Roll Number:______________________

Thapar Institute of Engineering & Technology

Department of Computer Science and Engineering

END SEMESTER EXAMINATION

B. E. (2nd Year COE/CSE, 3rd Year EM): Sem-IV/VI	Course Code: UCS415
20th May, 2023	Course Name: Design and Analysis of Algorithms
Saturday, 9:00 To 12:00 PM	Name of Faculty: Rajesh Mehta, Tarunpreet Bhatia,
Time: 3 Hours, Max Marks: 60	Randheer Negi, Anil Singh, Vaibhav Pandey, Manisha Panjeta, Shruti Aggarwal

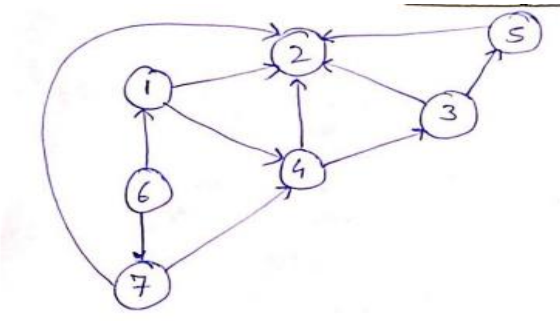
Note: Attempt all Questions in sequence. Answer all sub-parts of each question at one place. Do mention Page No. of your attempt at front page of your answer sheet. Assume missing data (if any).

Q.1 (a) Alice is a well organised person. Every day she makes a list of things which need to be done and enumerates them from 1 to n. However, some things need to be done before others. The input consists of m pairs of two distinct integers x and y, $(1 \leq x, y \leq n)$ describing that job x needs to be done before job y as given in **Table 1**. Your task is to find out whether Alice can solve all her duties and if yes, print the correct order. If there are multiple solutions print the one, whose first number is largest, if there are still multiple solutions, print the one whose second number is largest, and so on. You need to show the intermediate steps for finding the solution. (5)

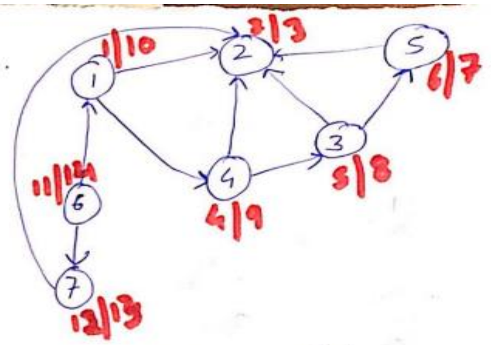
Table 1

x	1	1	4	4	3	5	3	6	7	7	6
y	4	2	2	3	2	2	5	7	4	2	1

Solution

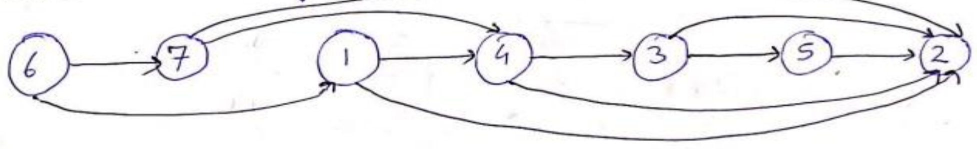


Initial Graph



Applying DFS

Linear Ordering will be



Marks Distribution

1 mark initial correct Graph or any other representation of question

2 marks for steps (either using DFS or Kahn's algorithm)

2 mark for final correct sequence (1 mark upto 6, 7, 1 and 1 mark for 4, 3, 5, 2).

Note: 2 solutions are possible 6, 7, 1, 4, 3, 5, 2 and 6, 1, 7, 4, 3, 5, 2 but we are choosing 6, 7, 1, 4, 3, 5, 2 as per condition given. If someone has written 6, **1, 7**, 4, 3, 5, 2 as answer then 0.5 mark is deducted.

(b) A floor plan of an art gallery is shown in **Fig 1.** Draw a graph that represent the floor plan, where each vertex represent a room and an edge connects two vertices if there is a doorway between the two rooms. Is it possible to walk through the art gallery and pass through each doorway without going through any doorway twice? Does it depend on whether you return to the room you started at? Justify your conclusion. Design an efficient algorithm to solve this problem. (5)

