)$

$\Rightarrow \theta(n)+\theta(n^2)+\theta(n^3)$

$\Rightarrow \theta(n^3)$

DIVIDE AND CONQUER :-

For a given function to compute n inputs, the divide and conquer strategy splits input into K distinct subsets when K is between $1 < K \leq n$, yielding the K subproblems. These sub problems must be solved and a method must be found to combine the solution of sub-problems into a solution of a given function.

CONTROL ABSTRACTION OF DIVIDE AND CONQUER :-

It is nothing but the algorithm description of divide and conquer.

```
                     divide and conquer
Algorithm  D AND C (P)
{                       └> problem
    if small (P) then return S(P);
    else {
        divide P into smaller instances P1, P2, ---- Pk  K>=1
        Apply D AND C to each of these subproblems;
        return combine (D AND C, D AND C (P2) ---- DANDC(Pk));
    }
}
```

The complexity of many divide & conquer algorithm is given by Recurrence of the Form $T(n) = \begin{cases} T(1) & n = 1 \\ aT(n/b) + f(n) & n > 1 \end{cases}$ where a & b are constants and $n = b^k$,

$K = \log_b n$

In order to find the time complexities for any divide and conquer problems, we have two techniques

i) Substitution Method ii) Recurrence / Master Method.

i) SUBSTITUTION METHOD

In substitution method, the following formulas and a tabular column is used to solve any kind of problem.

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

(2) $T(n) = n^{\log_b a}\,[T(1) + \mu(n)]$ universal length value

$h(n)$	$\mu(n)$
$\Theta(n^r), r<0$	$O(1)$
$\Theta((\log n)^i), i \geq 0$	$\Theta((\log n)^{i+1}/i+1)$
$\Omega(n^r), r>0$	$\Theta(h(n))$

1) $a=1 \quad b=2, f(n)=c$

$$h(n) = \frac{c}{n^{\log 1}_2} = \frac{c}{n^0} = c = c(\,)^0 = c$$

$$\mu(n) = \Theta((\log n)^{0+1}/_{0+1}) \qquad T(n) = n^{\log_b a}[T(1)+\mu(n)]$$

$$\mu(n) = \Theta(\log n) \qquad = n^{\log 1}_2 [T(1) + \Theta(\log n)]$$

$$= n^0 [T(1) + \Theta(\log n)]$$

$$T(n) = T(1) + \Theta(\log n)$$

(∵ according to asymptotic notations, $T(1)$ = constant)

$$\boxed{T(n) = \Theta(\log n)}$$

2) $a=5 \quad b=4 \quad f(n) = cn^2$

$$h(n) = \frac{cn^2}{n^{\log_4 5}} = \frac{cn^2}{n^{1.16}} = c^0 n^{0.84}$$

$$\mu(n) = \Theta(h(n))$$

$$= \Theta(n^{0.84})$$

$$T(n) = n^{\log_b a}\,[T(1) + \mu(n)] = n^{\log_4 5}\,[T(1) + \Theta(n^{0.84})]$$

$= n^{1.16}[T(1) + \theta(n^{0.84})]$

$= n^{1.16}[\theta(n^{0.84})] + T(1)(n^{1.16})$

$= \theta(n^{1.16(0.84)}) = \theta(n^2) + n^{1.16}$ $\boxed{T(n) = \theta(n^2) + n^{1.16}}$

3) $a=7 \quad b=2 \quad f(n) = 18n^2$

$$h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{18n^2}{n^{\log_2 7}} = \frac{18n^2}{n^{2.8}} = 18n^{2-2.8} = 18n^{-0.8}$$

$\mu(n) = O(1)$

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

$= n^{\log_2 7}[T(1) + O(1)]$ ($\because T(1)$ = constant according to asymptotic notations)

$\boxed{T(n) = n^{2.8}}$

4) $a=9 \quad b=3 \quad f(n) = 4xn^6$

$$h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{4xn^6}{n^{\log_3 9}} = \frac{4xn^6}{n^2} = 4n^4 = \mu(n) = \theta(h(n)) = \theta(n^4)$$

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

$= n^2[T(1) + \theta(n^4)]$

$= n^2[\theta(n^4)]$ ($\because T(1)$ = constant according to the asymptotic notations)

$\boxed{T(n) = \theta(n^6)}$

5) $a = 28 \quad b = 3 \quad f(n) = cn^3$

6) $a = 2, b = 2, T(1) = 2, f(n) = n$

7) $a = 2, b = 2 \quad f(n) = cn$

5) $a = 28 \quad b = 3 \quad f(n) = cn^3$

$$h(n) = \frac{cn^3}{n^{\log_3 28}} = \frac{cn^3}{n^3} = c \qquad \mu(n) = \theta(\log n)$$

$$T(n) = n^3 [T(1) + \theta(\log n)]$$

$$= \theta(n^3 \log n)$$

6) $a = 2 \quad b = 2 \quad T(1) = 2 \quad f(n) = n$

$$h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{n}{n^{\log_2 2}} = \frac{n}{n} = 1 = 1 \; (\;)^0 = 1$$

$$h(n) = \theta((\log n)^1)$$

$$T(n) = n^{\log_b a} [T(1) + \mu(n)] \quad (\because T(1) = \text{constant according to asymptotic Notations})$$

$$= n^{\lg 2} [T(1) + \theta(\log n)]$$

$$= n [T(1) + \theta(\log n)] = n\theta(\log n) + 2n$$

$$\boxed{T(n) = \theta(n \log n)}$$

7) $f(n) = cn \quad a = 2 \quad b = 2$

$$h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{cn}{n^{\log_2 2}} = \frac{cn}{n} = c \; (\;)^0 = c$$

$$\mu(n) = \theta(\log n) \quad (\because T(1) = \text{constant according to asymptotic Notations})$$

$$T(n) = n^{\log_2 2} [T(1) + \mu(n)]$$

$$= n [T(1) + \theta(\log n)] = \theta(n \log n)$$

5) $a = 2c \quad b = 3 \quad f(n) = cn^3$

$$h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{cn^3}{n^{\log_3 2c}} = \frac{cn^3}{n^{3.033}} = cn^{-0.033} < 0$$

$$\mu(n) = O(1)$$

$$T(n) = n^{\log_3 2c} [T(1) + O(1)] \quad (\because (T(1)), O(1) \text{ are constants according to asymptotic notations})$$

$$\boxed{T(n) = n^{3.033}}$$

ii) RECURSIVE METHOD

1) $a=2 \quad b=2 \quad f(n)=n$

$T(n) = aT(n/b)+f(n)$

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

$T(n/2) = 2T\left(\frac{n}{2}\cdot\frac{1}{2}\right)+\frac{n}{2}$

$T(n/2) = 2T(n/4)+n/2 \quad - (2)$

Substitute (2) in (1)

$T(n) = 2\left[2T(n/4)+n/2\right]+n$

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

$T(n) = 4T(n/4)+2n \quad - (3)$

$T(n/4) = 4T(n/4\cdot 1/2)+n/4$ (Put $n=n/4$ in equation (1) and calculate $T(n/4)$)

$T(n/4) = 4T(n/8)+n/4 \quad - (4)$

Substitute (4) in (3)

$T(n) = 4\left[2T(n/8)+n/4\right]+2n$

$= 8T(n/8)+4\cdot n/4+2n$

$T(n) = 8T(n/8)+3n \quad - (5)$

$\vdots$

$T(n) = 2^k T\left(n/2^k\right)+k\cdot n$

We know that

$n = b^k$

$k = \log_b n$

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

$T(n) = 2^{\log_2 n} T\left(\frac{2^k}{2^k}\right)+n\log_2 n$

$= n^{\log_2 2} T(1)+n\log_2 n$

$T(n) = n+n\log_2 n$

$\therefore T(n) = \theta(n\log_2 n)$

2> $a=2 \quad b=2 \quad f(n)=cn$.

$T(n) = aT(n/b) + f(n)$

$\boxed{T(n) = 2T(n/2) + cn}$ — ①

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

$T(n/2) = 2T(n/4) + \frac{cn}{2}$

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

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

$T(n/4) = 2T(n/4 \cdot 1/2) + c \cdot n/4$

$T(n/4) = 2T(n/8) + \frac{cn}{4}$

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

$T(n) = 8T(n/8) + 3cn$ — ③

$\boxed{T(n) = 2^k T\left(\frac{n}{2^k}\right) + k(cn)}$

$n = b^k \qquad k = \log_b n$

$n = 2^k \qquad k = \log_2 n$

$T(n) = 2^{\log_2 n} T\left(\frac{2^k}{2^k}\right) + (cn)(\log_2 n)$

$T(n) = n^{\log_2 2} T(1) + (cn)(\log_2 n)$

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

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

$T(n) = cn \log_2 n \Rightarrow \boxed{T(n) = \theta(n \log_2 n)}$

3) $a=1 \quad b=2 \quad f(n)=cn$

$$T(n) = aT(n/b) + f(n)$$

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

$$T(n/2) = T(n/4) + \frac{cn}{2}$$

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

$$T(n) = T(n/4) + \frac{3cn}{2} \quad \text{— (1)}$$

$$T(n/4) = T(n/4 \cdot 1/2) + c \cdot n/4$$

$$T(n/4) = T(n/8) + \frac{cn}{4}$$

$$T(n) = T(n/8) + \frac{cn}{4} + \frac{3cn}{2}$$

$$T(n) = T(n/8) + \frac{7cn}{4}$$

$$\vdots$$

$$T(n) = T(n/8) + cn/4 + \frac{cn}{2} + n$$

$$= T(n/8) + c\left[n/4 + n/2 + n + \dots\right]$$

$$= T(n/8) + cn\left[1 + 1/2 + 1/4 + \dots\right]$$

$$= T(n/8) + nc\left[\frac{1 - (1/2)^k}{1 - 1/2}\right]$$

$$= T(n/8) + 2nc\left[\frac{2^k - 1}{2^k}\right]$$

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

$$= T\left(\frac{2^k}{2^k}\right) + 2 \cdot 2^k \cdot c\left[\frac{2^k - 1}{2^k}\right]$$

$$= T(1) + 2c\left[2^k - 1\right]$$

$$= T(1) + c\left[2^k - 1\right]$$

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

$$= c \cdot 2^k - c \cdot 2$$

$$\log_2 n \qquad n^{\log_2 2}$$

$$\boxed{T(n) = \Theta(n)}$$

$1 + ar + ar^2 + \dots$ and $r < 1$

$$= \frac{a(1 - r^k)}{1 - r}$$

if $r > 1$

$$\frac{a(r^k - 1)}{r - 1}$$

$n = b^k$

$m = 2^k$

$k = \log_2 m$

$n = 2^k$

$k = \log_2 n$

4) $a=2 \quad b=3 \quad f(n)=n^3$

$T(n) = aT(n/b) + f(n)$

$\boxed{T(n) = 2T(n/3) + n^3}$

$T(n/3) = 2T(n/3 \cdot 1/3) + (n/3)^3$

$T(n/3) = 2T(n/9) + (n/3)^3$

$T(n) = 2\left[2T(n/9) + \frac{n^3}{3^3}\right] + n^3$

$T(n) = 4T(n/9) + 2(n/3)^3 + n^3$ — ①

$T(n/9) = 2T(n/9 \cdot 1/3) + \left(\frac{n}{9}\right)^3$

$= 2T(n/27) + \left(\frac{n}{3^2}\right)^3$ — ②

② in ①

$T(n) = 4\left[2T(n/27) + \left(\frac{n}{3^2}\right)^3\right] + 2\left(\frac{n}{3^1}\right)^3 + n^3$

$= 8T(n/27) + 2^2 \cdot \left(\frac{n}{3^2}\right)^3 + 2^1\left(\frac{n}{3^1}\right)^3 + n^3$

$T(n) = 2^K T\left(\frac{n}{3^K}\right) + n^3\left[2^2 \cdot \left(\frac{1}{3^2}\right)^3 + 2^1\left(\frac{1}{3}\right)^3 + 2^0\left(\frac{1}{3}\right)^6 + \cdots\right]$

$= 2^K T(n/3^K) + n^3\{$ constant, can be neglected.

$= 2^K T(n/3^K) + n^3$ $\quad n = b^K$, $n = 3^K$, $K = \log_3 n$

$T(n) = 2^K T(n/3K) + n^3$

$T(n) = 2^{\log_3 n} T\left(\frac{3^2}{3^2}\right) + n^3$

$\boxed{T(n) = \Theta(n^3)}$ (∵ According to order of growth $\log_3 n < n^3$)

4) $a=4 \quad b=2 \quad f(n)=n^2$

$T(n) = aT(n/b) + f(n)$

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

$T(n/2) = 4T(n/2 \cdot 1/2) + \frac{n^2}{2^2}$

$T(n/2) = 4T(n/4) + \frac{n^2}{2^2}$ — (2) $T(n) = 4^2 T(n/4) + 2n^2$ — (1)

$T(n/4) = 4T(n/4 \cdot 1/2) + \frac{n^2}{4^2}$

$T(n/4) = 4T(n/8) + (\frac{n}{4})^2$ — (2)

(2) in (1) $T(n) = 4^2[4T(n/8) + (n/4)^2] + 2n^2$

$= 4^3 T(n/8) + n^2 + 2n^2$

$= 4^3 T(n/8) + 3n^2$

$T(n) = 4^K T(n/2^K) + k \cdot n^2$

$n = b^K \Rightarrow n = 2^K \Rightarrow k = \log_2 n$

$T(n) = 4^k T(\frac{2^k}{2^k}) + \log_2 n \cdot n^2$

$= 4^k T(1) + n^2 \log_2 n$

$T(n) = 4^k T(1) + n^2 \log_2 n = n^{\log_2 4} T(1) + n^2 \log_2 n$

$T(n) = n^2 \log_2 n + 2n$ (∵ According to order of growth $2n < n^2 \log_2 n$)

$\boxed{T(n) = \theta(n^2 \log_2 n)}$

5. $T(n) = \begin{cases} k & n = 1 \\ 3T(n/2) + kn & n > 1 \end{cases}$

$T(n) = 3T(n/2) + kn$

$T(n/2) = 3T(n/2 \cdot 1/2) + k \cdot n/2$

$T(n/2) = 3T(n/4) + k \cdot n/2$

$T(n) = 3[3T(n/4) + k\frac{n}{2}] + kn$

$= 9T(n/4) + \frac{3kn}{2} + kn$

$T(n/4) = 3T(n/4 \cdot 1/2) + k \cdot \frac{n}{4} = 3T(n/8) + \frac{kn}{4}$

$$T(n) = 9\left[3T(n/8) + \frac{kn}{4}\right] + \frac{3kn}{2} + kn$$

$$= 3^3 T\left(\frac{n}{2^3}\right) + \frac{9}{4}kn + \frac{3}{2}kn + kn$$

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

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

$$= 3^k T\left(n/2^k\right) + 2kn\left[(3/2)^k - 1\right] \qquad n = b^k,\ n = 2^k,\ k = \log_2 n$$

$$= 3^k T\left(\frac{2^k}{2^k}\right) + 2kn\,(3/2)^k - 2kn \qquad 3^{\log_2 n},\ n^{\log_2 3}$$

$$= 3^k T(1) + 2kn\,(3/2)^k - 2kn$$

$$= k \cdot 3^k + 2kn\,(3/2)^k - 2kn$$

$$= k \cdot 3^{\log_2 n} + 2kn\,(3/2)^{\log_2 n} - 2kn \qquad n = 2^k,\ k = \log_2 n$$

$$= k \cdot 3^{\log_2 n} + 2kn\,\frac{3^{\log_2 n}}{2^{\log_2 n}} - 2kn \qquad a^{\log_b n} = n^{\log_b a}$$

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

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

6 $T(n) = mT(n/2) + an^2$ P.T $T(n) = O(n^{\log m})$

$$\boxed{T(n) = mT(n/2) + an^2}$$

$$T(n/2) = mT(n/2 \cdot 1/2) + a(n/2)^2$$

$$T(n/2) = mT(n/4) + a(n/2)^2$$

$$T(n) = m\left[mT(n/4) + a(n/2)^2\right] + an^2$$

$$T(n) = m^2 T(n/4) + ma(n/2)^2 + an^2 \quad \text{--- (1)}$$

$$T(n/4) = mT(n/4 \cdot 1/2) + a(n/4)^2$$

$$T(n/4) = mT\left(\frac{n}{8}\right) + a(n/4)^2 \quad \text{--- (2)}$$

$$T(n) = m^2\left[mT(n/8) + a(n/4)^2\right] + ma(n/2)^2 + an^2$$

$$= m^3 T(n/8) + m^2 a(n/4)^2 + ma(n/2)^2 + an^2$$

$$= m^k T(n/2^k) + a\left[(n/2)^2 + m(n/2)^2 + m^2\left(\frac{n}{2^2}\right)^2 + \cdots\right]$$

$$= m^k T\left(n/2^k\right) + an^2\left[1 + \left(\frac{m}{4}\right) + \left(\frac{m}{4}\right)^2 + \cdots\right]$$

$$= m^k T\left(\frac{n}{2^k}\right) + an^2 \cdot m\left(\frac{1-(m/4)^k}{1-1/4}\right)$$

let $m < 1$

$m > 1$

5/4

→ **BINARY SEARCH**

Divide and conquer solves the binary search problem in the following way:

If p is a problem which contains more than one element, it can be divided into new sub problems as

→ Pick the index q in the range i, l and compare x with a[q].

there are 3 possibilities

⇒ x = a[q] the p is immediately solved.

= If $x < a[q]$ x has to be searched for only sublist a_i can $(a_i, a_{i+1}, \ldots a_{q-1})$ and p reduces to $(a_{q-i}, a_i \ldots a_{q-1}, x)$ and
= If $x > a[q]$ the subset to be searched $a_{q+1}, \ldots a_l$ and p reduces to $p(l-q, a_{q+1}, \ldots a_l, x)$
The time taken for the above operations is $O(1)$
the Q is always choosen such that a_q is middle element i.e., $q = \frac{n+1}{2}$, the resultant search algorithm is called binary search.

ALGORITHM FOR BINARY SEARCH USING RECURSIVE AND NON RECURSIVE

i) Recursive Binary Search

```
Algorithm BinSrch(a, i, l, x)
{
    if(l=i) then {  // If p is small
        if(x=a[i]) then return i;
        else return 0;
    }
    else {
        mid := [(i+l)/2]; // reduce p into sub problems
        if(x=a[mid]) then return mid;
        else if(x<a[mid]) then
            return Binsrch(a, i, mid-1, x);
        else return Binsrch(a, mid+1, l, x);
    }
```

ii) Iterative / Non Recursive Binary Search

```
Algorithm BinSearch(a, n, x)
{
    low := 1; high := n;
    while (low ≤ high) do {
        mid := (low+high)/2;
        if(x < a[mid]) then high := mid-1;
        else if(x > a[mid]) then
            low := mid+1;
        else return mid;
    }
```

1) 1 2 3 4 5 6 7 8 9 10 11 12 13 14

-15, -6, 0, 7, 9, 23, 54, 82, 101, 112, 125, 131, 142, 151

x = 151

low	high	mid		
1	14	$\frac{15}{2} = 7$	x = 151 mid[7] = 54	151 > 54 low := mid + 1
8	14	$\frac{22}{2} = 11$	x = 151 mid[11] = 125	151 > 125 low := mid + 1
12	14	$\frac{26}{2} = 13$	x = 151 mid[13] = 142	151 > 142 low := mid + 1
14	14	$\frac{28}{2} = 14$	x = 151 mid[14] = 151	151 = 15

element found.

x = 9 x = -14

low	high	mid		
1	14	$\frac{15}{2} = 7$	x = 9 mid[7] = 54	high := mid - 1;
1	6	$\frac{7}{2} = 3$	9 < 54 x = 9 mid[3] = 0 9 > 0	low := mid + 1;
4	6	5	x = 9 mid[5] = 9	9 = 9

↳ element found.

x = -14

low	high	mid		
1	14	$\frac{15}{2} = 7$	x = -14 mid[7] = 54 -14 < 54	high := mid - 1;
1	6	3	x = -14 mid[3] = 0 -14 < 0	high = mid - 1;
1	2	1	x = -14 mid[1] = -15 -14 > -15	~~high := mid - 1;~~ low := md + 1;
2	2	2	x = -14 mid[2] = -6 -14 < -6	high := mid - 1;

2 | 1 | 1

a = -14

mid[1] = -15

-14 > -15 low = mid+1

2 | 1 | 1

a = -14

mid[1] = -15 low := mid+1

-14 > -15

* Note:

→ If n is in the range $(2^{k-1}, 2^k)$ then binary search makes atmost k element comparisions for a successful search

→ (k-1, k) comparision for an unsuccessful search.

→ Time taken for ~~un~~successful search is $\theta(\log n)$

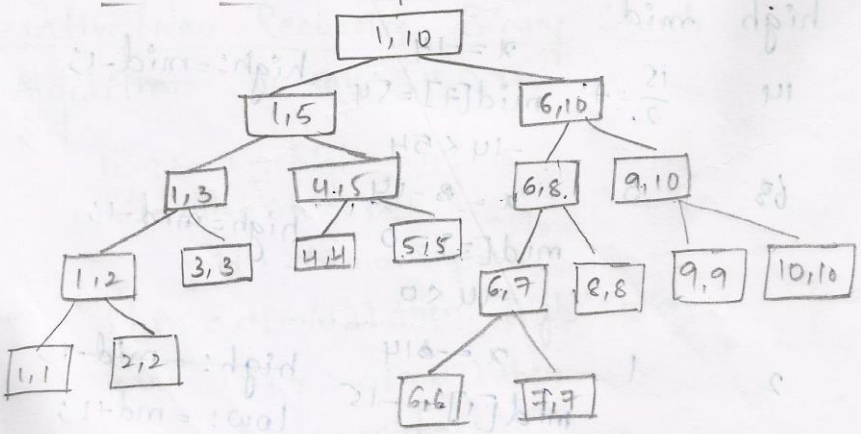
→ Time taken for unsuccessful search is $\theta(\log n)$

→ MERGE SORT:

For a given sequence of n elements a[1], ... a[n], the general idea is to split it into 2 sets i.e., a[1] to $a[n/2]$ and $a[n/2+1]$ ... a[n]. Each set is individually sorted and the resulting sorted sequences are merged to produce a single sorted sequence of n elements.

used in divide & conquer strategy

TREE CALLS OF MERGE SORT:



1) 310, 285, 179, 652, 351, 423, 861, 254, 450, 520

a[1] a[2] a[3]

The given array size elements are 1 to 10. So divide the n size by using mid = low+high/2 ; i,e, l=1 and h=10 and m=5. a[1]--- a[5] is one set. a[6] to a[10] is another set. first Sort the a[1] to a[5].

310	285	179	652	351
a[1]	a[2]	a[3]	a[4]	a[5]

Compare a[0] with a[1]

a[1] > a[2]. So merge a[1] & a[2].

285 310 a[1,2]

a[1,2] is compared with a[3]

So a[1,2] is merged with a[3] because a[1,2] > a[3]

179 285 310 a[1,3]

a[1,3] is compared with a[4,5]

652 351

a[4] a[5]

They are merged [a[4] > a[5]]

351 652

a[4,5]

179 285 310 351 652

↳ set① with Soreted list.

423	861	254	450	520
a[6]	a[7]	a[8]	a[9]	a[10]

Compare a[6] with a[7]. a[6] < a[7]

So no need to sort the elements

a[6,7] = 423 861

Now, a[6,7] is compared with a[8]

a[6,7] > a[8]. So merge, the list

254 423 861

a[6,8]

a[9] is merged with a[10].

a[9] < a[10]. No merge.

a[9,10] = 450 520

Now a[6,8] is merged with a[9,10]

254 423 861 | 450 520

254 423 450 520 861

↳ set② with sorted list

Combine set① & set②

179, 254, 285, ?

520, 652

2. 33, 29, 17, 66, 37, 44, 78, 26, 51, 56

a[1] a[2] a[3] a[4] a[5] a[6] a[7] a[8] a[9] a[10]

$mid = \frac{low + high}{2} = \frac{1+10}{2} = 11/2 = 5$

33 29 17 66 37 | 44 78 26 51 56

a[1] a[2] a[3] a[4] a[5] | a[6] a[7] a[8] a[9] a[10]

a[1] > a[2] So merge a[1] and a[2]

29 33 → a[1,2] is merged with a[3]

17 29 33 a[1,3]

a[4] > a[5] So merge a[4] & a[5]

37 66 → a[4,5]

a[1,3] is merged with a[4,5] to get a[1,5]

17	29	33	37	66

↳ set ① with the sorted list.

a[6] < a[7]. No need of Merging.

a[6,7] = 44 78

a[6,7] is compared with a[8]

26 44 78

a[6,8]

a[9] < a[10]. So no need to merge

a[9,10] = 51 56

a[6,8] is merged with a[9,10] to get a[6,10]

26	44	51	56	78

↳ set ② with the sorted list.

set ① & set ② are combined to give the required sorted elements.

17, 26, 29, 33, 37, 44, 51, 56, 66, 78.

→ are the required sorted elements.

The recurrence relation for merge sort is given as

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

c is constant.

$a=2 \quad b=2 \quad f(n)=cn$

$$h(n)=\frac{f(n)}{n^{\log_b a}}=\frac{cn}{n^{\log_2 2}}=\frac{cn}{n}=c \Rightarrow c(.)^0=c$$

$$\mu(n)=\underbrace{\theta(\log n)^{0+1}}_{0+1}=\theta(\log n)$$

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

$$=n^{\log_2 2}[a+\theta(\log n)]$$

$$=n[\theta(\log n)]=\theta(n\log n)$$

$$\boxed{T(n)=\theta(n\log n)}$$

The time complexity of a Merge Sort in best case, worst and average analysis is $O(n\log n)$

The recurrence relation for binary search is given as

$$T(n)=\begin{cases} a, & n=1 \\ T(n/2)+t & n>1 \end{cases}$$

Algorithm for Merge Sort :-

```
Algorithm Mergesort(low,high)
{
    if (low < high) then
    {
        mid := low+high/2;
        MergeSort(low, mid);
        MergeSort(mid+1,high);
        Mergesort(low, mid+1,high);
    }
}
Algorithm Mergesort (low, mid , high)
{
    h:=low, 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;
      }
      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];
   }
```

→ STRASSEN'S MATRIX MULTIPLICATION :

Let A, B be two matrix of n×n, the product matrix C = A×B is also an n×n matrix whose ith & jth element is found by taking elements in ith row of a and jth row of b by multiplying them to get $c[i,j] = \sum A(i,k) * B(k,j)$ $i \le k \le n$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$C = A * B$$

$$C = \begin{bmatrix} \underbrace{A_{11}B_{11} + A_{12}B_{21}}_{C_{11}} & \underbrace{A_{11}B_{12} + A_{12}B_{22}}_{C_{12}} \\ \underbrace{A_{21}B_{11} + A_{22}B_{21}}_{C_{21}} & \underbrace{A_{21}B_{12} + A_{22}B_{22}}_{C_{22}} \end{bmatrix}$$

$$T(n) = 8T(n/2) + an^2 \qquad n > 2$$

H.W (Recurrence)

The computing time of above matrix is $T(n) = 8T(n/2) + an^2$, $n>2$ i,e, $T(n) = \theta(n^3)$. The total computations it requires is 8 multiplication and 4 additions. In order to reduce the above computing time using divide & conquer policy Volker Strassen has introduced a method where you require 7 multiplications and 18 add/subtraction., i.e., $T(n) = 7T(n/2) + an^2$ $n>2$ H.W, $T(n) = \theta(n^{\sim 2.81})$. This method involves computing of 7 $n/2 * n/2$ matrixes i,e.,

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

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

$R = A_{11}(B_{12} - B_{22})$

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

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

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

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

$C_{11} = P + S - T + V$

$C_{12} = R + T$

$C_{21} = Q + S$

$C_{22} = P + R - Q + U$.

1) $A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$; $A_{11}=1$, $A_{12}=1$, $A_{21}=1$, $A_{22}=1$
$B_{11}=0$, $B_{12}=0$, $B_{21}=1$ $B_{22}=1$

$P = (1+1)(0+1)$

$P = 2+0 = 2$

$Q = (2)(0) = 0$

$R = 1(-1) = -1$

$S = 1(1) = 1$

$T = (2)(1) = 2$

$U = (1-1)(0+0) = 0$

$V = (1-1)(1+1) = 0$

$C_{11} = 2 + 1 - 2 + 0$

$C_{11} = +1$

$C_{12} = -1 + 2 = 1$

$C_{21} = 0 + 1 = 1$

$C_{22} = 2 - 1 - 0 + 0$

$= +1$

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

2) $A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 6 & 0 & 3 \\ 4 & 1 & 1 & 2 \\ 0 & 3 & 5 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 1 & 4 & 2 & 7 \\ 3 & 1 & 3 & 5 \\ 2 & 0 & 1 & 3 \\ 1 & 4 & 5 & 1 \end{bmatrix}$

$A_{11} = \begin{bmatrix} 1 & 2 \\ 0 & 6 \end{bmatrix}$ $A_{12} = \begin{bmatrix} 3 & 4 \\ 0 & 3 \end{bmatrix}$; $B_{11} = \begin{bmatrix} 1 & 4 \\ 3 & 1 \end{bmatrix}$ $B_{12} = \begin{bmatrix} 2 & 7 \\ 3 & 5 \end{bmatrix}$

$A_{21} = \begin{bmatrix} 4 & 1 \\ 0 & 3 \end{bmatrix}$ $A_{22} = \begin{bmatrix} 1 & 2 \\ 5 & 0 \end{bmatrix}$; $B_{21} = \begin{bmatrix} 2 & 0 \\ 1 & 4 \end{bmatrix}$ $B_{22} = \begin{bmatrix} 1 & 3 \\ 5 & 1 \end{bmatrix}$

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

$P = \begin{pmatrix} a_{11} & a_{12} \\ 2 & 4 \\ 5 & 6 \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} b_{11} & b_{12} \\ 2 & 7 \\ 8 & 2 \\ b_{21} & b_{22} \end{pmatrix} = \begin{bmatrix} 16 & 22 \\ 58 & 47 \end{bmatrix}$

$Q = \begin{bmatrix} 1 & 4 \\ 3 & 1 \end{bmatrix}\left[\begin{pmatrix} 4 & 1 \\ 0 & 3 \end{pmatrix} + \begin{pmatrix} 1 & 2 \\ 5 & 0 \end{pmatrix}\right] = \begin{bmatrix} 1 & 4 \\ 3 & 1 \end{bmatrix}\begin{bmatrix} 5 & 3 \\ 5 & 3 \end{bmatrix} = \begin{bmatrix} 25 & 15 \\ 20 & 12 \end{bmatrix}$

$R = \begin{bmatrix} 1 & 2 \\ 0 & 6 \end{bmatrix}\left[\begin{pmatrix} 2 & 7 \\ 3 & 5 \end{pmatrix} - \begin{pmatrix} 1 & 3 \\ 5 & 1 \end{pmatrix}\right] = \begin{bmatrix} 1 & 2 \\ 0 & 6 \end{bmatrix}\begin{bmatrix} 1 & 4 \\ -2 & 4 \end{bmatrix} = \begin{bmatrix} -3 \end{bmatrix}$

$p = (2+6)(4) = 8(4) = 32$

$q = 2(11) = 22$

$r = 2(5) = 10$

$s = 6(6) = 36$

$t = 6(2) = 12$

$u = 3(9) = 27$

$v = (-2)(10) = -20$

$C_{11} = P + S - T + V$

$= 32 + 36 - 12 - 20$

$= 68 - 32$

$C_{11} = 36$

$C_{12} = R + T = 22$

$C_{21} = Q + S = 22 + 36 = 58$

$C_{22} = P + R - Q + U = 32 + 10 - 22 + 27$

$C_{22} = 69 - 22 = 47$

$$P = \begin{bmatrix} 36 & 22 \\ 58 & 47 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1 & 4 \\ 3 & 1 \end{bmatrix}\left[\begin{pmatrix} 4 & 1 \\ 0 & 3 \end{pmatrix} + \begin{pmatrix} 1 & 2 \\ 5 & 0 \end{pmatrix}\right] = \begin{bmatrix} b_{11} & b_{12} \\ 1 & 4 \\ 3 & 1 \\ b_{21} & b_{22} \end{bmatrix}\begin{bmatrix} a_{11} & a_{12} \\ 5 & 3 \\ 5 & 3 \\ a_{21} & a_{22} \end{bmatrix}$$

$p = (A_{11}+A_{22})(B_{11}+B_{22}) = (2)(8) = 16$

$q = B_{11}(A_{21}+A_{22}) = 5(3) = 15$

$r = A_{11}(B_{12}-B_{22}) = 1(3-3) = 0$

$s = A_{22}(B_{21}-B_{11}) = 1(5-3) = 2$

$t = (A_{11}+A_{12})B_{22} = 3(5) = 15$

$u = (A_{21}-A_{11})(B_{11}+B_{12}) = (8)(2) = 16$

$v = (A_{12}-A_{22})(B_{21}+B_{22}) = 3(8) = 24$

$C_{11} = P+S-T+V = 16+6-8-0 = 14$

$C_{12} = R+T = 15+8 = 23$

$C_{21} = Q+S = 8+6 = 14$

$C_{22} = P+R-Q+U = 16+15-8+0 = 23$

$$R = \begin{bmatrix} 1 & 2 \\ 0 & 6 \end{bmatrix}\left[\begin{pmatrix} 2 & 7 \\ 3 & 5 \end{pmatrix} - \begin{pmatrix} 1 & 3 \\ 5 & 1 \end{pmatrix}\right] = \begin{bmatrix} a_{11} & a_{12} \\ 1 & 2 \\ 0 & 6 \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} b_{11} & b_{12} \\ 1 & 4 \\ -2 & 4 \\ b_{21} & b_{22} \end{bmatrix}$$

$p = (7)(5) = 35$

$q = (1)(6) = 6$

$r = 1(0) = 0$

$s = 6(-3) = -18$

$t = (3)(4) = 12$

$u = (-1)(4) = -4$

$v = (-4)(2) = -8$

2) $A=\begin{bmatrix}1&2&3&4\\0&6&0&3\\4&1&1&2\\0&3&5&0\end{bmatrix}$ $B=\begin{bmatrix}1&4&2&7\\3&1&3&5\\2&0&1&3\\1&4&5&1\end{bmatrix}$

$A_{11}=\begin{bmatrix}1&2\\0&6\end{bmatrix}$ $A_{21}=\begin{bmatrix}4&1\\0&3\end{bmatrix}$ $B_{11}=\begin{bmatrix}1&4\\3&1\end{bmatrix}$ $B_{12}=\begin{bmatrix}2&7\\3&5\end{bmatrix}$

$A_{12}=\begin{bmatrix}3&4\\0&3\end{bmatrix}$ $A_{22}=\begin{bmatrix}1&2\\5&0\end{bmatrix}$ $B_{21}=\begin{bmatrix}2&0\\1&4\end{bmatrix}$ $B_{22}=\begin{bmatrix}1&3\\5&1\end{bmatrix}$

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

$=\left[\begin{pmatrix}1&2\\0&6\end{pmatrix}+\begin{pmatrix}1&2\\5&0\end{pmatrix}\right]\left[\begin{pmatrix}1&4\\3&1\end{pmatrix}+\begin{pmatrix}1&3\\5&1\end{pmatrix}\right]$

$a_{11}\ a_{12}\quad b_{11}\ b_{12}$

$P=\begin{bmatrix}2&4\\5&6\end{bmatrix}\begin{bmatrix}2&7\\8&2\end{bmatrix}$ $a*b$

$a_{21}\ a_{22}\quad b_{21}\ b_{22}$

$p=4(2+6)=4(8)=32$

$q=2(11)=22$

$r=2(5)=10$

$s=6(6)=36$

$t=6(2)=12$

$u=3(9)=27$

$v=(-2)(10)=-20$

$C_{11}=P+S-T+V$

$C_{11}=32+36-12-20=36$

$C_{12}=R+T$

$C_{12}=10+12=22$

$C_{21}=Q+S$

$C_{21}=22+36=58$

$C_{22}=P+R-Q+U=32+10-22+27=47$

$P=\begin{bmatrix}36&22\\58&47\end{bmatrix}$

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

$a_{11}\ a_{12}\quad b_{11}\ b_{12}$

$Q=\left[\begin{pmatrix}4&1\\0&3\end{pmatrix}+\begin{pmatrix}1&2\\5&0\end{pmatrix}\right]\begin{bmatrix}1&4\\3&1\end{bmatrix}=\begin{bmatrix}5&3\\5&3\end{bmatrix}\begin{bmatrix}1&4\\3&1\end{bmatrix}$

$b_{21}\ a_{22}\quad b_{21}\ b_{22}$

$p=(8)2=16$

$q=1(8)=8$

$r=5(3)=15$

$s=3(2)=6$

$t=1(8)=8$

$u=0$

$v=0$

$Q_{11}=P+S-T+V=16+6-8+0=14$

$C_{12}=R+T=15+8=23$

$C_{21}=Q+S=8+6=14$

$C_{22}=P+R-Q+U=16+15-8+0=23$

$Q=\begin{bmatrix}14&23\\14&23\end{bmatrix}$

$R = A_{11}(B_{12} - B_{22})$

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

$a_{11}\ a_{12}\ b_{11}\ b_{12}$

$a_{21}\ a_{22}\ b_{21}\ b_{22}$

$p = 7(5) = 35$

$q = 6(1) = 6$

$r = 1(0) = 0$

$s = 6(-3) = -18$

$t = 3(4) = 12$

$u = (-1)5 = -5$

$v = (-4)2 = -8$

$C_{11} = 35 - 18 + 12 - 8 = -3$

$C_{12} = 0 + 12 = 12$

$C_{21} = 6 - 18 = -12$

$C_{22} = 35 + 0 - 6 - 5 = 35 - 11 = 24$

$$R = \begin{bmatrix} -3 & 12 \\ -12 & 24 \end{bmatrix}$$

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

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

$a_{11}\ a_{12}\ b_{11}\ b_{12}$

$a_{21}\ a_{22}\ b_{21}\ b_{22}$

$p = (1+0)(1+3) = 4$

$q = 5(1) = 5$

$r = 1(-4-3) = -7$

$s = 0$

$t = (1+2)3 = 9$

$u = 4(-3) = -12$

$v = 2(1) = 2$

$C_{11} = P + S - T + V = 4 + 0 - 9 + 2 = -3$

$C_{12} = R + T = -7 + 9 = 2$

$C_{21} = Q + S = 0 + 5 = 5$

$C_{22} = R + T\ \ P + R - Q + V$

$= 4 - 7 - 5 - 12 = 4 - 24 = -20$

$$S = \begin{bmatrix} -3 & 2 \\ 5 & -20 \end{bmatrix}$$

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

$\left(\begin{bmatrix} 1 & 2 \\ 0 & 6 \end{bmatrix} + \begin{bmatrix} 4 & 4 \\ 0 & 3 \end{bmatrix}\right)\begin{bmatrix} 1 & 3 \\ 5 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 6 \\ 0 & 9 \end{bmatrix}\begin{bmatrix} 1 & 3 \\ 5 & 1 \end{bmatrix} =$

$a_{11}\ a_{12}\ b_{11}\ b_{12}$

$a_{21}\ a_{22}\ b_{21}\ b_{22}$

$p = (5+9)(1+1) = 2(14) = 28$

$q = (9)(1) = 9$

$r = 5(3-1) = 10$

$s = 9(4) = 36$

$t = (5+3)(1) = 8$

$u = 4(-5) = -20$

$v = 6(-6) = -36$

$C_{11} = P + S - T + V$

$= 28 + 36 - 8 - 36$

$C_{11} = 20$

$C_{12} = R + T = 10 + 8 = 18$

$C_{21} = Q + S = 9 + 36 = 45$

$C_{22} = P + R - Q + V = 28 + 10 - 9 - 20$

$= 38 - 20 - 9 = 18 - 9$

$C_{22} = 9$

$$T = \begin{bmatrix} 34\ 20 & 18 \\ 45 & 9 \end{bmatrix}$$

$U = (a_{21} - a_{11})(b_{11} + b_{12})$

$= \left[\begin{pmatrix} 4 & 1 \\ 0 & 3 \end{pmatrix} - \begin{pmatrix} 1 & 2 \\ 0 & 6 \end{pmatrix}\right]\left[\begin{pmatrix} 1 & 4 \\ 3 & 1 \end{pmatrix} + \begin{pmatrix} 2 & 7 \\ 3 & 5 \end{pmatrix}\right]$

$\begin{matrix} a_{11} & a_{12} & b_{11} & b_{12} \end{matrix}$

$= \begin{bmatrix} 3 & -1 \\ 0 & -3 \end{bmatrix}\begin{bmatrix} 3 & 11 \\ 6 & 6 \end{bmatrix}$

$\begin{matrix} a_{21} & a_{22} & b_{21} & b_{22} \end{matrix}$

$P = (3-3)(9) = 0$

$q = 3(0-3) = -9$

$r = 3(11-6) = 15$

$s = (-3)(3) = -9$

$t = (3-1)6 = 12$

$u = (0-3)(14) = -3(14) = -42$

$v = (-1+3)(12) = 2(12) = 24$

$C_{11} = P+S-T+V = 24 - 21 = 3$

$C_{12} = R+T = 15+12 = 28$

$C_{21} = Q+S = -9-9 = -18$

$C_{22} = P+R-Q+U$

$= 0+15+9-42 = -18$

$U = \begin{bmatrix} 3 & 28 \\ -18 & -18 \end{bmatrix}$

$V = (a_{12} - a_{22})(b_{21} + b_{22})$

$V = \left[\begin{pmatrix} 3 & 4 \\ 0 & 3 \end{pmatrix} - \begin{pmatrix} 1 & 2 \\ 5 & 0 \end{pmatrix}\right]\left[\begin{pmatrix} 2 & 0 \\ 1 & 4 \end{pmatrix} + \begin{pmatrix} 1 & 3 \\ 5 & 1 \end{pmatrix}\right]$

$\begin{matrix} a_{11} & a_{12} & b_{21} & b_{12} \end{matrix}$

$V = \begin{bmatrix} 2 & 2 \\ -5 & 3 \end{bmatrix}\begin{bmatrix} 3 & 3 \\ 6 & 5 \end{bmatrix}$

$\begin{matrix} a_{21} & a_{22} & b_{21} & b_{22} \end{matrix}$

$p = 5(8) = 40$

$q = (-5+3)(3) = 3(-2) = -6$

$r = 2(3-5) = 2(-2) = -4$

$s = 3(3) = 9$

$t = 4(5) = 20$

$u = 6(-7) = -42$

$v = (-1)(11) = -11$

$C_{11} = P+S-T+V = 18$

$C_{12} = R+T = -4+20 = 16$

$C_{21} = Q+S = 3$

$C_{22} = P+R-Q+U$

$= 40-4+6-42$

$= 46-46 = 0$

$V = \begin{bmatrix} 18 & 16 \\ 3 & 0 \end{bmatrix}$

$C_{11} = P+S-T+V = \begin{bmatrix} 36 & 22 \\ 58 & 47 \end{bmatrix} + \begin{bmatrix} -3 & 2 \\ 5 & -20 \end{bmatrix} - \begin{bmatrix} 30 & 18 \\ 45 & 9 \end{bmatrix} + \begin{bmatrix} 18 & 16 \\ 3 & 0 \end{bmatrix}$

$- \begin{bmatrix} 17 & 22 \\ 21 & 18 \end{bmatrix}$

36
-3
33
-20
13
18

$C_{12} = R + T$

$$= \begin{bmatrix} -3 & 12 \\ -12 & 24 \end{bmatrix} + \begin{bmatrix} 34 & 18 \\ 45 & 9 \end{bmatrix} = \begin{bmatrix} 31 & 30 \\ 33 & 33 \end{bmatrix}$$

$C_{21} = Q + S$

$$= \begin{bmatrix} 14 & 23 \\ 14 & 23 \end{bmatrix} + \begin{bmatrix} -3 & 2 \\ 5 & -20 \end{bmatrix} = \begin{bmatrix} 11 & 25 \\ 19 & 3 \end{bmatrix}$$

$C_{22} = P + R - Q + U$

$$= \begin{bmatrix} 36 & 22 \\ 58 & 47 \end{bmatrix} + \begin{bmatrix} -3 & 12 \\ -12 & 24 \end{bmatrix} - \begin{bmatrix} 14 & 23 \\ 14 & 23 \end{bmatrix} + \begin{bmatrix} 3 & 27 \\ -18 & -18 \end{bmatrix} = \begin{bmatrix} 22 & 39 \\ 14 & 30 \end{bmatrix}$$

47
24
71

$$C = \begin{bmatrix} 17 & 22 & 31 & 30 \\ 21 & 18 & 33 & 33 \\ 11 & 25 & 22 & 39 \\ 19 & 3 & 14 & 30 \end{bmatrix}$$

1) $T(n) = 8T(n/2) + an^2$, $n > 2$ by Recurrence Method.

