Page No.
Date

Q.1(a)

Greedy approach	Dynamic programming
(1) Generates a single decision sequence	Many decision sequence may be generated.
(2) It is less reliable	It is highly reliable
(3) It follows top-down approach	It follows bottom up approach
(4) efficiency is more	efficiency is less
(5) Overlapping subproblems cannot be handled	chooses the optimal soln to subproblem.
(6) contain a particular set of feasible set of soln	There is no special set
(7) It is easier to engage in greedy procedure than to use Dijkstra shortest path algo.	Bellman ford algo. which is based on dynamic programming
(8) Fractional knapsack is an example of greedy algo.	0/1 knapsack problem is an example of
(9) No memorization is required	memorization is required
(10) Faster than dynamic programing	It is slower than greedy approach
(11) Each step is locally optimal	past soln are used to create new ones.

Page No.
Date

Q.1(b) Determine LCS of $x = (a, b, a, b, a, a, b)$ and $y = (a, b, a, b, b, a, a)$

→

		0	1	2	3	4	5	6	7
	x_i	a	b	a	b	a	a	b	
0	y_i	0	0	0	0	0	0	0	0
1	a	0	↖ (1)	← 1	↖ 1	← 1	↖ 1	↖ 1	← 1
2	b	0	↑ 1	↖ (2)	← 2	↖ 2	← 2	← 2	↖ 2
3	a	0	↖ 1	↑ 2	↖ (3)	← 3	↖ 3	↖ 3	← 3
4	b	0	↑ 1	↑ 2	↑ (3)	↖ 4	← 4	← 4	↖ 4
5	b	0	↑ 1	↖ 2	↑ 3	↖ (4)	↑ 4	↑ 4	↖ 5
6	a	0	↖ 1	↑ 2	↖ 3	↑ 4	↖ (5)	↖ 5	↑ 5
7	a	0	↖ 1	↑ 2	↖ 3	↑ 4	↖ 5	↖ (6)	← 6

LCS $(x, y) = (a, b, a, b, a, a)$

Date

2a) Find the order of parenthesization for the optimal chain multiplication where sequence of dimension d = (15, 5, 10, 20, 25)

→ Step 1: Compute first diagonal of matrix m

Given that product vector P is

P = 15 5 10 20 25

P_0 P_1 P_2 P_3 P_4

Therefore, matrices and their orders are as follows:

$A_1 = 15 \times 5$

$A_2 = 5 \times 10$

$A_3 = 10 \times 20$

$A_4 = 20 \times 25$

For first diagonal:

$m[i,j] = 0$ if $i = j$

$m[1,1] = 0$

$m[2,2] = 0$

$m[3,3] = 0$

$m[4,4] = 0$

Step 2:

For 2nd diagonal: (matrix m 45)

$m[i,j] = m[i,k] + m[k+1,j] + P_{i-1} P_k P_j$ for $i \leq k < j$

For m[1,2] only one value of k is possible: $1 \leq 1 < 2$

$\therefore m[1,2] = m[1,1] + m[2,2] + P_0 P_1 P_2$

$= 0 + 0 + 15 \times 5 \times 10$

$= 750$ for $k = 1$

$s[1,2] = 1$

Page No.
Date

For m[2,3] only one value of k is possible: $2 \le 2 < 3$

$\therefore m[2,3] = m[2,2] + m[3,3] + p_1 p_2 p_3$

$= 0 + 0 + 5 \times 10 \times 20$

$= 1000$ for $k = 2$

$s[2,3] = 2$

For m[3,4] only one value of k is possible: $3 \le 3 < 4$

$\therefore m[3,4] = m[3,3] + m[4,4] + p_2 p_3 p_4$

$= 0 + 0 + 10 \times 20 \times 25$

$= 5000$ for $k = 3$

$s[3,4] = 3$

Step 3:

For 3rd diagonal (matrix m 4s):

$m[i,j] = \min(m(i,k) + m[k+1,j] + p_{i-1} p_k p_j)$ for $i \le k < j$

For m[1,3] two values of k are possible as $k = 1, 2$ due to $1 \le 1, 2 < 3$

When $k = 1$:

