Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058, India

(Autonomous College Affiliated to University of Mumbai)

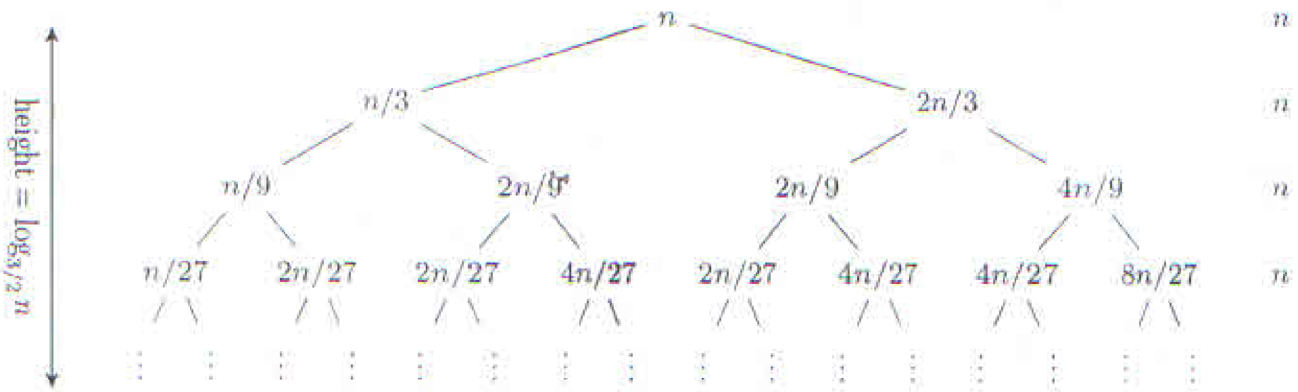
Synoptic

May 2018

Max. Marks: 100	Duration: 3 Hrs.
Class: S.Y.	Semester: IV
Course Code:MCA43	Branch: M.C.A.
Name of the Course: Design and Analysis of Algorithms	

Q. No.		Marks
Q.1 a)	Discuss the Role of Algorithms in Computing. five roles are expected. 1 mark each correct role	1 each

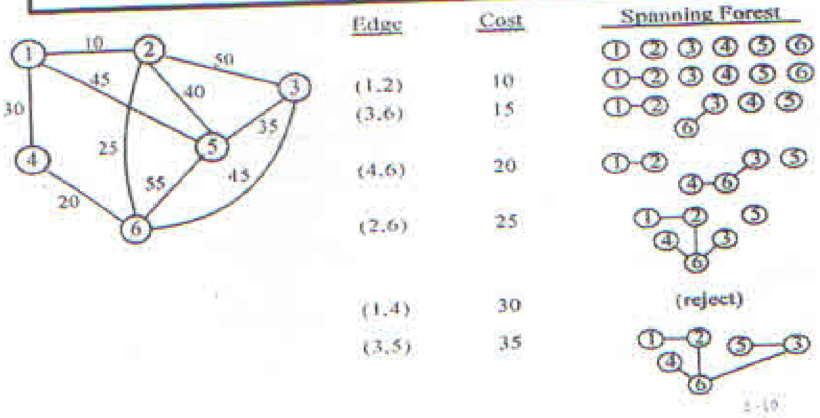
b)	What is the time complexity of following function fun ()? Explain
	
	int fun(int n)
	{
	for (int i = 1; i <= n; i++) …………..n times loop 1 condition check and 1 assignment (2n)
	{ for (int j = 1; j < n; j += i) ……..n-1 times loop 1 condition check and 1 assignment (2(n-1)
	{ Sum = Sum +i*j; ……….. 1 multiplication 1 addition and 1 assignment …3
	}
	}
	return(Sum); ………1 stmt
	}
	
	Time complexity : 2n*2(n-1)*3+1
		2
		2
		2
		
		1

c)	Draw the recurrence tree for *T(n) = T(n/3) + T(2n/3) + n* up to first 3 levels. Let's consider the example, *T(n) = T(n/3) + T(2n/3) + n.* Expanding out the first few levels, the recurrence tree is:  specify height…………………………………………………………………………… OR Compare P, NP, NP-complete problems. Defination	1 2 2 2 1 2

explanation
conditions for comparisons
examples ....2 marks for each comparison

). a)	Find the LCS of string 1: BACBAD string2: ABAZDC In this case, the LCS has length 4 and is the string ABAD. Table-4 marks,finding correct string -2 marks	8

b)



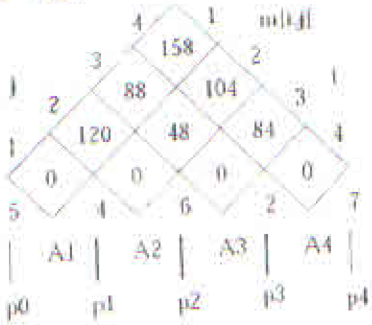
OR

Given a chain of four matrices ,A1, A2, A3, A4 (5,4,6,2,7). Find the cost of matrix multiplication.