$a = 8 \quad b = 2 \quad f(n) = an^2$

$T(n/2) = 8T(n/2 \cdot 1/2) + a(n/2)^2$

$T(n/2) = 8T(n/4) + \frac{an^2}{4}$

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

$= 8^2 T(n/4) + \frac{8^2 an^2}{4} + an^2$

$T(n) = 64T(n/4) + 3an^2$

$T(n/4) = 8T(n/4 \cdot 1/2) + a\left(\frac{n}{4}\right)^2$

$T(n/4) = 8T(n/8) + \frac{an^2}{16}$

$$T(n) = 64\left[8T\left(n/8\right) + \frac{an^2}{16}\right] + 3an^2$$

$$= 8^3 T\left(n/8\right) + \frac{64}{16} an^2 + 3an^2$$

$$T(n) = 8^3 T\left(n/8\right) + 4an^2 + 3an^2$$

$$T(n) = 8^3 T\left(n/2^3\right) + 3an^2 + 4an^2 = 8^3 T\left(n/2^3\right) + 7an^2$$

$$T(n) = 8^K T\left(n/2^K\right) + \left[an^2 + 3an^2 + 5an^2 + \cdots\right] \text{ (or)}$$

$$T(n) = 8^K T\left(n/2^K\right) + an^2\left[1 + 3 + 5 + 7 + \cdots\right]$$

$an^2 + 2an^2 + 3an^2 + 4an^2$

$= an^2\left[1+2+3+\cdots\right]$

$n = b^K$

$n = 2^K \Rightarrow K = \log_b n \Rightarrow K = \log_2 n$

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

$= \frac{an^3(n+1)}{2}$

$= 8^{\log_2 n} T\left(\frac{2^K}{2^K}\right) + \underbrace{an^2\left[1+3+5+\cdots\right]}_{\text{constant}}$

$= \underbrace{\text{constant}}$

$= n^3 + \text{constant}$

$$\boxed{T(n) = \Theta(n^3)}$$

2) $T(n) = 7T\left(n/2\right) + an^2, \quad n > 2$

$a = 7 \quad b = 2 \quad f(n) = an^2$

$$T\left(n/2\right) = 7T\left(n/2 \cdot 1/2\right) + a\left(n/2\right)^2$$

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

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

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

$$T\left(n/4\right) = 7T\left(n/4 \cdot 1/2\right) + a\left(n/4\right)^2$$

$$T\left(n/4\right) = 7T\left(n/8\right) + \frac{an^2}{16}$$

$$T(n) = 7^2\left[7T(n/8) + \frac{an^2}{16}\right] + 7/4\,an^2 + an^2$$

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

$$\vdots$$

$$T(n) = 7^k(n/2^k) + an^2\left[1 + 7/4 + (7/4)^2 + \cdots\right]$$

$n = b^k$

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

$$= 7^{\log_2 n}T\left(\frac{2^k}{2^k}\right) + an^2\left[\frac{(7/4)^k - 1}{7/4 - 1}\right]$$

$$= 7^{\log_2 n}T(1) + an^2\left[\frac{(7/4)^k - 1}{3/4}\right]$$

$$= n^{\log_2 7}T(1) + \frac{4}{3}an^2\left[\frac{7^k - 4^k}{4^k}\right]$$

(the bracketed term is constant)

$$= n^{2.80}T(1)$$

$$\boxed{T(n) = \Theta(n^{2.8})}$$

→ QUICK SORT :-

i) In merge sort sub arrays or sub problems are sorted independently and later merged.
Whereas in quicksort the division of sub problems is made so that the sorted sub problems need not to be merged.

ii) using divide & conquer strategy the quick sort uses partition of the elements and finding the solutions of element.

iii) let A[p---r] be an array which is divided into sub problems as

A[p] ... a[q-1] A[q] A[q+1] --- A[r] where A[q] is pivot element

$\leq a[q]$ (pivot) $\geq A[q]$

Conditions for quick sort :-

i) L ←——— R

↑ pivot

$p < r$?

if no → then swap pivot & right

$p < r$?

if yes → then move right one position towards left.

ii) L ⟶ R
↑ pivot

If p > left?

if NO → then swap right & left.

If p > left?

If yes → then move left one position towards right.

*. Show how quick sort sorts the following sequence of keys.

1) 1, 1, 1, 1, 1, 1, 1 2) 5, 5, 8, 3, 4, 3, 2

1, 1, 1, 1, 1, 1, 1
↑ P,L ↑ R

p < r ? no then swap

1, 1, 1, 1, 1, 1, 1
↑ L ↑ P,R

1, 1, 1, 1, 1, 1, 1
↑ P,L ↑ R

1, 1, 1

ALGORITHM :-

```
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)
{
    p := a[i];
    a[i] := a[j];
    a[j] := p;
}
Algorithm Quicksort(p,q)
{ if (p < q) then // if there are more than one element.
   {
      j := partition (a,p,q+1);   j = pivot element
   // divide p into sub problems
   // j is the position of partition element.
   Quicksort (p, j-1);
   Quicksort (j+1, q);
  }
}
```

When you go for analysis of quick sort we count only number of element comparision C(n).

Best Case :- Quick sort works best if each array is divided into two equal sub arrays of size $n/2$. This generates $\log(n)$ levels in the recursion tree. The recurrence relation for Best case is given by

$T(n) = 2T(n/2) + cn$ when $n > 2$.

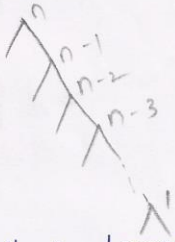
$$h(n) = \frac{cn}{n^{\log_2^2}} = c(\ )^0 = c$$

$$\mu(n) = \theta(\log n)$$

$$T(n) = n^{\log_2^2}[T(1) + \mu(n)] = n^1[\theta(\log n) + T(1)]$$

$$\boxed{T(n) = \theta(n \log n)}$$

Worst Case Analysis :- If the choosen pivot element is either smallest or largest in an array. In this case one part will be empty and other contains the remaining elements which generates $n-1$ levels in recursion tree.



If pivot is smallest element then the recursive relation is given as $T(n) = T(n-1) + cn$ where $n > 1$

$$T(n) = T(n-1) + cn$$
$$T(n-1) = T(n-2) + c(n-1)$$
$$T(n-2) = T(n-3) + c(n-2)$$
$$T(n-3) = T(n-4) + c(n-3)$$
$$\vdots$$
$$T(n) = T(n-1) + T(n-2) + T(n-3) + \cdots + T(2) + T(1) + c(n-1) + c(n-2) + c(n-3) \cdots + c(2) + c(1)$$
$$T(n) = T(1) + c[1 + 2 + 3 + \cdots + n]$$
$$T(n) = T(1) + c \cdot \frac{n(n+1)}{2}$$
$$T(n) = c \cdot \frac{n(n+1)}{2} \quad (\because T(1) = \text{constant})$$

$T(n) = n(n+1)$ (∵ 1/2 is constant)

$T(n) = n^2$

$\boxed{T(n) = \Theta(n^2)}$

2) 5, 5, 8, 3, 4, 3, 2

P,L ↑ R ↑

P < R ? NO

2, 5, 8, 3, 4, 3, 5 (P ↓, R ↑)

2, 5, 8, 3, 4, 3, 5 (L ↑, P ↓, R ↑)

2, 5, 8, 3, 4, 3, 5 (L ↑, P,R ↑)

2, 5, 5, 3, 4, 3, 8 (P,L ↓, R ↑)

2, 5, 5, 3, 4, 3, 8 (P,L ↓, R ↑)

2, 5, 3, 3, 4, 5, 8 (L ↑, P,R ↑)

2, 5, 3, 3, 4, 5, 8 (P,L ↓, R ↑)

2, 5, 3, 3, 4, 5, 8 (P,L ↓, R ↑)

2, 5, 3, 3, 4 | 5, 8 (P,L ↓, R ↑)

2, 5, 3, 3, 4 | 5, 8 (P,L ↓, R ↑)

2, 5, 3, 3, 4 | 5, 8 (P,L →, R ↑)

2, 5, 3, 3, 4 | 5, 8 (P,L →, R ↑)

2, 5, 3, 3, 4 | 5, 8 (P,L →, R ↑)

2 | 5, 3, 3, 4 | 5, 8 (P,L ↓, R ↑)

2 | 4, 3, 3, 5 | 5, 8 (L ↑, P ↓, R ↑)

2 | 4, 3, 3, 5 | 5, 8 (L ↑, P,R ↑)

2 | 4, 3, 3, 5 | 5, 8
↑L ↑P,R

2 | 4, 3, 3, | ↓P 5, 5, 8
↑P,L ↑R

2 | 3, 3, 4 | 5, 5, 8
↑L ↑P ↑R

2, 3, 3, 4, 5, 5, 8

* ** *** Imp :- Average Case Analysis of Quick Sort :-

$$T_A(n) = (n+1) + \frac{1}{n} \sum_{1 \le k \le n} [T_A(k-1) + T_A(n-k)]$$

n+1 = no. of repetitions.
A - Average case

Consider

$$\sum_{1 \le k \le n} [T_A(k-1) + T_A(n-k)] \Rightarrow$$

if $k=1 \Rightarrow T_A(0) + T_A(n-1)$
if $k=2 \Rightarrow T_A(1) + T_A(n-2)$
if $k=3 \Rightarrow T_A(2) + T_A(n-3)$
⋮
$k=n \Rightarrow T_A(n-1) + T_A(0)$

* n -①
n = n-1 -②
① - ② ÷ n(n+1) - ③ - b
n = n-1 n = n-2 - c

$$\Rightarrow T_A(n) = n+1 + \frac{1}{n}[2T_A(0) + 2T_A(1) + 2T_A(2) + \dots + 2T_A(n-2) + 2T_A(n-1)]$$

$$\Rightarrow T_A(n) = (n+1) + \frac{1}{n} \cdot 2[T_A(0) + T_A(1) + T_A(2) + \dots + T_A(n-2) + T_A(n-1)]$$

Multiply LHS & RHS by 'n'.

$$\Rightarrow nT_A(n) = n(n+1) + \frac{n}{n} \cdot 2[T_A(0) + T_A(1) + \dots + T_A(n-2) + T_A(n-1)]$$

$$\Rightarrow nT_A(n) = n(n+1) + 2[T_A(0) + T_A(1) + \dots + T_A(n-2) + T_A(n-1)] \quad \text{—①}$$

Put n = n-1 in equation ①

$$\Rightarrow (n-1)T_A(n-1) = (n-1)(n-2+1) + 2[T_A(0) + T_A(1) + \dots + T_A(n-3) + T_A(n-2)]$$

$$\Rightarrow (n-1)T_A(n-1) = n(n-1) + 2[T_A(0) + T_A(1) + \dots + T_A(n-3) + T_A(n-2)] \quad \text{—②}$$

$$\frac{T_A(n)}{n+1} = \frac{T_A(n-1)}{n} + \frac{T_A(n-2)}{n-1} + \frac{T_A(n-3)}{n-2} + \cdots + \frac{T_A(1)}{2} +$$

$$\frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \cdots$$

$$\frac{T_A(n)}{n+1} = \frac{T(1)}{2} + 2\left[\frac{1}{n+1} + \frac{1}{n} + \frac{1}{n-1} + \cdots\right]$$

$$= 2 \sum_{2 \le K \le n+1} 1/K \qquad (\because T(1) \text{ is constant})$$

$$\frac{T_A(n)}{n+1} = 2 \int_3^{n+1} 1/K \, dx$$

$$\frac{T_A(n)}{n+1} = 2 \left[\log K\right]_3^{n+1}$$

$$\frac{T_A(n)}{n+1} = 2\left[\log(n+1) - \log 3\right] = 2\log n + 2\log 1 - 2\log 3$$

or

$$\frac{T_A(n)}{n+1} = 2\left[\log(n+1)\right] \qquad (\because \log 3 \text{ is constant})$$

$$T_A(n) = 2(n+1)\log(n+1)$$

$$T_A(n) = n\log(n+1) + \log(n+1)$$

$$T_A(n) = n\log(n+1) \qquad \text{neglecting}$$

$$T_A(n) = n\log(n) \qquad (\because 1 \text{ is constant and we can neglect it})$$

$$\therefore \boxed{T_A(n) = O(n\log n)}$$

3/2/15

UNIT - 2

SEARCHING AND TRAVERSING TECHNIQUES

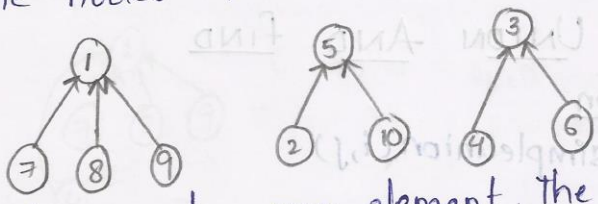
→ **DISJOINT SET OPERATION :-**

For n=10 the elements can be partitioned into 3 disjoint sets. $S_1 = \{1,7,8,9\}$

$S_2 = \{2,5,10\}$

$S_3 = \{3,4,6\}$

Each set is represented as a tree where each set is link the nodes from children to the parent.

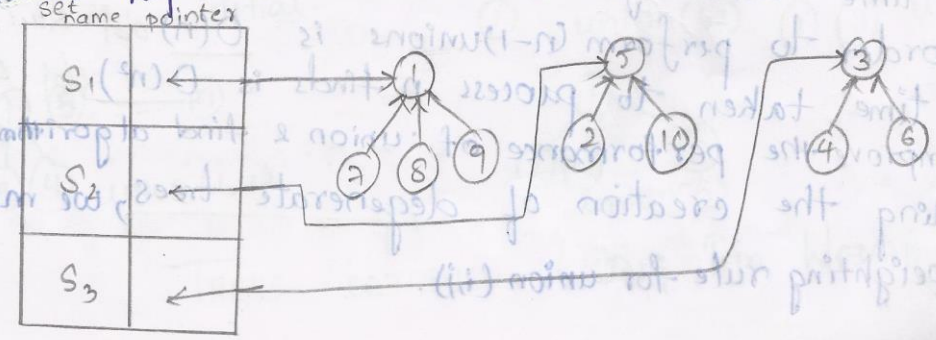


Root node can be any element. The operations to be performed on this sets are i) Disjoint set union :- if S(i) and S(j) are two disjoint set then their union $S_i \cup S_j$ = all elements x such that x is in S_i or S_j. i.e., $S_1 \cup S_2 = \{1,2,5,7,8,9,10\}$

ii) find (i) : Given the element i, find set containing i i.e., determines the root of tree containing element i. The determination of i is done until we reach a node with parent value -1.

iii) Data Representation And Array Representation of Sets :

a) Data Representation



b) Array Representation

i	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
P	-1	5	-1	3	-1	3	1	-1	1	5

In data representation the sets are represented in form of set table which contains two columns set name and the pointer which points to the tree representations of sets

In array representation the set elements are numbered from 1 to n i,e., P[1:n] where n is max. of n elements. The ith element of this array represents tree node that contains element i.

2M
→ * ALGORITHM FOR UNION AND FIND

i) Algorithm for union

```
Algorithm simpleunion(i,j)
{
    P[i]:=j;
}
Algorithm simpleFind(i)
{
    while(P[i]>=0) do
        i:=P[i];
    return i;
}
```

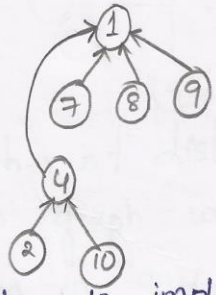
Sequence of union & Find operation

* union(1,2), union(2,3), union(3,4), union(4,5)- - - - - u(n-1,n)
find(1), find(2)- - - - find(n).

→ The time taken for one union is constant.
→ In order to perform (n-1) unions is O(n) obj Ques.
→ The time taken to process n finds is O(n²) obj Ques.
→ To improve the performance of union & find algorithm by avoiding the creation of degenerate trees, we make use of weighting rule for union(i,j).

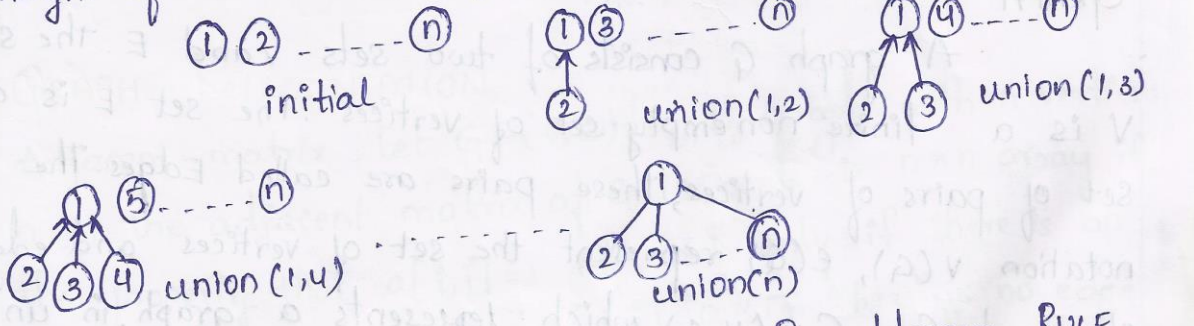
degenerating trees

iv) Weighting rule for union(i,j) :- If the number of nodes in the tree with the root 'i' is less than the number of nodes in the tree with root 'j', make 'j' as parent of 'i' otherwise 'i' as the parent of 'j'.



In order to implement the weighting rule, wee need to know how many nodes are there in each tree. We use a field called count in order to find the number of nodes in that tree.

Note : let T be a tree with m nodes created as a result of sequence of unions each performed using weight union. The height of a tree is not greater than $\log_2 m + 1$



TREES OBTAINED USING THE WEIGHT RULE

Algorithm for weighted rule :

```
Algorithm weightedunion(i,j)
{
    temp:=P[i]+P[j];
    if(P[i]>P[j])then
    {
        P[i]=j;
        P[j]=temp;
    }
    else {
        P[j]:=i;
        P[i]=temp;
    }
}
```

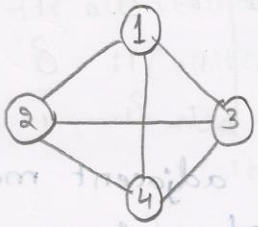
v) Collapsing rule :- If j is a node on path from i to its root and P[i] ≠ root[i], then set P[j] to root[i]

```
Algorithm collapsingFind(i)
{
    r:=i;
    while(p[r]>0)do
    r:=p[r]    // to find root
    while(i≠r) do   // collapse nodes from i to root r.
    {
        s:=p[i]
        p[i]:=r;
        i:=s;
    }
    return r;
}
```

GRAPH :-

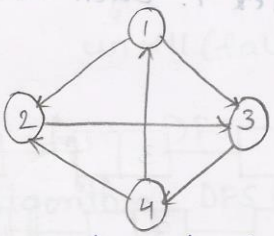
A graph G consists of two sets V and E. the set V is a finite non empty set of vertices. The set E is a set of pairs of vertices, these pairs are called Edges. The notation V(G), E(G) represent the set of vertices and edges of graph G.. G=(V,E) which represents a graph, in un-

directed graph the pair of vertices represents any edge in an unorder.



(1,2), (1,3) (1,4)
(2,1) (2,3) (2,4)
(3,1) (3,2) (3,4)
(4,1)(4,2),(4,3)

In a directed graph, each edge is represented by directed pair.

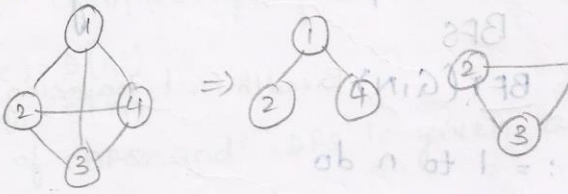


The number of distinct unordered pairs with $P(U,V)$ in addition $U \neq V$ in graph with n vertices is $n(n+1)/2$ i.e., max. no. of edges in any n vertex, of undirected graph.

SUB GRAPH :-

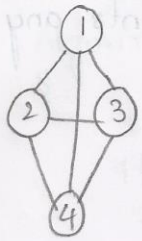
A subgraph of G is a sa graph G' such that $V(G') \subseteq V(G)$, $E(G') \subseteq E(G)$

Eg:



GRAPH REPRESENTATION

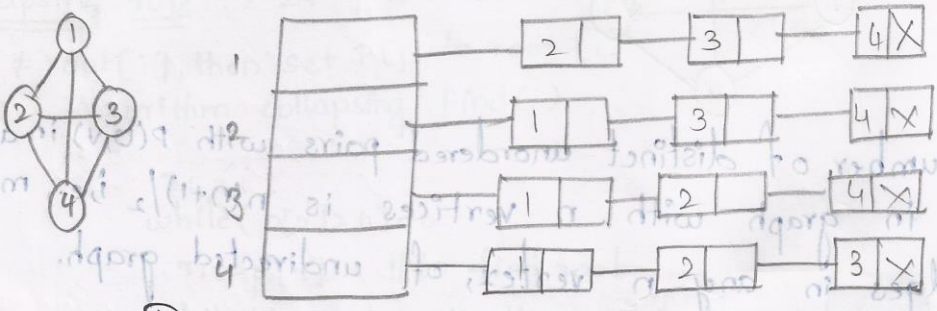
i) Adjacent matrix : Let $G=(V,E)$ be a graph with n vertices $n \geq 1$ the adjacent matrix of G is a 2-D $n \times n$ array if the property that $a[i,j]=1$ if and only if there is an edge between i and j. $a[i,j]=0$ if there is no edge.



Adjacent Matrix :-

	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

ii) Adjacent list: The 'n' rows of adjacent matrix are represented as n linked list. There is one list for each vertex. There is a node in list i represent the vertexes that are adjacent from vertex i. Each nodes have atleast a field vertex and link.



GRAPH TRAVERSALS :

There are two types of traversing graph.

i) BFS - which is implemented using queue data structure

ii) DFS - which is implemented using stack data structure.

→ Algorithm for BFS

```
Algorithm BFT(G,N) Breadth First Traversing
{
    for i := 1 to n do
        visited[i]=0;
    for i := 1 to n do
        if (visited[i]=0)
        then
            BFS(i)
}
Algorithm BFS(v) Breadth first Search
{
    u := v;
```

```
visited[v]:=1;
repeat
{
  for all vertices w adjacent from u do
  {
    if(visited[w]=0) then
    {
      Add w to q;
      visited[w]:=1;
    }
  }
  if 'q' is empty then return;
  Delete u from q;
}
until(false);
}
```

→ Algorithm for DFS

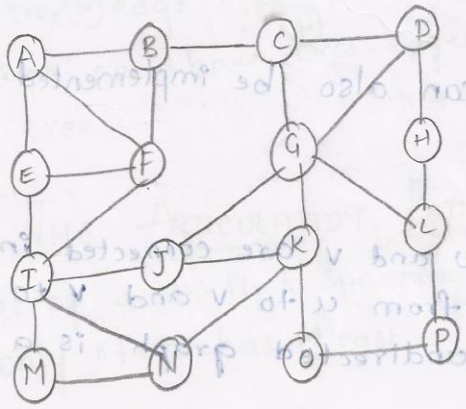
```
Algorithm DFS(v)
{
  visited[v]:=1;
  for each vertex w adjacent from v do
  {
    if(visited[w]=0)
    then
      DFS(w); same as above
  }
}
```

The time complexity of BFS and DFS is given by

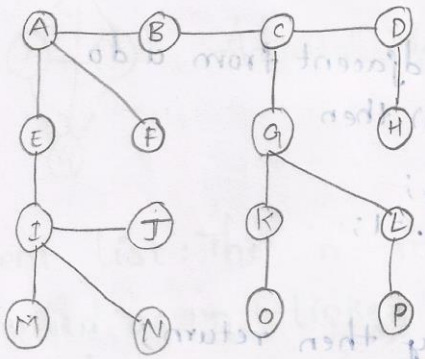
$T(n,e) = \theta(n^2)$

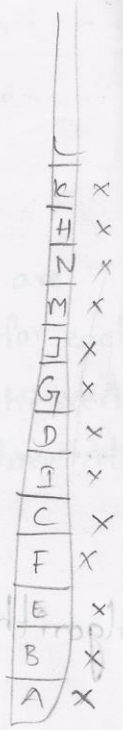
$s(n,e) = \theta(n)$

The space of BFS and DFS is given as ~~stack~~ stated as above.

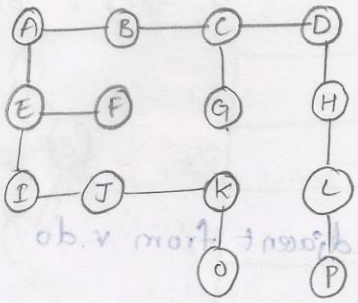


BFS





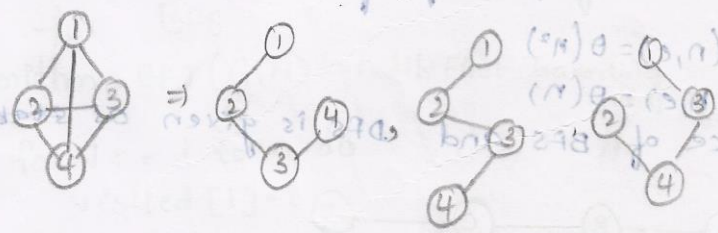
DFS



→ SPANNING TREE :

For a graph $G=(V, E)$ is said to be as a spanning if it contains all vertices with $n-1$ edges. It should not contain a loops or cycles. It must not be a unique.

Ex:



Spanning trees can also be implemented using DFA and BFA.

→ CONNECTED GRAPH :

Two vertices u and v are connected in undirected graph if there is a path from u to v and v to u. A connected component of undirected graph is a maximal

connected sub graphs.

→ STRONGLY CONNECTED GRAPH :

A directed graph G is said to be strongly connected if for every pair of distinct vertices u and v in G there is a directly path from u to v and v to u. A strongly connected component is a maximal subgraph that is strongly connected.

→ BI CONNECTED COMPONENTS :

*. A vertex V(G) is an articulation point if and only if the deletion of B, together with deletion of all edges incident to v leaves behind a graph that has atleast two connected components.

*. A biconnected graph is a connected graph that has no articulation points.

*. A biconnected component of connected graph G is a maximal biconnected subgraph of G i,e., G contains noother sub graphs i,e., both connected & biconnected.

Note : i) Two biconnected components of same graph can have atmost one verter in common.

ii) No edge can be into 2 or more biconnected components.

iii) The biconnected components of a connected, undirected graph can be found by using depth-first spanning tree of G.

iv) A non tree edge u,v is an back edge with respect to a spanning tree T.

v) A non tree edge that is not back edge is called Cross Edge.

vi) No graph can have cross edges with respect to any depth-First spanning tree.

→ FINDING THE ARTICULATION POINTS FOR GIVEN GRAPH :

i) The root of depth-first spanning tree is an articulation point if and only if it has atleast 2 children.

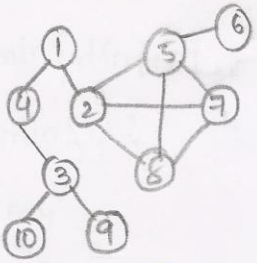
ii) u is an articulation point if and only if, u is either a spanning tree and has 2 or more childrens.

iii) u is not the root and u has a child w such that low(w) ≥ dfn(u) where dfn is depth first number.

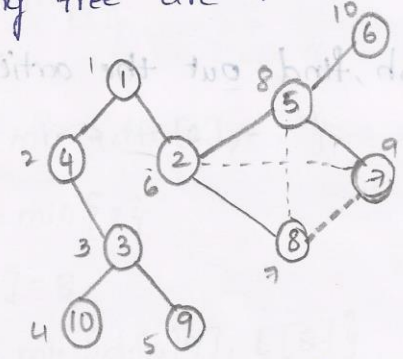
iv) L(v) = min{ dfn[u], min{ L[w] / w is child of u }, min{ dfn[w] / (u,w) is a back edge } };

ALGORITHM FOR BICONNECTED COMPONENTS :

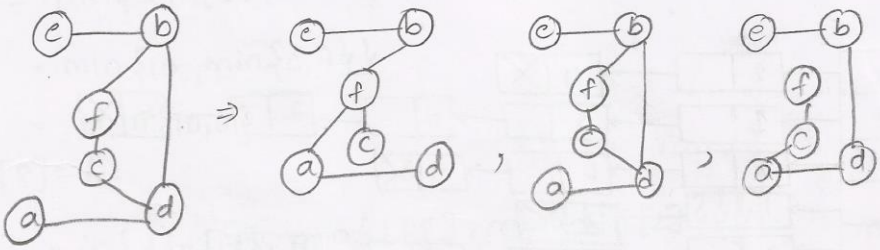
```
Algorithm Bicomp(u,v)
{
    dfn[u] := num;
    L[u] := num;
    num := num+1;
    for each vertex w adjacent from u do
    {
    if((v≠w) and (dfn[w]<dfn[u])) then
      add(u,w) to the top of the stack.
    if(dfn[w]=0) then
     {
       if(L[w]≥ dfn[u]) then
       {
          write("new bicomponent");
          repeat
          {
            Delete an edge from top of stack s;
            Let this edge be (x,y);
            write(x,y);
          }
          until((x,y)=(u,w)) or ((x,y)=(w,u)));
       }
       Bicomp(w,u); //w is unvisited
       L[u] := min[L[u],L[w]);
     }
     else if(w≠v) then
       L[u] := min[L[v], dfn[w]));
    }
}
```



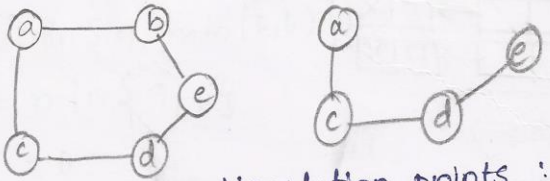
In order to find L[u] easily the vertices of depth-first spanning tree are visited in post order.



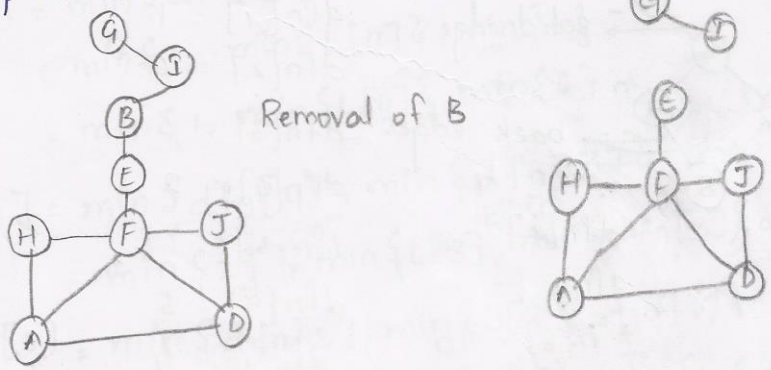
Example for connected components:



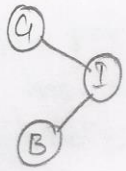
Examples for Biconnected Components:

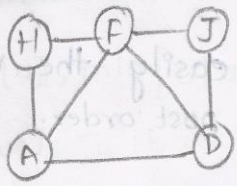


Examples for Articulation points:

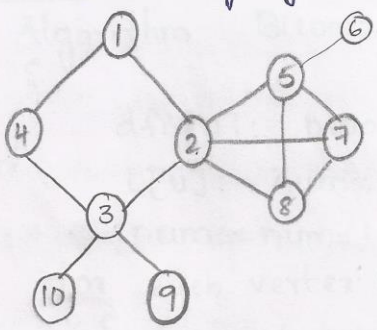


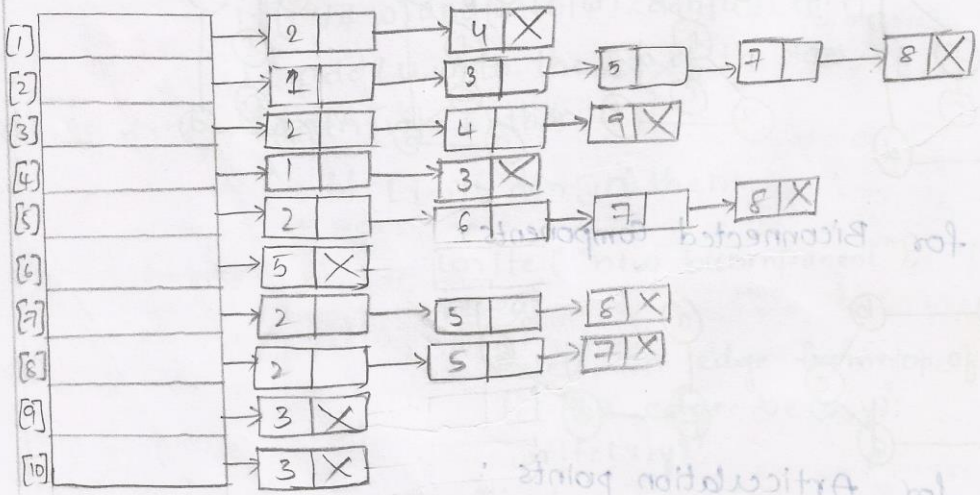
removal of ∈ ⇒

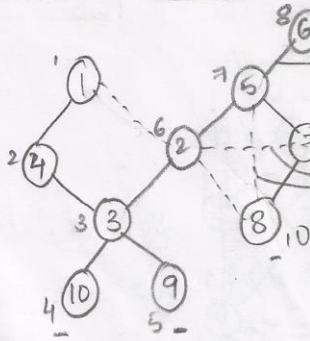




For the following given graph, find out the articulation points.







→ solid edges

⇢ back edges.

dfn[6] = 8
dfn[7] = 9
dfn[8] = 10

dfn[1] = 1
dfn[2] = 6
dfn[3] = 3
dfn[4] = 2
dfn[5] = 7
dfn[9] = 5
dfn[10] = 4

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

$= \min\{4\}$

$L[10] = 4$

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

$= \min\{5\}$

$L[9] = 5$

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

$= \min\{8\}$

$L[6] = 8$

$L[7] = \min\{dfn[7], L[8]\}$

$L[8] = \min\{dfn[8], \min\{dfn[2], dfn[5]\}$

$= \min\{10, \min\{6,7\}\}$

$= \min\{10, 6\}$

$L[8] = 6$

$L[7] = \min\{dfn[7], \min\{L[8], \min\{dfn[2]\}\}\}$

$= \min\{9, \min\{6\}, \min\{6\}\}$

$= \min\{9, \min(6,6)\}$

$= \min\{9, 6\}$

$L[7] = 6$

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

$= \min\{1, \min\{L[4]\}, \min\{6\}$

$= \min\{1, \min\{L[4]\}, \min\{6\}$

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

$= \min\{2\} = \min\{L[3], \min\{dfn[3]\}, dfn[1]\}$

$L[3] = \min\{dfn[3], \min\{L[9], L[10]\}, \min\{dfn[9], dfn[10]\}$

$= \min\{dfn[3], \min\{4, 5\}, \min\{4, 5\}\}$

$L[3] = \min\{3, \min\{4, 5\}\}$

$= \min\{3, 4\}$

$L[3] = 3$

$L[4] = \min\{2, \min\{3\}, \min\{3\}\}$

$= \min\{2, \min\{3\}\}$

$= \min\{2, 3\}$ ~~$L[x]$~~ $\min\{2, \min\{3, 1\}\}$

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

$L[4] = 1$

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

$= \min\{1, \min\{1, 6\}\}$

$= \min\{1, 1\}$

$L[1] = 1$

$L[2] = \min\{dfn(2), \min\{L[5]\}, \min\{dfn[1]\}\}$

$= \min\{6, \min\{L[5]\}, \min\{1\}\}$

$L[5] = \min\{dfn(5), \min\{L[6], L[7]\}, \min\{dfn[8]\}\}$

$= \min\{7, \min\{6, 8\}, \min\{10\}\}$

$= \min\{7, 6, 10\}$

$L[5] = 6$

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

$= \min\{6, 6, 1\}$

$L[2] = 1$

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

Vertex 4: $L[3] \geq dfn[4]$

$3 \geq 2$ ✓

Vertex 3: $L[10] \geq dfn[3]$

$4 \geq 3$ ✓

L[9] >= dfn[3]

5 >= 3 ✓

L[2] >= dfn[3]

1 >= 3 ×

Vertex 2

L[5] >= dfn[2]

6 >= 6 ✓

Vertex 5

L[6] >= dfn[5]

8 >= 7

L[7] >= dfn[5]

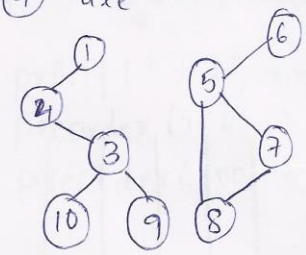
6 >= 7

Vertex 7

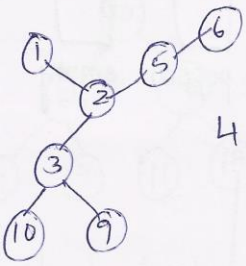
L[8] >= dfn[7]

6 >= 9 ×

(2) & (4) are



2 is an articulation point.



4 is not an articulation point.

EFFICIENT NON RECURSIVE BINARY TREE TRAVERSAL :

The binary tree traversals are divided into three types.

* → Inorder

The Recursion algorithm of inorder is given as follows

⟹

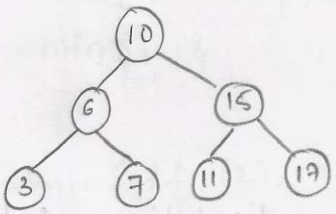
```
void Inorder (Node *root)
{
```

```
if (!root) return ; (if root == 0)
  Inorder (root → left);
  printf ("%d", root → data);
  Inorder(root → right);
}
```

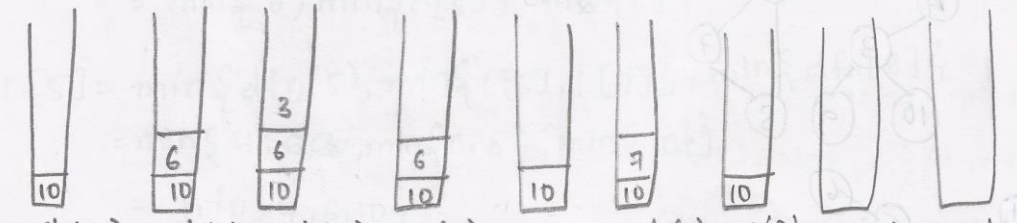
The above recursion method is replaced by using stack to implement inorder without recursion.

