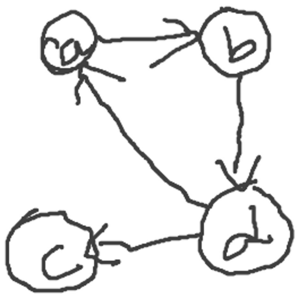
ALGORITHMS

Apply Warshall Algorithm to compute the transitive closure for the graph shown below



The adjacency matrix for the given graph is shown below R(0)

	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0
d	1	0	1	0

In the above table since (d,a)=1 and (a,b)=1 so make the entry (d,b)=1

The resulting table R(1)is shown below.

	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0
d	1	1	1	0

Now consider path through vertex b

	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0

d	1	1	1	0

Since (a,b)=1 and (b,d)=1 we make (a,d)=1

Since(d,b)=1 and (b,d)=1 we makke (d,d)=1

The resulting table R(2) is shown below

	a	b	c	d
a	0	1	0	1
b	0	0	0	1
c	0	0	0	0
d	1	1	1	1

Now consider path through vertex c

	a	b	c	d
a	0	1	0	1
b	0	0	0	1
c	0	0	0	0
d	1	1	1	1

There is no edge which originates at c so table R(3) remains unaltered and shown below

	a	b	c	d
a	0	1	0	1
b	0	0	0	1
c	0	0	0	0
d	1	1	1	1

Now consider path through vertex d

	a	b	c	d
a	0	1	0	1
b	0	0	0	1
c	0	0	0	0
d	1	1	1	1

Since (a,d)=1 and (d,a)=1. So, (a,a)=1

Since (a,d)=1 and (d,b)=1. So, (a,b)=1

Since (a,d)=1 and (d,c)=1. So, (a,c)=1

Since (a,d)=1 and (d,d)=1. So, (a,d)=1

Since (b,d)=1 and (d,a)=1. So, (b,a)=1

Since (b,d)=1 and (d,b)=1. So, (b,b)=1

Since (b,d)=1 and (d,c)=1. So, (b,c)=1

Since (b,d)=1 and (d,d)=1. So, (b,d)=1

The resultant table R(4) is shown below

	a	**b**	**c**	**d**
a	**1**	**1**	**1**	**1**
b	**1**	**1**	**1**	**1**
c	**0**	**0**	**0**	**0**
d	**1**	**1**	**1**	**1**

BOYER MOORE ALGORITHM

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	_
1	2	6	6	6	6	6	6	6	6	6	6	6	6	3	6	6	6	6	6	6	6	6	6	6	6	6

K	Pattern	d2
1	BAO**B**A**B**	**2**
2	**B**AOB**AB**	**5**
3	**B**AO**BAB**	**5**
4	**B**A**OBA**	**5**
5	**BAOBAB**	**5**

B	E	S	S		K	N	E	W		A	B	O	U	T		B	A	O	B	A	B	S	
B	A	O	B	A	B																		
						B	A	O	B	A	B												
											B	A	O	B	A	B							
																B	A	O	B	A	B		

FLOYD ALGORITHM

ALGORITHM Floyd(W[1..n, 1..n])