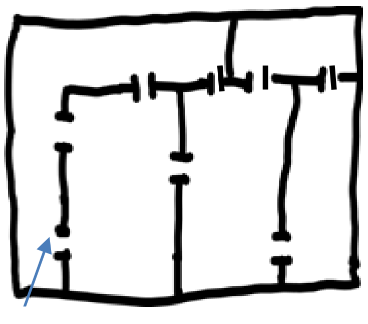
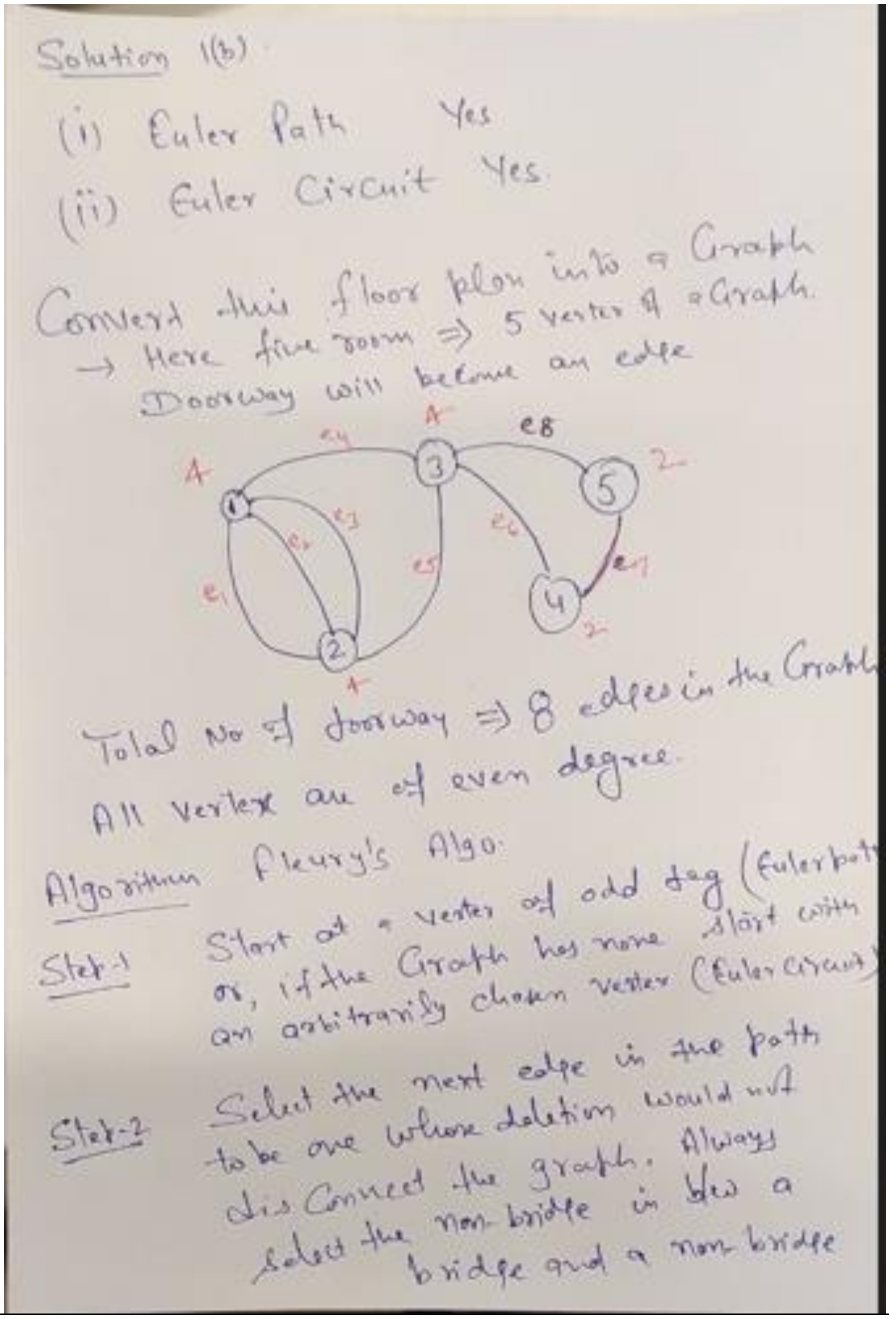


Fig. 1

Solution



Solution 1(b)

(i) Euler Path Yes

(ii) Euler Circuit Yes.

Convert this floor plan into a Graph
→ Here five room ⇒ 5 vertex of a Graph.
Doorway will become an edge

1
2
3
4
5
e1
e2
e3
e4
e5
e6
e7
e8
4
4
4
2
2

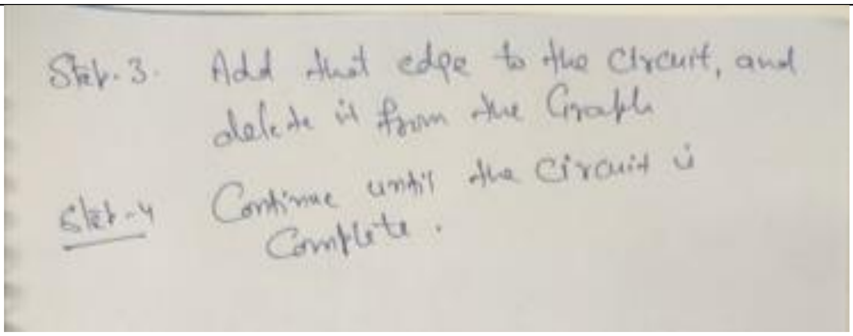
Total No of doorway ⇒ 8 edges in the Graph

All vertex are of even degree.

Algorithm Fleury's Algo.

Step-1 Start at a vertex of odd deg (Euler path) or, if the Graph has none start with an arbitrarily chosen vertex (Euler circuit)

Step-2 Select the next edge in the path to be one whose deletion would not disconnect the graph. Always select the non-bridge in b/w a bridge and a non-bridge



Step-3. Add that edge to the Circuit, and delete it from the Graph

Step-4 Continue until the Circuit is Complete.

Marks Distribution

1 mark for drawing graph

0.5 mark for telling Yes for Euler Path and 0.5 mark for telling Yes for Euler Circuit. These are given if graph is correct.