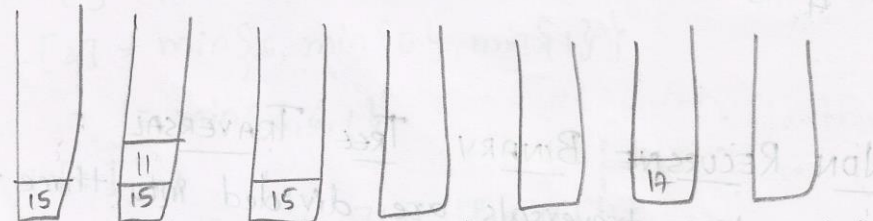
1. Start with root. If (root → left) push root on to the stack.
2. Move to root → left.
3. Print root → data. Move towards right.
4. Pop from the stack.

left, root, right



Inorder : 3, 6, 7, 10, 11, 15, 17





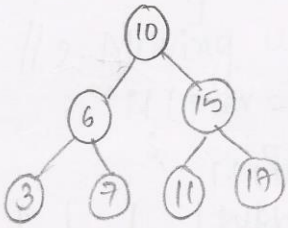
Without Recursion :

```
void Inorder (Node *root)
{   if (root != NULL) return;
```

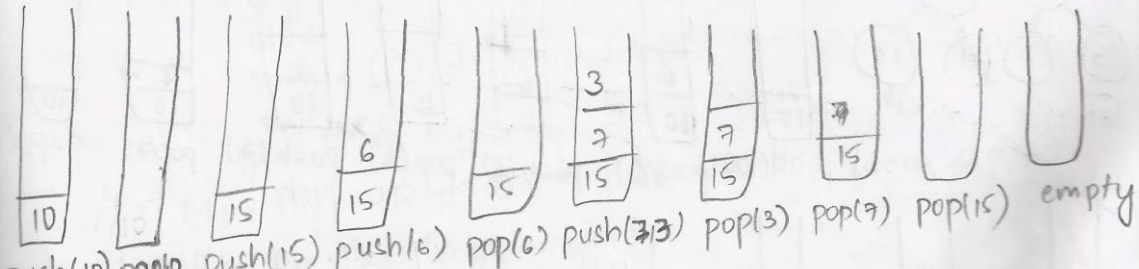
```
Node temp = root;
push (&stack, temp);
while (!is_empty(stack) && temp != null)
{
    if(temp)
    {
        push (&stack, temp);
        temp = temp->left;
    }
    else {
        temp = pop(stack);
        printf ("%d", temp->data)
        temp = temp->right;
    }
}
```

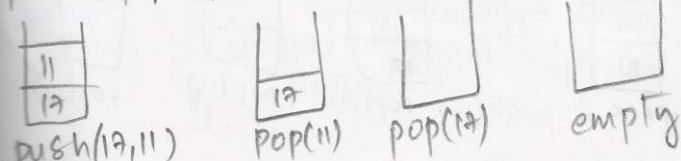
* → Preorder

```
void Preorder (Node *root)
{
    printf ("%d", root->data);
    preorder (root -> left);
    preorder (root -> right);
}
```



Preorder : 10, 6, 3, 7, 15, 11, 17





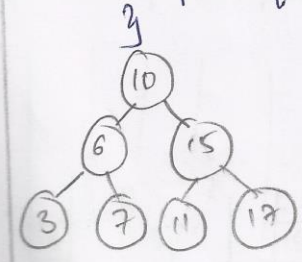
Without Recursion :-

```
void preorder (Node *root)
{
    Node *temp = root;
    if (!root) return;
    initialize (&stack);
    push(&stack, root);
    while (! is_empty (stack))
    {
        temp = pop( &stack);
        printf ("%d", temp->data);
        if (temp->right)
        push (&stack, temp->right);
        if ( temp->left)
        {
            push (&stack , temp->left);
        }
    }
}
```

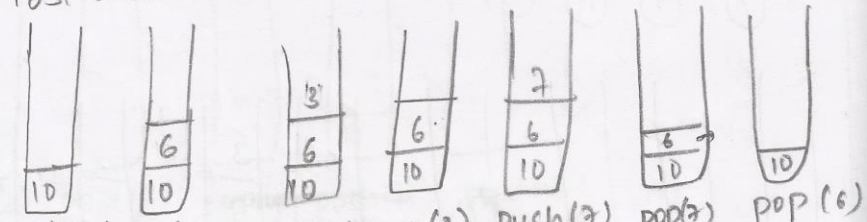
- While performing stack operation, first push the root, then for that root if it contains left & right child, pop the root and push the right child first into stack and then left child.

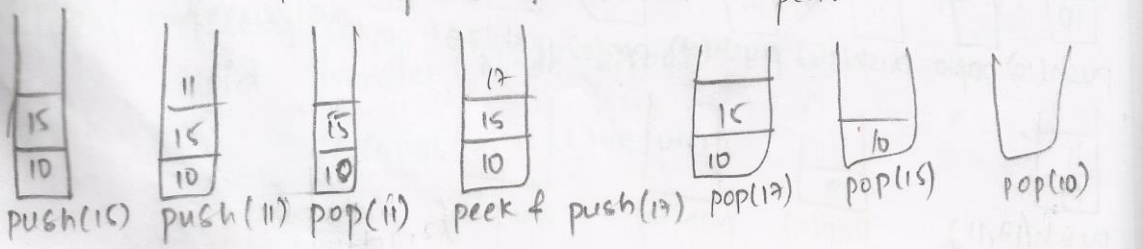
*. → Post Order

```
void postorder (Node *root)
{
    postorder ( root->left);
    postorder (root-> right);
    printf ( "%d", root->data);
}
```



Post order : 3, 7, 6, 11, 17, 15, 10





Without recursion :

```
void postorder (Node *root)
{
    Stack S;
    initialize (&S);
    prev = NULL;
    Node * temp = root;
    if(!root) return;
    push (&s, temp);
    while (! is_empty (s)) {
      temp = top (&s);
// 1. Moving down push into stack.
if(prev == NULL || prev->left == temp || prev->right = temp){
if (temp->left)
push (temp->left);
elseif (temp->right);
push (&s, temp->right);
else if (temp->left && !temp->right)
{
    printf ("%d", temp->data);
    pop(&s);
}
// 2. Moving up from leftside, peek operation.
if(prev == temp->left)
{
    if(temp->right)
    push(&s, temp->right);
}
else
printf ("%d", temp->data)
pop(&s);
}
// 3. Moving up from right side, pop from stack.
if (prev == temp->right)
{
    printf ("%d", temp->data);
    pop(&s);
}
```

```
        prev = temp;
    }
}
```

The peek operation is same as pop, but it returns the value in the stack but does not pop the value.

GAME TREES :-

A game tree is a type of recursive search function that examines all possible moves of a game and their results to obtain a optimal solution. They are very useful for artificial intelligence, making number of possible choice per play.

Ex: chess

We will classify all games in following three groomes.

→ Single Player Path Finding Problems.

Ex: Rubik's cube
travelling salesman problem

→ Two Player (Path Finding) Games

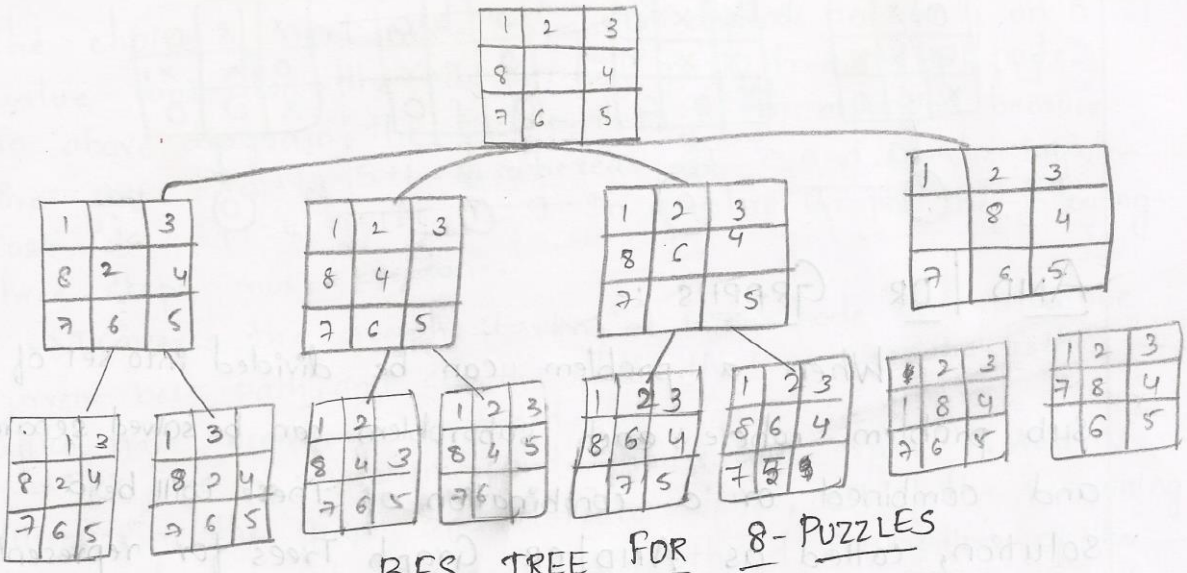
Ex: 1) Chess
2) Checkers

→ Constraint Satisfcation Problem

Ex: 1) Eight Queens 3) Rubik's cube
2) Sudoku.

Each game consists of problem faced, initial state, a single goal state. A problems face is a mathematical abstraction in the form of a tree.

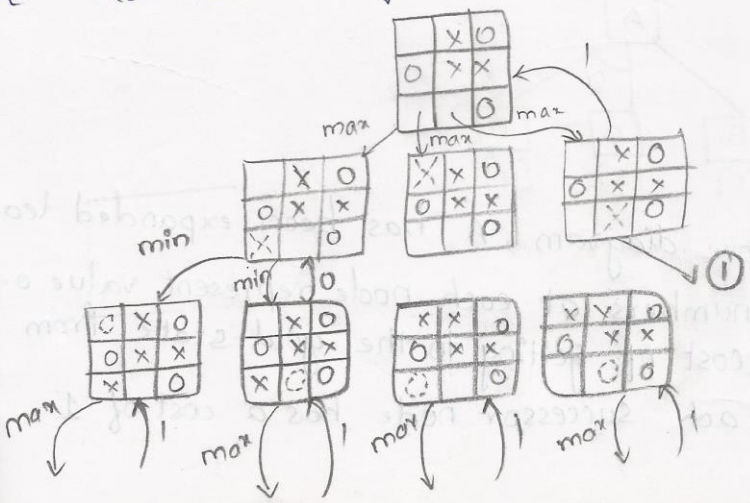
1). The root represents the current state.
2). Nodes represent the state of game.
3) Edges represent moves - Leaves represents final state. (win, loss, draw)

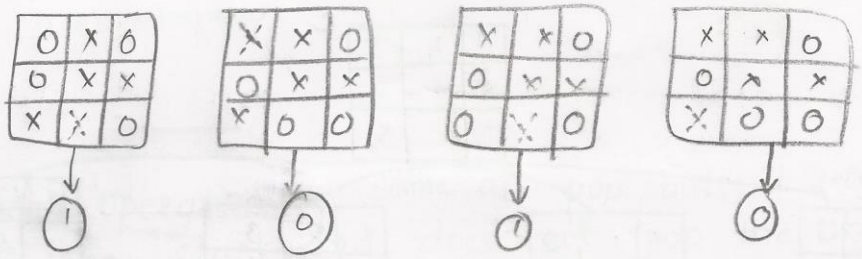


BFS TREE FOR 8-PUZZLES

→ MIN MAX ALGORITHM :-

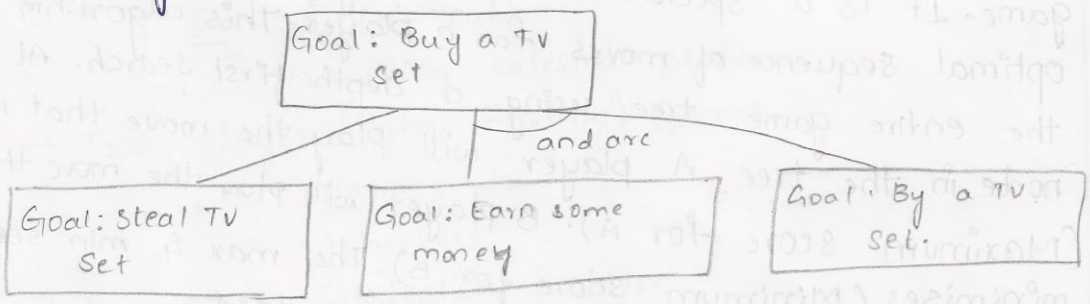
We consider games with 2 players in which one persons gains and another person looses is called Zero sum game. It is a specialised search algorithm which returns optimal sequence of moves for a player. this algorithm explores the entire game tree using a depth first search. At each node in the tree A player will play the move that maximizes (Maximum score for A). B player will play the move that minimises (Minimum score for B). The max & min scores are taken alternately at each level of tree.



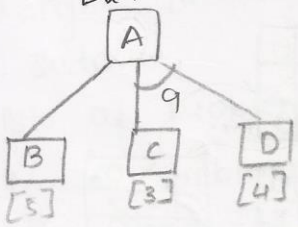


AND | OR GRAPHS :-

When a problem can be divided into set of sub problem where each subproblem can be solved separately and combined or a combination of these will be a solution, called as AND|OR Graph Trees for representing the solution. The decomposition of problem are problem reduction generates AND arcs. One and may point to any number of successor nodes. All these nodes must be solved so that and arcs may raise many arcs, i,e., indicating several possible solutions.



Ex: AndlOR Graphs.



In above diagram, A has been expanded leading to B, C, D, the numbers at each node represent value of f' at that node (cost of getting to the gold state from current state). Each successor node has a cost of 1

The choice of next node expansion depends not only on n value but also the current best path from initial node. In above diagram, the best path is B for expansion because the cost total is 5+1=6 whereas for C and D, the total cost is 3+1=4, 4+1=5, is 9. In ANDIOR Graphs the following three steps must be done.

→ Traverse the graph starting at initial node and follow the current best path, and set the nodes that are on the path which have been not expanded.

→ Pick one of these unexpanded nodes and expand it. Add as successor to the graph and compute f' i,e., cost of the remaining distance for expanded nodes. → Change the f' estimation of the newly expanded node to reflect the new information produced by its successor.

After following the above 3 steps, decide which is the best path.

11/2/15

UNIT-3

GREEDY METHOD

→ General Method : For a given n inputs of any problem, obtain a subset that satisfies some constraints. Any subset that satisfies these constraints is called a feasible solution.

*. A feasible solution that either maximises or minimises given objective function is called optimal solution.

→ Algorithm :-

```
Algorithm Greedy (a,n)
{
    solution := ∅;
    for i := 1 to n do
    {
        x := select(a);
        if Feasible(solution,x) then
            solution := union(solution,x);
    }
    return solution;
}
```

KNAPSACK PROBLEM :-

For a given n objects and a knapsack bag, object i has a weight w_i and capacity of knapsack is m. If a fraction x_i between $0 \le x_i \le 1$, when object i is placed into the knapsack then the profit of $p_i x_i$ is earned. The objective is to obtain filling of knapsack that maximises the total profit gain. The problem can be stated as

$$\text{Maximize} \sum_{1 \le i \le n} p_i x_i \quad \text{—} (1)$$

$$\text{Subject to} \sum_{1 \le i \le n} w_i x_i \le m \quad \text{—} (2)$$

and $0 \le x_i \le 1, \quad 1 \le i \le n$ — ③

The profits and weights are positive number. Eqn ② will give the feasible solution and eqn ① is an optimal solution which contains maximum value.

For ~~n=~~

Find solution of knapsack problem where $n=3$; $m=20$;

$P_1, P_2, P_3 = 25, 24, 15$; $w_1, w_2, w_3 = 18, 15, 10$;

S.NO	$\sum w_i x_i$	$\sum P_i x_i$
1) Considering the objects x_1 x_2 x_3 : $1/2$ $1/3$ $1/4$	$18 \times 1/2 + 15 \times 1/3 + 10 \times 1/4 = 16.5$	$25 \times \frac{1}{2} + 24 \times \frac{1}{3} + 15 \times \frac{1}{4} = 24.25$
2) Consider the maximum profit (25 24 15 / 18 15 10) x_1 x_2 x_3 : 1 $2/15$ 0	$18 \times 1 + \frac{2 \times 15}{15} + 10 \times 0 = \underline{20}$	$25 \times 1 + 15 \times 0 = 28.2$
3) Considering the ~~maxi~~ mini-mum profit x_1 x_2 x_3 : 0 $2/3$ 1 (25, 24, 15 – P; 18, 15, 10 – w)	$18 \times 0 + 15 \times 2/3 + 1 \times 10 = \underline{20}$	$25 \times 0 + 24 \times 2/3 + 15 \times 1 = 31$
4) Greedy approach $\frac{P_i}{w_i}$ ratio; $\frac{P_1}{w_1} = \frac{25}{18} = 1.38$, $\frac{P_3}{w_3} = \frac{15}{10} = 1.5$, $\frac{P_2}{w_2} = \frac{24}{15} = 1.6$; x_1 x_2 x_3 : 0 1 $1/2$	$18 \times 0 + 15 \times 1 + 10 \times 1/2 = 20$	$25 \times 0 + 24 \times 1 + 15 \times 1/2 = 31.5$

For the above problem, the optimal value is 31.5.

MID **

2) $n=7$, $m=15$, P_1 to $P_7 = 10, 5, 15, 7, 6, 18, 3$; w_1 to $w_7 = 2, 3, 5, 7, 1, 4, 1$

→ S.NO $\sum w_i x_i$ $\sum p_i x_i$

1) Considering the objects

x_1 x_2 x_3 x_4 x_5 x_6 x_7

$\frac{1}{2}$ $\frac{1}{3}$ $\frac{1}{4}$ $\frac{1}{5}$ $\frac{1}{6}$ $\frac{1}{7}$ $\frac{1}{8}$

$2\times\frac{1}{2} + \frac{3}{3} + \frac{5}{4} + \frac{7}{5} + \frac{1}{6} + \frac{4}{7} + \frac{1}{8}$

$= 5.51$

$\frac{10}{2} + \frac{5}{3} + \frac{15}{4} + \frac{7}{5} + \frac{6}{6} + \frac{18}{7} + \frac{3}{8}$

$= 15.76$

2) Considering the maximum profit

4	4/7	
6	2	
11	5	15
	4	

x_1 x_2 x_3 x_4 x_5 x_6 x_7

1 0 1 $\frac{7}{4}$ 0 1 0

$2 + 5 + \frac{7\times4}{7} + 4$

$= 15$

$10 + 15 + 9.4/7 + 18$

$= 47$

3) Considering the minimum profit

x_1 x_2 x_3 x_4 x_5 x_6 x_7

1 1 $\frac{4}{5}$ 0 1 1 1

$2 + 3 + \frac{4}{5} + 7 + 1 + 1$

$= 15$

$10 + 15 + \frac{15\times4}{5} + 6\times1 + 18\times1 + 3\times1$

$= 54$

4	4/5	
8	4	
11	3	15
13	2	
14	1	
	1	

4) Greedy Approach

$\frac{P_i}{w_i}$ (1) $\frac{10}{2} = 5$

(2) $5/3 = 1.6$, (3) $\frac{15}{5} = 3$ (4) $7/7 = 1$ (5) $6/1 = 6$ (6) $18/4 = 4.5$ (7) $3/1 = 3$

2	2/3	
3	1	
8	5	15
12	4	
14	2	
	1	

x_1 x_2 x_3 x_4 x_5 x_6 x_7

1 $\frac{2}{3}$ 1 0 1 1 1

$2 + 3.2/3 + 5 + 1 + 4 + 1$

$= 15$

$10 + \frac{5.2}{3} + 15 + 6 + 18 + 3$

$= 55.33$

18
15
10
43

Algorithm :-

```
Algorithm Greedyknapsack(m,n)
{
    for i:=1 to n do
        x[i]:=0.0;
    U:=m;  U - capacity
    for i:=1 to n do
    {
        if (w[i] > U) then break;
        x[i]:=1.0;
        U:=U-w[i];
    }
    if (i<=n) then
        x[i]:=U/w[i];
}
```

3) $m = 7$, $n = 3$, P_1 to $P_3 = 6, 5, 4$, w_1 to $w_3 = 5, 4, 3$

S.NO	$\sum w_i x_i$	$\sum P_i x_i$
1) Considering the objects x_1 x_2 x_3 $1/2$ $1/3$ $1/4$	$5/2 + 5/3 + 4/4$ $5/2 + 4/3 + 3/4$ $= \frac{30+16+9}{12} = \frac{55}{12} = 4.5$	$3 \cdot 6/2 + 5/3 + 4/4$ $= 5.66$
2) Considering the maximum profits x_1 x_2 x_3 1 $1/2$ 0	$5 + 4/2 = 7$	$6 + 5/2 = 29/4 = 8.5$
3) Considering minimum profits x_1 x_2 x_3 0 1 1	$4 \times 1 + 3 \times 1 = 7$	$5 + 4 = 9$

4) Greedy Approach

$\sum w_i x_i$ $\qquad$ $\sum P_i x_i$

$\frac{P_i}{w_i}$

1) $6/5 = 1.2$

2) $5/4 = 1.25$

3) $4/3 = 1.3$

$4\times1 + 3\times1$ $\qquad$ $5\times1 + 4\times1 = 9$

$= 7$.

x_1	x_2	x_3
0	1	1

The optimal value is 9.

Note:- All optimal solutions will fill the knapsack bag exactly.

*. If $P_1/w_1 \geq P_2/w_2 \geq \cdots P_n/w_n$ then greedy knapsack generates an optimal solutions to the given instance of the knapsack problem.

→ JOB SEQUENCING WITH DEAD LINES:

For a set of n jobs, each job i is associated with an integer dead line $d_i \geq 0$ and $P_i > 0$. For any job i the profit P_i is earned or gained if and only iff job is completed by its dead line.

A feasible solution for this problem is a subset J of jobs J_1 such that each job in the subset can be completed by its dead line. The value of a feasible solution J is the sum of profits of the jobs in J $\sum_{i \in J} P_i$.

An optimal solution is a feasible with maximal value.

1> $n = 4$ jobs

P_1 to P_4 = 100, 10, 15, 27.

d_1 to D_4 = 2, 1, 2, 1 → d → dead line

1 2 3 4
100, 10, 15, 27 d_1 to d_4 = 2, 1, 2, 1

Method 1 :-

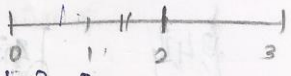
S.NO	Feasible solution	Processing Sequence	Value
1.	1, 2	2, 1	100+10=110
2.	1, 3	1, 3 or 3, 1	100+15=115
3.	1, 4	4, 1	100+27=127
4.	2, 3	2, 3	10+15=25
5.	2, 4	2, 4 or 4, 2	10+27=37
6.	3, 4	4, 3	15+27=42
7.	1	1	100
8.	2	2	10
9.	3	3	15
10	4	4	27.

For, above problem
The optimal value is 127.
Feasible solution is 1, 4
Prossessing Sequence is 4, 1.

For all The time complexity of this method is $O(n^2)$. To overcome above complexity Method 2 is followed.

Method 2 :-

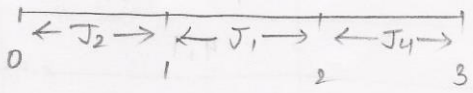
2) n=5, P1 to P5 = 20, 15, 10, 5, 1 (descending order) d_1 to d_4 = 2, 2, 1, 3, 3



J	assigned slots	Job considered	action	profit
∅	none	1	[1,2]	0
{1}	[1,2]	2	[0,1]	20
{1,2}	[0,1], [1,2]	3	rejected no slots	20+15=35
{1,2,4}	[0,1], [1,2]	4	[2,3]	35

{1, 2, 4}	[0,1] [1,2] [2,3]	5	rejected no slots	35+5 = 40
{1, 2, 4}	[0,1] [1,2] [2,3]			= 40

Optimal value is 40.



3. $n = 7$.

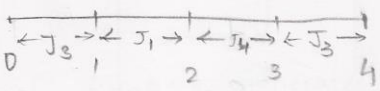
P_1 to P_7 = 3, 5, 20, 18, 1, 6, 30

d_1 to d_7 = 1, 3, 4, 3, 2, 1, 2.

→ P_1 to P_7 = 30, 20, 18, 6, 5, 3, 1

d_1 to d_7 = P_7 P_3 P_4 P_6 P_2 P_1 P_5

2 4 2 1 3 1 4



maximum dead line = 4.

J	assigned slots	Job considered	action	p: profit
∅	none	7	[1,2]	0
{7}	[1,2]	3	[3,4]	30
{3,7}	[1,2] [3,4]	4	[2,3]	30+20 = 50
{3,4,7}	[2,3] [1,2] [3,4]	6	rejected slots [0,1]	50+18 = 68
{3,4,7}	[2,3] [1,2] [3,4] [0,1]	2	rejected slot	68+6 = 74
{3,4,7,2}	[2,3] [1,2] [3,4] [0,1]	1	rejected slots	68+6 = 74
{3,4,7}	[2,3] [1,2] [3,4] [0,1]	5	[0,1]	74
{7,4,3}	[2,3] [1,2] [3,4] [0,1]			74

```
Algorithm JS(d,j,n)
{
    d[0] := J[0] := 0;
    J[1] := 1;
    k := 1;
  for i := 2 to n do
  {
    r := k;
   while ((d[J[r]] > d[i]) and (d[J[r]] ≠ r)) do
    r := r-1;
   if ((d[J[r]] ≤ d[i]) and (d[i] > r)) then
   {
       for q := k to (r+1) step -1 do
       J[q+1] := J[q];
       J[r+1] := i;
       k := k+1;
   }
   }
   return k;
}
```

PRIMS ALGORITHM :-

```
Algorithm prim(E,cost,n,t)
{
  Let (k,l) be an edge of minimum cost in E
  mincost := cost[k,l];
  t[1,1] := k; t[1,2] := l;
  for i := 1 to n do
   if (cost[i,l] < cost[i,k]) then
     near[i] := l;
    else near[i] := k;
   near[k] := near[l] := 0;
    for i := 2 to n-1 do
    {
```

Let j be an index
such that near[j] ≠ 0 and
cost(j, near[j]) is minimum;
t[i,1] := j; t[i,2] := near[j];
mincost := mincost + cost[j, near[j]];
near[j] := 0;
for k := 1 to n do
if ((near[k] ≠ 0) and (cost(k, near[k]) > cost[k,j]))
then near[k] := j;
}
return mincost;
}

Time Complexity is $O(n^2)$

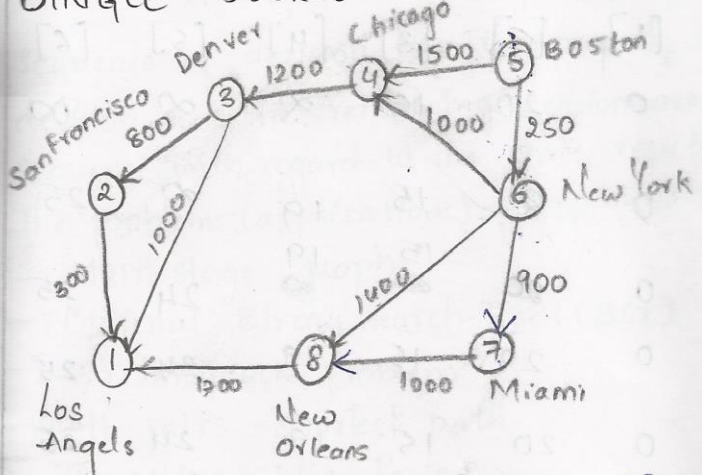
KRUSHKAL'S ALGORITHM :- All weights are tabulated in increasing order and drawn the graph for Krushkal

```
Algorithm Krushkal(E, cost, n, t)
{
    construct a heapout of the edges
    cost using heapify for i := 1 to n do
    parent[i] := -1;
    i := 0; mincost := 0.0;
    while ((i < n-1) and (heap.not empty)) do
    {
        Delete a minimum cost
            from the heap and verify
                using Adjust;
        j := Find(u);
        k := Find(v);
        if (j ≠ k) then
        {
            i := i+1;
            t[i,1] := u;
            t[i,2] := v;
            mincost := mincost + cost[u,v];
            union(j, k);
        }
```

```
if (i ≠ n-1) then
write ("no spanning tree");
else return mincost;
}
```

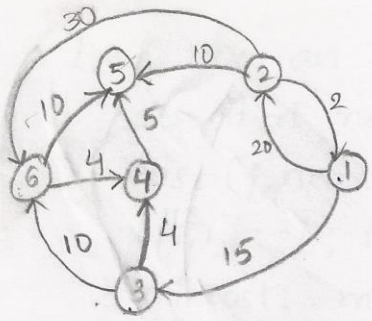
Time Complexity is $O(|E| \log |E|)$

SINGLE SOURCE SHORTEST PATH :



Iteration	S	Vertex Selected	LA [1]	SF [2]	DEN [3]	CHI [4]	BOST [5]	NY [6]	MA [7]	NO [8]
—	—	—	∞	∞	∞	1500	0	250	∞	∞
1	5	6	∞	∞	∞	1250	0	250	1150	1650
2	{5,6}	7	∞	∞	∞	1250	0	250	1150	1650
3	{5,6,7}	4	∞	∞	2450	1250	0	250	1150	1650
4	{5,6,7,4}	8	3350	∞	2450	1250	0	250	1150	1650
5	{6,5,7,4,8}	3	3350	3250	2450	1250	0	250	1150	1650
6	{5,6,7,4,8,3}	2	3350	3250	2450	1250	0	250	1150	1650
7	{5,6,7,8,3,2}	1	3350							

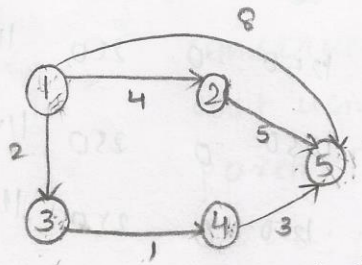
2)



Source 1
destination 6

Iteration	S	Vertex Selected	[1]	[2]	[3]	[4]	[5]	[6]
-	-	-	0	20	15	∞	∞	∞
1	1	3	0	~~20~~	15	19	∞	25
2	{1,3}	4	0	~~20~~	~~15~~ 15	~~19~~ 19	24	25
3	{1,3,4}	2	0	20	15	19	24	25
4	{1,3,4,2}	5	0	20	15	19	24	25
5	{1,3,4,2,5}	6	0	20	15	19	24	25

3)



Iteration	S	vertex selected	[1]	[2]	[3]	[4]	[5]
-	-	-	0	4	2	~~∞~~ ~~20~~ 8	8
1	1	3	0	4	2	~~∞~~ 3	8
2	{1,3}	~~3~~ 4	0	4	2	3	6
3	{1,3,4}	2	0	4	2	3	6
4	{1,3,4,2}	5	0	4	2	3	6

Dynamic Programming :-

It is a method that can be used when the solution to the problem can be viewed as a result of sequence of decision.

In greedy method, one decision sequence is generated where as in dynamic programming many decision sequence are generated.

In DP, the principle of optimately states that an optimal sequence of decision has the property that whatever the initial state & decision are, the remaining decision are must constitute an optimal decision sequence with regard to the state resulting from the first decision.

The problems (applications) of DP are :-

→ Multi-stage graph
→ Optimal Binary Search Tree (OBST)
→ 0/1 Knapsack Problem
→ All pairs - shortest path
→ Travelling Sales Person.
→ Reliability Design

All Pairs Shortest Path :-

Let $G = \{V, E\}$ be a directed graph with 'n' vertices, Let c be a cost of adjacent matrix for 'G' such that $cost(i,i) = 0$ where $1 \le i \le n$. $Cost(i,j)$ is the length of the edge (i,j) if $(i,j) \in \underset{\text{Edge}}{E}(G)$. $Cost(i,j) = \infty$ if $i \neq j$ & $(i,j) \notin E(G)$. The all pairs shortest path problem is to determine a matrix 'A' such that $A(i,j)$ is the length of shortest path from i to j The matrix 'A' can be obtained by solving 'n' single source problems using the algorithm shortest path.

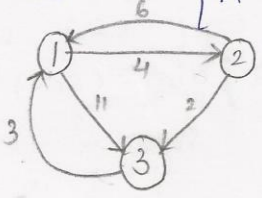
Since each application of this procedure requires $O(n^2)$ time, the matrix 'A' can be obtained in $O(n^3)$

Formulae :-

$$A^0(i,j) = cost(i,j) \quad 1 \le i \le n, \; 1 \le j \le n.$$

$$A^k(i,j) = \min\{A^{k-1}(i,j), A^{k-1}(i,k) + A^{k-1}(k,j)\} \quad k \ge 1$$

1>



Step 1 :-

$A^0(i,j) = cost(i,j)$

$A^0(1,1) = 0$ $\quad A^0(2,1) = 6$ $\quad A^0(3,1) = 3$

$A^0(1,2) = 4$ $\quad A^0(2,2) = 0$ $\quad A^0(3,2) = \infty$

$A^0(1,3) = 11$ $\quad A^0(2,3) = 2$ $\quad A^0(3,3) = 0$

Matrix :-

A^0	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

Step 2 :-

$k=1$, $i=1$, $j=1,2,3$

$A^k(i,j) = \min\{A^k(i,j), A^{k-1}(i,k) + A^{k-1}(k,j)\}$

$A^1(1,1) = \min\{A^{1-1}(1,1), A^{1-1}(1,1) + A^{1-1}(1,1)\}$

$= \min\{A^0(1,1), A^0(1,1) + A^0(1,1)\}$

$= \min\{0\}$

$A^1(1,1) = 0$

A^0	1	2	3
1	0		
2			
3			

$\Rightarrow k=1$ $\quad i=1$, $\quad j=2$

$A^1(1,2) = \min\{A^{1-1}(1,2), A^{1-1}(1,1) + A^{1-1}(1,2)\}$

$= \min\{A^0(1,2), A^0(1,1) + A^0(1,2)\}$

$= \min\{4, 0+4\}$

$= \min\{4,4\}$ $\quad A^1(1,2) = 4$

A^0	1	2	3
1	0	4	
2			
3			

$\Rightarrow k=1$ $\quad i=1$ $\quad j=3$

$A^1(1,3) = \min\{A^{1-1}(1,3), A^{1-1}(1,1) + A^0(1,3)\}$

$= \min\{A^0(1,3), A^0(1,1) + A^0(1,3)\}$

$= \min\{11, 0+11\}$

$A^1(1,3) = 11$

A^0	1	2	3
1	0	4	11
2			
3			

$\Rightarrow k=2$, $i=2$, $j=1$

$A^2(2,1) = \min\{A^{1}(2,1), A^{1}(2,2) + A^{1}(1,1)\}$

$= \min\{A^0(2,1), A^0(2,1) + A^0(1,1)\}$

$= \min\{6, 6+0\}$

$A^1(2,1) = \{6\}$

⇒ $k=1,\ i=2\ j=2$

$A^1(2,2) = \min\{A^{1-1}(2,2), A^{1-1}(2,1) + A^{1-1}(1,2)\} = \min\{0, 6+4\} = 0$

⇒ $k=1,\ i=2,\ j=3$

$A^1(2,3) = \min\{A^0(2,3), A^0(2,1) + A^0(1,3)\} = \min\{2, 6+4\} = 2$

⇒ $k=1,\ i=3,\ j=1$

$A^1(3,1) = \min\{A^0(3,1), A^0(3,1) + A^0(1,1)\} = \min\{3, 3+0\} = 3$

⇒ $k=1\quad i=3\quad j=2$

$A^1(3,2) = \min\{A^0(3,2), A^0(3,1) + A^0(1,2)\} = \min\{0, 7\} = 7.$

⇒ $k=1\quad i=3\quad j=3$

$A^1(3,3) = \min\{A^0(3,3), A^0(3,1) + A^0(1,3)\} = \min\{0, 3+11\} = 0$

A^1	1	2	3
1	0	4	11
2	6	0	2
3	3	7	0

Step 3 :-

⇒ $k=2,\ i=1,\ j=1$

$A^2(1,1) = \min\{A^{2-1}(1,1), A^{2-1}(1,2) + A^{2-1}(2,1)\}$

⇒ $k=2\ i=1, j=2 = \min\{A^1(1,1), A^1(1,2) + A^1(2,1)\} = \min\{0, 4+6\} = 0$

$A^2(1,2) = \min\{A^1(1,2), A^1(1,2) + A^1(2,2)\} = \min\{4, 4+0\} = 4$

⇒ ~~k~~ $k=2\quad i=1\quad j=3$

$A^2(1,3) = \min\{A^1(1,3), A^1(1,2) + A^1(2,3)\} = \min\{11, 4+2\} = 6$

⇒ $k=2\quad i=2\quad j=1$

$A^2(2,1) = \min\{A^1(2,1), A^1(2,2) + A^1(2,1)\}$

$= \min\{6, 0+6\} = 6$

⇒ $k=2\quad i=2\quad j=2$

$A^2(2,2) = \min\{A^1(2,2), A^1(2,2) + A^1(2,2)\}$

$= \min\{0, 0\}$

$= 0$

⇒ $k=2\quad i=2\quad j=3$

$A^2(2,3) = \min\{A^1(2,3), A^1(2,2) + A^1(2,3)\} = \min\{2, 0+2\}$

$= 2$

⇒ $k=2, i=3, j=1$

$A^2(3,1) = \min\{A^1(3,1), A^1(3,2)+A^1(2,1)\}$

$= \min\{3, 7+6\} = \min\{3, 13\} = 3$

⇒ $k=2, i=3, j=2$

$A^2(3,2) = \min\{A^1(3,2), A^1(3,2)+A^1(2,2)\}$

$= \min\{7, 7+0\} = 7$

⇒ $k=2 \quad i=3, j=3$

$A^2(3,3) = \min\{A^1(3,3), A^1(3,2)+A^1(2,3)\} = \min\{0, 7+2\} = 0$

A^2	1	2	3
1	0	4	6
2	6	0	2
3	3	7	0

⇒ $k=3, i=1, j=1$

$A^3(1,1) = \min\{A^2(1,1), A^1(1,3)+A^1(3,1)\}$

$= \min\{0, 6+3\} = 0$

⇒ $k=3, i=1, j=2$

$A^3(1,2) = \min\{A^2(1,2), A^2(1,3)+A^2(3,2)\} = \min\{4, 6+7\} = 4$

⇒ $k=3 \quad i=1 \quad j=3$

$A^3(1,3) = \min\{A^2(1,3), A^2(1,3)+A^2(3,3)\} = \min\{6, 0+6\} = 6$

⇒ $k=3 \quad i=2 \quad j=1$

$A^3(2,1) = \min\{A^2(2,1), A^2(2,3)+A^2(3,1)\} = \min\{6, 2+3\} = 5$

⇒ $k=3 \quad i=2 \quad j=2$

$A^3(2,2) = \min\{A^2(2,2), A^2(2,3)+A^2(3,2)\} = \{0\}$

⇒ $k=3 \quad i=2 \quad j=3$

$A^3(2,3) = \min\{A^2(2,3), A^2(2,3)+A^2(3,3)\} = \min\{2, 2+0\} = 2$

⇒ $k=3, i=3, j=1$

$A^3(3,1) = \min\{A^2(3,1), A^2(3,3)+A^2(3,1)\} = \min\{3, 0+3\} = 3$

⇒ $k=3, i=3, j=2$

$A^3(3,2) = \min\{A^2(3,2), A^2(3,3)+A^2(3,2)\} = \min\{7, 0+7\} = 7$

⇒ $k=3 \quad i=3 \quad j=3$

$A^3(3,3) = \min\{A^2(3,3), A^2(3,3)+A^2(3,3)\} = 0$

A^3	1	2	3
1	0	4	6
2	5	0	2
3	3	7	0

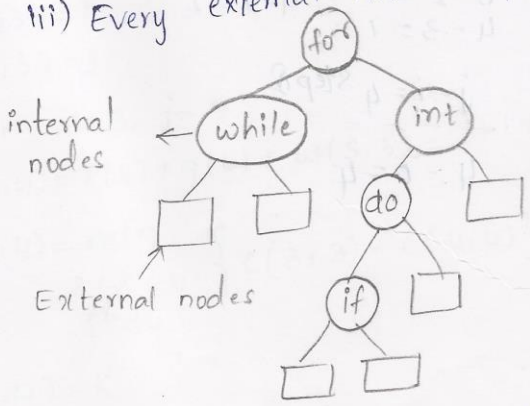
Algorithm Allpaths (cost, A, n)
{
    for i:1 to n do
    for j:1 to n do
    A[i,j]:=cost[i,j];
    for k:=1 to n do
    for i:=1 to n do
    for j:=1 to n do
        A[i,j]=min{ A[i,j], A[i,k]+A[k,j]};

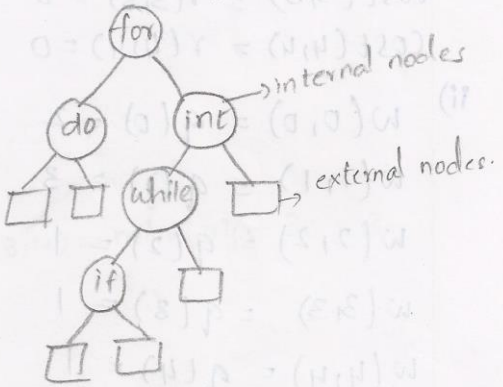
Optimal Binary Search Tree :

i) In a binary search tree for n identifiers there will be n internal nodes and n+1 external nodes.

ii) Every internal node represents a successful search

iii) Every external node represents a unsuccessful search.