$m[1,3] = m[1,1] + m[2,3] + p_0 p_1 p_3$

$= 0 + 1000 + 15 \times 5 \times 20$

$= 2500$ for $k = 1$

When $k = 2$:

$m[1,3] = m[1,2] + m[3,3] + p_0 p_2 p_3$

$= 750 + 0 + 15 \times 10 \times 20$

$= 3750$ for $k = 2$

Date

Now we have $m[1,3] = 2500$ for $k=1$

$m[1,3] = 3750$ for $k=2$

$\therefore m[1,3] = \min(2500, 3750)$

$= 2500$ for $k=1$

$\therefore s[1,3] = 1$

For $m[2,4]$ two value of k are possible as $k=2,3$ due to $2 \leq 2, 3 < 4$

when $k=2$

$m[2,4] = m[2,2] + m[3,4] + p_1 p_2 p_4$

$= 0 + 5000 + 5 \times 10 \times 25$

$= 6250$ for $k=2$

when $k=3$

$m[2,4] = m[2,3] + m[4,4] + p_1 p_3 p_4$

$= 1000 + 0 + 5 \times 20 \times 25$

$= 3500$ for $k=3$

$\therefore m[2,4] = \min(6250, 3500)$

$= 3500$ for $k=3$

$\therefore s[2,4] = 3$

Step 4:

For 4th diagonal

$m[i,j] = m[i,k] + m[k+1,j] + p_{i-1} p_k p_j$ for $i \leq k < j$

for $[1,4]$ three value of $k = 1, 2, 3$

Page No.
Date

When $k=1$:

$m[1,4] = m[1,1] + m[2,4] + p_0 p_1 p_4$

$= 0 + 3500 + 15 \times 5 \times 25$

$= 5375$ for $k=1$

When $k=2$:

$m[1,4] = m[1,2] + m[3,4] + p_0 p_2 p_4$

$= 750 + 5000 + 15 \times 10 \times 25$

$= 9500$ for $k=2$

When $k=3$,

$m[1,4] = m[1,3] + [4,4] + p_0 p_3 p_4$

$= 2500 + 0 + 15 \times 20 \times 25$

$= 10,000$ for $k=3$

Now,

$m[1,4] = \min [5375, 9500, 10000)$

$= 5375$ for $k=1$

$s[1,4] = 1$

Step 5 :

Optimal No. of multiplication $= m[1,n]$

$n \leftarrow$ length $(p) - 1$

$n \leftarrow 5 - 1$

$n = 4$

The optimal no. of multiplication $= m[1,4]$

$= 5375$

As we have only 4 matrix A_1, A_2, A_3, A_4

1	2	3	4	
0	750	2500	5375	1
	0	1000	3500	2
		0	5000	3
			0	4

m(i,j)

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

s(i,j)

Step 6:

Optimal parenthization

we have four matrix, first parentization will be as follows: (A_1, A_2, A_3, A_4)

check for breakage from lower limit to upper limit that is from 1 to 4

check for s[1,4]

s[1,4] = 1

Therefore,

$(A_1 (A_2, A_3 A_4))$ ... Optimal Parenthization

2b) What is purpose of matrix chain multiplication algo.?

→ ① The matrix chain multiplication problem generalizes to solve a more abstract problem.

② It is a method under dynamic programming in which previous output is taken as input for next.

③ We have to perform the matrix multiplication, which can be accomplished by a series of matrix multiplication.

④ matrix chain multiplication is an optimization problem concerning the most efficient way to multiply a given sequence of matrices.

⑤ Dynamic programming, Brute force, Recursion method can be used to solve the matrix chain multiplication problem.

⑥ For ex.

if the chain of matrices is (A_1, A_2, A_3, A_4) then we can fully parenthesize the product $A_1A_2A_3A_4$ in five distinct way:

a) $(A_1(A_2(A_3A_4)))$

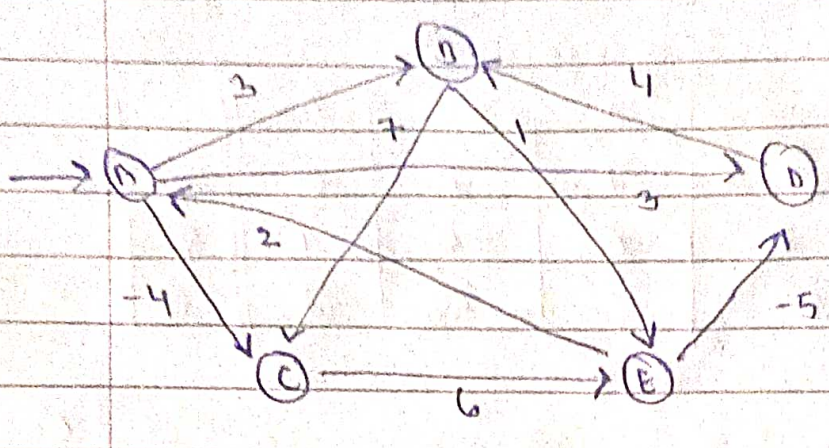
b) $(A_1(A_2A_3)A_4))$

c) $((A_1A_2)(A_3A_4))$

d) $((A_1(A_2A_3))A_4)$

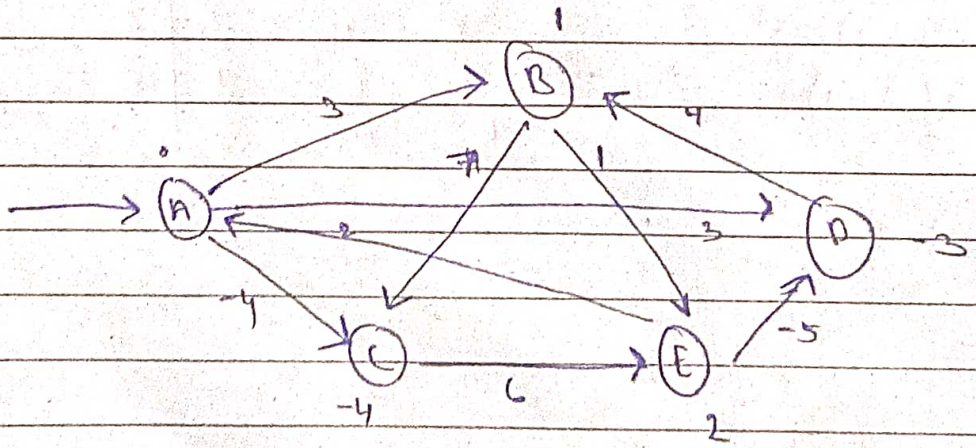
e) $(((A_1A_2)A_3)A_4)$

3b) find the shortest distance using Bellman-Ford algo.



edges : (A,B) (B,C) (D,B)
(A,D) (B,E) (E,A)
(A,C) (C,E) (E,D)

	A	B	C	D	E
0	∞	∞	∞	∞	∞
1	0	3	-4	3	∞
2	0	3	-4	-3	2
3	0	1	-4	-3	2
4	0	1	-4	-3	2



Page No.
Date

we have distance Vector as

A	B	C	D	E
0	-1	-4	-3	2

Shortest path from vertex A to E

$\therefore A \rightarrow C \rightarrow E$

and

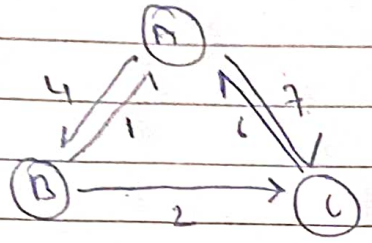
cost = d[E]