1 mark for printing Euler Circuit

2 marks for writing Fleury/Hierholzer's Algorithm

Q.2 (a) You are designing a board game for a computer program, and you need to implement the feature that checks whether a given board configuration is valid according to the rules of the game. In particular, you need to check whether the placement of N objects on an NxN board violates the rule that no two objects can attack each other by being in the same row, column, or diagonal. To do this, you decide to use the Backtracking Algorithm to find a valid placement of the N objects on the board. (4+1)

i) Draw the state space tree to find all the possible solutions for $N = 4$.

ii) What is the time complexity of the Backtracking Algorithm for this problem?

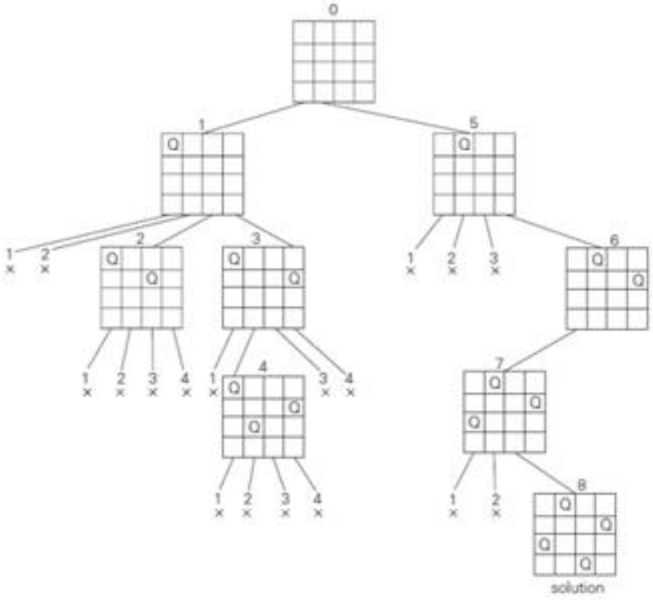
Solution: (5)

2 marks for each solution in state space tree

1 mark for complexity

Solution1: (2 4 1 3) State space tree is shown below

Solution 2: (3 1 4 2) state space tree similar to solution 1



Time complexity = O(N!)

(b) You are a network administrator responsible for assigning frequencies to N wireless devices in a wireless network. Each device can only operate on a single frequency channel at a time, and adjacent devices must operate on different channels to avoid interference. Your goal is to represent this network scenario as an undirected graph G and then assign frequencies to the devices such that the minimum number of channels are used. Design an appropriate algorithm to solve this problem and determine its time complexity.
Solution:

Algorithm : 4 marks

Complexity: 1 mark

i)

Each device is represented as a vertex. Channel acts as a colour of the graph.

Use the Backtracking Algorithm to find a proper k-colouring of G, where each color represents a frequency channel and no two adjacent devices are assigned the same color.

To find a proper k-colouring of G using the Backtracking Algorithm, we can use the following steps:

1. initiate k=1 and increment k until the graph is properly coloured.
2. Choose an uncoloured vertex v.
3. For each possible color c, check if v can be colored with c without violating the coloring constraint (i.e., no adjacent vertices have the same color).
4. If a valid color is found, recursively apply the algorithm to the next uncoloured vertex.
5. If all vertices are colored, the algorithm terminates and outputs the valid k-coloring.

ii) O(k^n), where k is the number of colour and n is the number of vertices in the graph.

Q.3

(a) Write the recursive equation for finding expected search costs in optimal binary search tree (OBST) using dynamic programming when both successful and unsuccessful search probability is given. Also determine the cost and structure of OBST for the following 4 search keys A, B, C and D (given in sorted order) with the success probabilities as 0.1, 0.2, 0.4 and 0.3 respectively.

(1+6)

Solution:

Recursive Equation:

$$e[i, j] = \begin{cases} q_{i-1} & \text{if } j = i-1 \\ \min \{e[i, r-1] + e[r+1, j] + w(i, j) & \\ i \le r \le j \} & \text{if } i \le j \end{cases}$$

where

$$w(i, j) = \sum_{m=i}^{j} p_m + \sum_{m=i-1}^{j} q_m$$

Pseudo code:

```
Algorithm OBST(p, q, n)
// e[1…n+1, 0…n ] : Optimal sub tree
// w[1…n+1,  0…n] : Sum of probability
// root[1…n, 1…n] : Used to construct OBST
for i ← 1 to n + 1 do
        e[i, i – 1] ← qi – 1
        w[i, i – 1] ← qi – 1
end
for m ← 1 to n do
        for i ← 1 to n – m + 1 do
        j ← i + m – 1
        e[i, j] ← ∞
        w[i, j] ← w[i, j – 1] + pj + qj
        for r ← i to j do
       t ← e[i, r – 1] + e[r + 1, j] + w[i, j]
       if t < e[i, j] then
         e[i, j] ← t
         root[i, j] ← r
       end
        end
        end
end
return (e, root)
```

Sol 3 Construct an optimal binary search tree for given
(a) set of keys:

Keys	A	B	C	D
probability	0.1	0.2	0.4	0.3

Initial tables will be

	0	1	2	3	4
1	0	0.1			
2		0	0.2		
3			0	0.4	
4				0	0.3
5					0

	0	1	2	3	4
1		1			
2			2		
3				3	
4					4
5					

Now,

$$e(1,2) = \min \begin{cases} r=1 \; e(1,0) + e(2,2) + \Sigma_{m=1}^{2} p_m \\ r=2 \; e(1,1) + e(3,2) + \Sigma_{m=1}^{2} p_m \end{cases}$$

$$= \min(0 + 0.2 + 0.3,\; 0.1 + 0 + 0.3)$$

$$= \min(0.5,\; 0.4) \Rightarrow 0.4$$

$$e(2,3) = \min \begin{cases} r=2 \; e(2,1) + e(3,3) + \Sigma_{m=2}^{3} p_m \\ r=3 \; e(2,2) + e(4,3) + \Sigma_{m=2}^{3} p_m \end{cases}$$

$$= \min(0 + 0.4 + 0.6,\; 0.2 + 0 + 0.6)$$

$$= \min(1.0,\; 0.8) \Rightarrow 0.8$$

$$e(3,4) = \min \begin{cases} r=3 \; e(3,2) + e(4,4) + \Sigma_{m=3}^{4} p_m \\ r=4 \; e(3,3) + e(5,4) + \Sigma_{m=3}^{4} p_m \end{cases}$$

$$= \min(0 + 0.3 + 0.7,\; 0.4 + 0 + 0.7)$$

$$= \min(1.0,\; 1.1) \Rightarrow 1.0$$

Now, the updated table values are:

	0	1	2	3	4
1	0	0.1	0.4		
2		0	0.2	0.8	
3			0	0.4	1.0
4				0	0.3
5					0

	0	1	2	3	4
1		1	2		
2			2	3	
3				3	3
4					4
5					

Now,

$$e(1,3) = \min \begin{cases} r=1 : e(1,0) + e(2,3) + \sum_{m=1}^{3} p_m \\ r=2 : e(1,1) + e(3,3) + \sum_{m=1}^{3} p_m \\ r=3 : e(1,2) + e(4,3) + \sum_{m=1}^{3} p_m \end{cases}$$

$$= \min(0+0.8+0.7, 0.1+0.4+0.7, 0.4+0+0.7)$$

$$= \min(1.5, 1.2, 1.1) \Rightarrow 1.1$$

$$e(2,4) = \min \begin{cases} r=2 : e(2,1) + e(3,4) + \sum_{m=2}^{4} p_m \\ r=3 : e(2,2) + e(4,4) + \sum_{m=2}^{4} p_m \\ r=4 : e(2,3) + e(5,4) + \sum_{m=2}^{4} p_m \end{cases}$$

$$= \min(0+1.0+0.9, 0.2+0.3+0.9, 0.8+0+0.9)$$

$$= \min(1.9, 1.4, 1.7) \Rightarrow 1.4$$

Now, the table becomes

	0	1	2	3	4
1	0	0.1	0.4	1.1	1.7
2		0	0.2	0.8	1.4
3			0	0.4	1.0
4				0	0.3
5					0

	0	1	2	3	4
1		1	2	3	3
2			2	3	3
3				3	3
4					4
5					

Now,

$$e(1,4) = \min \begin{cases} r=1 : e(1,0) + e(2,4) + \sum_{m=1}^{4} p_m \\ r=2 : e(1,1) + e(3,4) + \sum_{m=1}^{4} p_m \\ r=3 : e(1,2) + e(4,4) + \sum_{m=1}^{4} p_m \\ r=4 ; e(1,3) + e(5,4) + \sum_{m=1}^{4} p_m \end{cases}$$

$$= \min(0 + 1.4 + 1.0,\ 0.1 + 1.0 + 1.0,\ 0.4 + 0.3 + 1.0,\ 1.1 + 0 + 1.0)$$

$$= \min(2.4,\ 2.1,\ 1.7,\ 2.1)$$

$$= 1.7$$

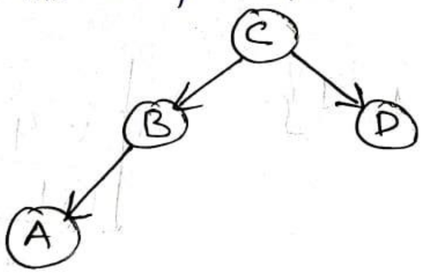
⇒ Thus, the avg. number of key comparisons in OBST is equal to 1.7.

⇒ Since R(1,4) = 3, the root of the optimal tree contains the third key i.e. C.

⇒ Since it is a binary search tree, Its left subtree is made up of keys A and B, and right contain D.

⇒ In the root table since R(1,2) = 2, the root of the optimal tree containing A and B is B, with A being its left child.

⇒ Since R(4,4) = 4, the root of this one-node optimal tree is its only key D.