In above diagram the probability of successful nodes or successful BST, varies from 1 to 2 diagrams. In order to overcome this we consider optimal binary search tree where we are going to find maximum cost for the respective routes.

Formulae :-

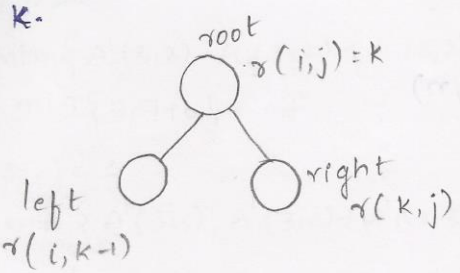
i) cost(i,i) = r(i,i) = 0

ii) w(i,i) = q(i) $0 \le i \le n$

3. $w(i,j) = p(j) + q(j) + w(i,j-1)$

4. $c(i,j) = \min_{i<k\le j} \{c(i,k-1) + c(k,j)\} + w(i,j)$

5. $r(i,j) = k$.



Problem :-

1. $n = 4$, $(a_1, a_2, a_3, a_4) = (do, if, int, while)$

$P(1:4) = 3, 3, 1, 1$

$q(0:4) = 2, 3, 1, 1, 1$ q values are $0 \le i \le n$ ($i = 0$ to 4)

$i = 0, 1, 2, 3, 4$

i) $cost(0,0) = r(0,0) = 0$
$cost(1,1) = r(1,1) = 0$
$cost(2,2) = r(2,2) = 0$
$cost(3,3) = r(3,3) = 0$
$cost(4,4) = r(4,4) = 0$

ii) $w(0,0) = q(0) = 2$
$w(1,1) = q(1) = 3$
$w(2,2) = q(2) = 1$
$w(3,3) = q(3) = 1$
$w(4,4) = q(4) = 1$

iii)

Step I	Step II	Step III
$j - i = 1$	$j - i = 2$	$j - i = 3$
$1 - 0 = 1$	$2 - 0 = 2$	$3 - 0 = 3$
$2 - 1 = 1$	$3 - 1 = 2$	$4 - 1 = 3$
$3 - 2 = 1$	$4 - 2 = 2$	
$4 - 3 = 1$		

$j - i = 4$ Step IV

$4 - 0 = 4$

⇒ For $j - i = 1$ (Step I)

3) $w(i,j) = p(j) + q(j) + w(i,j-1)$

* $i = 0$, $j = 1$ ($\because 1 - 0 = 1$)

$w(0,1) = p(1) + q(1) + w(0,0) = 3 + 3 + 2 = 8$ $\quad w(0,1) = 8$

4) $c(i,j) = \min_{i<k\le j} \{c(i,k-1) + c(k,j)\} + w(i,j)$

$$c(0,1) = \min_{0<k\le 1} \{c(0,0) + c(1,1)\} + w(0,1) = \min\{0+0\} + 8$$
$$= 8 + 0$$

$k = 1$

$$c(0,1) = 8$$

5) $r(i,j) = k$

$r(0,1) = 1$

* when $i = 1 \quad j = 2$

$w(1,2) = p(2) + q(2) + w(1,1)$

$= 3 + 1 + 3 = 7.$

$$c(1,2) = \min_{1<k\le 2} \{c(1,1) + c(2,2)\} + w(1,2) = \min\{0+0\} + w(1,2)$$

$k = 2 \qquad c(1,2) = 7.$

$r(1,2) = k$

$r(1,2) = 2.$

* when $i = 2 \quad j = 3$

$w(2,3) = p(3) + q(3) + w(2,2) = 1+1+1 = 3$

$$c(2,3) = \min_{2<k\le 3} \{c(2,2) + c(3,3)\} + w(2,3) = \min\{0+0\} + 3 = 3$$

$k = 3$

$r(2,3) = k$

$r(2,3) = 3$

* when $i = 3, \quad j = 4$

$w(3,4) = p(4) + q(4) + w(3,3) = 1+1+1 = 3.$

$$c(3,4) = \min_{3<k\le 4} \{c(3,3) + c(4,4)\} + w(3,4) = \min\{0\} + 3 = 3$$

$k = 4$

$r(3,4) = k$

$r(3,4) = 4$

Step II :- For $j - i = 2$

* $i = 0, \quad j = 2$

$w(0,2) = p(2) + q(2) + w(0,1)$

$= 3 + 1 + 8 = 12$

$$c(0,2) = \min_{0<k\le 2} \{c(0,0) + c(1,2), \; c(0,1) + c(2,2)\} + w(0,2) = \min\{7+0, \; 8+0\}$$

$k = 1, k = 2$

$= \min\{7, 8\} + 12$

$= 7 + 12 = 19$

$r(0,2) = 1$ ($\because k=1$ we got minimum value.

* when $i=1$ $j=3$

$w(1,3) = p(3) + q(3) + w(1,2)$

$= 1 + 1 + 7 = 9.$

$c(1,3) = \min_{1<k\le 3} \{c(1,1)+c(2,3), c(1,2)+c(3,3)\} + w(1,3)$

$k=2$, $k=3$

$= \min\{0+3, 7+0\} + 9$

$= 3 + 9 = 12$

$r(1,3) = k$

$r(1,3) = 2$

* when $i=2$ $j=4$

$w(2,4) = p(4) + q(4) + w(2,3)$

$= 1 + 1 + 3 = 5$

$c(2,4) = \min_{2<k\le 4} \{c(2,2)+c(3,4), c(2,3)+c(4,4)\} + w(2,4)$

$k=3$, $k=4$

$= \min\{0+3, 3+0\} + 5$

$= 3 + 5 = 8$

$r(2,4) = k$

$r(2,4) = 3$ (first 3 for $k=3$)

* when $i=$

Step III :-

* when $i=0$ $j=3$

$w(0,3) = p(3) + q(3) + w(0,2)$

$= 1 + 1 + 12 = 14$

$k=1$ $k=2$

$c(0,3) = \min_{0<k\le 3} \{c(0,0)+c(1,3), c(0,1)+c(2,3), c(0,2)+c(3,3)\} + w(0,3)$

$k=1,2,3$

$= \min\{0+12, 8+3, 19+0\} + 14 = \min\{12, 11, 19\} + 14 = 11 + 14$

12 11 19

$= 25$

$r(0,3) = k \qquad r(0,3) = 2$

* when $i = 1 \quad j = 4$

$w(1,4) = P(4) + q(4) + w(1,3)$

$= 1 + 1 + 9 = 11$

$c(1,4) = \min_{1 < k \le 4} \{ c(1,1) + c(2,4), c(1,2) + c(3,4), c(1,3) + c(4,4) \} + w(1,4)$

($k = 2$, $k = 3$, $k = 4$)

$= \min\{ 8 + 0, 7 + 3, 12 + 3 \} + 11$

$= 8 + 11 = 19$.

$r(1,4) = 2$

Step IV :-

* when $i = 0 \quad j = 4$

$w(0,4) = P(4) + q(4) + w(0,3)$

$= 1 + 1 + 14 = 16$

$c(0,4) = \min_{0 < k \le 4} \{ c(0,0) + c(1,4), c(0,1) + c(2,4), c(0,2) + c(3,4), c(0,3) + c(4,4) \} + w(0,4)$

($k = 1$, $k = 2$, $k = 3$, $k = 4$)

$= \min\{ 0 + 19, 8 + 8, 19 + 3, 25 + 0 \} + 16$

$= \min\{ 19, 16, 22, 25 \} = 16 + 11 = 32$

$r(0,4) = k$

$r(0,4) = 2$

	0	1	2	3	4
$j-i=0$	$w(0,0) = 2$ $c(0,0) = 0$ $r(0,0) = 0$	$w(1,1) = 3$ $c(1,1) = 0$ $r(1,1) = 0$	$w(2,2) = 1$ $c(2,2) = 0$ $r(2,2) = 0$	$w(3,3) = 1$ $r(3,3) = 0$ $r(3,3) = 0$	$w(4,4) = 1$ $c(4,4) = 0$ $r(4,4) = 0$
$j-i=1$	$w(0,1) = 8$ $c(0,1) = 8$ $r(0,1) = 1$	$w(1,2) = 7$ $c(1,2) = 7$ $r(1,2) = 2$	$w(2,3) = 3$ $c(2,3) = 3$ $r(2,3) = 3$	$w(3,4) = 3$ $c(3,4) = 3$ $r(3,4) = 4$	×
$j-i=2$	$w(0,2) = 12$ $c(0,2) = 19$ $r(0,2) = 1$	$w(1,3)$ 9 $c(1,3) = 12$ $r(1,3) = 2$	$w(2,4) = 5$ $c(2,4) = 8$ $r(2,4) = 3$	×	×
$j-i=3$	$w(0,3) = 14$ $c(0,3) = 25$ $r(0,3) = 2$	$w(1,4) = 11$ $c(1,4) = 19$ $r(1,4) = 2$	×	×	×

UNIT : 4

BACK TRACKING :

In case of greedy & dynamic programming technique, we use Brute Force approach i.e., evaluating all possible solution for a given n value & then selecting 1 solution as optimal. In this we will get the same optimal solution with less than no. of n trials. It is more efficient compared to greedy & dynamic programming. In this, we use bounding functions (or, criterian Function)

1. Explicit constraints : The rules that restrict each x_i to take a values only from a given set

Ex: $x_i \geq$ or $S_i = \{$ all non-negative real numbers $\}$

$x_i = 0$ or 1 or $S_i = \{0,1\}$

It completely depends upon the particular instance of the problem. All tuples must satisfy explicit constraints in order to define a possible solution.

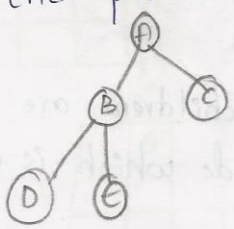
2. Implicit constraints : Rules that determine which of tuples in solution space of i satisfies the criterian function. It describes the way the which x_i must relate to each other

Ex: 0/1 knapsack problem.

Bounding Function.

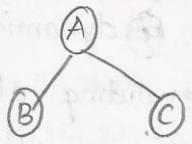
3. Criterian Function : It is a function $P(x_1, x_2 \ldots x_n)$ which needs to be maximized or minimized for a given problem.

4. Solution Space : All touples that satisfies the explicit constraints defines a possible solution for a particular instance for i in the problem.



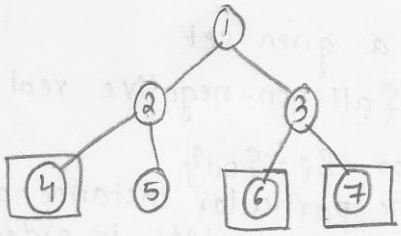
In above diagram ABD, ABE, AC are the tuples of solution space.

Problem State : Each node in the tree organization defines a problem state.



A, B, C are nodes of problem state.

a) Solution State : These are those problem state S for which the path from root to S defines a tuple in the solution space. □ indicates the solution space.

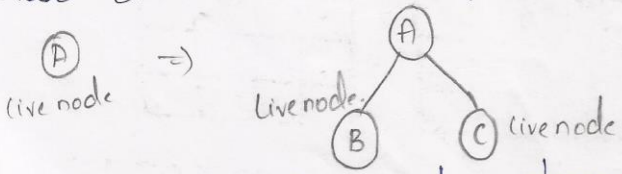


In above diagram there are 3 solution states which are represented in form of tuples i,e., (1,2,4), (1,3,6) (1,3,7)

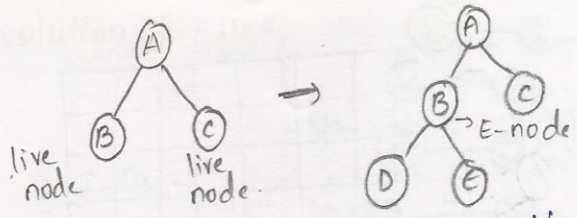
b) Answers State : The solution space 's' for which the path from root to S defines tuple which is a member of set of solutions (which satisfies implicit constraints of the problem)

In above diagram, answer states are 4,6,7.

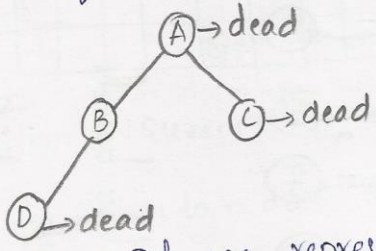
c) Live Node : A node which has been generated and all of those children have not yet been generated is called Live node.



d) E-node : The live nodes whose childrens are currently being generated is called E-node (the node which is expanded)



Dead Node : It is generated that is either not to be expanded further or one for which all of his childrens have been generated.



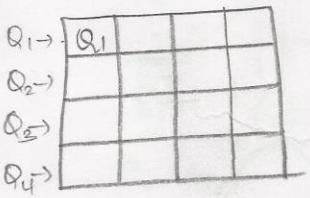
e) State Space Tree : If we represent a solution space in form of a tree, then the tree is called as state space tree.

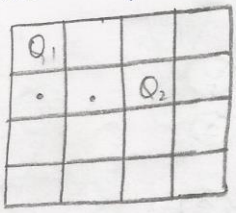
* n - Queens Problem :

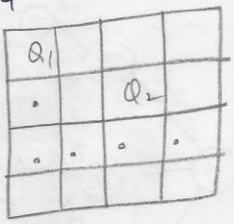
The n queens are to be placed on nxn chess board so that no two queens should be on the same row, same column, same diagonal. The solution space consists of n! permutations of n tuples. A tree with this permutation is called permutation tree. Edges are labelled by possible values of x_i. Edges from level i to level $i+1$ nodes specifies the values of x_i. Solution space is defined by all paths from root node to leaf node.

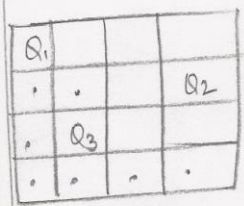
→ Find solution for n=4.

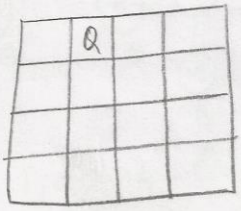
Since n=4 we require 4x4 chess board.

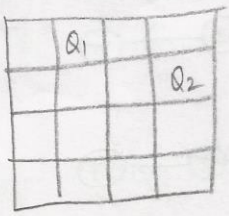


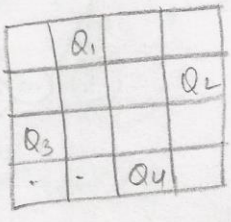












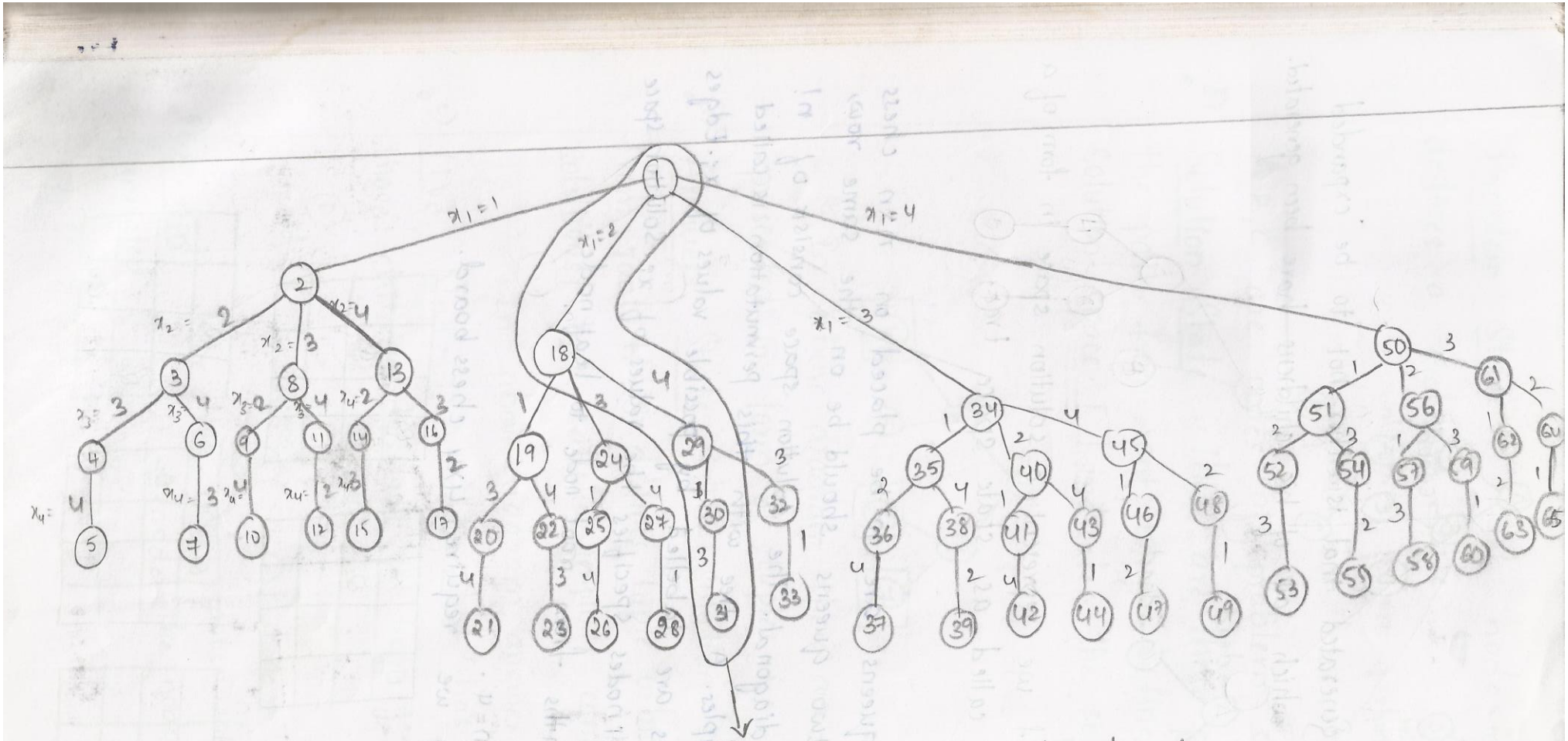
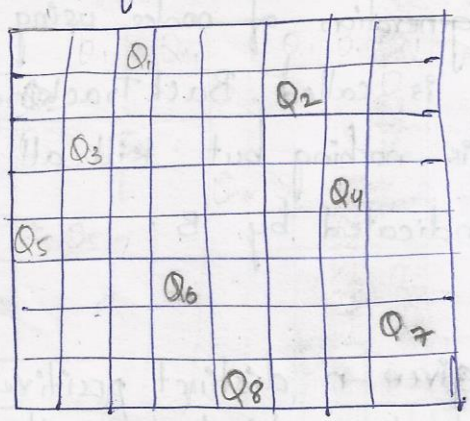


Fig: Solution Space tree for n=4 Using depth first Search.

Find the solution for n=8

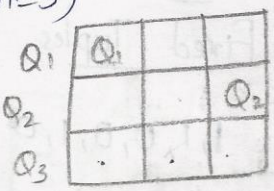


3, 6, 2, 7, 1, 4, 8, 5

Algorithm:

```
Algorithm NQueens(k,n)
{
    for i := 1 to n do
    {
        if place(k,i) then
        {
            x[k] := i;
            if (k=n) then write(x[1:n]);
            else NQueens(k+1,n);
        }
    }
}
```

Justify your answer for which value of n, the n queens problem will fail. (n=3)



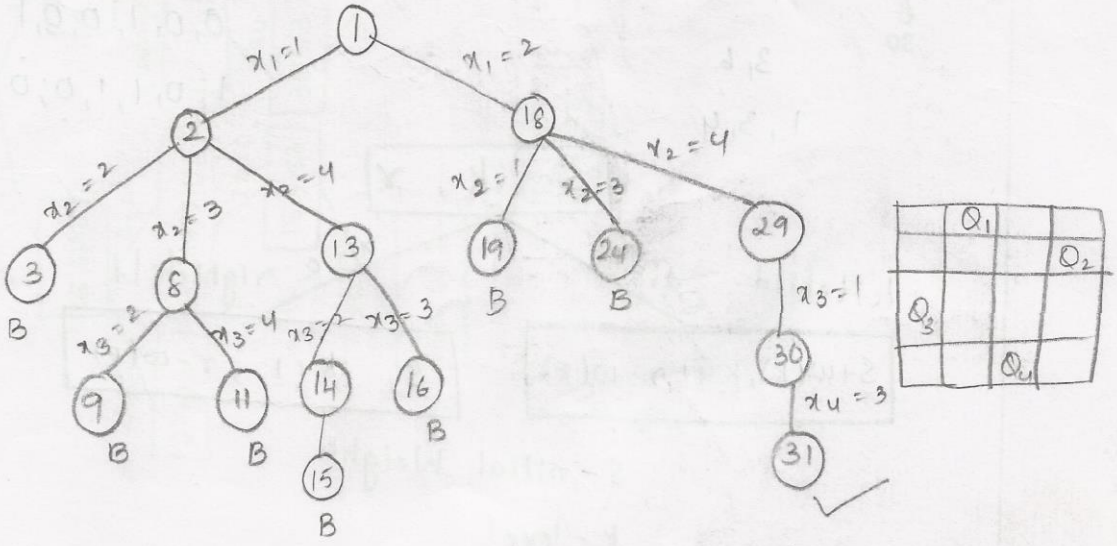


Fig: 4-Queens using BackTracking.

The above 4 queens generation of nodes using depth first search with bounding function is called Backtracking.
The bounding function is nothing but kill all nodes which is not generating solution, indicated by B.

Sum of Subsets :-

For a given n distinct positive numbers usually called weights, to find all combination of these weights whose sums are m, this is called the sum of subsets problem. It is solved using the fixed sized tuples and variable sized tuples. We consider the backtracking solution using fixed sized tuples. The element x_i of solution vector is either 0 or 1. ($\because 0 \rightarrow$ weight is not consider, $1 \rightarrow$ weight is considered). For a node at level i the left child corresponds to $x_i = 1$ & right child corresponds to $x_i = 0$. A simple choice for bounding is $B_k(x_1 \dots x_k)$ = true iff

$$\sum_{i=1}^{k} w_i x_i + \sum_{i=k+1}^{n} w_i \geq m \quad \& \quad \sum_{i=1}^{k} w_i x_i + w_{k+1} \leq m.$$

17 Find out solution for a given $n = 6$, $m = 30$, $w[1:6] = \{5, 10, 12, 13, 15, 18\}$

Variable tuples

(5, 10, 15) 1, 2, 5
↓
30

3, 6

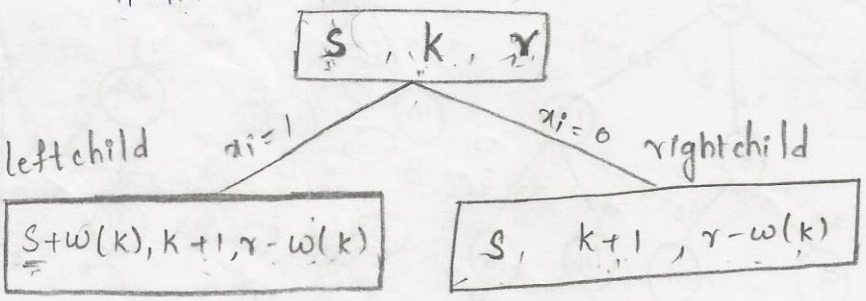
1, 3, 4

Fixed Tuples

1, 1, 0, 0, 1, 0 — (A)

0, 0, 1, 0, 0, 1 — (B)

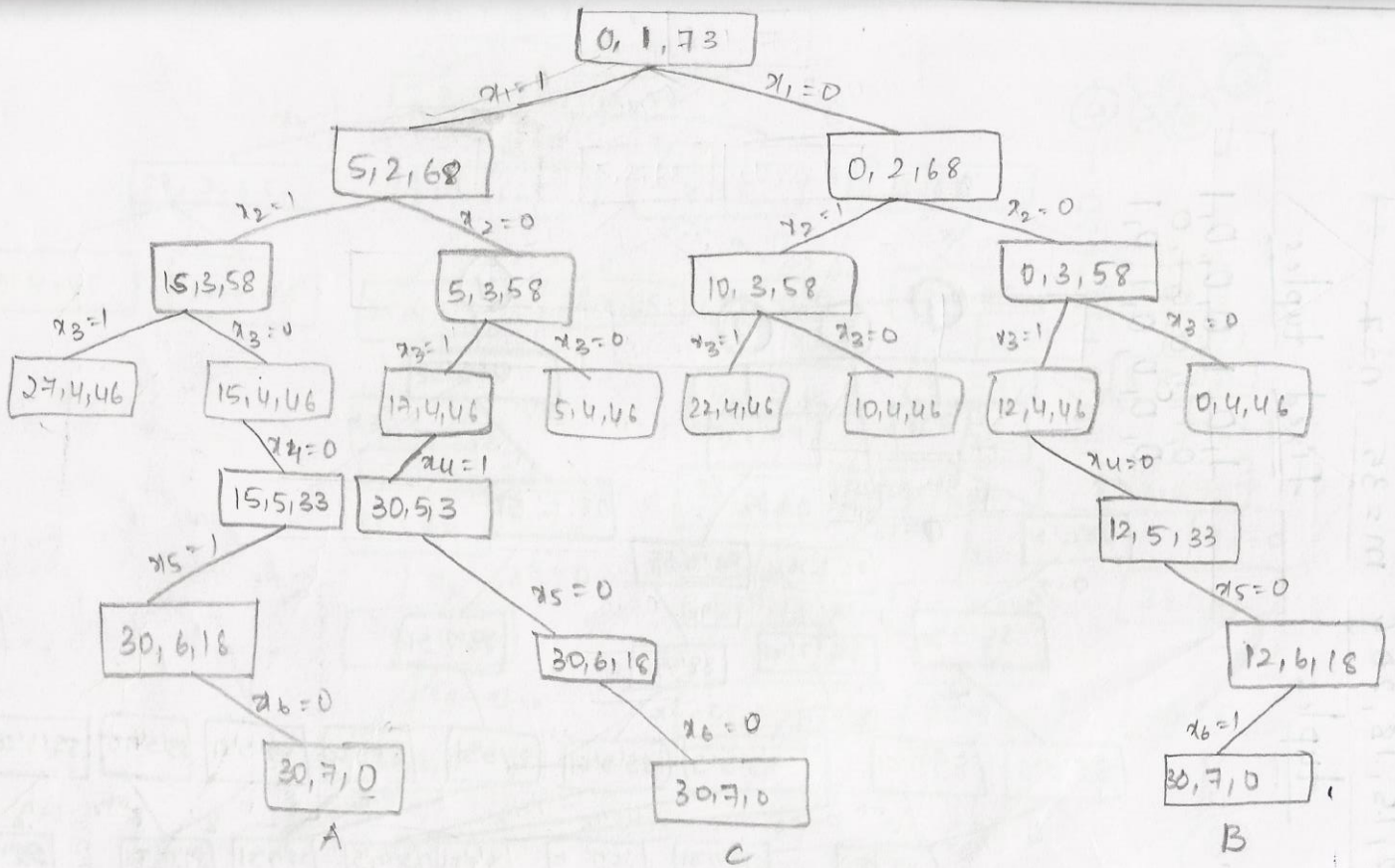
1, 0, 1, 1, 0, 0 — (C)



s - initial weight

k - level

r - sum of all weights.


0, 1, 73
$x_1=1$
$x_1=0$
5, 2, 68
0, 2, 68
$x_2=1$
$x_2=0$
$x_2=1$
$x_2=0$
15, 3, 58
5, 3, 58
10, 3, 58
0, 3, 58
$x_3=1$
$x_3=0$
$x_3=1$
$x_3=0$
$x_3=1$
$x_3=0$
$x_3=1$
$x_3=0$
27, 4, 46
15, 4, 46
12, 4, 46
5, 4, 46
22, 4, 46
10, 4, 46
12, 4, 46
0, 4, 46
$x_4=0$
$x_4=1$
$x_4=0$
15, 5, 33
30, 5, 3
12, 5, 33
$x_5=1$
$x_5=0$
$x_5=0$
30, 6, 18
30, 6, 18
12, 6, 18
$x_6=0$
$x_6=0$
$x_6=1$
30, 7, 0
30, 7, 0
30, 7, 0
A
C
B

2. $w = \{5, 7, 10, 12, 15, 18, 20\}$ $m = 35$ $n = 7$.

3. $w = \{15, 7, 20, 5, 18, 10, 12\}$ $m = 35$ $n = 7$

4. $w = \{20, 18, 15, 12, 10, 7, 5\}$ $m = 35$ $n = 7$.

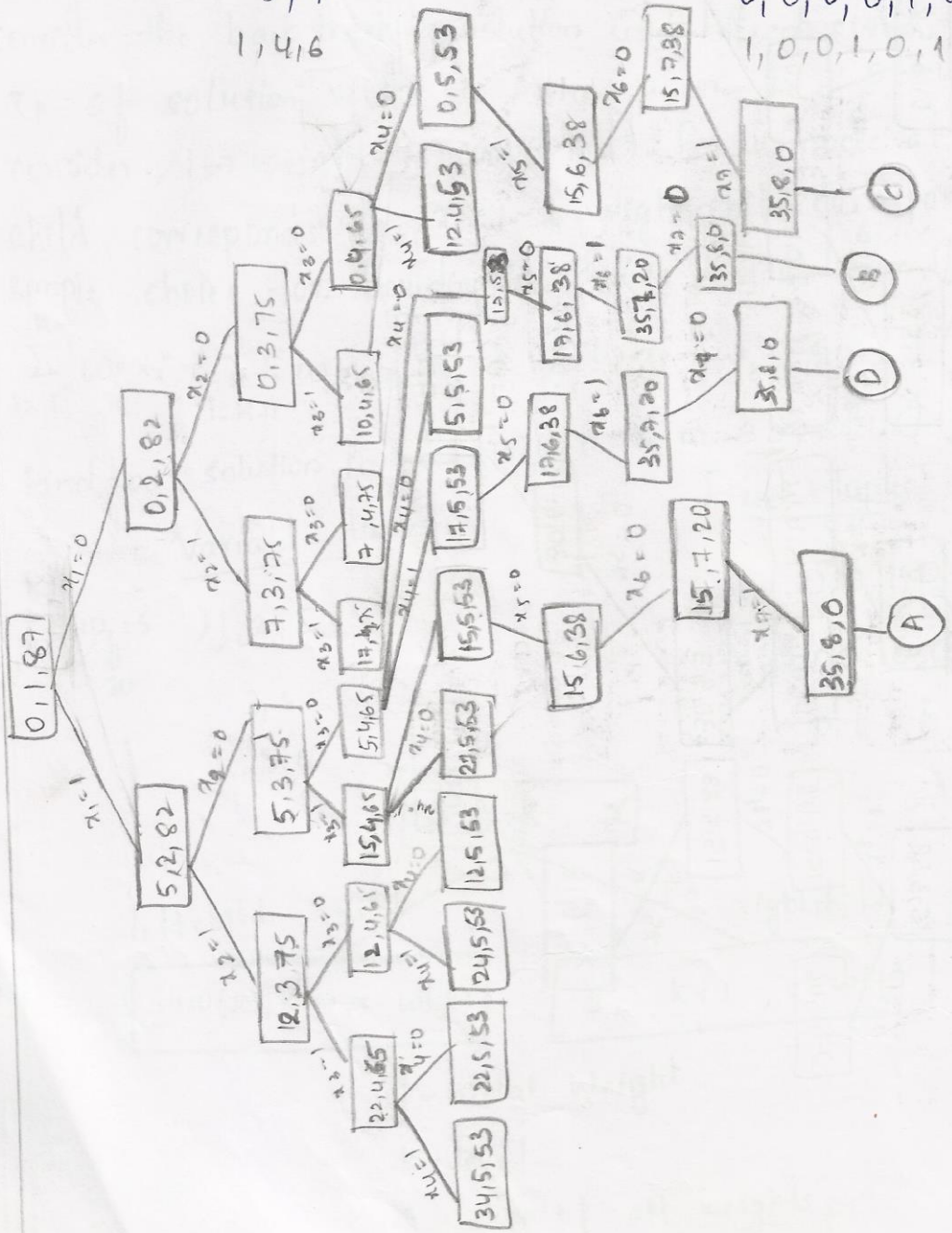
2. $w = \{\overset{1}{5}, \overset{2}{7}, \overset{3}{10}, \overset{4}{12}, \overset{5}{15}, \overset{6}{18}, \overset{7}{20}\}$ $m = 35$ $n = 7$. 87

Variable tuples

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

Fixed tuples

1, 0, 1, 0, 0, 0, 1 — (A)
0, 1, 1, 0, 0, 1, 0 — (B)
0, 0, 0, 0, 1, 0, 1 — (C)
1, 0, 0, 1, 0, 1, 0 — (D)



$w = \{15, 7, 20, 5, 18, 10, 12\}$ (1 2 3 4 5 6 7) $m = 35$ $n = 7$.

Variable tuples

1, 3

4, 5, 7.

3, 4, 6

2, 5, 6

Fixed tuples

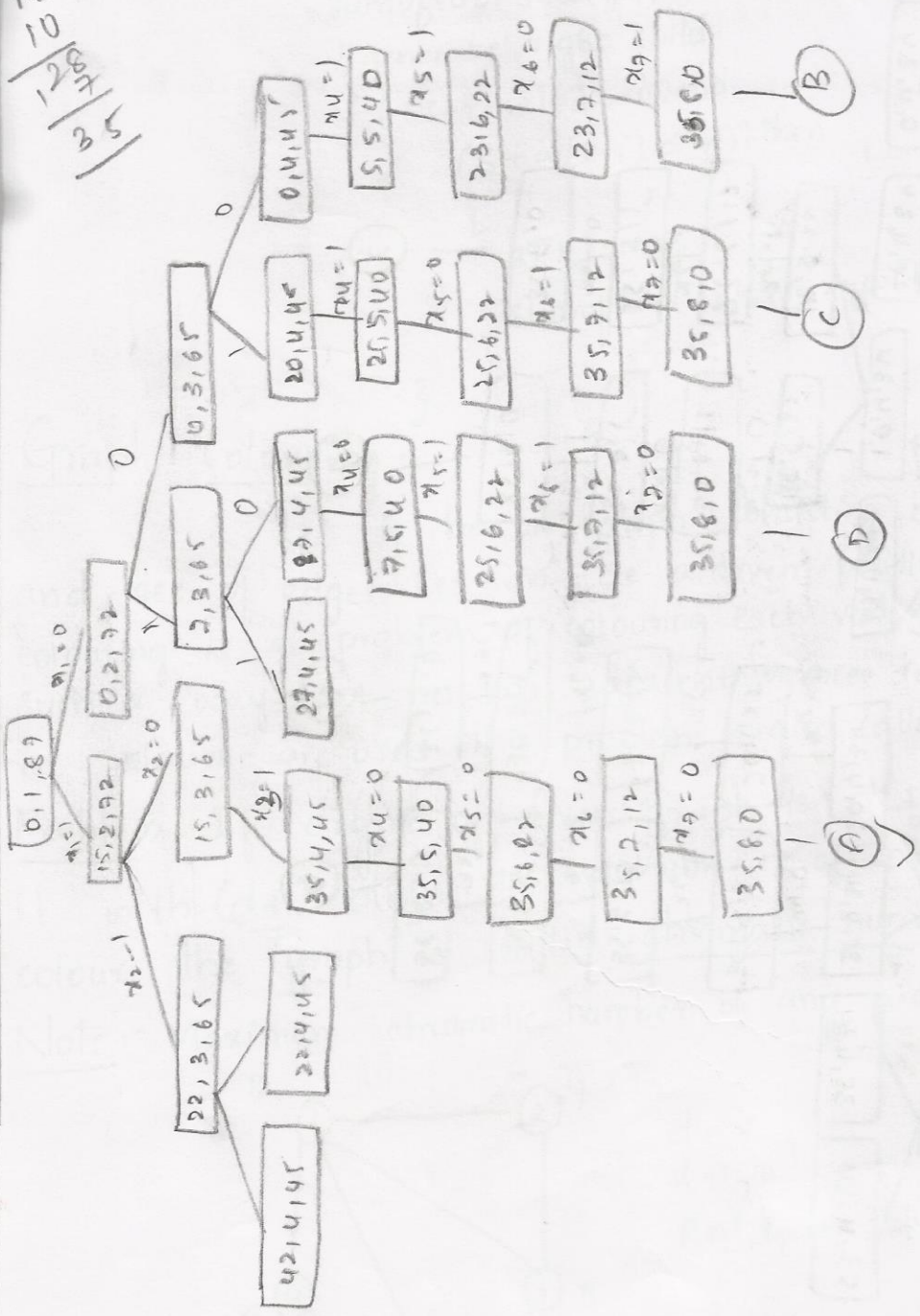
1, 0, 1, 0, 0, 0, 0 — (A)

0, 0, 0, 1, 1, 0, 1 — (B)

0, 0, 1, 1, 0, 1, 0 — (C)

0 1 0 0 1 1 0 — (D)
1 2 3 4 5 6 7

1
1
10
128
35



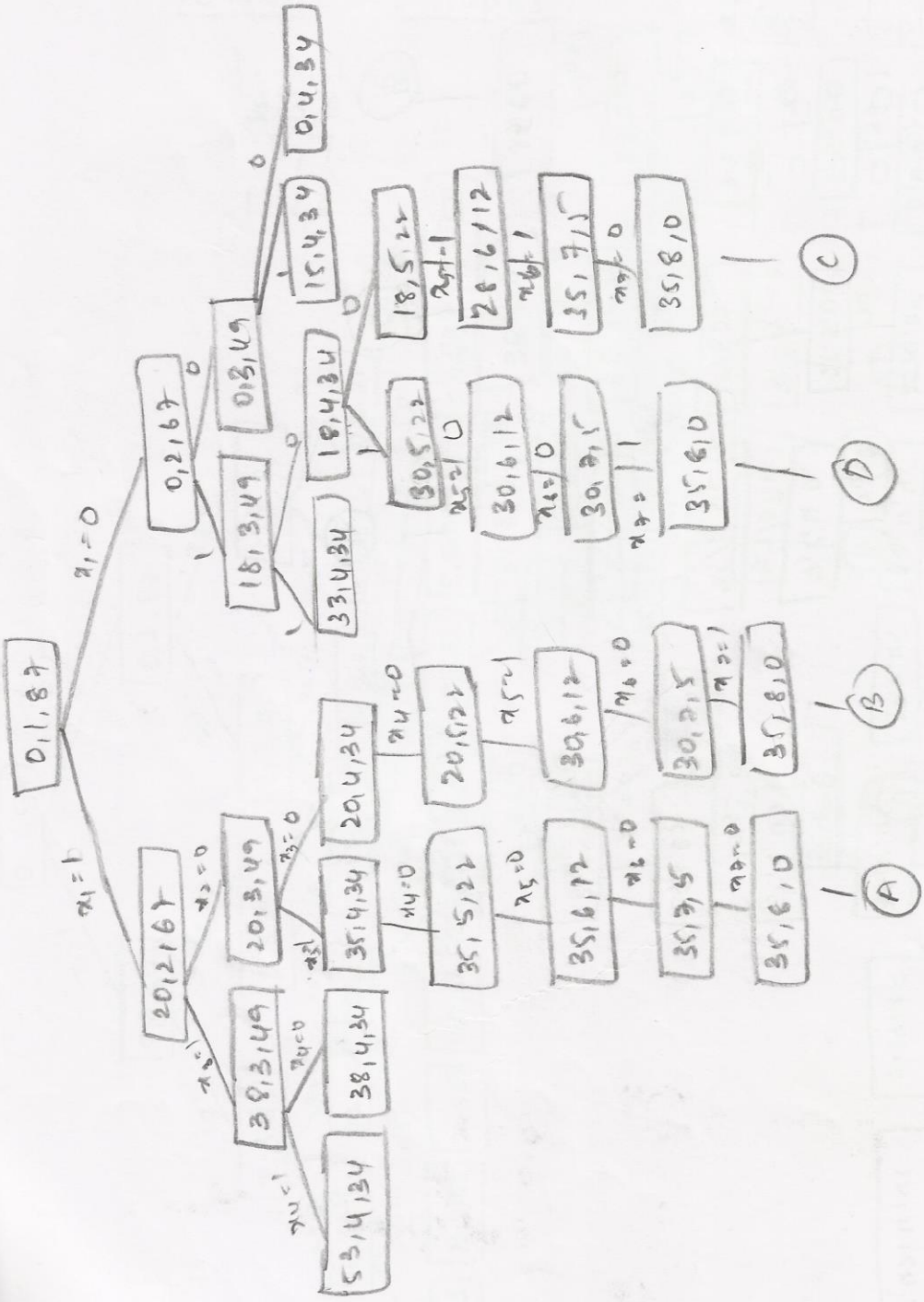
4 $w = \{20, 18, 15, 12, 10, 7, 5\}$ $n = 7$ $m = 35$

Variable tuples

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

Fixed tuples

1, 0, 1, 0, 0, 0, 0 — A
1, 0, 0, 0, 1, 0, 1 — B
0, 1, 0, 0, 1, 1, 0 — C
0, 1, 0, 1, 0, 0, 1 — D



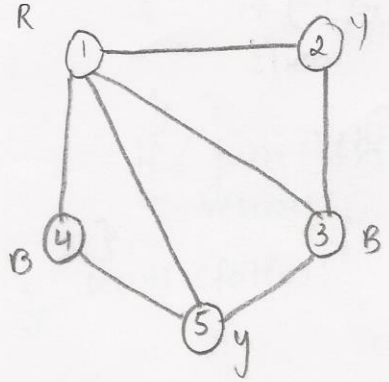
Algorithm:

```
Algorithm sumofsub(s, k, r)
{
    x[k] := 1; // generate leftchild.
    if (s+w[k] = m) then
      write (x[1:k]); // subsetfound
      else if (s+w[k]+w[k+1] ≤ m)
      then
      sumofsub(s+w[k], k+1, r-w[k]);
      // generate right child.
      if ((s+r-w[k] ≥ m) and
            (s + w[k+1] ≤ m)) then
            {
              x[k] := 0;
              sumofsub(s, k+1, r-w[k]);
            }
}
```

Graph Colouring :-

Let G be a graph consisting of set of vertices and set of edges. Let 'm' be a given positive integer. Graph colouring is a problem of colouring each vertex in a graph in such a way that no two adjacent vertices have same colour and 'm' colours are used. This problem is also called as M-Colouring Problem. If degree of given graph is d, then we can colour it with (d+1) colours. The minimum no. of colours required to colour the graph is called chromatic number.