$$m[1,4] = \min_{1\le k<4} (m[1,k] + m[k+1,4] + p_0 p_k p_4)$$

$$= \min \left\{ \begin{array}{l} m[1,1] + m[2,4] + p_0 p_1 p_4 \\ m[1,2] + m[3,4] + p_0 p_2 p_4 \\ m[1,3] + m[4,4] + p_0 p_3 p_4 \end{array} \right\}$$

$$= 158.$$



c)	Describe the Dynamic 0/1 Knapsack Problem. Descrition – 2 marks problem solution: 4 marks i. In 0/1 Knapsack problem, items can be entirely accepted or rejected. ii. Given a knapsack with maximum capacity W, and a set S consisting of n items. iii. Each item i has some weight wiand benefit value bi(all wiand W are integer values). iv. The problem is how to pack the knapsack to achieve maximum total value of packed items.	6

or solving the knapsack problem we can generate the sequence of decisions in order to obtain the optimum selection.

vi. Let Xn be the optimum sequence and there are two instances {Xn} and {Xn-1, Xn-2... X1}.

vii. So from {Xn-1, Xn-2... X1} we will choose the optimum sequence with respect to Xn.

viii. The remaining set should fulfill the condition of filling Knapsack of capacity W with maximum profit.

ix. Thus, 0/1 Knapsack problem is solved using the principle of optimality

To solve this problem using dynamic programming method we will perform following steps:

Steps:

1. Let, fi (yj)be the value of optimal solution.
2. Using formula: $f_i(y_j)=max f_{i-1}(y), f_{i-1}(y-w_i)+p_i$

to solve problem.
Then Si is a pair (p, w) where $p=f(y_j)$ and $w=y_j$
Initially $S_0=(0,0)$
Then $S_{i1}=(P,W)|(P-p_i,W-w_i)S_i$
$S_{i+1}1$ can be computed by merging S_i *and* S_{i1}
This is used for obtaining optimal solution.

Find an optimal solution for the dynamic programming 0/1 knapsack instance for n=3, m=6, profits are (p1, p2, p3) = (1,2,5), weights are (w1,w2,w3)=(2,3,4).

Q. 3 a) Discuss the 8-Queen Problem. 2
What technique is used to solve the problem? (Backtracking) 1

Write the algorithm to solve above problem. 4
1) Start in the leftmost column
2) If all queens are placed
   return true
3) Try all rows in the current column. Do following for every tried row.
   a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing queen here leads to a solution.
   b) If placing queen in [row, column] leads to a solution then return true.
   c) If placing queen doesn't lead to a solution then umark this [row, column] (Backtrack) and go to step (a) to try other rows.
3) If all rows have been tried and nothing worked, return false to trigger backtracking.

OR

"Least cost Branch and Bound reduces the state space search" comment 2
Yes: 2 marks 5

Justification ....5 marks

b) Discuss the Hamiltonian Cycles Problem -2 marks.
What technique is used to solve the problem? – 1 mark
Write the algorithm to solve above problem-4 marks

Backtracking

Hamiltonian Path in an undirected graph is a path that visits each vertex exactly once. A Hamiltonian cycle (or Hamiltonian circuit) is a Hamiltonian Path such that there is an edge (in graph) from the last vertex to the first vertex of the Hamiltonian Path. Determine whether a given graph contains Hamiltonian Cycle or not. If it contains, then print the path. Following are the input and output of the required function.

Input:
A 2D array graph[V][V] where V is the number of vertices in graph and graph[V][V] is adjacency matrix representation of the graph. A value graph[i][j] is 1 if there is a direct edge from i to j, otherwise graph[i][j] is 0.

Output:
An array path[V] that should contain the Hamiltonian Path. path[i] should represent the ith vertex in the Hamiltonian Path. The code should also return false if there is no Hamiltonian Cycle in the graph.