Marks Distribution

Recursive Equation or Pseudo code (1 mark)

Two table (2 + 2 marks)

OBST structure (2 marks)

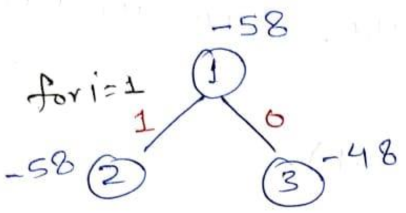
(3)

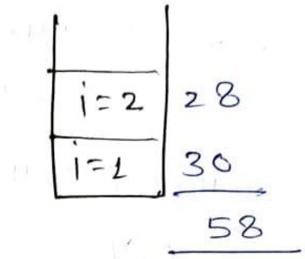
(b) Given two integer arrays $p = \{30, 28, 20, 24\}$ and $wt = \{5, 7, 4, 2\}$ that represent profits and weights associated with $n = 4$ items respectively and a knapsack capacity of 12 kg. With the help of state space tree, solve 0/1 knapsack problem using Least Cost (LC) branch and bound technique.

Sol 3

(b) Given M = 12

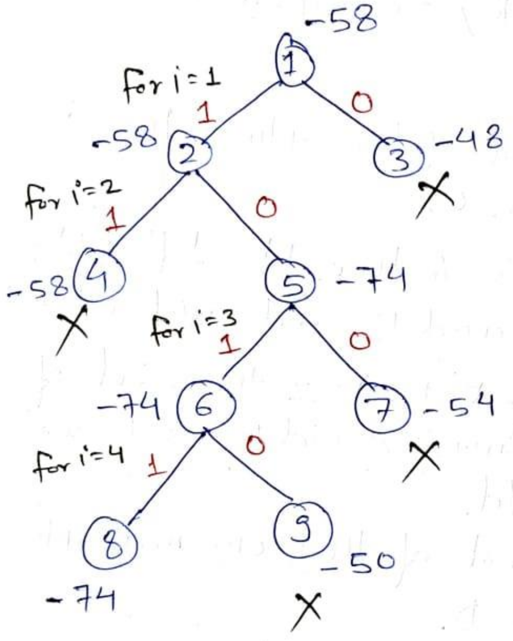
i	1	2	3	4
p	30	28	20	24
w	5	7	4	2

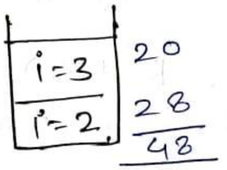




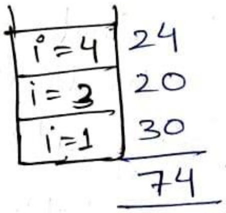
m = ~~12~~ ~~7~~ 0

Apply LCBB.



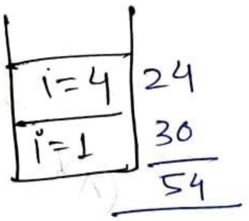


m = ~~12~~ ~~8~~ 1



m = ~~12~~ ~~7~~ ~~3~~ 1

Solution = { 1, 0, 1, 1 }



M = ~~12~~ ~~7~~ 5

Marks Distribution

State space tree (2 marks)

Final solution (1 marks)

Q.4

(a) Write the recursive equation for finding length of longest common subsequence (LCS) between two strings. Consider two strings A = "abcbdab" and B = "bdcaba". You need to determine the length of LCS between A and B (show LCS table using DP tabulation method) and the number of such longest common subsequences between A and B.

(4)

Solution

Recursive Equation

$$c[i,j] = \begin{cases} c[i-1, j-1]+1 & \text{if } x[i] = y[j], \\ \max(c[i, j-1], c[i-1, j]) & \text{otherwise} \end{cases}$$

OR

```
if $x_i == y_j$
        $c[i, j] = c[i□1, j□1] + 1$
else if $c[i□1, j] \geq c[i, j□1]$
                $c[i, j] = c[i□1, j]$
            else $c[i, j] = c[i, j□1]$
```

Marks Distribution

Recursive Equation (1 mark)

LCS table 2 marks

(5+1)

DP table for strings A = "abcbdab" and B = "bdcaba":

		b	d	c	a	b	a
	0	0	0	0	0	0	0
a	0	0	0	0	1	1	1
b	0	1	1	1	1	2	2
c	0	1	1	2	2	2	2
b	0	1	1	2	2	3	3
d	0	1	2	2	2	3	3
a	0	1	2	2	3	3	4
b	0	1	2	2	4	4	4

Length of LCS is 4. (0.5 mark)

Number of such LCS subsequence is 3 /4 (0.5 mark)

bdab -1/2

bcba-1

bcab-1

(b) Given a set of towns and each town has its own bus-stop at locations (A, B, C, E, and M). There is a school in town S which is connected to all these bus-stops via roads. The average time taken by a school bus from S to every bus-stop and between every pair of bus-stops is shown in **Fig 2.**

i) There is a school bus starting at S that needs to pick the students from all the bus-stops in minimum time such that it visits every bus-stop exactly once and returns back to the school using 2-Approximation algorithm. Show intermediate steps and the final tour with the help of schematic diagram.

ii) Out of Branch and Bound and Approximation algorithms, which one is more suitable for finding optimal tour when there are large number of towns? Justify your answer.