Note :- Maximum chromatic number of any planar graph is 4.



d = 3
d+1 = 4
Red, Green, Blue, Yellow

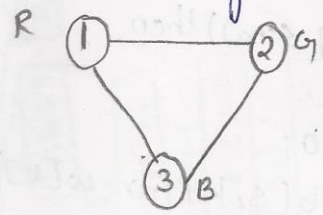
We use backtracking problem ~~using~~ for graph colouring as follows.

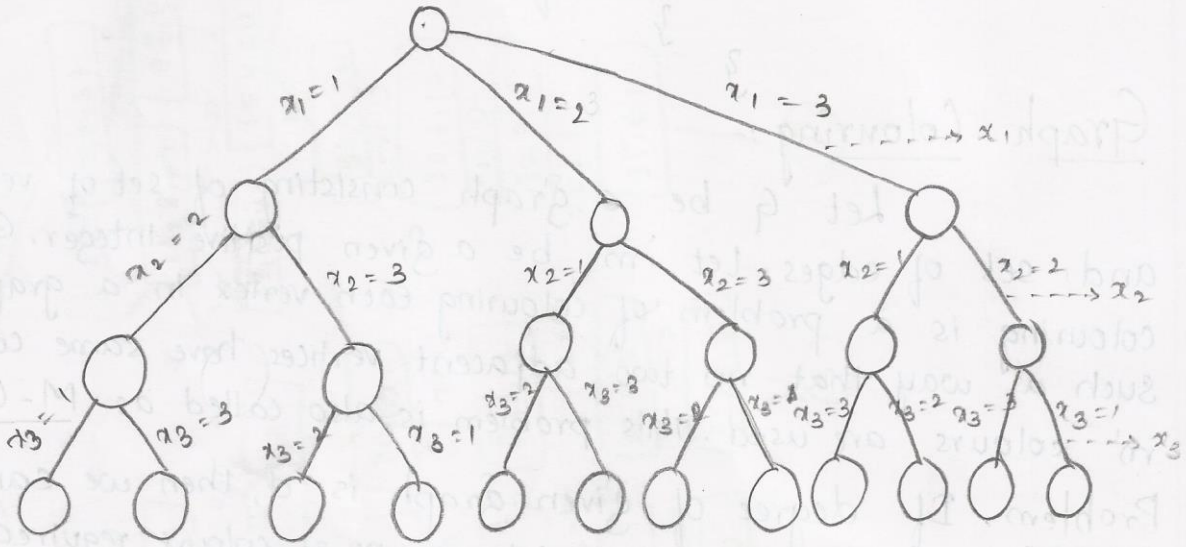
→ Let g be a graph consisting of 'n' vertices with adjacent matrices $A = [a_{ij}]_{n \times n}$ where $a_{ij} = 1$ if $(i,j) \in E(G)$

$= 0$ if $(i,j) \notin E(G)$

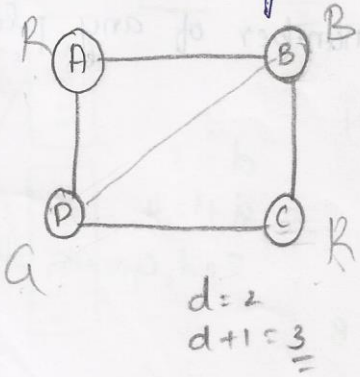
→ Let colours are represented by integers $(1, 2, \ldots m)$ and $(x_1, x_2, \ldots x_n)$ be solution where x_i = colour of vertex i.

*. Draw the m-colouring solution state space tree for n=3 m=3.



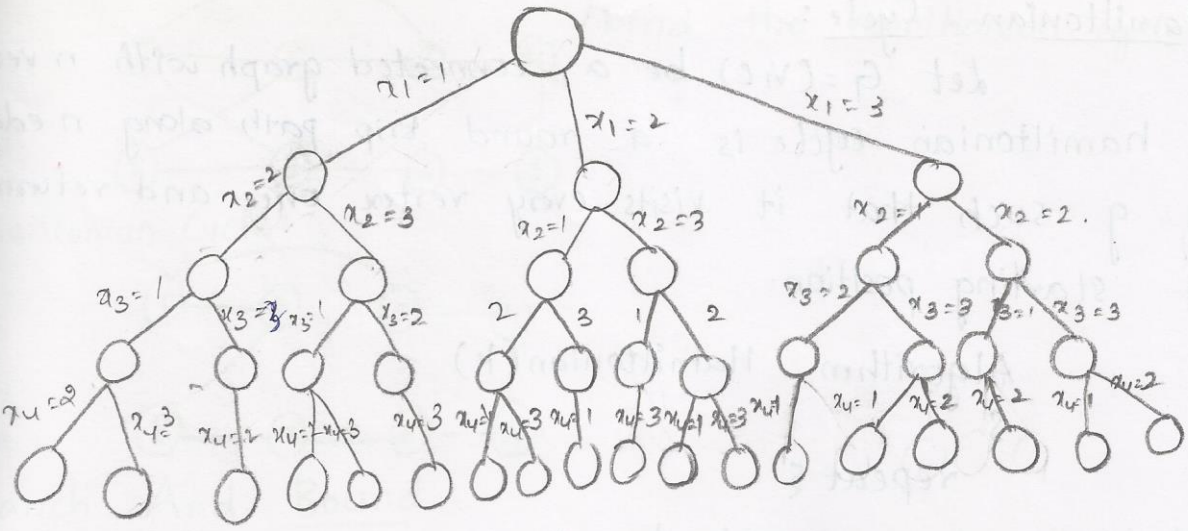


* Draw the m-colouring solution state space tree for n=4 and m=3.



d = 2

d+1 = 3



Algorithm :

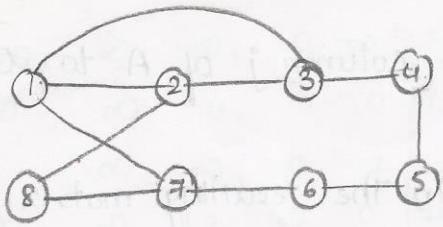
```
Algorithm m_colouring(k)
{
  repeat
  {
    nextvalue(k);
    if(x[k]=0) then return;
    if(k=n) then
     write(x[1:n]);
    else mcolouring(k+1);
  }
  until(false);
}
Algorithm nextvalue(k)
{
    repeat
    {
     x[k]:=(x[k]+1)mod(m+1);
     if(x[k]=0) then return;
     for j:=1 to n do
       {     if ( (G[k,j] ≠ 0) and (x[k]=x[j]))
             then
       }
      if ( j=n+1) then
          return;
    }
    until(false);
}
```

Hamiltonian Cycle:

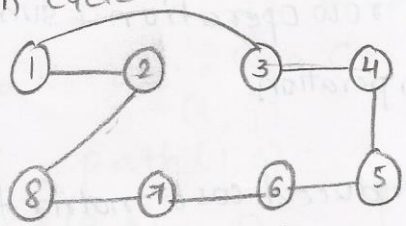
Let G = (V, E) be a connected graph with n vertices A hamiltonian cycle is a round trip path along n edges of G such that it visits every vertex once and returns to it starting position.

```
Algorithm Hamiltonian(k)
{
  repeat {
        nextvalue(k);
        if(x[k]=0) then return;
        if(k=n) then
          write (x[1:n]);
          else Hamiltonian(k+1);
        }
        until false;
    }
Algorithm nextvalue(k)
{
    repeat {
        x[k]:=(x[k+1])mod(n+1);
        if (x[k]=0) then return;
        if (G[x[k-1]], x[k]] ≠0) then
        {
            for j:=1 to k-1 do
            if(x[j]:=x[k]) then
              break;
            if (j=k) then
            if((k<n) or (k=n) and G(x[n], x[1])≠0))
            then return;
            }
        }
        until (false);
        }
```

Find the hamiltonian cycle?



Hamiltonian Cycle:



Branch And Bound:

The term Branch & Bound refers to all state space search methods in which all childrens of ~~Each~~ node are generated before any other live node can become the E-node. In Branch & Bound, BFS, state space search are called FIFO technique and DFS state space search are called LIFO. In back tracking, we generate solution state space tree using only DFS, whereas in branch & Bound we use both BFS & DFS techniques. In backtracking bounding functions are used to avoid generation of subtrees that do not contain an answer node. Backtracking is more efficient for decision problems not for optimization problems. The main difference between branch & bound and backtracking is that if we get the solution then we will terminate the search procedure in backtracking, whereas BB we will continue the process until we get a optimal solution. BB is applicable only for minimum values.

Travelling Sales Person Problem:-

Let $G=(V,E)$ be a directed graph. Let $(C(i,j))C_{ij}$ $=\infty$ if $i,j \notin E$. TSP can be calculated based upon the least cost Branch And Bound (LCBB).

Procedure :-

Let A be reduced cost matrix for root noode R.

Let S be a child of R such that the tree edge R,S corresponds to include edge i, j in the tool.

3. Change all n-trees in row i & column j of A to ∞.

4. Set A(j,1) to ∞

5. Reduce all rows & columns in the resulting matrix.

6. Cost of R => $\hat{c}$(R) = sum of row operation + sum of column operation
   ↙ reduced matrix.

7. If s is not a leaf then the reduced cost matrix for s can be obtained as

$$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$$

↓ cost of reduced matrix. ; ↓ ith row of reduced cost matrix ; → jth column of reduced cost matrix ; ↳ reduced cost matrix.

*.
$$\begin{bmatrix} \infty & 20 & 30 & 10 & 11 \\ 15 & \infty & 16 & 4 & 2 \\ 3 & 5 & \infty & 2 & 4 \\ 19 & 6 & 18 & \infty & 3 \\ 16 & 4 & 7 & 16 & \infty \end{bmatrix} \begin{matrix} r_1 = 10 \\ r_2 = 2 \\ r_3 = 2 \\ r_4 = 3 \\ r_5 = 4 \end{matrix}$$

$c_1 =$ $c_2 =$ $c_3 =$ $c_4 =$ $c_5 =$

$$\begin{bmatrix} \infty & 10 & 20 & 0 & 1 \\ 13 & \infty & 14 & 2 & 0 \\ 1 & 3 & \infty & 0 & 2 \\ 16 & 3 & 15 & \infty & 0 \\ 12 & 0 & 3 & 12 & \infty \end{bmatrix} \Rightarrow \begin{bmatrix} \infty & 10 & 17 & 0 & 1 \\ 12 & \infty & 11 & 2 & 0 \\ 0 & 3 & \infty & 0 & 2 \\ 15 & 3 & 12 & \infty & 0 \\ 11 & 0 & 0 & 12 & \infty \end{bmatrix} = A$$

$c_1 = 0$ $c_2 = 0$ $c_3 = 3$ $c_4 = 0$ $c_5 = 0$

$$\hat{c}(R) = 10 + 2 + 2 + 3 + 4 + 1 + 3$$

$$\hat{c}(R) = \underline{25}$$

<u>Step 1</u> :- path(i, j) (1, 2)

1) (j, j) = ∞ ; ith row & jth row

2) A(j, 1) = ∞ ; A(2, 1) = ∞

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 11 & 2 & 0 \\ 0 & \infty & \infty & 0 & 2 \\ 15 & \infty & 12 & \infty & 0 \\ 11 & \infty & 0 & 12 & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix}$$

$c_1=0 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 25 + A(1,2) + r$

$= 25 + 10 + 0 = 35$ //

(i, j)

Step 2:- path (1, 3)

1) $A(3,1) = \infty$

2) $(1,3) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & \infty & 2 & 0 \\ 0 & 3 & \infty & 0 & 2 \\ 15 & 3 & \infty & \infty & 0 \\ 11 & 0 & \infty & 12 & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix} \Rightarrow$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & 2 & 0 \\ \infty & 3 & \infty & 0 & 2 \\ 4 & 3 & \infty & \infty & 0 \\ 0 & 0 & \infty & 12 & \infty \end{bmatrix}$$

$c_1=11 \quad c_2=0 \quad c_3=\infty \quad c_4=\infty \quad c_5=\infty$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 25 + 19 + 11$

$= 53$

Step 3: path (1, 4)

1) $1,4 = \infty$

2) $A(4,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & 11 & \infty & 0 \\ 0 & 3 & \infty & \infty & 2 \\ 15 & 3 & 12 & \infty & 0 \\ 11 & 0 & 0 & \infty & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix}$$

$c_1=0 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 25 + A(1,4) + 0$

$= 25 + 0 + 0 = 25$

Step 4:- path (1,5)

$1,5 = \infty$

$A(5,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & 11 & 2 & \infty \\ 0 & 3 & \infty & 0 & \infty \\ 15 & 3 & 12 & \infty & \infty \\ 11 & 0 & 0 & 12 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=2 \\ r_3=0 \\ r_4=3 \\ r_5=0 \end{matrix}$$

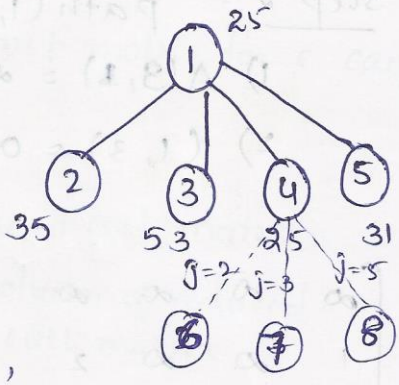
$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 10 & \infty & 9 & 0 & \infty \\ 0 & 3 & \infty & 0 & \infty \\ 12 & 0 & 9 & \infty & \infty \\ 11 & 0 & 0 & 12 & \infty \end{bmatrix} = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 10 & \infty & 9 & 0 & \infty \\ 0 & 3 & \infty & 0 & \infty \\ 12 & 0 & 9 & \infty & \infty \\ 11 & 0 & 0 & 12 & \infty \end{bmatrix}$$

$r_1=0 \quad r_2=0 \quad r_3=0 \quad r_4=0 \quad r_5=0$

$$\hat{c}(s) = \hat{c}(R) + A(1,5) + r$$

$$= 25 + 1 + 5$$

$$\hat{c}(s) = 31$$



Now

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & 11 & \infty & 0 \\ 0 & 3 & \infty & \infty & 2 \\ 3 & 3 & 12 & \infty & 0 \\ 11 & 0 & 0 & \infty & \infty \end{bmatrix} = A$$

$\hat{c}(R) = 25$

Step 5 :- path $(i,j) = (4,2)$

$4,2 = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 11 & \infty & 0 \\ 0 & \infty & \infty & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & 0 & \infty & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix}$$

$c_1=0 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=0$

$$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$$

$$= 25 + 3 + 0 = 28$$

Step 6 :- path $(i,j) = (4,3)$

1) $4,3 = \infty$

2) $A(3,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & \infty & \infty & 0 \\ \infty & 3 & \infty & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & \infty & \infty & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=2 \\ r_4=\infty \\ r_5=0 \end{matrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & \infty & \infty & 0 \\ \infty & 3 & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & \infty & \infty & \infty \end{bmatrix} = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & 0 \\ \infty & 1 & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & \infty & \infty & \infty \end{bmatrix}$$

$C_1 = 11 \quad C_2 = 0 \quad C_3 = \infty \quad C_4 = \infty \quad C_5 = 0$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 25 + 12 + 13$

$\hat{c}(7) = 50$

Step 7 :- $(i,j) = (4,5)$

$4,5 = \infty$

$A(5,1) = \infty$

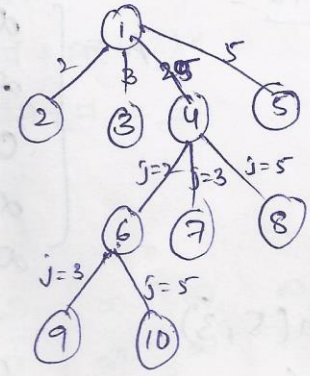
$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 12 & \infty & 11 & \infty & \infty \\ 0 & 3 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & 0 & \infty & \infty \end{bmatrix} \begin{matrix} r_1 = \infty \\ r_2 = 11 \\ r_3 = 0 \\ r_4 = 0 \\ r_5 = 0 \end{matrix} \Rightarrow \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & 0 & \infty & \infty \\ 0 & 3 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & 0 & \infty & \infty \end{bmatrix}$$

$C_1 = 0 \quad C_2 = 0 \quad C_3 = 0 \quad C_4 = 0 \quad C_5 = \infty$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 25 + 0 + 11$

$= 36.$



Ex.

$$A = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 11 & \infty & 0 \\ 0 & \infty & \infty & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty \\ 11 & \infty & 0 & \infty & \infty \end{bmatrix}$$

$\hat{c}(R) = 28$

Step 8 :- path $(i,j) = (2,3)$

$2,3 = \infty$

$A(3,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty \\ 11 & 0 & \infty & \infty & \infty \end{bmatrix} \begin{matrix} r_1 = \infty \\ r_2 = \infty \\ r_3 = 2 \\ r_4 = \infty \\ r_5 = 0 \end{matrix} = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & \infty & \infty & \infty \end{bmatrix}$$

$C_1 = 11 \quad C_2 = 0 \quad C_3 = \infty \quad C_4 = 0 \quad C_5 = 0$

$C_1 = 11$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 28 + 11 + 13 = 52//$

28
13
41
11
52

Step 9:- (i, j) = (2, 5)

2,5 = ∞

A(5,1) = ∞

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

$C_1 = 11 \quad C_2 = 0 \quad C_3 = 0 \quad C_4 = 0 \quad C_5 = 0$

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

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 28 + 0 + \infty$

$= 28$

Step 10:-

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

Path(5,3)

(5,3) = ∞

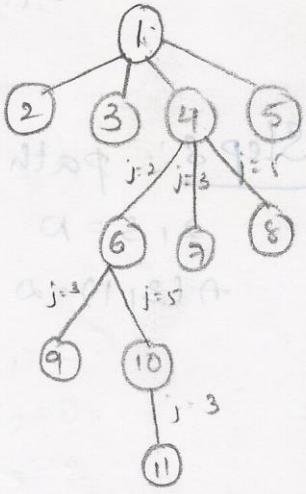
(3,1) = ∞

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 28 + 0 + 0 = 28$

$1 \rightarrow 4 \rightarrow 2 \rightarrow 5 \rightarrow 3 = 28$

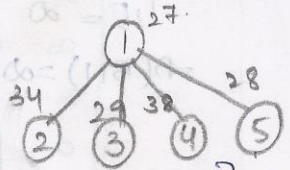


$$\begin{bmatrix} \infty & 7 & 3 & 12 & 8 \\ 3 & \infty & 6 & 14 & 9 \\ 5 & 8 & \infty & 6 & 18 \\ 9 & 3 & 5 & \infty & 11 \\ 18 & 14 & 9 & 8 & \infty \end{bmatrix} \begin{matrix} r_1=3 \\ r_2=3 \\ r_3=5 \\ r_4=3 \\ r_5=8 \end{matrix} \Rightarrow \begin{bmatrix} \infty & 4 & 0 & 9 & 5 \\ 0 & \infty & 3 & 11 & 6 \\ 0 & 3 & \infty & 1 & 13 \\ 6 & 0 & 2 & \infty & 8 \\ 10 & 6 & 1 & 0 & \infty \end{bmatrix} \Rightarrow \begin{bmatrix} \infty & 4 & 0 & 9 & 0 \\ 0 & \infty & 3 & 11 & 1 \\ 0 & 3 & \infty & 1 & 8 \\ 6 & 0 & 2 & \infty & 3 \\ 10 & 6 & 1 & 0 & \infty \end{bmatrix}$$

$C_1=0 \quad C_2=0 \quad C_3=0 \quad C_4=0 \quad C_5=5$ ⇓ A'

$\hat{c}(R) = 5+3+5+3+8+0+0+0+0+5$

$\hat{c}(R) = 27.$



Step 1:- (i,j) $(1,2)$

$1,2 = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 6 & 14 & 9 \\ 5 & \infty & \infty & 6 & 18 \\ 9 & \infty & 5 & \infty & 11 \\ 18 & \infty & 9 & 8 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2= \end{matrix} \qquad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 3 & 11 & 1 \\ 0 & \infty & \infty & 1 & 8 \\ 6 & \infty & 2 & \infty & 3 \\ 10 & \infty & 1 & 0 & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=1 \\ r_3=0 \\ r_4=2 \\ r_5=0 \end{matrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 10 & 0 \\ 0 & \infty & \infty & 1 & 8 \\ 4 & \infty & 0 & \infty & 1 \\ 10 & \infty & 1 & 0 & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 27 + A(1,2) + 3$

$\hat{c}(S) = 29 + 7 = 34$

$C_1=0 \quad C_2=\infty \quad C_3=0 \quad C_4=0 \quad C_5=0$

Step 2:- (i,j) $(1,3)$

$1,3 = \infty$

$A(3,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & 1 \\ \infty & 3 & \infty & 1 & 8 \\ 6 & 0 & \infty & \infty & 3 \\ 10 & 6 & \infty & 0 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=1 \\ r_4=0 \\ r_5=0 \end{matrix} \qquad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & 1 \\ 0 & 2 & \infty & 0 & 7 \\ 6 & 0 & \infty & \infty & 3 \\ 10 & 6 & \infty & 0 & \infty \end{bmatrix}$$

$C_1=0 \quad C_2=0 \quad C_3=\infty \quad C_4=0 \quad C_5=1$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & 0 \\ 0 & 2 & \infty & 0 & 6 \\ 6 & 0 & \infty & \infty & 2 \\ 10 & 6 & \infty & 0 & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 27 + 1 + 1 = 29$

$C_1=0 \quad C_2=0 \quad C_3=0 \quad C_4=0 \quad C_5=1$

Step 3: $(1,4)$

$1,4 = \infty$

$A(1,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 3 & \infty & 1 \\ 0 & 3 & \infty & \infty & 8 \\ \infty & 0 & 2 & \infty & 3 \\ 9 & 5 & 0 & \infty & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=1 \end{matrix} \qquad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 3 & \infty & 1 \\ 0 & 3 & \infty & \infty & 8 \\ \infty & 0 & 2 & \infty & 3 \\ 9 & 5 & 0 & \infty & \infty \end{bmatrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 3 & \infty & 0 \\ 0 & 3 & \infty & \infty & 7 \\ \infty & 0 & 2 & \infty & 2 \\ 9 & 5 & 0 & \infty & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$\hat{c}(4) = 27 + 9 + 2$

$= 27 + 11 = 38.$

Step 4 :- $(1,5) = \infty$

$1,5 = \infty$

$A(5,1) = \infty$

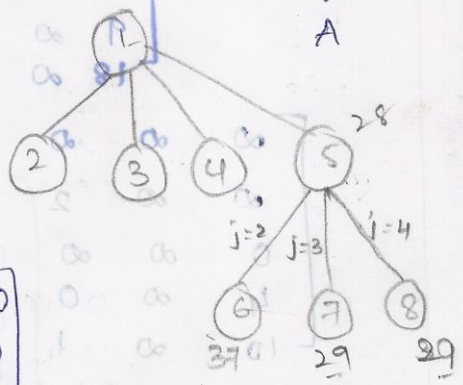
$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 3 & 11 & \infty \\ \infty & 3 & \infty & 1 & \infty \\ 6 & 0 & 2 & \infty & \infty \\ 10 & 6 & 1 & 0 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix} \quad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & 2 & 11 & \infty \\ 0 & 3 & \infty & 1 & \infty \\ 6 & 0 & 1 & \infty & \infty \\ 10 & 6 & 0 & 0 & \infty \end{bmatrix} \Downarrow A$$

$c_1=0 \quad c_2=0 \quad c_3=1 \quad c_4=0 \quad c_5=\infty$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 27 + A(1,5) + 1$

$= 27 + 1 = 28$



Step 5 :- $(5,2)$

$(5,2) = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 11 & \infty \\ 0 & \infty & \infty & 1 & \infty \\ 6 & \infty & 1 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \quad A =$$

$$\begin{bmatrix} \infty & \infty & 0 & 9 & 0 \\ 0 & \infty & 3 & 11 & 1 \\ 0 & \infty & \infty & 1 & \infty \\ 4 & \infty & 0 & \infty & 1 \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \quad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 11 & \infty \\ 0 & \infty & \infty & 1 & \infty \\ 6 & \infty & 1 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \begin{matrix} \\ r_2=2 \\ \\ r_4=1 \\ \end{matrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 9 & \infty \\ 0 & \infty & \infty & 1 & \infty \\ 5 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \quad c_4=1$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 8 & \infty \\ 0 & \infty & \infty & 0 & \infty \\ 5 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 28 + 6 + 4 = 38$

Step 6 :- path $(5,3)$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & \infty \\ \infty & 3 & \infty & 1 & \infty \\ 6 & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ 0 \\ 1 \\ 0 \\ 0 \end{matrix} \Rightarrow \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & \infty \\ \infty & 2 & \infty & 0 & \infty \\ 6 & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 28 + 0 + 1 = \underline{29}$

Step 7:- (5,4) path

$5,4 = \infty$

$A(4,1) = \infty$

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

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 28 + 0 + 1 = 29.$

Step 8:- $A = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 11 & \infty \\ 0 & 2 & \infty & 0 & \infty \\ 6 & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$

(3,2) path

$3,2 = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 11 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 6 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix} \begin{matrix} \\ r_2=11 \\ \\ r_4=6 \\ \end{matrix} \Rightarrow \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r = 29 + 17 = 46$

Step 9:- (3,4) path

$3,4 = \infty$

$A(4,1) = \infty$

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

$\hat{c}(s) = A(3,4) + r + \hat{c}(P) = \underline{29}$

Step 10:- (4,2) $= \infty$

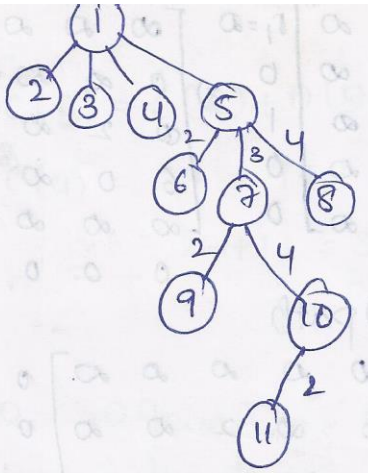
$4,2 = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

$\hat{c}(s) = 29$

$1 \to 5 \to 3 \to 4 \to 2$



1,5,3,4,2

$\Rightarrow 26$

$$\begin{bmatrix} \infty & 11 & 0 & 9 & 6 \\ 8 & \infty & 7 & 3 & 4 \\ 8 & 4 & \infty & 4 & 8 \\ 11 & 10 & 5 & \infty & 5 \\ 6 & 9 & 5 & 5 & \infty \end{bmatrix} \begin{matrix} r_1=0 \\ r_2=3 \\ r_3=4 \\ r_4=5 \\ r_5=5 \end{matrix} \quad \begin{bmatrix} \infty & 11 & 0 & 9 & 6 \\ 5 & \infty & 4 & 0 & 1 \\ 4 & 0 & \infty & 0 & 4 \\ 6 & 5 & 0 & \infty & 0 \\ 1 & 4 & 0 & 0 & \infty \end{bmatrix}$$

$c_1=1 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=0$

$$\begin{bmatrix} \infty & 11 & 0 & 9 & 6 \\ 4 & \infty & 4 & 0 & 1 \\ 3 & 0 & \infty & 0 & 4 \\ 5 & 5 & 0 & \infty & 0 \\ 0 & 4 & 0 & 0 & \infty \end{bmatrix} \Rightarrow A$$

$\hat{c}(R) = 3+4+5+5+1+0+0+0+0$

$\hat{c}(R) = 18$

Step 1:- path$_{(i,j)}$(1,2)

$1,2 = \infty$

$A(2,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 4 & 0 & 1 \\ 3 & \infty & \infty & 0 & 4 \\ 5 & \infty & 0 & \infty & 0 \\ 0 & \infty & 0 & 0 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=0 \\ r_4=0 \\ r_5=0 \end{matrix}$$

$c_1=0 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=0$

$\hat{c}(B) = \hat{c}(R) + A(i,j) + r$

$= 18 + A(1,2) + 0$

$= 18 + 11 = 29$

$\hat{c}(s) = 29$

$\hat{c}(2) = 29$

Step 2: path$_{(i,j)}$(1,3)

$1,3 = \infty \quad A(3,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & \infty & 0 & 1 \\ \infty & 0 & \infty & 0 & 4 \\ 5 & 5 & \infty & \infty & 0 \\ 0 & 4 & \infty & 0 & \infty \end{bmatrix} \begin{matrix} r_1 = \infty \\ r_2 = 0 \\ r_3 = 0 \\ r_4 = 0 \\ r_5 = 0 \end{matrix}$$

$C_1=0$ $C_2=0$ $C_3=0$ $C_4=0$ $C_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$\hat{c}(3) = 18$

$= 18 + 0 + 0 = 18$

Step 3: path (1, 4)
(i, j)

$1,4 = \infty$

$A(4,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & 4 & \infty & 1 \\ 3 & 0 & \infty & \infty & 4 \\ \infty & 5 & 0 & \infty & 0 \\ 0 & 4 & 0 & \infty & \infty \end{bmatrix} \begin{matrix} r_1 = \infty \\ r_2 = 1 \\ r_3 = 0 \\ r_4 = 0 \\ r_5 = 0 \end{matrix} \quad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 3 & \infty & 3 & \infty & 0 \\ 3 & 0 & \infty & \infty & 4 \\ \infty & 5 & 0 & \infty & 0 \\ 0 & 4 & 0 & \infty & \infty \end{bmatrix}$$

$C_1=0$ $C_2=0$ $C_3=0$ $C_4=0$ $C_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$\hat{c}(4) = 18 + A(1,4) + 1 \quad = 18 + 9 + 1 = 28$

Step 4: path (1, 5)

$1,5 = \infty$

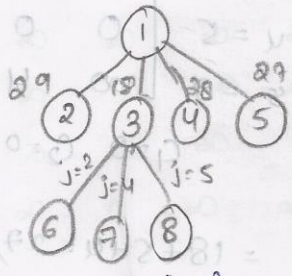
$A(5,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & 4 & 0 & \infty \\ 3 & 0 & \infty & 0 & \infty \\ 5 & 5 & 0 & \infty & \infty \\ \infty & 4 & 0 & 0 & \infty \end{bmatrix} \begin{matrix} r_1 = \infty \\ r_2 = 0 \\ r_3 = 0 \\ r_4 = 0 \\ r_5 = 0 \end{matrix}$$

$C_1=3$ $C_2=0$ $C_3=0$ $C_4=0$ $C_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 18 + 6 + 3 = 24 + 3 = 27$



i, j

Step 5: path (3, 2)

$3,2 = \infty$

$A(2,1) = \infty$

$$A = \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & \infty & 0 & 1 \\ \infty & 0 & \infty & 0 & 4 \\ 5 & 5 & \infty & \infty & 0 \\ 0 & 4 & \infty & 0 & \infty \end{bmatrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 1 \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & 0 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=\infty \\ r_4=0 \\ r_5=0 \end{matrix}$$

$c_1=0 \quad c_2=\infty \quad c_3=0 \quad c_4=0 \quad c_5=0$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r = 18 + A(3,2) + 0$

$\hat{c}(6) = 18+0 = 18$

Step 6 :- path $(3,4)$ (i,j)

$3,4 = \infty$

$A(4,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & \infty & \infty & 1 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 5 & \infty & \infty & 0 \\ 0 & 4 & \infty & \infty & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=1 \\ r_3=\infty \\ r_4=0 \\ r_5=0 \end{matrix}$$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 3 & \infty & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 5 & \infty & \infty & 0 \\ 0 & 4 & \infty & \infty & \infty \end{bmatrix} \quad c_2=4$$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 18 + A(3,4) + 1$

$= 18 + 0 + 4 = 23$

Step 7 :- path $(3,5)$ (i,j)

$3,5 = \infty$

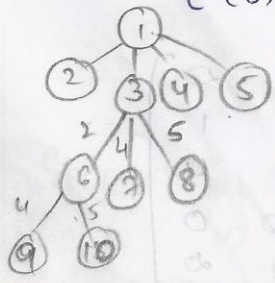
$A(5,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & 5 & \infty & \infty & \infty \\ \infty & 4 & \infty & 0 & \infty \end{bmatrix} \begin{matrix} r_1=\infty \\ r_2=0 \\ r_3=\infty \\ r_4=5 \\ r_5=0 \end{matrix} \quad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & \infty & \infty & \infty \\ \infty & 4 & \infty & 0 & \infty \end{bmatrix}$$

$c_1=0 \quad c_2=0 \quad c_3=0 \quad c_4=0 \quad c_5=\infty$

$\hat{c}(s) = \hat{c}(R) + A(i,j) + r$

$= 18 + 5 + A(3,5) = 18+5+4 = 27$



Step 8 :- path(i,j) path$(2,4)$

$$A=\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 1 \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & 0 & \infty \end{bmatrix}$$

$2,4 = \infty$

$A(4,1) = \infty$

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

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 18 + A(2,4)$

$= 18 + 0 = 18.$

Step 9 :- path(i,j) path$(2,5)$

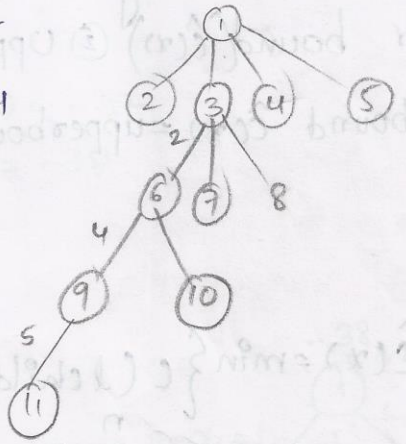
$2,5 = \infty$

$A(5,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 0 & \infty \end{bmatrix} \quad \begin{matrix} \\ \\ \\ 5 \\ 0 \end{matrix} \quad \begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 0 & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(2,5) + r$

$= 18 + 1 + 5 = 24$



Step 10 :- path(i,j) path$(4,5)$

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

$4,5 = \infty$

$A(5,1) = \infty$

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

$\hat{c}(S) = \hat{c}(R) + A(i,j) + r$

$= 18 + 0 = 18$ //

$1 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 5$

$= 18$

0/1 Knapsack Problem:-

→ A maximization problem can be easily converted into the minimization problem by changing the sign of the objective function

→ for 0/1 knapsack problem, techniques are used.

(1) FIFOBB expanding the nodes using BFS technique.

(2) LIFOBB expanding the nodes using DFS technique

We expanding the node by taking the $x_i = 1$ for left child and $x_i = 0$ for right child.

1) LIFOBB :- / LCBB (Least Cost Branch & Bound)

Inorder to find the solution for a given problem using LIFO BB for 0/1 knapsack problem.

The following 2 methods are used.

① Lower bound $(\hat{c}(x))$ ② Upper bound $(\mu(x))$.

Lower bound $\hat{c}(x)$ = upperbound - remaining weight in knapsack ~~bag~~ bag × remaining profit

(remaining weight which is not considered)

$$\hat{c}(x) = \min\{c(lchild(x)), c(rchild(x))\}$$

Upper bond $\mu(x) = \sum_{i=1}^{n} P_i$

→ Inorder to expand the next level of nodes, we consider ~~lower bound values~~. the lower bound values.

→ If both the ~~bar~~ nodes consists of same lower bound values then select the minimum upper bound values.

1. $n = 4$ $m = 15$ $P = 10, 10, 12, 18$ $w = 2, 4, 6, 9$.

For node (1)

$w_1 + w_2 + w_3 + w_4 \le m$

$2 \le 15$

$2+4 \le 15$

$6 \le 15$ ✓

$6+6 \le 15$

$12 \le 15$ ✓

$12+9 \le 15$ ✗

consider only w_1, w_2, w_3 weights and their profits.

$w_1 + w_2 + w_3 = 12$

$\hat{c}(1) = ?$

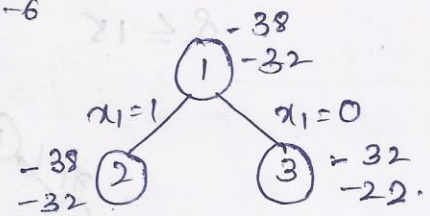
$u(1) = -\sum_{i=1}^{4} p_i = -(p_1 + p_2 + p_3 + p_4)$ (p_4: ✗)

$= -(10+10+12)$

$= -32$

$\hat{c}(1) = -32 - \frac{3}{9} \times 18 = -32 - 6$

$= -38$



For node (2)

$w_1 + w_2 + w_3 + w_4 \le m$

$2 \le 15$

$2+4 \le 15$

$2+4+6 \le 15$

$w_1 + w_2 + w_3 = 12$

$u(2) = -(10+10+12) = -32$

$\hat{c}(2) = -32 - \frac{3}{9} \times 18 = -38$

For node (3)

$w_1 + w_2 + w_3 + w_4 \le m$

w_1: 0

$4 \le 15$

$4+6 \le 15$

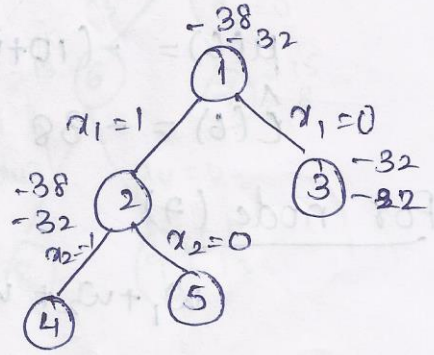
$10+9 \not\le 15$

$w_2 + w_3 = 10$

$u(3) = -(10+12) = -22$

$\hat{c}(3) = -22 - \frac{5}{9} \times 18$

$= -32$



For node (4)

$w_1 + w_2 + w_3 + w_4 \le m$

$2 \le 15$

$2+4 \le 15$

$2+4+6 \le 15$

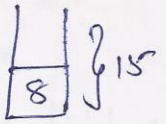
$w_1 + w_2 + w_3 = 12$

$\mu(4) = -(10+10+12) = -32$

$\hat{c}(4) = -32 - \frac{3}{9} \times 18 = -38$

For node (5)

$w_1 = 1 \quad w_2 = 0$



$2 \leq 15$

$w_1 \; w_3$

$2+6 \leq 15$

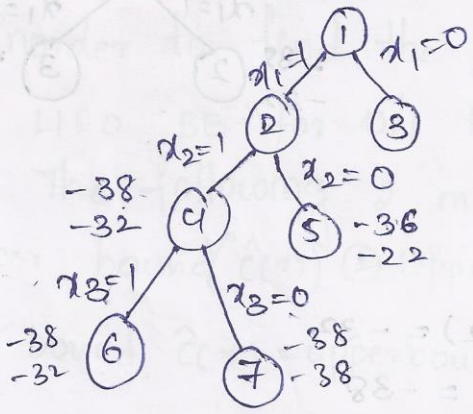
$8 \leq 15$

$w_1 + w_3 = 8$

$\mu(5) = -(10+12) = -22$

$\hat{c}(5) = -22 - \frac{7}{9} \times 18 = -22 - 14$

$= -36$



For node (6)

$w_1 + w_2 + w_3 = 12$

$\mu(6) = -(10+10+12) = -32$

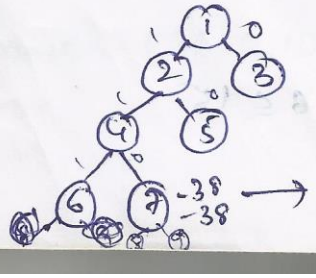
$\hat{c}(6) = -38$

For node (7)

$w_1 + w_2 + w_4 \leq 15$

$\mu(7) = -(10+10+18) = -38$

$\hat{c}(7) = -38 - \frac{0}{9} \times 18 = -38$



→ LB & U.B are equal. So kill that node.