= 2

4a) Find all pairs Shortest paths using floyd warshall algo. for given graph

	A	B	C
A	0	4	7
B	1	0	2
C	6	10	0

→



$D^{(0)}$

	A	B	C
A	0	4	7
B	1	0	2
C	6	10	0

$\Pi^{(0)}$ =

	A	B	C
A	N	A	A
B	B	N	B
C	C	N	N

$D^{(1)}$ =

	A	B	C
A	0	4	7
B	1	0	2
C	6	10	0

$\Pi^{(1)}$ =

	A	B	C
A	N	A	A
B	B	N	B
C	C	A	N

$D^{(2)}$ =

	A	B	C
A	0	4	6
B	1	0	2
C	6	10	0

$\Pi^{(2)}$ =

	A	B	C
A	N	A	B
B	B	N	B
C	0	A	N

$D^{(2)} =$

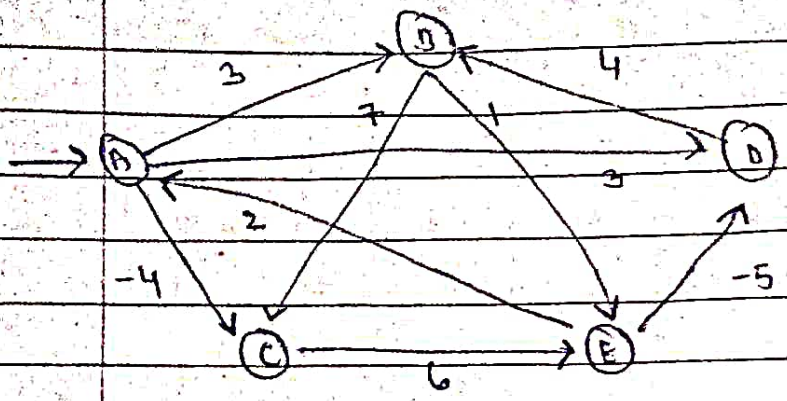
	A	B	C
A	0	4	6
B	1	0	2
C	6	10	0

$\Pi^{(2)} =$

	A	B	C
A	M	A	B
B	B	M	B
C	C	A	M

Page No.
Date

3b) Find the shortest distance using Bellman-Ford algo.



① Initialization:

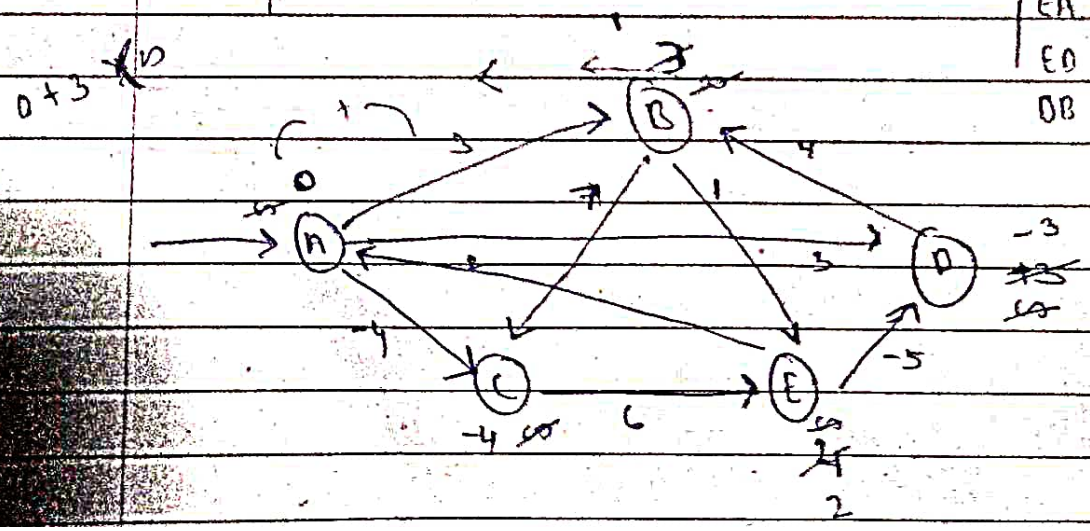
for each vertex $v \in V[G]$
do $d(v) \leftarrow \infty$
$\pi(v) \leftarrow NIL$
$d(s) \leftarrow 0$

edges: (A,B) (B,C) (D,B)
(A,D) (B,E) (E,A)
(A,C) (C,E) (E,D)

	A	B	C	D	E
0	∞	∞	∞	∞	∞
1	0	3	-4	3	∞
2	0	3	-4	-3	2
3	0	1	-4	-3	2
4	0	1	-4	-3	2

OR

Edge	i=1	i=2	i=3	i=4
AB	✓	X	q	q
AC	✓	q	q	q
AD	✓	q	q	q
BC	X	q	q	q
BE	✓	q	q	q
CE	✓	q	q	q
EA	X	q	q	q
ED	✓	X	q	q
DB	✓	q	q	q



we have distance Vector as

A	B	C	D	E
0	-1	-4	-3	2

Shortest path from vertex A to E

$\therefore A \rightarrow C \rightarrow E$

and

cost = d[E]

= 2