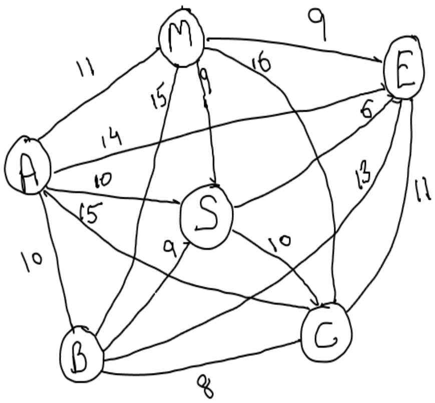


Fig 2

Marks Distribution

4 marks for correct execution of 2-Approximation Algorithm (2 marks for computing MST + 1 mark for pre-order walk + 1 mark for telling order of vertices)

MST Creation - 2 Marls
DFA Tour - 1 Marks
Optimal Solution starting from S is SMEBCAS - 1 marks

1 mark for making Final tour (showing vertices and edges) that is traversed by a bus

0.5 for choosing correct algorithm (Approximation Algorithms) and 0.5 mark for justification (approximation takes less time than BnB)

Q.5 (a) Suppose you have a text string T = "*abcabcbcbabcabcabcbc*" and a pattern string P = "*abcabcb*", and you want to find all occurrences of P in T using the KMP algorithm. (2+6)

i) Calculate the prefix function for the pattern P.

Marks Distribution
1 mark for steps
1 mark for right answer

ii) Apply KMP algorithm to determine the indices where P appears in T. Show intermediate steps. (2)

Marks Distribution
2 Marks for intermediate steps
2 Marks for correctly findling first occurrence of P
2 Marks for correctly findling second occurrence of P

Solution:

1. To calculate the prefix function for P using the KMP algorithm, we need to find the length of the longest proper prefix of P that is also a suffix of P for each position in P. The prefix function for P is:

P: a b c a b c b
P1 = {a}, P_k = {} No matching prefix and suffix of P1.
P2 = {a,b}. P_k = {}. No matching prefix and suffix of P1.
P3 = {a b c}. P_k = {}. No matching prefix and suffix of P3.
P4 = {a b c a}. P_k = {a}. k=1
P5 = {a b c a b}. P_k = {a b}. k=2
P6 = {a b c a b c}. P_k = {a b c}. k=3
P7 = {a b c a b c b}. P_k = {}. No matching prefix and suffix of P7.

P	a	b	c	a	b	c	b
K	0	0	0	1	2	3	0

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a	b	c	a	b	c	b	c	b	a	b	c	a	b	c	a	b	c	b	c

i=1, q=0 T[1] = P[1] (shift = 0), q = 1
i=2, q=1 T[2] = P[2] (shift = 0), q = 2
i=3, q=2 T[3] = P[3] (shift = 0), q = 3
i=4, q=3 T[4] = P[4] (shift = 0), q = 4
i=5, q=4 T[5] = P[5] (shift = 0), q = 5
i=6, q=5 T[6] = P[6] (shift = 0), q = 6
i=7, q=6 T[7] = P[7] (shift = 0), q = 7
q == m (m=length of P)

Pattern occurs with shift (i-m i.e 7-7 =0)

q= π[q] = π[7] =0

i=8, q=0 T[8] != P[1] (shift = 7), q = 0

i=9, q=0 T[9] != P[1] (shift = 8), q = 0

i=10, q=0 T[10] = P[1] (shift = 9), q = 1

i=11, q=1 T[11] = P[2] (shift = 9), q = 2

i=12, q=2 T[12] = P[3] (shift = 9), q = 3

i=13, q=3 T[13] = P[4] (shift = 9), q = 4

i=14, q=4 T[14] = P[5] (shift = 9), q = 5

i=15, q=5 T[15] = P[6] (shift = 9), q = 6

i=16, q=6 T[16] != P[7] (shift = 9), q = π[6] =3

i=16, q=3 T[16] = P[4] (shift = 12), q = 4

i=17, q=4 T[17] = P[5] (shift = 12), q = 5

i=18, q=5 T[18] = P[6] (shift = 12), q = 6

i=19, q=6 T[19] = P[7] (shift = 12), q = 7

q == m (m=length of P)

Pattern occurs with shift (i-m i.e 19-7 =12)

q= π[q] = π[7] =0

i=20, q=0 T[20] != P[1] (shift = 12), q = 0

Found pattern at index 0

Found pattern at index 12

(b) Solve the recurrence relation $T(n) = 4\,T(n/2) + \frac{n}{loglog\, n}$ using master method.

Marks Distribution

1 mark for steps

1 mark for correct answer (if steps are correct)

Solution: It can be solved by master theorem for dividing function:

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

a>=1, b>1 and f(n) = theta($n^k log^p n$)

if $log_b a$ > k then complexity is theta($n^{log_b a}$)

Here k is 1, p is -1, a is 4 and b is 2,

$log_b a$ = 2 which is greater than k(=1).

Hence complexity is theta(n^2).

Q.6

You are given a flow network with capacity of each edge as shown in the **Fig 3.** You need to find a min-cut and its capacity corresponding to the maximum flow in the given flow network from source **s** to destination **t** using Ford-Fulkerson algorithm. Show all the intermediate steps.