For node (8)

$w_1 + w_2 + w_4 \le m$

$2 + 4 + 9 \le 15$

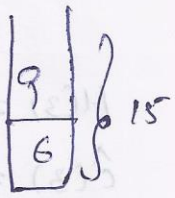
$\mu(8) = -38$

$\hat{c}(8) = -38 - \frac{0}{9} \times 18 = -38$

For node (9)

$w_1 + w_2 \le m$

$6 \le 15$



$\mu(9) = -20$

$\hat{c}(8) = -20 - \frac{0}{9} \times 18$

$= -20$

The solution is 1101

2. $n = 5$, $m = 15$, $w = 4, 4, 5, 8, 9$. $P = 4, 4, 5, 8, 9$

For node (1)

$w_1 + w_2 + w_3 + w_4 + w_5 \le m$

$4 \le 15$

$4 + 4 \le 15$

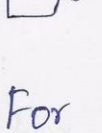
$4 + 4 + 5 \le 15 \Rightarrow 13 \le 15$ ✓

$13 + 8 \le 15$ ×

$w_1 + w_2 + w_3 \le 15$

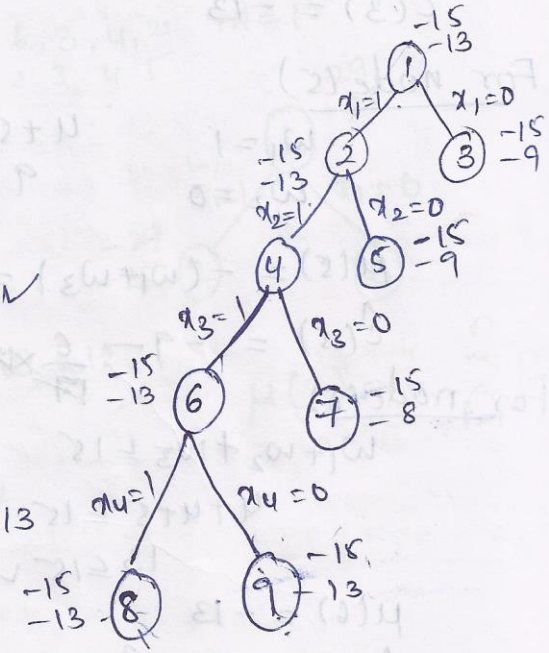
$13 \le 15$

$\mu(1) = -\sum_{i=1}^{n} P_i = -(4+4+5) = -13$



$\hat{c}(1) = -13 - \frac{2}{9} \times 9$

$\hat{c}(1) = -15$



For node (2)

$w_1 + w_2 + w_3 \le 15$

$4 + 4 + 5 \le 15$ $\quad 13 \le 15$ ✓

$\mu(2) = -\sum_{i=1}^{n} P_i = -13$

$$\hat{c}(2) = -13 - \frac{2}{9} \times 9 = -15$$

For node(3)

$w_1 = 0$

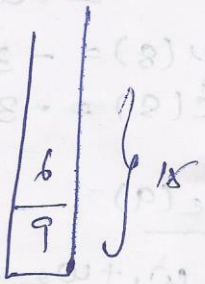
$4 \leq 15$

$4+5 \leq 15$

$9 \leq 15$

$9+8 \leq 15$

$17 \leq 15$ ×



$w_2 + w_3 \leq 15$

$9 \leq 15$

$\mu(3) = -(4+5) = -9$

$\hat{c}(3) = -9 - \frac{6}{9} \times 9 = -15$

For node(4)

$w_1 = 1 \quad w_2 = 1$

$4+4+5 \leq 15$

$13 \leq 15$

$\mu(3) = -15$

$\hat{c}(3) = -13$

For node(5)

$w_1 = 1$

$w_2 = 0$

$4+5 \leq 15$

$9 \leq 15$

$\mu(5) = -(w_1 + w_3) = -(9)$

$\hat{c}(5) = -9 - \frac{6}{9} \times 9 = -15$

For node(6)

$w_1 + w_2 + w_3 \leq 15$

$4+4+5 \leq 15$

$13 \leq 15$ ✓

$\mu(6) = -13$

$\hat{c}(6) = -13 - \frac{2}{18} \times 18 = -15$

For node(7)

$w_1 = 1 \quad w_2 = 1 \quad w_3 = 0$

$4+4 \leq 15$

$8 \leq 15$ ✓ $\quad 8+9 \leq 15$

$\mu(7) = -(4+4) = -8$

$$\hat{c}(4) = -8 - \frac{7}{9} \times 9 = -15$$

For node(8)

$$w_1 + w_2 + w_3 + w_4 \leq 15$$

$$w_1 + w_2 + w_3 \leq 15$$

$$\mu(8) = -13$$

$$\hat{c}(8) = -13 - \frac{2}{18} \times 18 = -15$$

For node(9)

$$w_4 = 0$$

$$w_1 + w_2 + w_3 + w_4 \leq 15$$

$$4 \leq 15$$

$$8 \leq 15$$

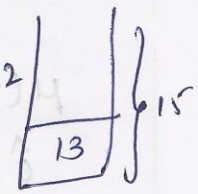
$$8+5 \leq 15$$

$$13 \leq 15$$

$$w_1 + w_2 + w_3 \leq 15$$

$$\mu(9) = -(4+4+5) = -13$$

$$\hat{c}(9) = -15$$



3. $n = 5$ $P = 10, 15, 6, 8, 4$ (1 2 3 4 5) $w = 4, 6, 3, 4, 2$ (1 2 3 4 5) $m = 12$

For node(1)

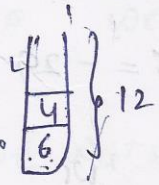
$$w_5 + w_1 + w_2 + w_3 + w_4 \leq 12$$

$$4 \leq 12$$

$$4+6 \leq 12$$

$$4+6+3 \leq 12 \times$$

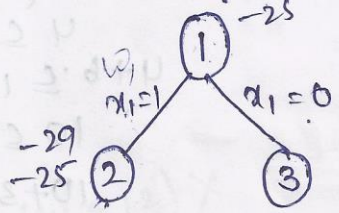
$$w_1 + w_2 = 12$$



$$\mu(x) = (10+15) = -25$$

$$\hat{c}(x) = -25 - \frac{2}{9} \times 18$$

$$= -25 - 4 = -29$$



$$\mu(x) = -\sum_{i=1}^{n} P_i$$

For node(2)

$$w_5 + w_1 + w_2 + w_3 + w_4 \leq 12$$

$$4 \leq 12$$

$$4+6 \leq 12$$

$$\mu(x) = -25 \qquad \hat{c}(x) = -29$$

For node (3)

$w_5 + w_1 + w_2 + w_3 + w_4 \leq 12$

$x_1 = 0 \Rightarrow w_1 = 0$ (we should not consider w_1 weight)

$w_2 + w_3 + w_4 \leq 12$

$6 \leq 12$

$6 + 3 \leq 12$

$9 \leq 12$

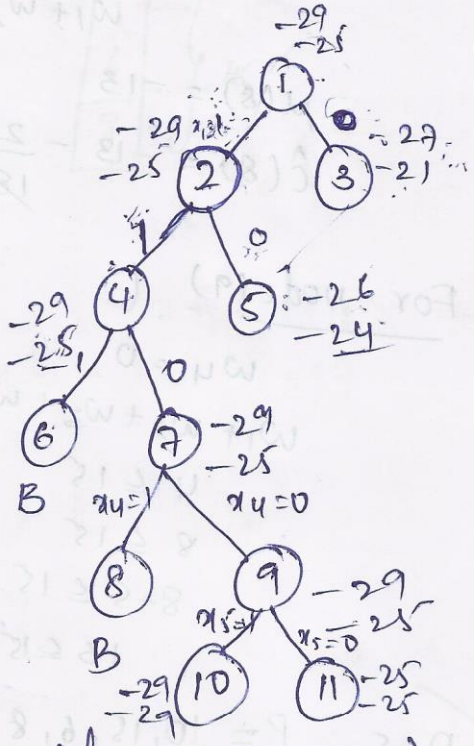
$9 + 4 \leq 12$ ×

$w_2 + w_3 = 12$

6 3

$\mu(x) = -(15+6) = -21$

$\hat{c}(x) = -21 - \frac{3}{6} \times 12 = -27$



For node (4)

$w_5 + w_1 + w_2 + w_3 + w_4 \leq 12$

$w_1 = 1$ $w_2 = 1$ (we have to consider w_1 & w_2 wts)

$4 \leq 12$

$4 + 6 \leq 12$ $\quad w_1 + w_2 \leq 12$

$10 \leq 12$

$10 + 3 \leq 12$ ×

$\mu(x) = -(10+15) = -25$

$\hat{c}(x) = -25 - \frac{2}{9} \times 18 = -29$

For node (5)

$w_1 + w_2 + w_3 + w_4 \leq 12$ (w_5)

$4 \leq 12$ $\quad 11 + 2 \leq 12$ ×

$4 + 3 \leq 12$ $\quad w_1 + w_3 + w_4 \leq 15$

$7 \leq 12$

$7 + 4 \leq 12$ $\quad \mu(x) = -(10+6+8) = -24$

$11 \leq 12$ ✓ $\quad \hat{c}(x) = -24 - \frac{1}{2} \times 4$

$= -26$

For node(6)

$w_1 + w_2 + w_3 + w_4 + w_5 \leq 12$

$4 \leq 12$

$4 + 6 \leq 12$

$10 + 3 \leq 12$

$13 \leq 12$ ✗

Kill the node. Because weight > m.

For node(7)

$w_1 + w_2 + w_3 + w_4 + w_5 \leq 12$

$4 + 6 \leq 12$

$10 \leq 12$

$u(x) = -(25) = -25$

$\hat{c}(x) = -25 - \frac{2}{6} \times 12$

$= -25 - 4 = -29$

For node(8)

$w_1 + w_2 + w_3 + w_4 + w_5 \leq 12$

$4 + 6 \leq 12$

$10 \leq 12$

$10 + 4 \leq 12$ ✗ $(14 \leq 12)$ ✗

Kill the node 8 weight > m.

For node(9)

$w_1 + w_2 + w_3 + w_4 + w_5 \leq 12$

$4 + 6 \leq 12$ $\quad 12 \leq 12$

$10 \leq 12$

$u(x) = -25 - 4 = -29$

$\hat{c}(x) = -25 - \frac{2}{2} \times 4$

$= -25 - 4 = -29$

For node (10)

$$w_1 + w_2 + w_5 \leq 12$$

$$4 + 6 + 2 \leq 12$$

$$12 \leq 12$$

$$\hat{u}(x) = -29$$

$$\hat{c}(x) = -29$$

For node (11)

$$w_1 + w_2 \leq m$$

$$4 + 6 \leq 12$$

$$\hat{u}(x) = -25$$

$$\hat{c}(x) = -25 - \frac{9}{0}$$

$$\hat{c}(x) = -25$$

The solution is 11001

Path generated is $1 \to 2 \to 4 \to 7 \to 9 \to 10$

upperbond $= -29$

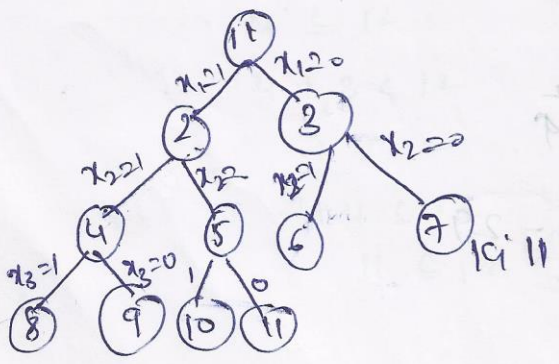
FIFOBB — 0/1 Knapsack Problem.

(1) $\hat{c}(x) \geq U(x)$ kill the node

BFS

$n = 4,\ m = 15,\ P = 10, 10, 12, 18$

$w = 2, 4, 6, 9$



For node (1)

$w_1+w_2+w_3+w_4 \le m$

$2 \le 15$

$2+4 \le 15$

$6 \le 15$

$6+6 \le 15$

$12 \le 15$

$12+9 \le 15$ ×

$w_1+w_2+w_3 \le m$

$12 \le 15$

$\mu(1) = -(10+10+12) = -32$

$\hat{c}(1) = -32 - \frac{3}{9} \times 18^{2} = -38$

For node (2)

$w_1+w_2+w_3+w_4 \le m$

$w_2 = 0$

$2 \le 15$

$2+4 \le 15$

$6 < 15$

$6+6 \le 15$

$12 \le 15$

$12+9 \le 15$ ×

$\mu(2) = -(10+10+12) = -32$

$\hat{c}(2) = -32 - \frac{3}{9} \times 18^{2} = -38$

For node (3)

$w_1+w_2+w_3+w_4 \le m$

$4+6 \le 15$

$10 \le 15$

$\mu(3) = -(10+12) = -22$

$\hat{c}(3) = -22 - \frac{5}{9} \times 18^{2} = -32$

For node (4)

$w_1+w_2+w_3+w_4 \le m$

$2 \le 15$

$2+4 \le 15$

$6 \le 15$

$6+6 \le 15$

$12 \le 15$

$12+9 \le 15$ ×

$\mu(4) = -32$

$\hat{c}(4) = -38.$

For node (5)

$w_1+w_2+w_3+w_4+w_5 \le m$

$\mu(5) = -22$

$\hat{c}(5) = -36$

For node (6)

$w_1+w_2+w_3+w_4 \le m$

$4 \le 15$

$4+6 \le 15$

$10 \le 15$

$\mu(6) = -22$

$\hat{c}(6) = -22 - \frac{5}{9} \times 18^{2} = -32$

For node (7)

$w_1+w_2+w_3+w_4 \le m$

$6 \le 15$

$6+9 \le 15$

$15 \le 15$

$\mu(7) = -30$

$\hat{c}(7) = -30 - \frac{0}{0}$

$\hat{c}(7) = -30$

$\hat{c}(x) = \mu(x)$: kill the node.

For node (8)

$2+4+6 \le 15$

$12 \le 15$

$\mu(8) = -32$

$\hat{c}(8) = -38$

For node(9)

$2 \leq 15$

$2+4 \leq 15$

$6 \leq 15$

$\mu(9) = -(10+10+18)$

$= -38$

$\hat{c}(9) = -38 - 0 = -38$

kill node

For node(10)

$2+6 \leq 15$

$8 \leq 15$

$8+9 \leq 15$ X

$\mu(10) = -(+32) = -32$

$\hat{c}(10) = -32 - \frac{7}{9} \times 18^{2}$

$= -32 - 14$

$= -36$

For node(11)

$2 \leq 15$

$2+9 \leq 15$

$11 \leq 15$

$\mu(11) = -(+28) = -28$

$\hat{c}(11) = -28 - 4$

$\hat{c}(11) = -28$

For node(12)

011

$4+6 \leq 15$

$10 \leq 15$

$\mu(12) = -22$

$\hat{c}(12) = -32$

For node(13)

010

$4+9 \leq 15$

$13 \leq 15$

$\mu(13) = -28$

$\hat{c}(13) = -28$

kill node

For node(14)

$2+4+6+9 \leq 15$ X

Bound.

For node 15

$12 \leq 15$

$\mu(15) = -32$

$\hat{c}(15) = -32 - \frac{3}{00}$

$\hat{c}(15) = -32$

kill node.

For node(16)

1011

$2+6 \leq 15$

$8 \leq 15$

$8+9 \leq 15$ X

$\mu(16) = -22$

$\hat{c}(16) = -22 - 0 = -22$

Bound ~~kill node~~

For node(17)

1010

$2+6 \leq 15$

$8 \leq 15$

$\mu(17) = -22$

$\hat{c}(17) = -22$

kill node

For node (18)

0111

$4+6+9 \leq 15$

$19 \leq 15$

Bound.

For node (19)

0110

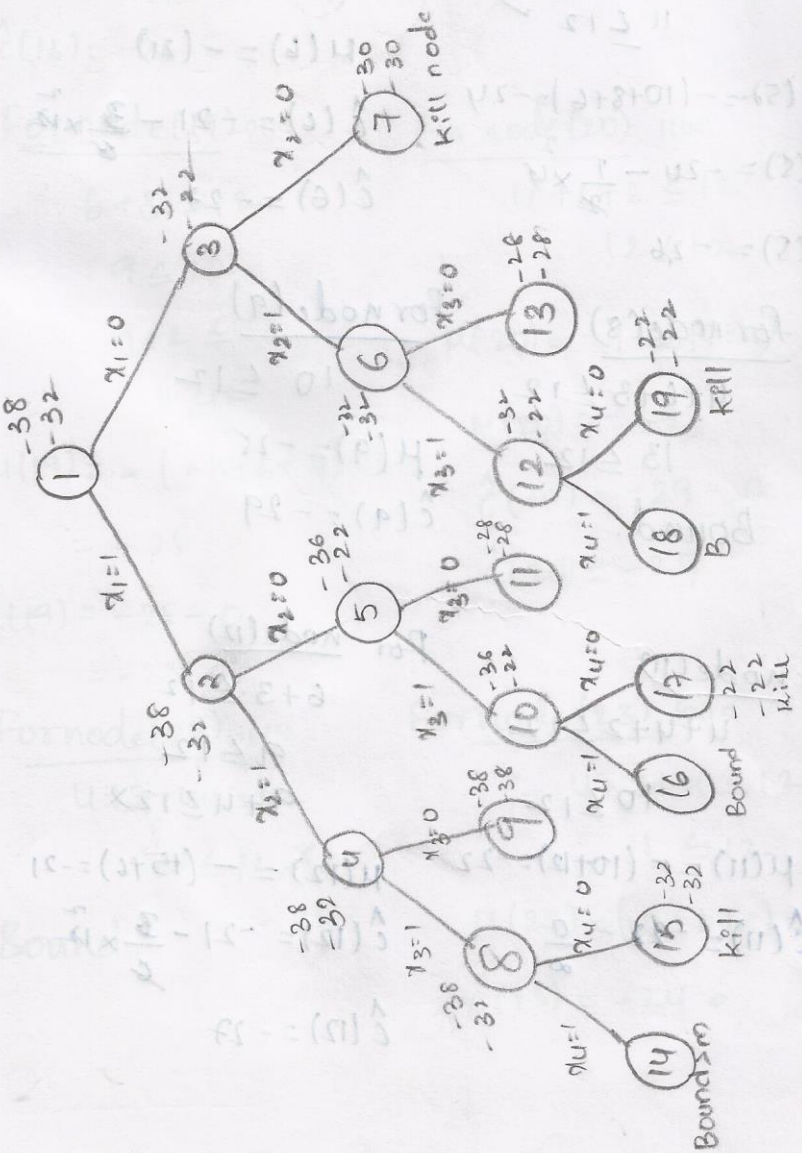
$4+6 \leq 15$

$10 \leq 15$

$\mu(19) = -22$

$\hat{c}(19) = -22$

Kill node.



2. $n=5$ $P = 10, 15, 6, 8, 4$ (1 2 3 4 5) $w = 4, 6, 3, 4, 2$ (1 2 3 4 5) $m = 12$

For node (1)

$4 \le 12$

$4+6 \le 12$

$4+6+3 \le 12$ ×

$\mu(1) = -25$

$\hat{c}(1) = -25 - \frac{2}{9} \times 18^2$

$\hat{c}(1) = -29$

For node (2)

$4 \le 12$

$4+6 \le 12$

$10 \le 12$

$\mu(2) = -25$

$\hat{c}(2) = -25 - \frac{2}{9} \times 18^2$

$\hat{c}(2) = -29$

For node (3)

$6 \le 12$

$9 \le 12$ ✓

$13 \le 12$ ×

$\mu(3) = -(15+6) = -21$

$\hat{c}(3) = -21 - \frac{3}{9} \times 18^2$

$\hat{c}(3) = -27$.

For node (4)

$4+6 \le 12$

$10 \le 12$

$\mu(4) = -(10+15) = -25$

$\hat{c}(4) = -25 - \frac{2}{9} \times 18^2$

$\hat{c}(4) = -29$

For node (5)

$4+3+4 \le 12$

$11 \le 12$ ✓

$\mu(5) = -(10+8+6) = -24$

$\hat{c}(5) = -24 - \frac{1}{2} \times 4^2$

$\hat{c}(5) = -26$

For node (6)

$6+3 \le 12$

$9 \le 12$

$\mu(6) = -(21)$

$\hat{c}(6) = -21 - \frac{3}{6} \times 12^2$

$\hat{c}(6) = -27$

For node (7)

$3+4+2 \le 12$

$9 \le 12$

$\mu(7) = -(6+8+4) = -18$

$\hat{c}(7) = -18$.

Kill the node.

For node (8)

$4+6+3 \le 12$

$13 \le 12$

Bound.

For node (9)

$10 \le 12$

$\mu(9) = -25$

$\hat{c}(9) = -29$

For node (10)

$4+3+4 \le 12$

$11 \le 12$

$\mu(10) = -(10+6+8) = -24$

$\hat{c}(10) = -24 - \frac{1}{2} \times 4^2$

$\hat{c}(10) = -26$

For node (11)

$4+4+2 \le 12$

$10 \le 12$

$\mu(11) = -(10+12) = -22$

$\hat{c}(11) = -22 - \frac{0}{\infty}$

For node (12)

$6+3 \le 12$

$9 \le 12$

$9+4 \le 12$ ×

$\mu(12) = -(15+6) = -21$

$\hat{c}(12) = -21 - \frac{3}{6} \times 12^2$

$\hat{c}(12) = -27$

For node(13) 010

$6+4+2 \leq 12$

$12 \leq 12$

$\mu(13) = -(15+8+4) = -27$

$\hat{c}(13) = -27 - 0$

$\hat{c}(13) = -27$

For node(14) 1101

$4+6+4 \leq 12$ ✗

Bound

For node(15) 1000

$4+6 \leq 12$

$19 \leq 12$

$\mu(15) = -25$

$\hat{c}(15) = -25 - 0 = -25$

For node(16) 1011

$4+3+4 \leq 12$

$11 \leq 12$

$\mu(16) = -(10+6+8) = -24$

$\hat{c}(16) = -24 - \frac{1}{2} \times 4^2$

$= -26$

$\hat{c}(16) = -26$

For node(17) 1010

$4+3 \leq 12$

$7^{2-7} \leq 12 \Rightarrow 9 \leq 12$

$\mu(17) = -(10+6) = -20$

$\hat{c}(17) = -20 - 0$

$\hat{c}(17) = -20$

For node(18) 0111

$6+3+4 \leq 12$

$13 \leq 12$ ✗

$\mu(18) = -($

$\hat{c}(18) =$

For node(19) 0110

$6+3 \leq 12$

$9 \leq 12$

$9+2 \leq 12$

$11 \leq 12$

$\mu(19) = -(+15+6+4)$

$= -25$

$\hat{c}(19) = -25 - 0$

$= -25$

For node(20) 11001

$4+6+2 \leq 12$

$12 \leq 12$

$\mu(20) = -(10+15+4)$

$\mu(20) = -29$

$\hat{c}(20) = -29 - 0$

$\hat{c}(20) = -29$

For node (21) 11000

$4+6 \leq 12$

$10 \leq 12$

$\mu(21) = -25$

$\hat{c}(21) = -25 - 2$

$\hat{c}(21) = -25$

For node

For node(22) 10111

$4+3+4+2 \leq 12$

$13 \leq 12$ ✗

Bound.

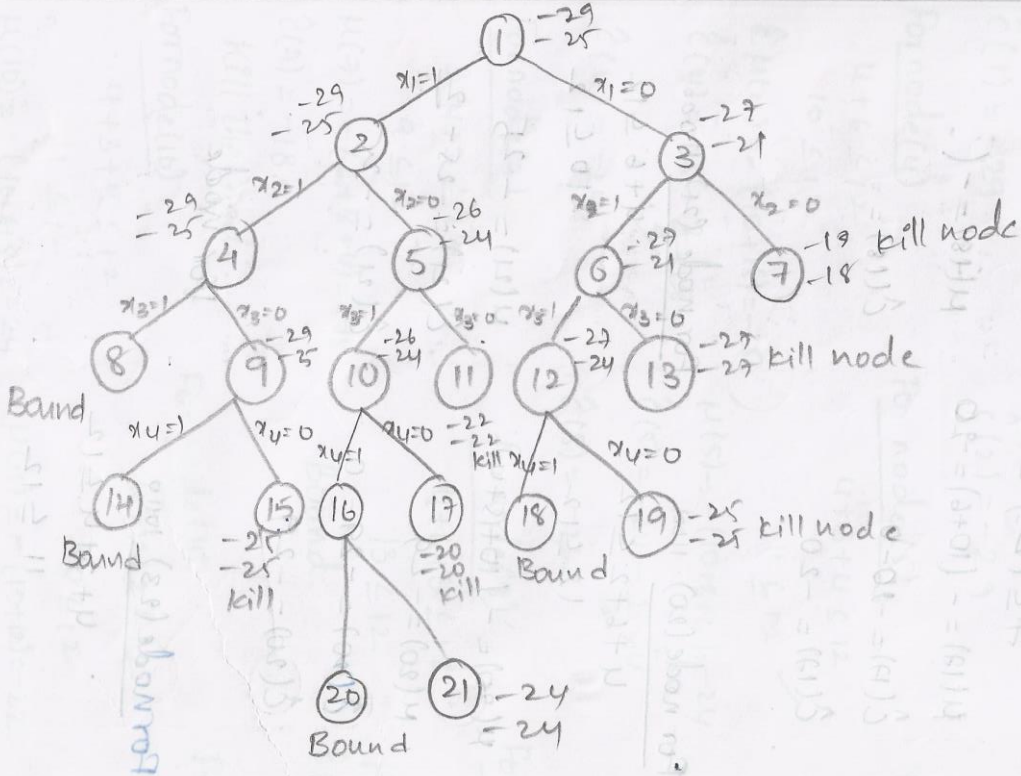
For node(23) 10110

$4+3+4 \leq 12$

$11 \leq 12$

$\mu(23) = -(10+6+8) = -24$

$\hat{c}(23) = -24$ o


1
-29
-25
$x_1=1$
$x_1=0$
2
-29
-25
3
-27
-21
$x_2=1$
$x_2=0$
$x_2=1$
$x_2=0$
4
-29
-25
5
-26
-24
6
-27
-21
7
-19
-18
kill node
$x_3=1$
$x_3=0$
$x_3=1$
$x_3=0$
$x_3=1$
$x_3=0$
8
Bound
9
-29
-25
10
-26
-24
11
12
-27
-24
13
-27
-27
kill node
$x_4=1$
$x_4=0$
$x_4=1$
$x_4=0$
$x_4=1$
$x_4=0$
14
Bound
15
-25
-25
kill
16
17
-20
-20
kill
-22
-22
kill
18
Bound
19
-25
-25
kill node
20
Bound
21
-24
-24

3. $n=5 \quad m=15 \quad P = W = 4, 4, 5, 8, 9.$

For node (1)

$4 \le 15$

$4+4+5 \le 15$

$13 \le 15$ ✓

$\mu(1) = -(4+4+5) = -13$

$\hat{c}(1) = -13 - \frac{2}{17} \times 17 = -15$

For node (2)

$4+4+5 \le 15$

$\mu(2) = -13$

$\hat{c}(2) = -15$

For node (3)

$4+5 \le 15$

$9 \le 15$

$8+9 \le 15$ X

$\mu(3) = -(4+5) = -9$

$\hat{c}(3) = -9 - \frac{6}{17} \times 17$

$\hat{c}(3) = -15$

For node (4)

$4+4 \le 15$

$8 \le 15$

$8+5 \le 15$

$13 \le 15$

$\mu(4) = -(4+4+5) = -13$

$\hat{c}(4) = -13 - \frac{2}{17} \times 17$

$\hat{c}(4) = -15$

For node (5)

$4+5 \le 15$

$9 \le 15$

$9+8 \le 15$ X

$\mu(5) = -(4+5) = -9$

$\hat{c}(5) = -9 - \frac{6}{17} \times 17$

$\hat{c}(5) = -15$

For node (6)

$4+5 \le 15$

$9 \le 15$

$9+8 \le 15$ X

$\mu(6) = -(4+5) = 9$

$\hat{c}(6) = -9 - \frac{6}{17} \times 17$

$\hat{c}(6) = -15$

For node (7)

$5+8 \le 15$

$13 \le 15$

$\mu(7) = -(5+8) = -13$

$\hat{c}(7) = -13 - \frac{2}{9} \times 9$

$= -15$

For node (8) 111

$4+4+5 \le 15$

$13 \le 15$

$\mu(8) = -(4+4+5) = -13$

$\hat{c}(8) = -13 - \frac{2}{17} \times 17$

$\hat{c}(8) = -15$

For node (9) (110)

$4+4 \le 15$

$8 \le 15$

$8+8 \le 15$ X

$\mu(9) = -(8)$

$\hat{c}(9) = -8 - \frac{7}{17} \times 17$

$\hat{c}(9) = -15$

For node (10) (101)

$4+5 \le 15$

$9 \le 15$

$9+8 \le 15$ X

$\mu(10) = -(9)$

$\hat{c}(10) = -9 - \frac{6}{12} \times 12 = -15$

For node (11)

$4+8 \le 15$

$12 \le 15$

$\mu(11) = -12$

$\hat{c}(11) = -12 - \frac{3}{9} \times 9$

$= -15$

For node (12) 011

$4+5 \le 15$

$9 \le 15$

$\mu(12) = -9$

$\hat{c}(12) = -9 - \frac{6}{17} \times 17$

$\hat{c}(12) = -15$

```
Algorithm UBound(cp, cw, k, m)
{
    b := cp, c := cw;
    for i := k+1 to n do
    {
        if (c + w[i] ≤ m) then
        {
            c := c + w[i];
            b := b - p[i];
        }
    }
    return b;
}
```

UNIT - 5

NP-HARD, NP-COMPLETE

There are 2 types of groups to solve the problems. i) Problems that can be solved in polynomial time. (NP- Complete)

ii) Problems that cannot be solved in polynomial time (NP-Hard)

NP- Complete: A problem that is NP-complete has the property that it can be solved in polynomial time if & only if all other NP- complete problems can also be solved in polynomial time.

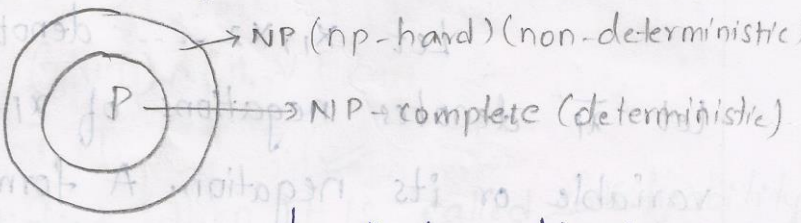
Ex: quick sort, binary search etc.

If an NP-Hard problems can be solved in polynomial time then all NP-complete problems can be solved in polynomial time.

All NP- complete problems are NP - Hard.

NP-complete $\subseteq$ NP-Hard.

But some NP-Hard problems are not NP-complete



An algorithm with the property that result of every operation is uniquely defined is called Deterministic

Algorithms

→ An algorithms whose result of every operation of is not uniquely defined is called non-deterministic Algorithm

→ In order to specify non-deterministic problems, then we consider 3 functions

choice(s) : chooses one of element in given set s

failure() : returns unsuccessful completion

success() : returns successful completion

Ex:
```
j := choice(1, n);
if A[j] = x then
{
    write(j);
    success();
}
write(0);
Failure();
```

CNF ⇒ $\bigwedge_{i=1}^{k} = c_i$ → $V_{L_{ij}}$ → literals

DNF ⇒ $\bigvee_{i=1}^{k} = c_i$ → $\bigwedge L_{ij}$ → literals

Satisfisability :

Let $x_1, x_2 \ldots$ denote boolean variables let $\bar{x_i}$ denotes negations of x_i. A literal is either a variable or its negation. A formula in the proposational calculus is an expression that can be constructed using literals and operations ~~and, or~~ "$\wedge$" or "$\vee$"

The symbol 'V' denotes OR ; '∧' denotes 'AND'. A formula in conjunctive normal form if & only if it is represented as $\bigwedge_{i=1}^{k} = C_i$ where C_i are the clauses each represented as $\bigvee_{\text{Literals}} L_{ij}$ where L_{ij} are literals.

A formula is in disjunctive normal form if and only if it is represented as $\bigvee_{i=1}^{k} = C_i$ where C_i are clauses each represented as $\bigwedge_{L_{ij}}$ where L_{ij} are literals.

The satisfisability problem is to determine whether the formula is true for some assignment of truth values to variables.

Ex: $(x_1 \wedge x_2) \vee (x_3 \wedge \bar{x}_4)$ [DNF]

$(x_3 \vee \bar{x}_4) \wedge (x_1 \vee \bar{x}_2)$

$(x_1 \wedge x_2) \vee (x_3 \wedge \bar{x}_4) \rightarrow$ DNF

$(x_1 \vee \bar{x}_2) \wedge (x_3 \vee \bar{x}_4) \rightarrow$ CNF

$x_1 = T$

$x_2 = T$ For DNF,

$x_3 = T$ $(T \wedge T) \vee (T \wedge T)$

$x_4 = F$ $T \vee T = T$ ✓

For CNF,

$(T \vee T) \wedge (T \vee F)$

$T \wedge T = T$ ✓

If CNF satisfies, then it is called satisfisability problem for CNF formulas & vice-versa

Algorithm :- (Non-deterministic satisfiability)

```
Algorithm Eval(E,n)
{
    for i := 1 to n do
        x_i = choice(false, true);
    if E(x_1, x_2 ---- x_n) then success();
    else Failure();
}
```

* Classes Of NP-HARD & NP-COMPLETE :-

(i) → P is a set of all decision problems soluble by the deterministic algorithms in polynomial time.

→ NP is a set of all decision problems soluble by the non-deterministic algorithms in polynomial time

$$P \subseteq NP \;;\; P = NP \text{ (or) } P \neq NP$$

Sorting, searching, all pairs shortest path

TSP, graph colouring.

ii) Let L_1, L_2 be problems. If problem L_1 reduces to L_2 i.e., $L_1 \propto L_2$ if and only if there is a way to solve L_1 by a deterministic polynomial time algorithm using a deterministic algorithm that solves L_2 in polynomial time. i.e., if we have a polynomial time algorithm for

l_2 then we can solve l_1 in polynomial time.

$$l_1 \xrightarrow{\alpha} l_2 \qquad l_1 \alpha l_2$$

$$l_2 \xrightarrow{\alpha} l_3 \qquad l_2 \alpha l_3$$

$$l_1 \alpha l_3$$

$\therefore l_1 \xrightarrow{\alpha} l_3$ (According to transitive property)

iii) A problem L is NP-Hard if and only if satisfiability reduces to L.

A problem L is NP-complete if and only if L is NP-Hard & L ∈ NP. ↪ NP-Hard & L ∈ NP

iv) Two problems L_1 and L_2 are said to be polynomially equilant if and only if l_1 reduces to l_2 and l_2 reduces to l_1 i,e, a problem l_2 is NP-Hard since l_1 is some problem already known to be NP-Hard. Since it is using transitive relation it follows that satisfiability $\underset{\text{reduces}}{\alpha}$ l_1

$$l_1 \alpha l_2$$

then satisfiability α l_2.

COOK'S THEOREM:-

0. * It "states that satisfisability is in P if and only if P = NP". According to definition of satisfiability we already seen that satisfiability is in NP. Hence, P = NP. i.e., satisfiability is also in P. In order to prove this following steps are considered.

1. * To show how to obtain from any polynomial time non deterministic decision algorithms "A" and input "I" a formula Q(A, I) such that Q is satisfiable if and only if A has

a successful termination with input I.

*. If length of I is 'n', and time complexity of A is p(n) for some polynomial time, then length of queue is given as $O(p^3(n)\log n) = O(p^4(n))$. The time needed to construct Q is $O(p^3(n)\log n)$.

2. *. A deterministic algorithm to determine outcome of A on any input I can be easily obtained.

→ Algorithm Z simply computes Q and then uses a deterministic algorithm for satisfiability problem to determine whether queue is satisfiable. If $O(q(m))$ is a time needed to determine whether a formula of length m is satisfiable then complexity of Z is $O(p^3(n)\log n) + q(p^3(n)\log n)$

→ If satisfiability is in P then $q(m)$ is a polynomial function of M and complexity of Z becomes $O(r(n))$ for some polynomial R

→ Hence, if satisfiability is in P, then for every non-deterministic algorithm A in NP can obtain a deterministic Z in P so the above construction shows that if satisfisability is in P, then P=N.

Differences :-

1. Greedy Method & Dynamic Binding.

Dynamic Progr.	Greedy Method.
1. It is a method in which the solution to a problem can be viewed as result of sequence of decisions.	1. It is a most forward technique for constructing solutions to an optimal problem through sequence of steps.
2. It More than one decision is made at a time	2. Only one decision made at a time.
3. It consideres all possible sequences in order to obtain optimal solution.	3. It considers an optimal solutions without revising previous solution
4. Due to principles of optimality the solution to problem is guaranteed.	4. Solution to a problem is not guaranteed.
5. It is an bottom up approach b/c the problem cannot be solved until we find solutions of sub problems	5. It is a top down approach, we solve sub problems by first finding the choice for a problem then reduce it into an smaller one.
6. DP is more expensive b/c we have to try each and every possibility in order to hold principles of optimality.	6. It is less expensive than DP.

2. Brute Force Approach & Back Tracking

Back Tracking	Brute Force Approach.
1. It is more efficient than Brute force Approach b/c of less no. of 'n' trials.	1. It is less efficient b/c for 'n' input all the trials has to be done.
2. Back Tracking solves large no. of problems based up of the constraints.	2. It is a straight forward approach for solving problems based upon the Contstraints.
3. In backtracking, the nodes which do not generate answer node are killed, to reduce the time.	3. All solutions for nodes are generated either it is a answer node or not.
4. Examples → n queens → Sum of subset → 0/1 knapsack using Brauch & Bound.	4. Examples → Selection sort → Sequential search
5. This method is not as simple as Brute Force.	5. This method is simple but time taking.

n-queens
sum of subsets
0/1

** HALTING PROBLEM:-

******

It is an example of NP-Hard decision problem that is not NP-complete. The main aim is to determine whether an deterministic algorithm A and input I terminates finite no. of steps & enters an infinite loop. This problem is undecidable i,e., there is no algorithm to solve this problem. Based upon above problem is not NP.

To show satisfiability a halting problem simply constructs a algorithm a whose input is proportional to the formula. If a has 'n' variables then 'A' tries 2^n possible truth values & verifies 'x' is satisfiable. If 'A' stops then 'x' is not satisfiable & 'A' enters into an infinite loop(i.e) If 'A' satisfies - 'x' is satisfied & there is a polynomial time algorithm for an algorithm 'A', having 'x' as an input. The halting problem is completely NP Hard problem but not NP Complete.

Questions:-

→ Write an algorithm for max clue?

→ Write non-deterministic algorithm for 0/1 knapsack problem & sum of subsets?

16.Question Bank

UNIT I

Short Questions:

1. If $f(n)=a_m n^m+\ldots..a_1 n+a_o$.then prove that $f(n)=O(n^m)$
2. Establish the relationship between Big-oh and Omega
3. What do you mean by Algorithm?
4. Define Big-oh notation?
5. What do you mean by Divide and Conquer Strategy?
6. What are the properties of Big-oh notations?
7. What is called substitution Method?
8. Differentiate Time complexity from Space Complexity?
9. What is Recurrence Relation?
10. What do you mean by Amortized Analysis?
11. How is the efficiency of the algorithm defined?
12. How is an algorithm's time efficiency measured?
13. Define direct recursive and indirect recursive algorithms?
14. What are the characteristics of an algorithm?
15. Analyze the time complexity of the following segment:

```
for(i=0;i<N;i++)
for(j=N/2;j>0;j--)
sum++;
```

Long Questions:

1. Prove that every polynomial $p(n)=a_k n^k+a_{k-1}n^{k-1}+\_\ \_\ \_\ \_\ \_ =a0$ with $a_k>0$ belongs to $\theta(n^k)$

2. Prove that exponential functions a^n have different order of growth for different values of base $a>0$

3. Set up and solve a recurrence relation for the number of calls made by F(n), the recursive algorithm for computing n!

4. Assume that T satisfies $T(n)\leq a^T([n/b])+f(n)$, $f(n)=O(n^k)$, and $a<b^k$. Choose a suitable constant C and then use induction to prove that $T(n)\ )\leq cn^k\log n+T(1)+T(2)+\ \_\ \_\ \_\ \_T([b])$ for all $n\geq 1$. Prove that $T(n)=O(n^k\log n)$.

5. With an example problem derive amortized complexity of an algorithm?

6 Show the complexity of Fibonacci function, F(n) is $\Omega((3/2)^n)$

7. Give a recursive algorithm to compu7te the produce of two positive integers m and n using only addition.

8. Develop an algorithm which converts a Roman numeral into an Arabic integer.

10. What is an algorithm? What are the properties of an algorithm? Explain with example?

11. What are the rules that are to be followed when writing an algorithm. Explain with an example?

12. What is performance analysis of an algorithm? Explain various notations used?

13. When a mortised analysis use used to be measure the performance of an algorithm? Explain in detail?

14. What happens to the worst case runtime of quick sort if the median of the given key is used as spitter key? Derive the relation?

15. Input is an array of numbers where each number is an integer in the range [O,N} (for some N>>n) present algorithm that runs in the worst case in time O(n(logn/logn)) and check whether these n numbers are distinct, and the algorithm should use only O9n) space? Design.

16. The sets A and B have m and n elements from a linear order. These sets are not necessarily sorted. Also assume mεn. Show how to compute A∪B and A∩B in O(nlogn) time.

17. Give a proof which shows that the recurrence relation $T(n)=mT(n/2)+an^2$ is satisfied by T(n)=O(nlogn).

18. Show how quick sort procedure sorts the following sets of keys (1,1,1,1,1,1,1) and (5,5,8,3,4,3,2).

19. Write an algorithm for quick sort and derive a recurrence relation to show the performance?

20. What is worst case complexity of merge sort? Derive the recurrence relation with help of an algorithm?

21. Give recursive and non recursive algorithms to binary search? Give their performances?

22. Give the control abstraction of divide and conquer strategy with an example?

23 What is the advantage of Strassens matrix multiplication? Explain?

24. Compare the quicksort and merge sort algorithms?

25. Compute 2101*1130 by applying the divide and conquer algorithm

UNIT II

Short Questions:

1. What is collapsing rule?
2. What are AND/OR graphs?
3. Explain about Game trees?
4. Write the non recursive algorithms for post order traversals?
5. What is connected and Biconnected components?
6. Explain about articulation point?
7. What is spanning tree?Explain with examples?
8. Write the algorithm for Breadth first search?
9. What is weighting rule for union?
10. Write the algorithm for union and find?