Algorithm:

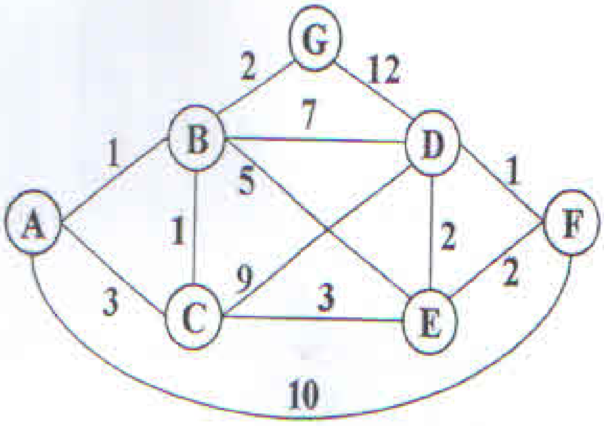
```
function check_all_permutations(adj[][], n)
    for i = 0 to n
        p[i]=i
    while next permutation is possible
        valid = true
        for i = 0 to n-1
            if adj[p[i]][p[i+1]] == false
                valid = false
                break
        if valid == true
            return true
        p = get_next_permutation(p)
    return false
bool check_all_permutations(bool adj[][MAXN], int n){
        vector<int>v;
        for(int i=0; i<n; i++)
            v.push_back(i);
        do{
            bool valid=true;
            for(int i=0; i<v.size()-1; i++){
                if(adj[v[i]][v[i+1]] == false){
                    valid=false;
                    break;
                }
            }
            if(valid)
                return true;
        }while(next_permutation(v.begin(), v.end()));
        return false;
}
```

c) Find the time complexity of "the subset sum problem". Algo-3 marks, 6 complexity- 3marks

OR

efine the 15-puzzle problem. 1 mark
Suggest the technique to solve the problem. 1 mark
According to you does it give the optimal solution? 1 mark
Justify- 3 marks

Q. 4 a) Find the single-source shortest paths from A to every other vertex using Dijkstra's algorithm. 7



Known vertices (in order marked known): A B C G E D or F D or F

Vertex	Known	Cost	Path
A	Y	0	
B	Y	1	A
C	Y	3 2	A B
D	Y	8 7	B E
E	Y	6 5	B C
F	Y	10 7	A E
G	Y	3	B

Lowest-cost path from A to F: A to B to C to E to F

b) Write Floyd Warshalls algorithm for all pair shortest path 6

```
{ for i = 1 to n do                          initialize
      for j = 1 to n do
      { D^0[i, j] = w[i, j];
        pred[i, j] = nil;
      }

   for k = 1 to n do                         dynamic programming
     for i = 1 to n do
       for j = 1 to n do
         if (d^(k-1)[i, k] + d^(k-1)[k, j] < d^(k-1)[i, j])
            {d^(k)[i, j] = d^(k-1)[i, k] + d^(k-1)[k, j];
            pred[i, j] = k;}
         else d^(k)[i, j] = d^(k-1)[i, j];
   return d^(n)[1..n, 1..n];
}
```

:)	Derive the complexity of Knuth Morris Pratt string matching algorithm. Algorithm-4 marks complexity-3 marks OR Derive the complexity of Rabin Carp string matching algorithm. Algorithm-4 marks complexity-3 marks	7
). 5 1)	Derive the Best, Worst and Average time complexities of Quick sorting technique. Algortim: 2 marks Best, Worst and Average: 1 mark each OR definition "Vertex cover Problem". 2 mark Justification of type of problem -3 marks	5
:)	Compare Backtracking and Branch and Bound techniques (Definition-1 Mark, Working 1 Mark, Performance 1 Mark , Analysis 1 Mark, Example 1 Mark)	5

5 d)

$Q = \{a, b, c\}$

$S = a$

$F = c$

$\Sigma = \{0, 1\}$

2 marks

δ = Transition function.

$\{ \delta(a,0) = a \quad \delta(a,1) = b$

$\delta(b,0) = 1 \quad \delta(b,1) = -$

$\delta(c,0) = - \quad \delta(c,1)\ c \}$

1.5 mar

Accepted strings

S1 = 010

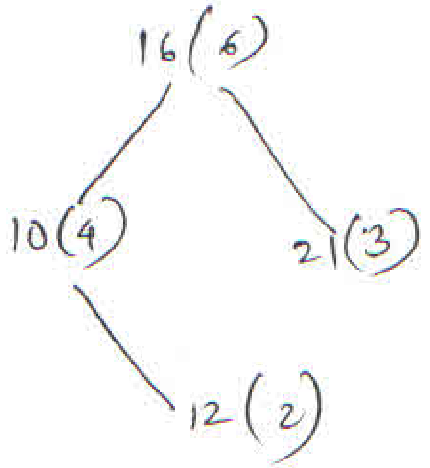
S2 = ~~001~~ 001110

S3 = 00011111000111

1.5 mark

Q5-b. OBST.

	0	1	2	3
0	4	$8^{(0)}$	$20^{(2)}$	$26^{(2)}$
1		2	$10^{(2)}$	$16^{(2)}$
2			6	$12^{(2)}$
3				3



$6 + 4 \times 2 + 3 \times 2 + 2 \times 3 = 26.$

A ~~5~~ ~~4~~ ~~8~~ 20

4 2 6 3