(10)

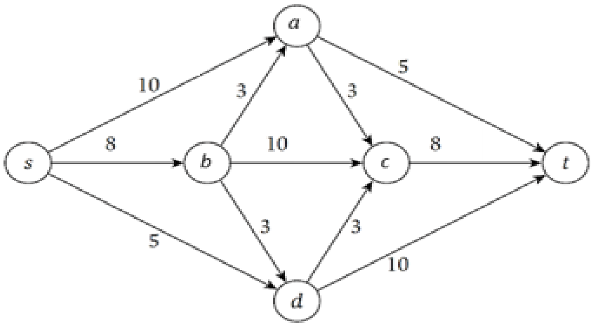
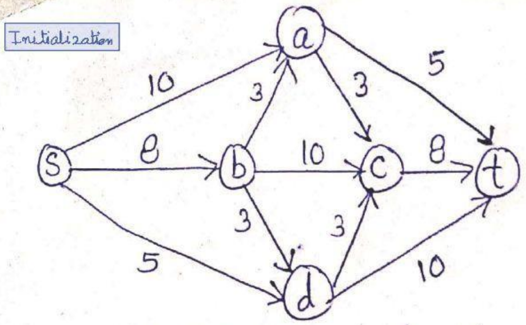
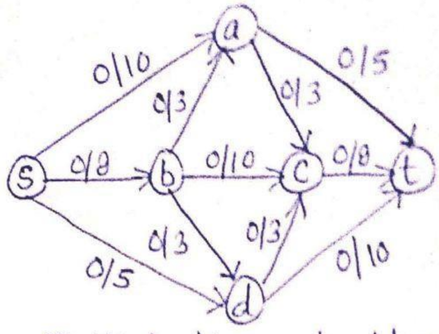