Long Questions:

1. Experimentally compare the performance of simple union and simple find with weighted union and collapsing find and generate a random sequence of union and find operations?

2. What that any connected = undirected graph G = (V_2,E) satisfy |E|≥|V|-|

3 Shoat that it a directed graph or undirected graph contain path between two vertices u and v, then it contains a simple path between u and V, show that if a directed graph contains a cycle, then it contains a simple cycle.

4. Suppose if a set of n elements contains distinct elements show that at most n-1 unions can be performed before the number of sets becomes 1.

5. With an example write the algorithm for simple union and measure the algorithms performance?

6. With an example write the algorithm for simple find and give the algorithms performance?

7. Bring out the differences between simple and weighted union? Explain with example?

8. Give the differences between simple find and collapsing Find? Give a suitable example?

10. What are connected components explain with an example.

11. What is meant by biconnected components, with example give how it is different from components.

12. What is degenerate tree? Explain with an example?

13. Show that is u unions are performed then atleast max{n-2u, 0} singleton sets remain?

14. Explain about AND/OR graphs with an example?

15. What are Game trees explain for an Tic-Tac-Toe problem?

UNIT III

Short Questions:

1. Explain the greedy method.
2. Define feasible and optimal solution.
3. Write the control abstraction for greedy method.
4. What are the constraints of knapsack problem?
5. What is a minimum cost spanning tree?
6. Specify the algorithms used for constructing Minimum cost spanning tree.
7. State single source shortest path algorithm (Dijkstra's algorithm).
8. Write any two characteristics of Greedy Algorithm?
9. What is the Greedy approach?
10. What are the steps required to develop a greedy algorithm?

11. Write the difference between the Greedy method and Dynamic programming.
12. Define dynamic programming.
13. What are the features of dynamic programming?
14. Define principle of optimality.
15. Define OBST
16. Write the general procedure of dynamic programming.
17. Define multistage graph
18. Define All pair shortest path problem
19. State time and space efficiency of OBST
20. What are the drawbacks of dynamic programming?

Long Questions:

1. Write a pseudocode of the greedy algorithm for change making problem with an amount η and coin denominations $d_1>d_2>d_3$_ _ _ >dm as its input. What is the efficiency class of your algorithm as a function of η.
2. Is the notation of a minimum cost spanning tree is applicable to a connected graph> Do we have to check the graph's connectivity before applying the algorithms (prism's/Kruskal's) or can these algorithms does it by themselves? Justify?
3. Design a linear-time algorithm for solving the single source shortest path problem for DAG represented by their adjancency linked list?
4. By considering the complete graph with n vertices, show that the number of spanning trees in an n vertex graph can be greater than 2^{n-1}-2.
5. How minimum cost spanning tee is constructed? Explain with the help of prism's algorithm?
6. Write Kuskal's algorithm for minimum spanning tree?
7. What is 0/1 knapsack problem? Explain with an example?
8. How job sequencing is implemented by using dead links? Explain
9. Give the general method of greedy algorithms? Explain with an example?
10. What are Feasible solution, optimal solutions and the condition of optimality explain with the help of 0/1 knapsack problem?
11. If the set of job are 4 and profits are (100,20,15,27) and dead links are (2,1,2,1) the find out optimal solutions?
12. What is single source shortest path problem,? Explain with an example?
13. Write an algorithm to multiply two matrices of sizes nxm, mxp and prove number of multiplications needed is nmp?

14. Write a dynamic programming algorithm to compute a^n based on the formulas
    1. an = $a^{n/2}a^{n/2}$, n even
       an = $a^{(n/-1)2}a^{n-1/2}$, n odd

15. Define $F = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$, show that $F = \begin{bmatrix} f_{n-1} & f_n \\ f_n & f_{n+1} \end{bmatrix}$, n≥1 where f_n is the nth Fibonacci number nad f_0 is defined as 0.

16. Show that the number of different binary trees with n nodes is $\frac{1}{n+1}\binom{2n}{n}$

17. What is the need of matrix chain multiplication? How it is implemented?

18. Give the control abstraction of dynamic programming with an example?

19. Can the same problem solved by using dynamic programming and greedy method? Explain with example?

20. Show that the computing time of OBST is $O(n^2)$?

21. What is Travelling sales person problem and what are its applications.

22. Write a pseudocode of the dynamic programming algorithm for solving Optimal Binary search tree and determine its time and space efficiencies.

23. Write algorithms corresponding to ADJUST, HEAPIFY, INSERT and DELETE for the case of a min-heap represented as a complete binary tree. Explain the time complexity of HEAPIFY

24. Write the implementation of DELETE (b,s) in which an element b found at vortex v of a binary Search tree whose elements belong to Set S.

UNIT-VI

Short Questions:

1. What are the requirements that are needed for performing Backtracking?
2. Define explicit constraint and Implicit Constraints?
3. Define state space tree.
4. Define answer states.
5. Define a live node.
6. Define a E – node.
7. Define a dead node.
8. What are the factors that influence the efficiency of the backtracking algorithm?
9. Define Branch-and-Bound method.
10. What are the searching techniques that are commonly used in Branch-and-Bound

11. State 8 – Queens problem.
12. State Sum of Subsets problem.
13. State m – colorability decision problem.
14. Define chromatic number of the graph
15. What are dynamic trees?

Long Questions:

1. Design a new control abstraction of Backtracking by combining the Recursive and non-recursive control abstraction of Back tracking?
2. Design n-queue algorithm and run it for n=8,9,10 and give all the possible solutions?
3. Determine the order of magnitude of the worst case complexity of backtracking procedure which finds all Hamiltonian cycles?
4. Prove the size of the set of all subsets of n elements is $2^{n.}$
5. Let W=(5,7,10,12,15,18) and M=35, find all possible subsets of which sum to M. Draw the portion of state space tree.
6. Give solution to 4Queue problem by design and algorithm?
7. What is meant by Graphs coloring? Explain map and its planar graph with help of an algorithm?
8. Is it possible to draw Hamiltonian cycle to every graph. Explain with example.
9. What is graph coloring? Present an algorithm which finds m-coloring of a graph.
10. Write the iterative Backtracking algorithm.
11. Write an algorithm of n-queens problem.
12. Explain the depth first search algorithm for an undirected graph.
13. Design a complete LC branch-and-bound algorithm for the job sequencing with dead lines problem. Use the fixed tuple size formulation?
14. Prove the goal of the given figure is reachable from the initial state iff $\sum_{i=1}^{16} \text{LESS}(i) + X$ is even.

    Figure:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

15. If T is a tour in H then T is a tour in exactly one of the graphs (V,E_i), $1 \le i \le r$ prove?
16. Prove that if a better function is used in LC branch-and-bound algorithm, the number of nobles generated may increase.
17. Draw the portion of the state space tree generated by LCKnap for the instance.

n=5 (P_1_ _ _ _P_5) = (10, _ _ _ 45, (w_1 _ _ _ _ 45) = (4,6,3,122) and m=12

18. Obtain the reduced cost matrix for TSP instance

$$\begin{bmatrix} \infty & 7 & 3 & 12 & 8 \\ 3 & \infty & 6 & 14 & 9 \\ 5 & 8 & \infty & 6 & 18 \\ 9 & 3 & 5 & \infty & 11 \\ 18 & 14 & 9 & 8 & \infty \end{bmatrix}$$

19. Solve the Job sequencing with dead lines? Where profits =(6,3,4,8,5) line = (2,1,2,1,1), deadlines=(3,1,4,2,14) and n=5.

20. Write a program schema DFBB?

21. Present a program schema for a FIFO Branch & Bound search for a Least-Cost answer node.

22. Explain how state space trees are used for programming nim, tic tac toe, checkers games

23. Define the term Branch & Bound and explain with an example.

24. Explain live node, E-node and dead node with an example.

UNIT-V

Short Questions:

1. What are NP- hard and Np-complete problems?
2. What is a decision problem?
3. what is approximate solution?
4. what is promising and non-promising nodes?
5. Write formula for bounding function in Knapsack problem
6. Write about traveling salesperson problem
7. Differentiate decision problem and optimization problem
8. what is class P and NP?
9. Define NP-Hard and NP-Complete problems
10. Give the time complexity and space complexity of traveling salesperson problem

Long Questions:

1. Convert the Boolean formula B=$(x_1 \overline{\leftrightarrow} x_2).(\overline{x_3}+x_4x_5)\overline{.}(x_1x_2+ x_3x_4)$ into CNF?

2. Design Pseudocode description of the branch and bound algorithm for TSP.

3. Show that every language L in P is polynomial time reducible to the language m={5} that is the language that simply asks if the binary encoding of the input equal 5.

4. Show the knapsack problem is solvable in polynomial time if the i/p is given in a unary encoding. That is show that the knapsack is not strongly NP-hard.

5. $P(x)=x^n+x^{n+1}+\_\ \_\ \_\ \_\ \_ +x+1$ how many arithmetic operations does this algorithm do?

6. Show that bin packing problem is NP?

7. Prove that polynomial reduction is a transitive relation.

8. Give the P,Np, NP hard problems?

9. Explain the satisfiability problem and write the algorithm for the same.

10. Differentiate between NP-complete and NP-Hard.

11. What is meant by Halting problem explain with an example.

12. State and Explain cook's theorem.

UNIT-I

1. The running time of an algorithm is represented by the following recurrence relation: [GATE 2009]

   $T(n)= n \quad n\leq 3$

   $T(n/3)+cn \quad$ otherwise

   Which one of the following represents the time complexity of the algorithm?

   a) $\theta(n)$ b) $\theta(n\log n)$

   c) $\theta(n^2)$ d) $\theta(^{n2\log n})$

2. The minimum number of comparisons required to determine if an integer appears more than n/2 times in a sorted array of n integers is [GATE 2008]

   a) $\theta(n)$ b) $\theta(\log n)$

   c) $\theta(\log^* n)$ d) $\theta(1)$

3. Consider the following functions:

   $f(n) = 2^n$

   $g(n) = n!$

   $h(n) = h^{\log n}$

   Which of the following statements about the symptotic behaviour of f(n), g(n), and h(n) is true?

a) $f(n) = o(g(n)); g(n) = O(h(n))$ b) $f(n) = \Omega(g(n)); g(n) = O(h(n))$ [GATE 2008]

c) $g(n) = O(f(n)); h(n) = O(f(n))$ d) $h(n) = O(f(n)); g(n) = \Omega(f(n))$

4. Consider the following C code segment: [GATE 2007]

```
Int IsPrime (n)
{
int i, n;
for (I = 2; I < = sqrt (n) ; i++)
uf*b%u = = 0)
if (n%i = = 0)
{printf('notprime\n"); return 0;)
return 1;
}
```

Let T(n) denote the number of times the for loop is executed by the program on input n. Which of the following is TRUE?

a) $T(n) = O(\sqrt{n})$ and $T(n) = \Omega(\sqrt{n})$ b) $T(n) = O(\sqrt{n})$ and $T(n) = \Omega(1)$

c) $T(n) = O(n)$ and $T(n) = \Omega(\sqrt{n})$ d) None of these

5. In the following C function, let $n \geq m$ [GATE 2007]

```
Int gcd (n,m)
{
If (n%m = = 0) return m;
n = n%m
return gcd (m,n);
}
```

How may recursive calls are made by this function?

a) $\otimes(\log_2 n)$ b) $\otimes(n)$

c) $\otimes(\log_2 \log_2 n)$ d) $\log_2(\sqrt{n})$

6. Consider the following segment of C-code [GATE 2007]

```
int J,n;
j = 1;
while (j<=n)
j = j + 2;
```

The number of comparisons made in the execution of the loop for any n>0 is

a) $[\log_2 n]+1$ b) n

c) $[\log_2 n]$ d) $[\log_2 n]+1$

7. The median of n elements can be found in O(n)time. Which one of the following is correct about the complexity of quick sort, in which median is selected as pivot? [GATE 2006]

a) $\theta(n)$ b) $\theta(n\log n)$

c) $\theta(n^2)$ d) $\theta(n^3)$

8. Consider the following C-function in which a[nl and b[mlare two sorted integer arrays and c[n + m] be another integer array. [GATE 2006]

void xyz(int a[], mt b [1, int c []){

int i, j, k;

i=j=k=O;

while ((i<n) && (j<m))

if (a[i] < b[j]) c[k++] = a[i++];

else c[k++] = b[j++]'

Which of the following condition(s) hold(s) after the termination of the while loop?

(i) j<m,k=n+j—1, and a[n—i1<b[jl ifi=n

(ii) i<n,k=m+i—1, and b[m—i1a[i1 ifj=m

a) only (i)
b) only (ii)
c) either (i) or (ii) but not both
d) neither (i) nor (ii)

9. Consider the following recurrence: [GATE 2006]

T(n)=2T(|√n|)+1,T(1) = 1

Which one of the following is true?

a) T(n) = θ(log log n)
b) T(n) = θ(logn)
c) T(n)=θ(√n)
d) T(n)=θ(n)

10. A set X can be represented by an array x[nl as follows:
[GATE 2006]

X[i] = 1 if i€X

0 otherwise

Consider the following algorithm in which x,y and z are Boolean arrays of size n:

algorithm zzz(x[ ], y[ ], z [ ]) {

int i;

for (i=O; i<n; ++i)

z[i] = (x[i] Ay[i]) V (x[i] A y[i])

}

The set Z computed by the algorithm is:

a) (XUY)
b) (xflY)

c) (X-Y)fl(Y-X)

d) (x-Y)U(Y-x)

11. Consider the following C-program fragment in which i,j and nare integer variables. for (i = n,j = 0; i >0; i 1= 2,j +=i); Let val(j)denote the value stored in the variable jafter termination of the for loop. Which one of the following is true? [GATE 2006]

a) val(j) = 8(logn)

b) vaI(j)=8(fi)

c) val(j)=8(n)

d) val(j)= 8(nlogn)

12. Let T(n) be a function defined by the recurrence
[GATE 2005]

T(n) = 2T(n/2)+√n for n≥2 and

T(1) = 1

Which of the following statements is TRUE?

a) T(n)=8(logn)

c) T(n)=8(n)

b) T(fl)r=8(fi)

d) T(n) = 8(nlogn)

13. In the following table, the left column contains the names of standard graph algorithm, and the right column contains the time complexities of the algorithms. Match each algorithm with its time complexity.

(1) Bellman Ford algorithm	(A) O (mlog n)	[GATE 2005]
(2) Kruskal's algorithm	(B) O (n3)	
(3) Floyd-Warshall algorithm	(C) O (nm)	
(4) Topological Sorting	(D) O (n + m)	

a) 1-C2-A3-B4-D

b) 1-B2-D3-C4-A

c) 1-C2-D3-A4-B

d) 1-B2-A3-C4-D

14. Linked Answer Questions: Q.14a to Q.14b carry two marks each. Statement for Linked Answer Questions 15a & 15b:

Consider the following C-function: -

```
double foo (mt n) { I-

int I;

double sum;

if (n==O) return 1.0;

else {
```

```
sum = 0.0;
for (i =0; ±<n; i++)
sum +=foo(i);
return sum;
} }
```

a. a) The space complexity of the above function is:

a) 0(1) b) 0(n)

c) 0(n!) d) $0(n^n)$

b. b) Suppose we modify the above function foo() and store the values of foo (±), 0<=I<n, as and when they are computed. With this modification, the time complexity for function foo() is significantly reduced. The space complexity of the modified function would be:
[GATE 2005]

a) 0(1) b) 0(n)

c) $0(n^2)$ d) 0(n!)

15. Suppose T(n) =2T (n/2)+n, T(0) =T(i)=1 [GATE 2005]

Which one of the following is FALSE?

a) $T(n)=O(n^2)$ b) $T(n)=\theta(nlogn)$

c) $T(n)=\Omega(n2)$ d) T(n)=O(nlogn)

16. The time complexity of computing the transitive closure of set of n elements is known to be: [GATE 2005]

a) 0(n) b) 0(n log n)

c) $O(n^{3/2})$ d) $O(n^3)$

17. Consider a list of recursive algorithms and a list of recurrence relations as shown below. Each recurrence relation corresponds to exactly one algorithm and is used to derive the time [GATE 2004]

COMPLEXITY OF THE ALGOI	**RECURSIVE ALGORITHM RECURRENCE RELATION**
(P) Binary search	(I) T(n)=T(n—k)+T(k)+cn
(Q)	(II) T (n) = 2T (n — 1) + 1
	(III) T (n) = 2T(n /2) + cn

(R) Quick sort

(S) Tower of Hanoi

Which of the following is the correct match between the algorithms and their recurrence relations?

a) P-IIQ-III R-IVS-I

b) P-IVQ-IIIR-I S-Il

c) P-III Q-II R-IVS-I

d) P-IV Q-II R-I S-Ill

18. Let f(n),g(n) and h(n) be functions defined for positive integers such that f(n) = O(g(n),g(n)) O(f (n)),g(n) = O(h(n)), and h(n) = O(g(n)). Which oneof the following statements is FALSE? [GATE 2004]

a) f(n)+g(n)=O(h(n))+h(n)) b) f(n)=O(h(n))

c) h(n)≠ O(f(n)) d) f(n)h(n)≠ O(g(n)h(n))

19. Let A[i,...,n] be an array storing a bit (1 or 0) at each location, and f(m) is a function whose time complexity is 0(m). Consider the following program fragment [GATE 2004]

written in a C like language:

```
counter = 0;
for (i =1; i = n; i++)
{if (a[i] == 1) counter++;
else {f (counter); counter = 0;)
}
```

The complexity of this program fragment is

a) Ω(n2) b) Ω (nlogn) and 0(n2)

c) θ(n) d) 0(n)

20. The time complexity of the following C function is (assume n > 0) [GATE 2004]

```
int recursive (int n) {
if (n == 1)
return (1);
```

else

return (recursive (n-i)± recursive (n-i);

}

a) 0(n)
b) 0(nlog n)
c) 0(n2)
d) $0(2^n)$

21. The recurrence equation [GATE 2004]

T(1)=1i

T(n)= 2T(n—1)+n,n>2

evaluates to

a) 2^{n-1} — n — 2
b) 2^n — n
c) 2^{n+1} — 2n — 2
d) 2^n + n

22. What does the following algorithm approximate? (Assume m > 1, E > 0).[GATE 2004]

X = m;

Y = 1;

While (x — y > E)

{ x=(x+y)/2;

y = m/x;

}

print (x);

a) log m
b) m^2
c) $m^{1/2}$
d) $m^{1/3}$

23 .If all permutations are equally likely, what is the expected number of inversions in a randomly chosen permutation of 1….n? [GATE 2003]

a) n(n-1) /2
b) n(n-1)/4
c) n(n+1)/4
d) $2n(\log_2 n)$

24. Consider the following three claims [GATE 2003]

I. $(n+k)^m = \theta(n^m)$ where k and m are constants

II. $2^{n+1} = O(2^n)$

III> $2^{2n+1} = O(2^n)$

Which of these claims are correct?

a) I and II
b) I and III
c) II and III
d) I, II and III

25. The running time of the following algorithm [GATE 2002]

Procedure A(n)

If n<=2 return (1) else return (A(√n));

Is bet described by

a) O(n)
b) O(log-n)
c) O(log log n)
d) O(1)

26 The solution to the recurrence $T\{2^k\} = 3T\{2^{k+1}\} + 1,\ T(1) = 1$ is [GATE 2002]

a) 2^k
b) $(3^{k+1}-1)/2$
c) $3^{\log_2}$
d) $2^{\log_2}$

27. let $f(n) = n^2 \log n$ and $g(n) = n(\log n)^{10}$ be two positive functions of n. which of the following statements is correct? [GATE 2001]

a) $f(n) = O(g(n)$ and $g(n) \neq O(f(n))$
b) $g(n) = O(f(n)$ and $f(n) \neq O(g(n))$
c) $f(n) = O(g(n))$ and $g(n) \neq O(f(n))$
d) $f(n) = O(g(n))$ and $g(n) = O(f(n))$

28. Let S be a sorted array of n integers Let t(n) denote the time taken for the most efficient algorithm to determined if there are two elements with sum less than 10000 in s. Which of the following statements is true? [GATE 2000]

a) t(n) is 0(1)
b) $n \leq t(n) \leq n \log_2 n$
c) $n \log_2 n \leq t(n) < (n/2)$
d) $t(n) = (n/2)$

29. Consider the following functions [GATE 2000]

$F(n) = 3n^{\sqrt{n}}$

$G(n) = 3n^{\sqrt{n}\log_2 n}$

$h(n) = 2!$

Which of the following is true?

a) h(n) is O(f(n)) b) h(n) is O(g(n))

c) h(n) is O(f(n)) d) f(n) is O(g(n))

30 It T1 = O(1) give the correct matching for the following pairs: [GATE 1999]

(M) $T_n = T_{n-1} + n$ U) $T_n = O(n)$

(N) $T_n = T_n + n$ V) $T_n = O(n\log n)$

(O) $T_n = T_{n/2} + n \log n$ W) $T_n = O(n^2)$

(P) $T_n = T_{n-1} + \log n$ X) $T_n = O(\log^2 n)$

a) M-W N-V O-u P-X b) M-W N-U O-X P-V

c) M-V N-W O-X P-U d) M-W N-U O-V P-X

31. If n is a power of 2, then the minimum number of multiplications needed to compote a* is[GATE 1999]

a) $\log_2 n$ b) c $\sqrt{n}$

c) n – 1 d) n

32. Which of the following statements is false? [GATE 1998]

(a) A tree with a n nodes has (n-1) edges

(b) A labeled rooted binary tree can be uniquely constructed given its post order and preorder traversal results.

(c) A complete binary tree with n internal nodes has (n+1) leaves.

(d) The maximum number of nodes in a binary tree of height h is $(2^{n+1} -1)$

33. Consider the following function. [GATE 1997]

Function F(n,m:integer):integer;

Begin

If(n<=0) or (m<=0) then F:=1

Else

F:F(n-1, m) + F(n, m-1)

End;

Use the recurrence relation $\binom{n}{k}=\binom{n-1}{k}=\binom{n-1}{k-1}$ to answer the following questions. Assume that n,m are positive integers. Write only the answers without any explanation.

a) What is the value of F(n,2)?

b) What is the value of F(n,m)?

c) How many recursive calls are made to the function F, including the original call, when evaluating F(n,m)

34. Let T(n) be the function defined by T(1) = 1, T(n) = 2T(⌊n/2⌋)=√2 for n≥2. Which of the following statements is true? [GATE 1997]

a) T(n) = O√n

b) T(n) = O(n)

c) T(n) = O (log n)

d) None of the above

35. The correct matching for the following pairs is
[GATE 1997]

a) All pairs shortest paths 1) Greedy

b) Quick sort 2) Depth-First search

c) Minimum weight spanning tree 3) Dynamic programming

d) Connected components 3) Divide and conquer

a) A-2 B-4 C-1 D-3

b) A-3 B-4 C-1 D-2

c) A-3 B-4 C-2 D-1

d) A-4 B-1 C-2 D-3

36. The recurrence relation [GATE 1996]

T(1) = 2

$T(n) = 3T(n/4) + n$

Has the solution T(n) is equal to

a) O(n)
b) O (logn)
c) $O(n^{3/4})$
d) None of the above

37. Consider the following two functions: [GATE 1994]

$$g_1(n) = \begin{cases} n^3 \text{ for } 0 \le n \le 10,000 \\ n^3 \text{ for } n \ge 10,000 \end{cases}$$

$$g_1(n) = \begin{cases} n^3 \text{ for } 0 \le n \le 100 \\ n^3 \text{ for } n \ge 100 \end{cases}$$

Which of the following is true?

a) $g_1(n)$ is $O(g_2(n))$
b) $g_1(n)$ is $O(n^3)$
c) $g_2(n)$ is $O(g_1(n))$
d) $g_2(n)$ is O(n)

38. $\sum_{t \le k \le n} o(n)$ where o(n) stands for order n is : [GATE 1993]

a) o(n)
b) $o(n^2)$
c) $o(n^3)$
d) $o(3n^2)$
e) $o(1.5\ n^2)$

39. Give an optimal algorithm in pseudo-code for sorting a sequence of n numbers which has only k distinct numbers (k is not known as priori. Give a brief analysis for the time-complexity of your altorithm)

[GATE 1991]

UNIT I(part b)

1. In quick sort, for sorting n elements, the (n/4)th smallest element is selected as pivot using an O(n) time algorithm, what is the worst case time complexity of the quick sort [GATE 2009]

a) θ(n)
b) θ(nlogn)
c) $\theta(n^2)$
d) $\theta(^{n2logn})$

2. Consider the Quicksort algorithm, suppose there is a procedure for finding a pivot element which splits the list into two sub-lists each of which contains at least one-fifth of the elements. T(n) be the number of comparisons required to sort n elements. Then

[GATE 2008]

a) $T(n) \leq 2T(n/5)+n$ b) $T(n) \leq T(n/5)+T(4n/5)+n$

c) $T(n) \leq 2T(4n/5)+n$ d) $T(n) \leq 2T(n/2)+n$

3. Which of the following sorting algorithms has the lowest worst-case complexity? [GATE 2007]

a) Merge sort b) Bubble sort
c) Quick sort d) Selection sort

4. Let a and b be two sorted arrays containing n integers each, in non-decreasing order. Let c be a sorted array containing 2n integers obtained by merging the two arrays a and b. assuming the arrays are indexed starting from 0, consider the following four statements.
[GATE 2005]

I. $a[i] \geq b[i] \Rightarrow c[2i] \geq a[i]$

II. $a[i] \geq b[i] \Rightarrow c[2i] \geq b[i]$

III. $a[i] \geq b[i] \Rightarrow c[2i] \leq a[i]$

IV. $a[i \geq b[i] \Rightarrow c[2i] \leq b[i]$

Which of the following is TRUE?

a) only I and II b) only I and IV

c) only II and III d) only III and IV

5. Suppose the numbers 7,5,1,8,3,6,0,9,4,2 are inserted in that order into an initially empty binary search tree. The binary search tree uses the usual ordering on natural numbers. What is the in-order traversal sequence of the resultant tree [GATE 2003]

a) 7510324689 b) 0243165987

c) 0123456789 d) 9864230157

6. What would be the worst case time complexity of the Insertion Sort algorithm, if the inputs are restricted to permutations 1…n with at most n inversions?
[GATE 2003]

a) $\theta(n^2)$ b) $\theta(n \log n)$

c) $\theta(n^{1.5})$ d) $\theta(n)$

7. The usual $\theta(n^2)$ implementation of insertion sort to sort an array uses linear search to identify the postion where an element is to be inserted into the already sorted part of the array. If, instead, we use binary search to identify the position, the worst case running time will
[GATE 2003]

a) remain $\theta(n^2)$ b) become $\theta (n (\log n)^2)$

c) become $\theta(n \log n)$ d) become $\theta(n)$

8. Consider an array multiplier for multiplying two n bit numbers. If each gage in the circuit has a unit delay, the total delay of the multiplier is [GATE 2003]

a) $\theta(1)$ b) $\theta(\log n)$

c) $\theta(n)$ d) $\theta(n^2)$

9. Consider the following algorithm for searching for a given number x in an unsorted array A[1..n] having n distinct values: [GATE 2002]

1. Choose an I uniformly at random from 1..nl
2. If A[i] = x then stop else goto 1;

Assuming that x is present A, what is the expected number of comparisons made by the algorithm before it terminates?

a) n b) n-1

c) 2n d) n/2

10. In the worst case, the number of comparisons needed to search a singly linked list of length n for a given element is [GATE 2002]

a) log n b) n/2

c) log(n/2) – 1 d) n

11. Randomized quicksort is an extension of quicksort where the pivot is chosen randomly. What is the most worst case complexity of sorting n numbers using randomized quicksort? [GATE 2001]

a) O(n) b) O(n log n)

c) $O(n^2)$ d) O(n!)

12. Suppose we want to arrange the n numbers stored in any array such that all negative values occur before all positive ones. Minimum number of exchanges required in worst case is [GATE 1999]

a) n – 1
b) n
c) n + 1
d) none of these

13. A sorting technique is called stable if [GATE 1999]

a) It takes O (nlong n) time

b) It maintains the relative order of occurrence of non-distinct elements

c) It uses divide and conquer paradigm

d) it takes O(n) space

14. If one uses straight two-way merge sort algorithm to sort the following elements in ascending order:

20,47,15,8,9,4,40,30,12,17 [GATE 1999]

Then the order of these elements after second pass of the algorithm is :

a) 8,9,15,20,47,4,12,17,30,40
b) 8,15,20,47,4,9,30,40,12,17
c) 15,20,47,4,8,9,12,30,40,17
d) 4,8,9,15,20,47,12,17,30,40

15. Quick-sort is run on two inputs shown below to sort in ascending order [GATE 1996]

(i) 1,2,3……n

(ii) n, n-1, n-2, ……2,1

Let C_1 and C_2 be the number of comparisons made for the inputs (i) and (ii) respectively, then,

a) $C_1 < C_2$
b) $C_1 > C_2$
c) $C_1 = C_2$
d) We cannot say anything for arbitrary n

16. The average number of key comparisons done on a successful sequential search in list of length n is

a) log n
b) (n-1) / 2 [GATE 1996]
c) n / 2
d) (n+1) / 2

17. For merging two sorted lists of sizes m and n into a sorted list of size m+n, we required comparisons of

a) O(m)
b) O(n) [GATE 1995]

c) O(m+n) d) O(logm+logn)

18. Merge sort uses [GATE 1995]

a) Divide and conquer strategy b) Backtracking approach

c) Heuristic search d) Greedy approach

19. Which of the following algorithm design techniques is used in the quicksort algorithm?[GATE 1994]

a) Dynamic programming b) Backtracking

c) Divide and conquer d) Greedy method

20. The recursive relation that arises in relation with the complexity of binary search is [GATE 1994]

a) $T(n) = T\left(\frac{n}{2}\right) + k,\ k \text{ a constant}$ b) $T(n) = 2T\left(\frac{n}{2}\right) + k,\ k \text{ a constant}$

c) $T(n) = T\left(\frac{n}{2}\right) + \log n$ d) $a^2_2 = \frac{1}{2}\ T(n) = T\left(\frac{n}{2}\right) + n$

21. Following algorithms can be used to sort n integers in the range [1 . . . . n^3] in 0(n) time[GATE 1992]

a) Heap sort b) Quick sort

c) Mergesort d) Radixsort

UNIT-II

1. What is the chromatic number of an n-vertex simple connected graph which does not contain any odd length cyckle? Assume n≥2 [GATE 2009]

a) 2 b) 3

c) n-1 d) n

2. Which one of the following is TRUE for any simple connected undirected graph with more than 2 vertices?

a) no Two vertices have the same degree [GATE 2009]

b) Atleast two vertices have the same degree

c) At least three vertices have the same degree

d) All vertices have the same degree.

3. G is a graph on n vertices and 2n-2 edges. The edges of G can be partitioned into two edge-disjoint spanning trees. Which of the following is NOT true for G? [GATE 2008]

a) For every subset of k vertices, the induced subgraph has atmost 2k-2 edges.

b) The minimum cut in G has atleast two edges

c) There are two edge-disjoint paths between every pair of vertices

d) There are two vertex-disjoint paths between every pair of vertices.

4. The most efficient algorithm for finding the number of connected components in an undirected graph on n vertices and m edges has time complexity [GATE 2008]

a) $\theta(n)$ b) $\theta(m)$

c) $\theta(m+n)$ d) $\theta(mn)$

5. Let w be the minimum weight among all edge weights in an undirected connected graph. Let e be a specific edge of weight w. Which of the following is FALSE? [GATE 2007]

a) There is a minimum spanning tree containing e
b) if e is not in a minimum spanning tree T. then in the cycle by adding e to T, all edges have the same weight.
c) Every minimum spanning tree has an edge of weight w.
d) e is present in every minimum spanning tree.

Common Data Questions:

Common Data for Questions 71, 72, 73:

The 2 vertices of a graph G corresponds to all subsets of a set of size n, for $n < 6$. Two vertices of G are adjacent if and only if the corresponding sets intersect in exactly two elements.

6. The number of vertices of degree zero in G is:
[GATE 2006]

a) 1 b) n

c) n+1 d) 2^n

The maximum degree of a vertex in G is: [GATE 2006]

a) $\{n/2/2\}2^{n/2}$ b) 2^{n-2}

c) $2^{n-3}x3$ d) 2^{n-1}

7. The number of connected components in G is:
[GATE 2006]

a) n
b) n+2
c) $2^{n/2}$
d) $2^n/n$

8. Let G be a simple connected planar graph with 13 vertices and 19 edges. Then, the number of faces in the planar embedding of the graph is: [GATE 2005]

a) 6
b) 8
c) 9
d) 13

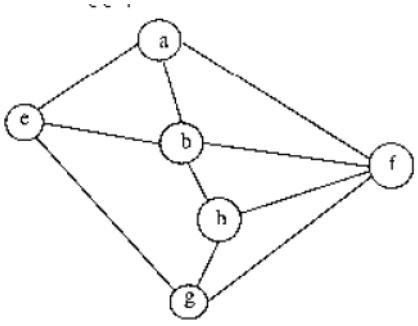
9. Let G be a simple graph with 20 vertices and 100 edges. The size of the minimum vertex cover of G is 8. then, the size of the maximum independent set of G is:
[GATE 2005]

a) 12
b) 8
c) Less than 8
d) More than 12

10. Let G_1 (V,E_1) and G_2 (V,E_2) be connected graphs on the same vertex set V with more than two vertices. If G_1 nG_2 (V,E_1 nE_2) is not a connected graph, then the graph G1 uG2 (V,E_1 uE_2)
[GATE 2004]

a) cannot have a cut vertex
b) must have a cycle
c) must have a cut-edge (bridge)
d) has chromatic number strictly greater than those of G_1 and G_2

11. Consider the following graph [GATE 2003]



Among the following sequences

I abeghf II abfehg III abfhge IV afghbe

Which are depth first traversals of the above graph?

a) I, II and IV only
b) I and IV only
c) II, III and IV only
d) I, III and IV only

12. Let G be an arbitrary graph with n nodes and k components. If a vertex is removed from G, the number of components in the resultant graph must necessarily lie between [GATE 2003]

a) k and n
b) k-1 and k+1
c) k-1 and n-1
d) k+1 and n-k

13. A weight-balanced tree is a binary tree in which for each node, the number of nodes in the let sub tree is at least half and at most twice the number of nodes in the right sub tree. The maximum possible height (number of nodes on the path from the root to the furthest leaf) of such a tree of n nodes is best described by which of the following? [GATE 2002]

a) $\log_2 n$
b) $\log_{4/3} n$
c) $\log_3 n$
d) $\log_{3/2} n$

14 The number of leaf nodes in a rooted tree of n nodes, with each node having 0 or 3 children is :

a) n/2
b) (n-1) / 3 [GATE 2002]
c) (n-1) / 2
d) (2n+1) / 3

15. Let G be an undirected connected graph with distinct edge weight. Let e_{max} be the edge with maximum weight and e_{min} the edge with minimum weight. Which of the following statements is false? [GATE 2000]

a) Every minimum spanning tree of G must contain e_{min}
b) If e_{max} is in a minimum spanning tree, then its removal must disconnect G.
c) No minimum spanning tree contains e_{max}
d) G has a unique minimum spanning tree

16. Let t be an undirected graph, Consider a depth-first traversal of G, and let T be the resulting depth-first search tree. Let U be a vertex in G and let V be the first new (unvisited) vertex visited after visiting U in the traversal. Which of the following statements is always true? [GATE 2000]

a) {u,v} must be an edge in G, and u is a descendant of v in T

b) {u,v}must be an edge in G, and v is a descendant of u in T

c) If {u,v} is not an edge in G then u is a leaf in T

d) If {u,v} is not an edge in G then u and v must have the same parent in T

17. Let G be a connected, undirected graph. A cut in G is a set of edges whose removal results in G bveing broken into two or more components, which are not connected with each other. The size of a cut is called its cardinality. A min-cut of G is a cut in G of minimum cardinality. Consider the following graph.

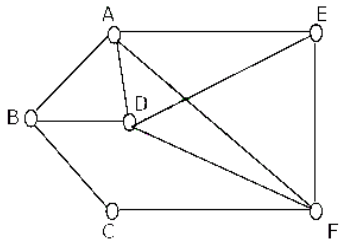
a) Which of the following sets of edges is a cut?
[GATE 1999]

i) {(A,B), (E,F), (B,D), (A,E), (A,D)}

ii) {(B,D), (C,F), (A,B)

b) What is the cardinality of min-cut in this graph?

c) Prove that if a connected undirected graph G with n vertices has a min-cut of cardinality k, then G has atleast (nk/2) edges.



18. How many sub strings of different lengths (non-zero) can be found formed from a character string of length n?
[GATE 1998]

a) n

b) n^2

c) 2^n

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

19. A binary Tree T has n leaf nodes, the number of nodes of degree 2 in T is : [GATE 1995]

a) $\log_2 n$

b) n-1

c) n

d) 2^n

20. The minimum number of edges in a connected cyclic graph on n vertices is [GATE 1995]

a) n-1 b) n

c) n + 1 d) None of the above

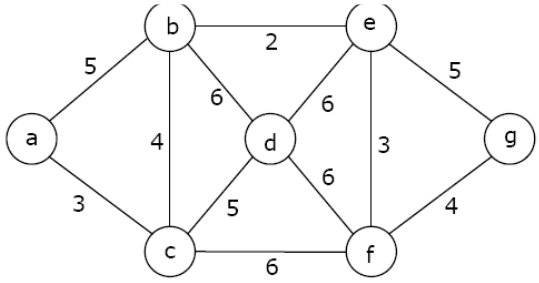
UNIT-III

1. Consider the following graph: [GATE 2009]

Which one of the following is NOT the sequence of edges added to the minimum spanning tree using Kruskal's algorithm?

a) (b,e) (e,f) (a,c) (b,c) (f,g) (c,d) b) (b,e) (e,f), (a,c) (f,g) (b,c) (c,d)

c) (b,e) (a,c) (e,f) (b,c) (f,g) (c,d) d) (b,e) (e,f) (b,c) (a,c) (f,g) (c,d)



2. Consider a weighted complete graph Gon the vertex set {v1,v2 ,v} such that the weight of the edge (v,,v) is 2i-j. The weight of a minimum spanning tree of G is:
[GATE 2006]

a) n—I b) 2n—2

c) (n/2) d) n^2

3. Let G be a weighted undirected graph and e be an edge with maximum weight in G. Suppose there is a minimum weight spanning tree in G containing the edge e. Which of the following statements is always TRUE? [GATE 2004]

a) There exists a cutest in G having all edges of maximum weight

b) There exits a cycle in G having all edges of maximum weight

c) Edge e cannot be contained in a cycle

d) All edges in G have the same weight

Statement for Linked Answer Questions 82a & 82b:

Let s and t be two vetices in a undirected graph G=(V,E) having distinct positive edge weights. Let [X,Y] be a partition of V such that s E X and T € Y. Consider the edge e having the minimum weight amongst all those edges that have one vertex in X and one vertex in Y.