Fig 3

Initialization

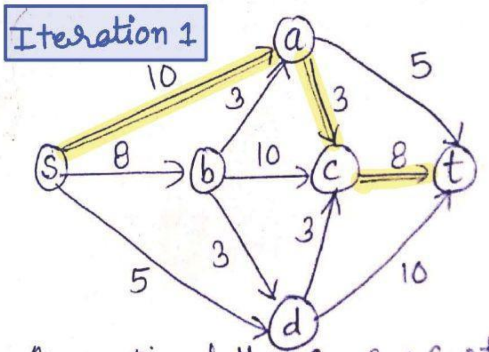


Initial flow network

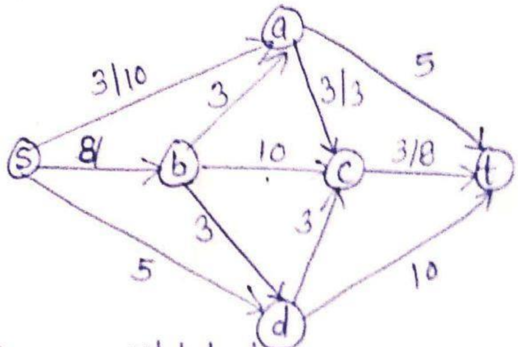


Initial flow network with flow f = 0

Iteration 1

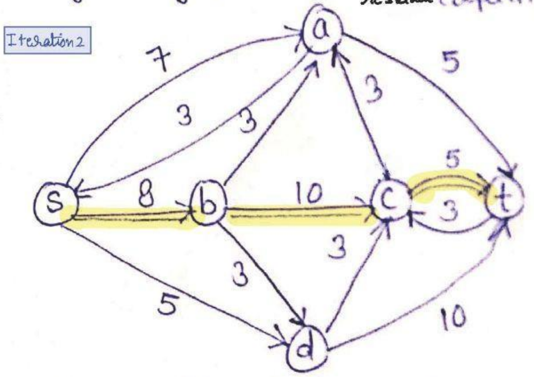


Augmenting path: s → a → c → t residual capacity: 3

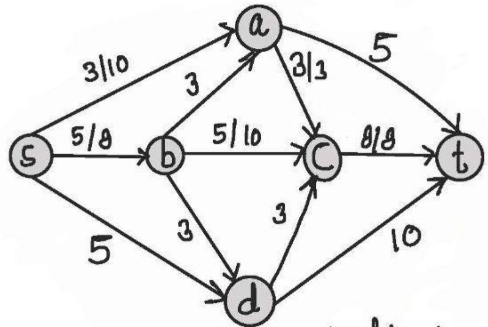


update flow

Iteration 2

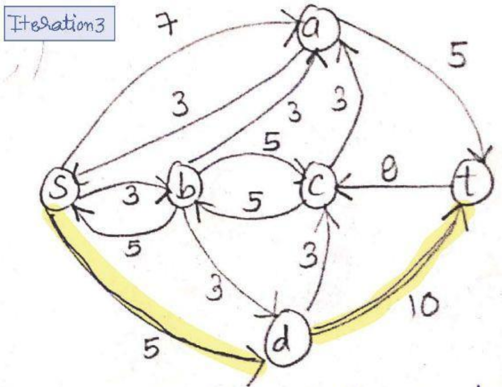


Augmenting path: s → b → c → t
residual capacity: 5

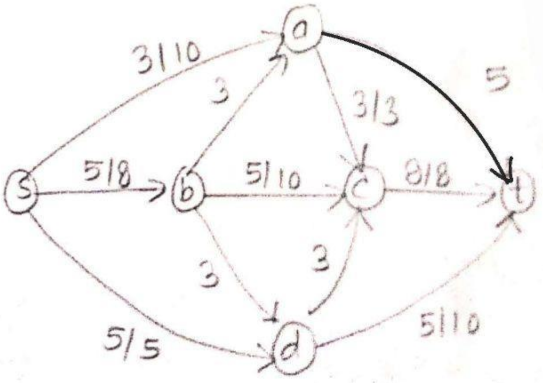


Updated flow

Iteration3

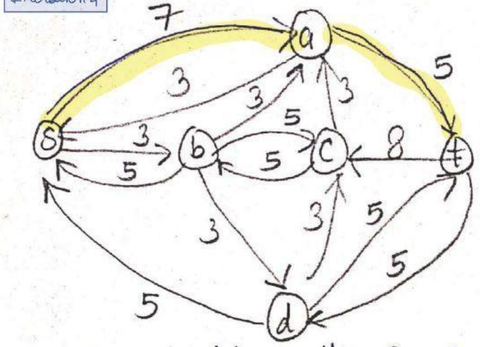


Augmenting path: s → d → t
residual capacity: 5

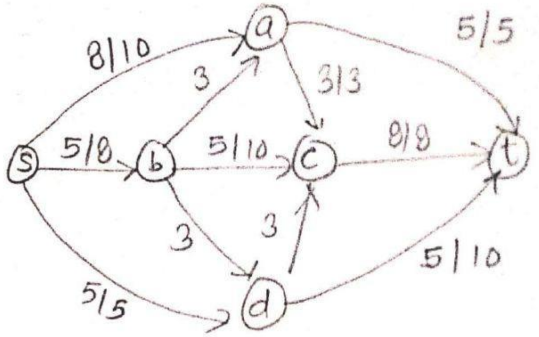


updated flow

Iteration 4

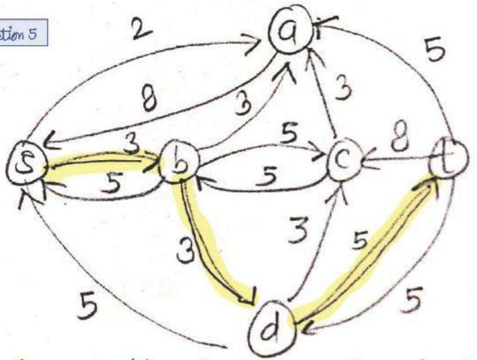


Augmenting path: S→a→t
residual capacity: 5

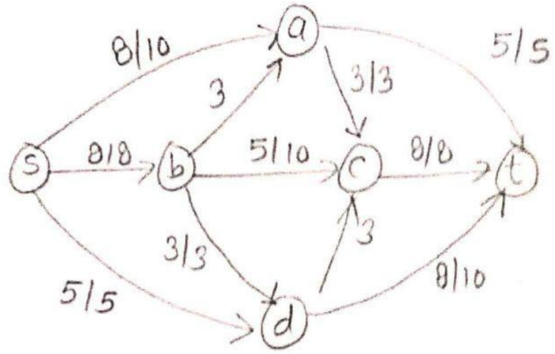


Updated flow

Iteration 5

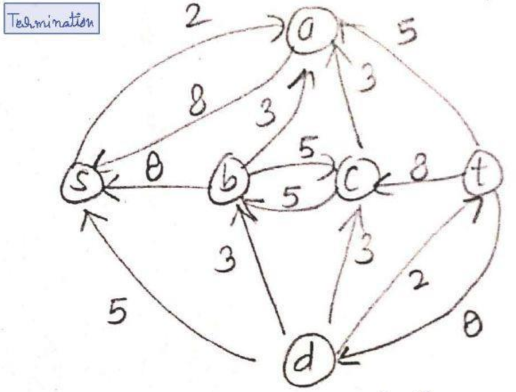


Augmenting path: S→b→d→t
residual capacity: 3

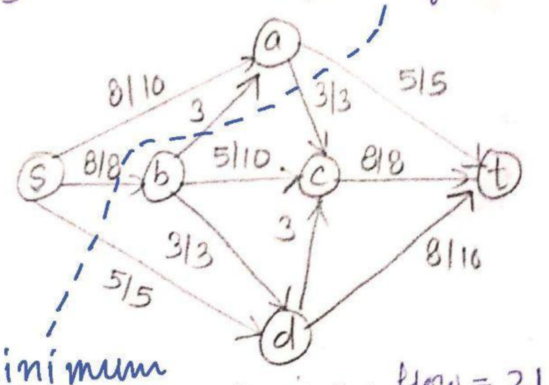


updated flow

Termination



No augmenting path



minimum cut

maximum flow = 21

Due to min-cut max-flow theorem, value of minimum-cut will also be 21.

Let min-cut = (A, B)
where A = { set of all these vertices which are reachable from vertex S (including s) in the last residual network)

B = { set of all other vertices}

A = {s, a}
B = {b, c, t, d}

Mark Distribution

7 marks for computing the correct maximum flow with all intermediate steps using the Ford-Fulkersion algorithm. (**Note: Marks have been deducted for not showing the residual graphs at each iterations. And, in case of incorrect final answers, 0-3 marks have been awarded for intermediate steps)**
2 marks for computing the minimum cut
1 mark for the value of minimum cut