4. a) The edge e must definitely belong to:

   a) the minimum weighted spanning tree of G
   b) the weighted shortest path from s to t
   c) each path from s to t
   d) the weighted longest path from s to t

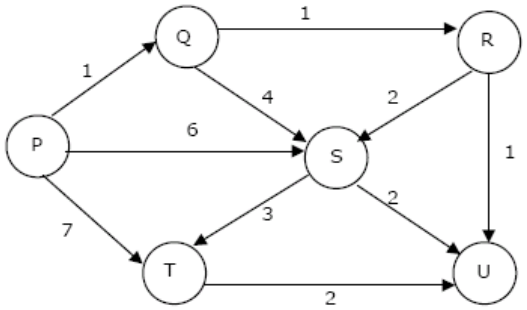
   b) Let the weight of an edge e denote the congestion on that edge. The congestion on a path is defined to be the maximum of the congestions on the edges of the path. We wish to find the path from s to t having minimum congestion. Which one of the following paths is always such a path of minimum congestion?

   a) a path from s to tin the minimum weighted spanning tree
   b) a weighted shortest path from s to t
   c) an Euler walk from s to t
   d) a Hamiltonian path from s to t [GATE 2005]

5. An undirected graph G has n nodes. Its adjacency matrix is given by an n x n square matr4ix whose (i) diagonal elements are 0's and (ii) non-diagonal elements are l's. which one of the following is TRUE?

   a) Graph G has no minimum spanning tree (MST)
   [GATE 2005]
   b) Graph G has a unique MST of cost n-1
   c) Graph G has multiple distinct MSTs, each of cost n-1
   d) Graph G has multiple spanning trees of different costs

6. Suppose we run Dijkstra's single source shortest-path algorithm on the following edge weighted directed graph with vertex P as the source. [GATE 2004]



a) P,Q.R.S.T.U
b) P,Q,R,U,S,T

c) P,Q,R.U,T,S
d) P,Q,T,R,U,S

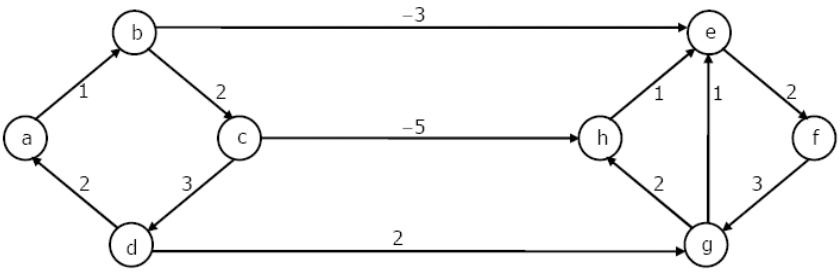
7. Maximum number of edges in a planar graph with n vertices is _____[GATE 1993]

8. Complexity of Kruskal's algorithm for finding the minimum snapping tree of an undirected graph containing n vertices and m dedges if the edges are sorted is _____ [GATE 1992]

UNIT-III(part b)

1. Dijkstra's single source shortest path algorithm when run from vertex a in the above graph, computes the correct shortest path distance to [GATE 2008]
a) only vertex a
b) only vertices a,e,f,g,h

c) only vertices a,b,c,d
d) all the vertices



2. In an unweighed, undirected connected graph, the shortest path from a mode S to every other node is computed most efficiently, in terms of the time complexity, by [GATE 2007]

a) Dijkstra's algorithm starting from S
b) Warshall's algorithm
c) Performing a DFS starting from
d) Performing a BFS starting from S

3. To implement Dijkstra's shortest path algorithm on unweighted graphs so that it runs in linear time, the data structure to be used is: [GATE 2006]

a) Queue
b) Stack

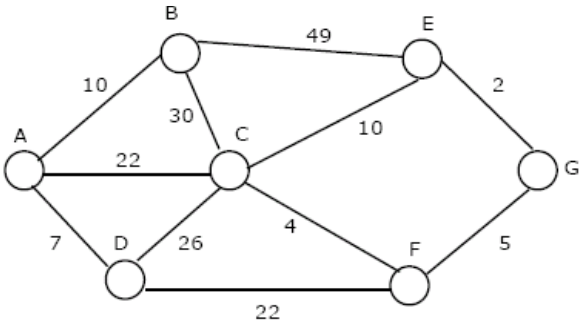
c) Heap
d) B-Tree

4. Let G(V,E) be an undirected graph with positive edge weights. Dijkstra's single source shortest path algorithm can be implemented using the binary heap data structure with time complexity: [GATE 2005]

a) o ($|v|^2$)
b) O(|E|+|V|log|V|)

c) O(|V|log|V|
d) O((|E|+|V|)log|V|)

5. Consider the undirected graph below:



Using Prim's algorithm to construct a minimum spanning tree starting with node A, which one of the following sequences of edges represents a possible order in which the edges would be added to construct the minimum spanning tree? [GATE 2004]

a) (E, G), (C, F), (F, G), (A, D), (A, B), (A, C)

b) (A, D), (A, B), (A, C), (C, F), (G, E), (F, G)

c) (A, B), (A, D), (D, F), (F, G), (G, E), (F, C)

d) (A, D), (A, B), (D, F), (F, C), (F, G), (G, E)

6. The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree (the height is the maximum distance of a leaf node from the root)? [GATE 2004]

a) 2 b) 3

c) 4 d) 6

7. Fill in the blanks in the following template of an algorithm to complete all pairs shortest path lengths in a directed graph G with n*n adjacency matrix A. A[I,j] equals if thre is an edge in G from i to j, and O otherwise. Your aim in filling in the blanks is to ensure that the algorithm is correct. [GATE 2002]

INITIALIZATION: For i = 1 . . . n

{For j=1..n

{if A[i,j]=0 then P[i,j}=_____ else P{i,j]=_____;}

ALGORITHM: For i = 1 . . n

{ For j = 1 . . . n

{ For k = 1 . . . n

{P[_____,_____] = min {_____, _____};}

}

}

a) Copy the complete line containing the blanks in the initialization step and fill in the blanks:

b) Copy the complete line containing the blanks in the algorithm step and fill in the blanks.

c) Fill in the blank: The running time of the algorithm is O(____).

UNIT-IV

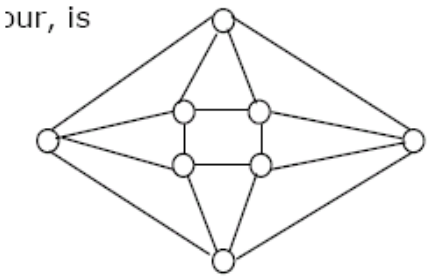
1. What is the maximum number of edges in an acyclic undirected graph with n vertices? [GATE 2004]

a) n-I
b) n
c) n + 1
d) 2n - 1

2. What is the number of vertices in an undirected connected graph with 27 edges, 6 vertices of degree 2, 3 vertices of degree 4 and remaining of degree 3?
[GATE 2004]

a) 10
b) 11
c) 18
d) 19

3. The minimum number of colours required to colour the following graph, such that no two adjacent vertices are assigned the same colour, is [GATE 2004]



a) 2
b) 3
c) 4
d) 5

4 The number of leaf nodes in a rooted tree of n nodes, with each node having 0 or 3 children is :

a) n/2
b) (n-1) / 3 [GATE 2002]
c) (n-1) / 2
d) (2n+1) / 3

5. The maximum number of colours required to colour the vertices of a cycle with n nodes in such a way that no two adjacent nodes have the same color is
[GATE 2001]

a) 2
b) 3
c) 4
d) n-2(n/2)+2

6 Give the correct matching for the following pairs:
[GATE 1998]

a) O (log n)	(P) Selection	
b) O (n)	(Q)	Insertion sort
c) O (n log n)	(R)	Binary Search
d) O (n^2)	(S) Merge sort	

a) A – R B – P C- Q D - S
b) A – R B – P C – S D –Q
c) A – P B – R C – S D – Q
d) A –P B-S C-R D-Q

7. Which of the following algorithm design technique is used in finding all pairs of shortest distances in a graph? [GATE 1998]

a) Dynamic programming
b) Backtracking
c) Greedy
d) Divide and conquer

8. The number of edges in a regular graph of degree d and n vertices is _____ [GATE 1994]

9. Consider a simple connected graph G with n vertices and n-edges (n>2). Then which of the following statements are true? [GATE 1993]

a) G has no cycles
b) The graph obtained by removing any edge from G is not connected
c) G has atleast one cycle
d) The graph obtained by removing any two edges from G is not connected
e) None of the above.

UNIT VII

10. A non-planar graph with minimum number of vertices has
[GATE 1992]

a) 9 edges, 6 vertices
b) 6 edges, 4 vertices
c) 10 edges, 5 vertices
d) 9 edges, 5 vertices

UNIT- V

1. The subset-sum problem is defined as follows: Given a set S of n positive integers W, determine whether there is a subset of S whose elements sum to W.
[GATE 2008]

An algorithm Q solves this problem in O(nW) time, which of the following statements is false?

a) Q soves the subset-sum problem in polynomial time when the input is encoded in unary.
b) Q solves the subset-sum probem in polynomial time when the input is encoded in binary.
c) The subset sum problem belongs to the class NP
d) The subset sum problem is NP-hard.

2. Let S be an NP-complete problem and Q and R be two other problems not known to be in NP. Q is polynomial time reducible to S and S is polynomial-time reducible to R. Which one of the following statements is true? [GATE 2006]

a) R is NP-complete
b) R is NP-hard
c) Q is NP-complete
d) Q is NP-hard

3. The problem 3-SAT and 2-SAT are [GATE 2004]

a) both in P
b) both NP complete
c) NP-complete and in P respectively
d) Undecidable and NP-complete respectively

4. Ram and Shyam have been asked to show that a certain problem II is NP-complete. Ram shows a polynomial time reduction from the 3-SAT problem to II, and shyam shows a polynomial time reduction from II to 3-SAT. Which of the following can be inferred from these reductions? [GATE 2003]

a) II is NP-hard but not NP-complete
b) II is in NP, but is not NP-complete
c) II is NP-complete
d) II is neither NP-hard, nor in NP

5. Which of the following problems is not NP-hard?
[GATE 1992]

a) Hamiltonian circuit problem
b) The 0/1 knapsack problem
c) Finding bi-connected components of a graph
d) The graph colouring problem

17.Internal Examination Question Papers

CMR INSTITUTE OF TECHNOLOGY
Kandlakoya (V), Medchal Road, Hyderabad-501401
MID EXAMINATION – 1 Set-1

Course : II B.TECH (CSE)- II SEM	Branch : CSE(A,B,C,D)
Subject : Design and Analysis of Algorithm	Duaration : 1 hrs
Date : 16-02-2015(AN)	Max.Marks : 10

Answer any TWO of the following FOUR Questions

1.a)What is an algorithm?Write the pseucode steps for an algorithm

b)Compute space complexity for Matrix multiplication

2.a)Discuss control abstraction for divide and conquer strategy.

b)Solve the given recurrence relation

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

3 .a)Define Connected and Biconnected components with example

b)Write and explain the algorithm for weighted& collapse rule with example.

4 .Find the maximum profit of knapsack. capacity m=20,(p1,p2,p3)=(25,24,15) and (w1,w2,w3)=(18,15,10)

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID EXAMINATION – 1 Set-2

Course :II B.TECH (CSE)- II SEM	Branch : CSE(A,B,C,D)
Subject : Design and Analysis of Algorithm	Duaration : 1 hrs
Date : 16-02-2015(AN)	Max.Marks : 10

Answer any TWO of the following FOUR Questions

1. a)Explain performance analysis of an algorithm.

   b)Find time complexity of factorial of given number using frequency method.

2. a)Compare and contrast divide and conquer and greedy method.

   b) Solve the given recurrence relation, If K is a non-negative constant the show that the solution to give recurrence eqation For n a power of 2 is

$$T(n) = K \qquad where \;\; n = 1$$
$$= 3T(n/2) + Kn \qquad where \;\; n > 1$$

3.a)Define an articulation point in graph. Write algorithm for finding articulation point.

   b)Write short notes on AND/OR graphs.

4.Explain the job sequencing with deadlines algorithm and also solve for the instance

   n=7,(p1…p7)=(3,5,20,18,1,6,30)&(d1….d7)=(1,3,4,3,2,1,2)

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID EXAMINATION – 1 Set-3

Course : II B.TECH (CSE)- II SEM	Branch : CSE(A,B,C,D)
Subject : Design and Analysis of Algorithm	Duaration : 1 hrs
Date : 16-02-2015(AN)	Max.Marks : 10

Answer any TWO of the following FOUR Questions

1. a)Write a short notes on Amortized analysis?

   b)Find Big oh notation and Omega notation for $f(n)=7n^3+4n^2+10$

2. a)Explain about Strassen's matrix multiplication?Show its time complexity isO($n^{2.81}$)

   b) Compare quick sort and merge sort

3. a)Write short notes on Game trees

   b) Write algorithm for Depth First Search.

4. a) Compare Kruskal's and Primes algorithm.Give their time complexities

   b)Write Kruskals algorithm that generates minimum spanning tree for every connected undirected graph.

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID EXAMINATION –1 Set-4

Course : II B.TECH (CSE)- II SEM	Branch : CSE(A,B,C,D)
Subject : Design and Analysis of Algorithm	Duaration : 1 hrs
Date : 16-02-2015(AN)	Max.Marks : 10

Answer any TWO of the following FOUR Questions

1 a)Explain Big Oh,Omega and Theta notations.

b)Find Big Oh and theta notation for the following functions

i)$100n+6$ ii)$10n^2+4n+2$

2. a)Derive average case time complexity for quick sort.

b)Solve the given recurrence relation

$T(n)=\{b \quad n=1$

$8T(n/2)+n^3 \quad n>2\}$

3. a)Write algorithm for non recursive preorder binary tree traversal.

b)Inorder sequence is 2,8,4,14,1,7,16,9,10,3 find the postorder and preorder of a binary tree traversal.

4. Find the feasible solution for job sequencing with deadlines for instance

n=5,(p1….p5)=(20,15,10,5,1) and (d1…d5)=(2,2,1,3,3)

Code No: 54016 **Set No. 1**

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

II B.Tech. II Sem., I Mid-Term Examinations, February – 2013

DESIGN AND ANALYSIS OF ALGORITHMS

Objective Exam

Name: _____ **Hall Ticket No.** | | | | | | A | | | | |

Answer All Questions. All Questions Carry Equal Marks. Time: 20 Min. Marks: 10.

I Choose the correct alternative:

1. Theta notation expresses [ ]
   (A) tight bounds (B) upper bounds (C) lower bounds (D)worst cases

2. Consider the following functions:
   $f(n) = 2^n$
   $g(n) = n!$
   $h(n) = n^{\log n}$
   Which of the following statements about the asymptotic behavior of f(n), g(n) and h(n) is true? [ ]
   (A) $f(n) = O(g(n))$; $g(n) = O(h(n))$ (B) $f(n) = \Omega(g(n))$; $g(n) = O(h(n))$
   (C) $g(n) = O(f(n))$; $h(n) = O(f(n))$ (D)$h(n)=O(f(n))$;$g(n)=\Omega(f(n))$

3. Function g is an upper bound on function f iff for all x, [ ]
   (A) $g(x) \leq f(x)$ (B) $g(x) \geq f(x)$; (C) $g(x) = f(x)$ (D) $f(x)<g(x)$

4. Time taken to perform (n-1) UNIONS is [ ]
   (A)$O(n^2)$ (B)$O(n^3)$ (C)O(n) (D)O(1)

5. How many spanning trees does the following graph have? [ ]

   (A)1 (B)2 (C)3 (D)4

6. Which of the following is the best sorting algorithm [ ]
   (A) Merge Sort (B) Heap Sort (C) Bubble Sort (D)Quick Sort

7. Which of the following sorting technique have same O (nlogn) complexity in all the cases? [ ]
   (A) Insertion Sort (B) Quick Sort (C)Merge Sort (D)Selection sort

8. The running time of quick sort depends heavily on the selection of [ ]
   (A) No of inputs (B) Arrangement of elements in array (C) Size of elements (D) Pivot element

9. Dijkstra's algorithm : [ ]
   (A) Has greedy approach to find all shortest paths
   (B) Has both greedy and Dynamic approach to find all shortest paths
   (C) Has greedy approach to compute single source shortest paths to all other vertices
   (D) Has both greedy and dynamic approach to compute single source shortest paths to all other vertices.

Cont.....2

10. Which of the following statements is true about Kruskal's algorithm. [ ]
(A) It is an inefficient algorithm, and it never gives the minimum spanning tree.
(B) It is an efficient algorithm, and it always gives the minimum spanning tree.
(C) It is an efficient algorithm, but it doesn't always give the minimum spanning tree.
(D) It is an inefficient algorithm, but it always gives the minimum spanning tree.

II Fill in the blanks:

11. The running time of an algorithm is represented by following recurrence relation:

$$T(n) = \begin{cases} T\left(\frac{n}{3}\right)^{a} + cn & n \le 3 \\ \text{Otherwise} \end{cases}$$

_____represents the time complexity of the algorithm.

12. $\sum_{1 \le k \le n} O(n)$, where O(n) stands for order n is:_____.

13. _____ rule is used for FIND algorithm.

14. _____ Search is used for testing connected components.

15. A graph G is said to be bi-connected if and only if it contains _____

16. The complexity of Binary search algorithm is _____

17. Merging 4 sorted files containing 50, 10, 25 and 15 records will take____time

18. _____ algorithm technique is used in the implementation of Kruskal solution for the MST.

19. property satisfies the optimal-substructure property, then a locally optimal solution is globally optimal.

20. The Knapsack problem belongs to the domain of _____ problems.

Code No: 54016 **Set No. 1**

DESIGN AND ANALYSIS OF ALGORITHMS
Keys

I Choose the correct alternative:

1. A
2. B
3. B
4. C
5. C
6. D
7. C
8. D
9. C
10. B

II Fill in the blanks:

11. $\Theta(n)$
12. $O(n^2)$
13. Collapsing rule
14. Breadth First
15. No Articulation Point
16. O(log n)
17. O (100)
18. Greedy Technique
19. Greedy Choice
20. Optimization

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID–II A.Y 2014-2015 SET-1

Course :IIB.Tech, II Sem	Branch : CSE-A.B.C.D
Subject :DAA	Duaration : 60 mins
Date :21-04-2015	Max.Marks : 10

Answer any two of the following four Questions. All Carry Equal marks

1. Explain the Principle of Optimality?
2. Find the solution for 0/1 Knapsack problem for n=3,m=6, Profits=1,2,3 and weights=2,3,4 using dynamic programming.
3. Find the solution for 4-Queens problem and Draw the solution state space tree using Backtracking
4. Explain the following terms with examples

   a)E-node B)Answer-state c)Live-node d)Solution state e)State space tree

   b) Explain about Cook's theorem?

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID–II A.Y 2014-2015 SET-2

Course :IIB.Tech, II Sem	Branch : CSE-A.B.C.D
Subject :DAA	Duaration : 60 mins
Date :21-04-2015	Max.Marks : 10

Answer any two of the following four Questions. All Carry Equal marks

1. Draw the portion of state space tree generated by LCBB by the following Knapsack problem for n-5,profits=10,15,6,8,4 and weights=4,6,3,4,2 and m=12
2. Find the all possible subsets of W that sum m .Draw the protion of the state space tree m=35,W=15,7,20,5,18,10,12
3. Explain the following terms with examples

   a)Deterministic algorithm B)Non-deterministic algorithm c)NP-hard d)NP-Complete
4. Write the Algorithm for optimal binary search tree of dynamic programming?

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID–II A.Y 2014-2015 SET-3

Course :IIB.Tech, II Sem	Branch : CSE-A.B.C.D
Subject :DAA	Duaration : 60 mins
Date :21-04-2015	Max.Marks : 10

Answer any two of the following four Questions. All Carry Equal marks

1. Write the differences between Backtracking and Brute force approach?
2. Write the algorithm for N-queens Problem?
3. Find the solution to design a 3 Stage System with device types D1,D2,D3. The costs are C1=$30,C2=$15,C3=$20 respectively.The total cost of the System is C=105. And Reliability r1=0.9 ,r2=0.8 ,r3=0.5.
4. Generate all possible 3- coloring for the following graph with 4-nodes using a state space tree.

CMR INSTITUTE OF TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad-501401

MID–II A.Y 2014-2015 SET-4

Course :IIB.Tech, II Sem	Branch : CSE-A.B.C.D
Subject :DAA	Duaration : 60 mins
Date :21-04-2015	Max.Marks : 10

Answer any two of the following four Questions. All Carry Equal marks

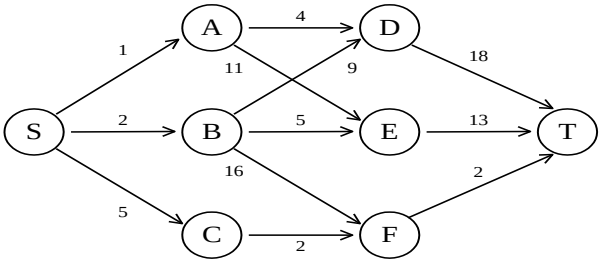
1. a)Explain the Classes of NP-hard and NP-complete?
   b)Write the control abstraction algorithm for LC-search?

2. Solve the following instance of travelling sales person problem using LCBB?

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} - & 10 & 8 & 9 & 7 \\ 10 & - & 10 & 5 & 6 \\ 8 & 10 & - & 8 & 9 \\ 9 & 5 & 8 & - & 6 \\ 7 & 6 & 9 & 6 & - \end{pmatrix} \end{array}$$

3. a)Explain the Satisfiability problem and Write the algorithm for the same?
   b) Write the algorithm for m-coloring of a graph?
4. Find the minimum cost path from s to t in the multistage graph of the following fig. in Backward approach

.



Code No: 54016 **Set No. 1**

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

II B.Tech. II Sem., II Mid-Term Examinations, April – 2012

DESIGN AND ANALYSIS OF ALGORITHMS

Objective Exam

Name: _____ **Hall Ticket No.** | | | | | | A | | | | |

Answer All Questions. All Questions Carry Equal Marks. Time: 20 Min. Marks: 10.

I Choose the correct alternative:

1. The name backtrack was first coined by [
   a) D.H Lehmar b) J.D. Ullman c) K.Thompson d) R.E. Bellman

2. If M=15, n=4, =(10,10,12,18) and =(2,4,6,9) of 0/1 knapsack problem then the maximum profit is [
   a)32 b)34 c)36 d)38

3. In branch-and-bound method, for nodes representing infeasible solutions, C(x) = [
   a)Infinite b)0 c)1 d)2

4. _____ Functions find a live node with least cost function in LC search [
   a) minimum() b) Maximum() c) Least() d) Search()

5. What is the relation between P and NP? [
   a) P ? NP ,but no one knows P=NP b) P=NP c) P>NP d) P<NP

6. We start at a particular node in the graph, visiting all nodes exactly once and come back to initial node with minimum cost is known as [
   a) 0/1 knapsack problem b) Optimal storage on tapes
   c) Minimum cost spanning tree d) Travelling sales person problem

7. For infinite state space trees with no answer nodes [
   a) LC search will terminate b) LC search will not terminate
   c) LC search can not be conducted d) Termination of LC search depends on cost function

8. The FIFO search coupled with bounding functions is called as [
   a) FIFOBB b) Least count search
   c) Cumulative reduction function d) Column reduction

9. NP complete stands for [
   a) Natural polynomial time complete b) Non polynomial time complete
   c) N-power time complete d) Non deterministic polynomial time complete

10. Only _____ can be NP complete [
    a) Linear problems b) Non linear problems
    c) Decision problems d) Hard problems

Cont.....2

II Fill in the blanks:

11. Tractability means _____.
12. _____ is the set of decision problems that can be solved by a determinstic machine in a polynomial time.
13. _____ process in one whose behavior is non deterministic i.e, the next state of the environment is not fully determined by the previous state of the environment.
14. In FIFOBB square node indicates _____.
15. The LC branch and bound search of a tree will begin with upper _____.
16. An optimal solution is a feasible solution with _____.
17. In branch and bound method the three common search strategies are _____, _____ and _____.
18. In FIFOBB initially the queue of live nodes is _____.
19. A problem is intractable if all algorithms to solve that problem are of atleast _____.
20. In _____ algorithms, the result of every operation is uniquely defined.

-o0o-

Code No: 54016 **Set No. 1**

DESIGN AND ANALYSIS OF ALGORITHMS
Keys

I Choose the correct alternative:

1) a
2) d
3) a
4) c
5) a
6) d
7) a
8) a
9) d
10) c

II Fill in the blanks:

11) Practically useful algorithm
12) Complexity class P
13) Stochastic
14) Answer nodes
15) 0
16) Maximum value
17) FIFO, LIFO and LC
18) Empty
19) Exponential time complexity
20) deterministic

-oOo-

18.External/End Examination Question Papers

B.TECH DEGREE EXAMINATION

Second Semester

Branch: Computer Science Engineering

DESIGN AND ANALYSIS OF ALGORITHMS (CSE)

(Model Question Paper)

Time: Three Hours **Maximum : 100 Marks**

Part – A

Answer all questions.
Each question carries 3 marks

1. Illustrate the steps involved in analyzing algorithm using an example.
2. Explain a sorting algorithm that use divide and conquer method.
3. Write the Kruskal's algorithm for minimum spanning tree.
4. Explain any one branch and bound technique.
5. Discuss planar graph coloring.

(5 × 3 = 15 marks)

Part – B

Answer all questions.
Each question carries 5 marks

6. Solve the recurrence relation, where T(1)=1 and T(n) for n>=2 satisfies T(n)=3T(n/2)+n.
7. Explain matrix multiplication using divide and conquer.
8. Explain the characteristics of a problem that can be solved efficiently using Dynamic programming technique.
9. Differentiate between generating function and bounding function.
10. Write a short note on string matching algorithms.

(5 × 5 = 25 marks)

PART – C

Answer any one full question from each module.
Each question carries 12 marks

11. (i) Why worst case analysis of algorithms is most important than average case analysis? (4 marks)

    (ii) Explain various asymptotic methods used to represent the rate of growth of running time of algorithms. (8 marks)

Or

12. Write an algorithm to search an item in a linear list. If there are n nodes in the list, what is the running time of your algorithm.

13. Explain a search procedure using divide and conquer technique. Prove that the procedure works correctly. Give the time complexity of the algorithm.

Or

14. Write an algorithm for quick sort. Explain with an example and show the analysis for the algorithm.

15. Explain Prim's algorithm for finding minimum spanning tree.

Or

16. Suggest an approximation algorithm for traveling sales person problems using Minimum spanning tree algorithm. Assume that the cost function satisfies the triangle inequality.

17. Explain the concept of backtracking using fixed and variable tuple formation.

Or

18. Explain N-Queens Problem.

19. Establish lower bounds for sorting by comparison of keys (both average and worst case).

Or

20. Explain with example Las Vagas algorithm for search.

(5 × 12 = 60 marks)

B.TECH DEGREE EXAMINATION

Second Semester

Branch: Computer Science Engineering

DESIGN AND ANALYSIS OF ALGORITHMS (CSE)

(Model Question Paper)

Time: Three Hours **Maximum : 100 Marks**

SECTION –A (10x2=20)

Q.1.

(a) Define algorithm validation.

(b) Define Big oh notation.

(c) List out two drawbacks of binary search algorithm.

(d) What is the time complexity of binary search??

(e) What are the drawbacks of Greedy Algorithm?

(f) What are the applications of dynamic programming?

(g) Define Brute force approach?

(h) What is the difference between a Live Node and Dead Node?

(i) Define feasible and optimal solution.

(j) What is a Recurrence Equation?

SECTION –B (4x5=20)

Q.2. Explain how analysis of linear search is done with a suitable illustration.

Q.3. Write a pseudo code for divide & conquer algorithm for merging two sorted arrays in to a single sorted one. Explain with example.

Q.4. Describe binary search tree with three traversal patterns? Give suitable example with neat diagram for all three traversal of binary search tree

Q.5. What is topological sorting? Where it is required?

SECTION –C (2x10=20)

Q.6. Explain quick sort Algorithm. (b) What are the steps involved in proving a problem NP-complete? Specify the problems already proved to be NP-complete.

Q.7. Solve the following 0/1 knapsack problem using dynamic programming. P=[11,21,31,33] w=[2,11,22,15] c=40, n=4.

Q.8. Explain how you can use greedy technique for Huffman coding.

Anna University

B.E./B.Tech. DEGREE EXAMINATION, APRIL/MAY 2010

Fourth Semester

Computer Science and Engineering

CS2251 — DESIGN AND ANALYSIS OF ALGORITHMS

(Regulation 2008)

Time: Three hours Maximum: 100 Marks

Answer ALL Questions

PART A — ($10 \times 2 = 20$ Marks)

1. Differentiate Time Complexity from Space complexity.

2. What is a Recurrence Equation?

3. What is called Substitution Method?

4. What is an Optimal Solution?

5. Define Multistage Graphs.

6. Define Optimal Binary Search Tree.

7. Differentiate Explicit and Implicit Constraints.

8. What is the difference between a Live Node and a Dead Node?

9. What is a Biconnected Graph?

10. What is a FIFO branch-and-bound algorithm?

PART B — ($5 \times 16 = 80$ Marks)

11. (a) Explain how Time Complexity is calculated. Give an example.

Or

(b) Elaborate on Asymptotic Notations with examples.

12. (a) With a suitable algorithm, explain the problem of finding the maximum and minimum items in a set of n elements.

Or

(b) Explain Merge Sort Problem using divide and conquer technique. Give an example.

13. (a) Write down and explain the algorithm to solve all pairs shortest paths problem.

Or

(b) Explain how dynamic programming is applied to solve travelling salesperson problem.

14. (a) Describe the backtracking solution to solve 8-Queens problem.

Or

(b) With an example, explain Graph Coloring Algorithm.

15. (a) Explain in detail the Graph Traversals.

Or

(b) With an example, explain how the branch-and-bound technique is used to solve 0/1 knapsack problem.

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2010

Fourth Semester

Computer Science and Engineering

CS 2251 — DESIGN AND ANALYSIS OF ALGORITHMS

(Regulation 2008)

Time : Three hours Maximum : 100 Marks

Answer ALL questions

PART A — (10 × 2 = 20 Marks)

1. If $f(n) = a_m n^m + ...a_1 n + a_0$, then prove that $f(n) = O(n^m)$.
2. Establish the relationship between O and Ω.
3. Trace the operation of the binary search algorithm for the input –15, –6, 0, 7, 9, 23, 54, 82, 101, 112, 125, 131, 142, 151, if you are searching for the element 9.
4. State the general principle of greedy algorithm.
5. State the principle of optimality.
6. Compare divide and conquer with dynamic programming and dynamic programming with greedy algorithm.
7. Explain the idea behind backtracking.
8. Define and solve the graph colouring problem.
9. Define NP Hard and NP Completeness.
10. What is a Minimum spanning tree?

PART B — (5 × 16 = 80 Marks)

11. (a) (i) Explain briefly the Time Complexity estimation, space complexity estimation and the trade off between Time and Space complexity. (6)

(ii) Solve the following recurrence equations completely

(1) $T(n) = \sum_{i=1}^{n-1} T(i) + 1$, if $n \geq 2$

$T(n) = 1$, if $n = 1$. (4)

(2) $T(n) = 5T(n-1) - 6T(n-2)$ (3)

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

Or

(b) (i) Write the linear search algorithm and analyse for its best, worst and average case time complexity. (10)

(ii) Prove that for any two functions $f(n)$ and $g(n)$, we have $f(n) = \theta\ (g(n))$ if and only if $f(n) = O\,(g(n))$ and $f(n) = \Omega(g(n))$. (6)

12. (a) (i) Write a recursive algorithm to determine the max and min from a given element and explain. Derive the time complexity of this algorithm and compare it with a simple brute force algorithm for finding max and min. (10)

(ii) For the following list of elements trace the recursive algorithm for finding max and min and determine how many comparisons have been made.

22, 13, –5, –8, 15, 60, 17, 31, 47. (6)

Or

(b) (i) Write the container loading greedy algorithm and explain. Prove that this algorithm is optimal. (8)

(ii) Suppose you have 6 containers whose weights are 50, 10, 30, 20, 60, 5 and a ship whose capacity is 100. Use the above algorithm to find an optimal solution to this instance of the container loading problem. (8)

13. (a) (i) Write and explain the algorithm to compute the all pairs source shortest path using dynamic programming and prove that it is optimal (8)

(ii) For the following graph having four nodes represented by the matrix given below determine the all pairs source shortest path (8)

$$\begin{matrix} 0 & \infty & 3 & \infty \\ 2 & 0 & \infty & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & \infty & 0 \end{matrix}$$

Or

(b) (i) Write the algorithm to compute the 0/1 knapsack problem using dynamic programming and explain it. (8)

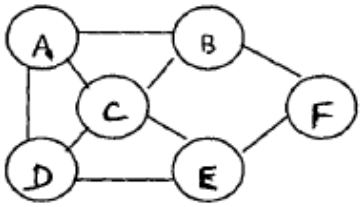
(ii) Solve the following instance of the 0/1, knapsack problem given the knapsack capacity is W = 5 (8)

Items	Weight	Value
1	2	12
2	1	10
3	3	20
4	2	15

14. (a) Write an algorithm to determine the Sum of Subsets for a given Sum and a Set of numbers. Draw the tree representation to solve the subset sum problem given the numbers set as {3, 5, 6, 7, 2} with the Sum = 15. Derive all the subsets. (8 + 8)

Or

(b) Write an algorithm to determine Hamiltonian cycle in a given graph using back tracking. For the following graph determine the Hamiltonian cycle. (8 + 8)



15. (a) Explain with an algorithm as to how 0/1 knapsack problem is solved using branch and bound technique. Apply branch and bound technique to solve the following 0/1 knapsack instance if W = 10. (8 + 8)

Items	Weight	Value
1	4	40
2	[illegible]	2
3	5	[illegible]5
[illegible]	[illegible]	12

Or

(b) Write an algorithm and explain to determine Biconnected Components. Prove the theorem that two biconnected components can have at most one vertex as common and this vertex is an articulation point. (10 + 6)

Code: 9A05403

B.Tech II Year II Semester (R09) Regular & Supplementary Examinations June 2014

DESIGN & ANALYSIS OF ALGORITHMS

(Common to CSS, IT & CSE)

Time: 3 hours Max. Marks: 70

Answer any FIVE questions

All questions carry equal marks

*****

1 Discuss about randomized algorithms in detail.

2 (a) Write a pseudo code for the implementation of UNION instruction using linked list. Explain the working of the implementation.

(b) Give the trees for the set {1, 2, 3, 4, 5, … n} by using weighting rule.

3 Discuss in detail the Stressen's matrix multiplication concept with an example.

4 (a) Prove that if $P_1/W_1 \geq P_2/W_2 \geq \ldots\ldots \geq Pn/Wn$, then greedy knapsack generates an optimal solution to the given instance of knapsack problem.

(b) Let n = 5 , P [1: 5] = (20,15,10,5,1) and d [1: 5] = (2,2,1,3,3). Generate the trees defined by the P (i)'s for the first three iterations.

5 Design a three stage system with device types D_1, D_2, D_3. The costs are the Rs 30. Rs 15 and Rs 20 respectively. The cost of the system is to be not more than Rs. 105. The reliability of each device type is 0.9, 0.8 and 0.5 respectively.

6 Draw the portion of the state space tree generated by SUMOFSUB procedure while working on the instance n = 6; M = 30; w (1:6) = (5, 10, 12, 13, 15, 18).

7 Explain the principles of:
(a) LIFO branch and bound.
(b) FIFO branch and bound.

8 (a) Show that any language in NP can be decided by an algorithm running in time $2_{o(nk)}$ for some constant k.

(b) Prove that if then .

Code: 9A05403

R09

B.Tech II Year II Semester (R09) Supplementary Examinations December/January 2014/2015

DESIGN & ANALYSIS OF ALGORITHMS

(Common to CSE, IT & CSS)

Time: 3 hours Max. Marks: 70

Answer any FIVE questions

All questions carry equal marks

****

1 a) Present an algorithm to finding Fibonacci sequence upto a given number
b) Discuss about Space complexity in detail

2 a) Describe UNION and FIND algorithms
b) What are Disjoint Sets and its Operations? Explain

3 a) Explain the principle of Divide- and – Conquer Technique
b) Draw the tree of calls of Merge foe the following set of elements

(20, 30, 10, 40, 5, 60, 90, 45, 35, 25, 15, 55)

4 a) Find optimal solution for the following Knapsack problem n-3 , m=20 (p1,p2,p3)=(25, 24, 15) and (w1, w2, w3) = (18, 15, 10)
b) Prove that P1/W1>=P2/W2>= ……>=Pn/Wn then the Greedy Knapsack generates optimal solution to the given instance of knapsack problem

5 a) Write an algorithm for all pairs shortest path
b) Explain the matrix chain multiplication with an example

6 Write a Back Tracking algorithm for

a) The Hamiltonian circuit problem
b) M-Coloring problem

7 Draw the position of the state space tree generated by LC branch and bound for an instance n=4 , (p1, p2, p3, p4) = (10, 10, 12, 18) , (w1, w2, w3, w4) = (2, 4, 6, 9) and m=15.

8 a) Write a short notes on

i) Classes of NP-Hard
ii) Classes of NP-Complete

b) How are P and NP problems related

R13

Code No: 114CT

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

B.Tech II Year II Semester Examinations, May - 2015

DESIGN AND ANALYSIS OF ALGORITHMS

(Information Technology)

Time: 3 Hours

Max. Marks: 75

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 25 marks. Answer all questions in Part A.
Part B consists of 5 Units. Answer any one full question from each unit.
Each question carries 10 marks and may have a. b. c as sub questions.

PART- A

(25 Marks)

1.a) What are the properties of an algorithm? [2M]
b) Write control abstraction algorithm of divide and conquer approach. [3M]
c) What is optimal solution? [2M]
d) Write Greedy method control abstraction for the subset paradigm. [3M]
e) What is dominance rule? [2M]
f) Explain how to estimate the minimum cost path in multistage graph by using forward approach. [3M]
g) What is E-node and answer node? [2M]
h) Distinguish between FIFOBB and LIFOBB. [3M]
i) What is deterministic algorithm? [2M]
j) Write non deterministic algorithm for satisfiability. [3M]

PART-B

(50 Marks)

2.a) Derive the time complexity of Quick sort in an average case.
b) Distinguish between Amortized analysis and Probabilistic analysis. [5+5]

OR

3.a) Explain how to reduce the complexity in union algorithm.
b) Derive the time complexity of Merge sort. [5+5]

4.a) Find the optimal solution by using job sequencing with deadlines problem n=5, (p1…..p5)=(20, 15,10, 5, 1) and (d1,…..d5)=(2, 2, 1, 3,3)
b) Write an algorithm of Kruskal's minimum cost spanning tree. [5+5]

OR

5.a) Explain about the single source shortest problem with an example.
b) Write an algorithm of job sequencing deadlines. [5+5]

6.a) Find the minimum cost path from the source(6) to destination(3) by using backward approach in multistage graph shown in figure 1.

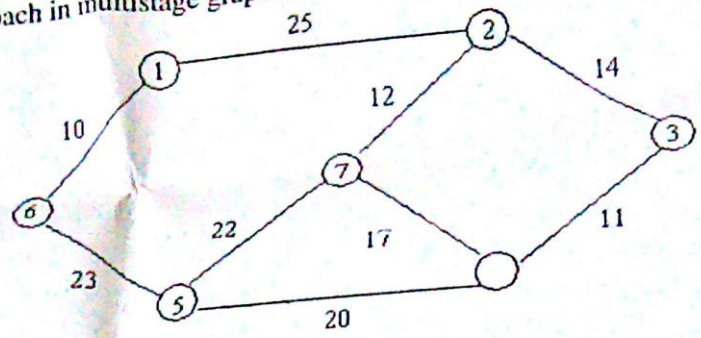


Figure: 1

[7+3]

b) What are the applications of dynamic programming?

OR

7. Design a three stage system with device types D_1, D_2, D_3. The costs all Rs. 20, Rs.15 and Rs 25 respectivesly. The cost of the system is to be no more than Rs.100. The reliability of each device type is 0.7, 0.6 and 0.5 respectively.[10]

8. Solve the traveling sales man problem for the following graph shown in figure 2 by using branch and bound. [10]

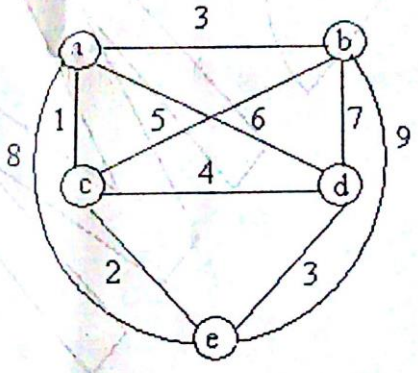


Figure: 2

OR

9.a) w= {15, 7, 20, 5, 18, 10, 12} m=35. Find all possible subsets of w that sum to m. Solve this using sum of subsets. Draw the portion of state space tree that is generated.

b) *Write an algorithm* to estimate the efficiency of backtracking. [5+5]

10.a) Explain the cook's theorem with an example.

b) Show that the knapsack optimization problem reduces to the knapsack decision problem when all the p's w's and m are integer and the complexity is measured as a function of input length.

OR

11.a) Show that the Hamiltonian-path problem is NP-complete.

b) Explain the classes of NP-hard and NP-Complete. [5+5]

---ooOoo---

Code No: 114CS **R13**

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD
B.Tech II Year II Semester Examinations, May - 2016
DESIGN AND ANALYSIS OF ALGORITHMS
(Computer Science and Engineering)

Time: 3 Hours **Max. Marks: 75**

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 25 marks. Answer all questions in Part A.
Part B consists of 5 Units. Answer any one full question from each unit.
Each question carries 10 marks and may have a, b, c as sub questions.

PART- A **(25 Marks)**

1.a) List the asymptotic notations. [2]
b) Explain the time complexity of merge sort. [3]
c) Define graph. [2]
d) Explain the properties of strongly connected components. [3]
e) Give brief description on greedy method. [2]
f) What is multistage graph? [3]
g) Write the applications of Branch and Bound problem. [2]
h) What is sum of subsets problem? [3]
i) What is NP-Hard? [2]
j) Explain non-deterministic algorithm. [3]

PART-B **(50 Marks)**

2.a) What is an algorithm? Explain its characteristics.
b) Explain the strassen's matrix multiplication. [5+5]

OR

3.a) Discuss about space complexity in detail.
b) Write an algorithm for quick sort. Explain with an example. [5+5]

4.a) Describe Union and Find algorithms.
b) Explain the BFS algorithm with example. [5+5]

OR

5.a) Write a nonrecursive algorithm for preorder traversal of a binary tree T.
b) Explain game tree with an example. [5+5]

6.a) Write a greedy algorithm to the job sequencing with deadlines.
b) Define merging and purging rules in 0/1 knapsack problem. [5+5]

OR

7.a) Differentiate between greedy method and dynamic programming.
b) Explain the Kruskal's algorithm with an example. [5+5]

8. Draw the portion of the state space tree generated by LCBB for the following instances:
n=5, m=12, $(P_1 \ldots P_5) = (10, 15, 6, 8, 4)$ $(w_1 \ldots w_5) = (4, 6, 3, 4, 2)$ [10]

OR

9.a) Describe Backtracking technique to m-coloring graph.
b) Briefly explain n-queen problem using backtracking. [5+5]

10.a) Explain the classes of NP-Hard and NP-Complete.
b) Explain the satisfiability problem. [5+5]

OR

11.a) Explain the strategy to prove that a problem is NP hard.
b) Explain the non-deterministic sorting problem. [5+5]

---ooOoo---