//Implements Floyd's algorithm for the all-pairs shortest-paths problem

//Input: The weight matrix W of a graph with no negative-length cycle

//Output: The distance matrix of the shortest paths' lengths

D ←W //is not necessary if W can be overwritten

for k←1 to n do

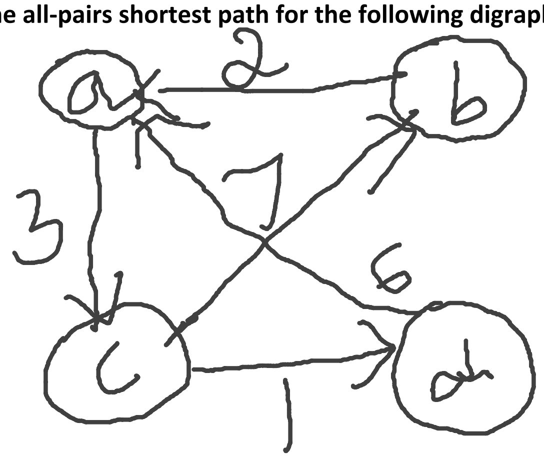
for i ←1 to n do

for j ←1 to n do

D[i, j]←min{D[i, j], D[i, k]+ D[k, j]}

return D

Find the all-pairs shortest path for the following digraph



Cost adjacency matrix of above digraph is shown below

	a	b	c	d
a	0	∞	3	∞
b	2	0	∞	∞
c	∞	7	0	1
d	6	∞	∞	0

Step1: Consider the shortest distance through vertex a

	a	b	c	d

a	0	∞	3	∞
b	2	0	∞	∞
c	∞	7	0	1
d	6	∞	∞	0

(b,a)=2 and (a,c)=3 so (b,c)=min{(b,c),(b,a)+(a,c)}
=min{∞,2+3}
=5

(d,a)=6 and (a,c)=3 so (d,c)=min{(d,c),(d,a)+(a,c)}
=min{∞,6+3}
=9

The resultant matrix is shown below

	a	b	c	d
a	0	∞	3	∞
b	2	0	5	∞
c	∞	7	0	1
d	6	∞	9	0

Step2: Consider the shortest distance through vertex b

	a	b	c	d
a	0	∞	3	∞
b	2	0	5	∞
c	∞	7	0	1
d	6	∞	9	0

(c,b)=7 and (b,a)=2 so (c,a)=min{(c,a),(c,b)+(b,a)}
=min{∞,7+2}
=9

(c,b)=7 and (b,c)=5 so (c,c)=min{(c,c),(c,b)+(b,c)}
=min{0,7+5}
=0

The resultant matrix is shown below

	a	b	c	d
a	0	∞	3	∞
b	2	0	5	∞
c	9	7	0	1
d	6	∞	9	0

Step3: Consider the shortest distance through vertex c

	a	b	c	d
a	0	∞	3	∞
b	2	0	5	∞
c	9	7	0	1
d	6	∞	9	0

(a,c)=3 and (c,a)=9 so (a,a)=min{(a,a),(a,c)+(c,a)}
=min{0,3+9}
=0

(a,c)=3 and (c,b)=7 so (a,b)=min{(a,b),(a,c)+(c,b)}
=min{∞,3+7}
=10

(a,c)=3 and (c,d)=1 so (a,d)=min{(a,d),(a,c)+(c,d)}
=min{∞,3+1}
=4

(b,c)=5 and (c,a)=9 so (b,a)=min{(b,a),(b,c)+(c,a)}
=min{2,5+9}
=2

(b,c)=5 and (c,b)=7 so (b,b)=min{(b,b),(b,c)+(c,b)}
=min{0,5+7}
=0

(b,c)=5 and (c,d)=1 so (b,d)=min{(b,d),(b,c)+(c,d)}
=min{∞,5+1}
=6

(d,c)=9 and (c,a)=9 so (d,a)=min{(d,a),(d,c)+(c,a)}
=min{6,9+9}
=6

(d,c)=9 and (c,b)=7 so (d,b)=min{(d,b),(d,c)+(c,b)}
=min{∞,9+7}
=16

(d,c)=9 and (c,d)=1 so (d,d)=min{(d,d),(d,c)+(c,d)}
=min{0,9+1}
=0

The resultant matrix is shown below

	a	b	c	d
a	0	10	3	4
b	2	0	5	6
c	9	7	0	1
d	6	16	9	0

Step 4: Consider shortest distance through vertex d

	a	b	c	d
a	0	10	3	4
b	2	0	5	6
c	9	7	0	1
d	6	16	9	0

(a,d)=4 and (d,a)=6 so (a,a)=min{(a,a),(a,d)+(d,a)}
=min{0,4+6}
=0

(a,d)=4 and (d,b)=16 so (a,b)=min{(a,b),(a,d)+(d,b)}
=min{10,4+16}
=10

(a,d)=4 and (d,c)=9 so (a,c)=min{(a,c),(a,d)+(d,c)}
=min{3,4+9}
=3

(b,d)=6 and (d,a)=6 so (b,a)=min{(b,a),(b,d)+(d,a)}
=min{2,6+6}
=2

(b,d)=6 and (d,b)=16 so (b,b)=min{(b,b),(b,d)+(d,b)}
=min{0,6+16}
=0

(b,d)=6 and (d,c)=9 so (b,c)=min{(b,c),(b,d)+(d,c)}
=min{5,6+9}
=5

(c,d)=1 and (d,a)=6 so (c,a)=min{(c,a),(c,d)+(d,a)}
=min{9,1+6}
=7

(c,d)=1 and (d,b)=16 so (c,b)=min{(c,b),(c,d)+(d,b)}

=min{7,1+16}
=7

(c,d)=1 and (d,c)=9 so (c,c)=min{(c,c),(c,d)+(d,c)}
=min{0,1+9}
=0

The final matrix is

	a	b	c	d
a	0	10	3	4
b	2	0	5	6
c	7	7	0	1
d	6	16	9	0

KNAPSACK PROBLEM-BOTTOM UP DYNAMIC PROGRAMMING TECHNIQUE

$$F(i,j) = \begin{cases} \max(F(i-1,j), V_i + F(i-1, j-w_i)) & \text{if } j-w_i \geq 0 \\ F(i-1,j) & \text{if } j-w_i < 0 \end{cases}$$

Initial conditions

$$F(0,j) = 0 \text{ for } j \geq 0$$

$$F(i,0) = 0 \text{ for } i \geq 0$$

item	Weight	value
1	2	12
2	1	10
3	3	20
4	2	15

W=5 Knapsack Capacity

	0	1	2	3	4	5
0						
1						
2						
3						
4						

STEP1

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

STEP2

When i=1 weight(i)=2 and value(i)=12
We have to compute the results for various values of j ->(1,2,3,4,5)
F(1,1)=F(0,1)=0 because j<weight(i)
F(1,2)=max(F(0,2),F(0,0)+12)=max(0,0+12)=12
F(1,3)=max(F(0,3),F(0,1)+12)=max(0,0+12)=12
F(1,4)=max(F(0,4),F(0,2)+12)=max(0,0+12)=12
F(1,5)=max(F(0,5),F(0,3)+12)=max(0,0+12)=12

The resultant table is

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0					
3	0					
4	0					

Step 3: when i=2 weight(i)=1 value(i)=10

We have to compute the results for various values of j ->(1,2,3,4,5)
F(2,1)=max(F(1,1),F(1,0)+10)=max(0,0+10)=10
F(2,2)=max(F(1,2),F(1,1)+10)=max(12,0+10)=12
F(2,3)=max(F(1,3),F(1,2)+10)=max(12,12+10)=22
F(2,4)=max(F(1,4),F(1,3)+10)=max(12,12+10)=22
F(2,5)=max(F(1,5),F(1,4)+10)=max(12,12+10)=22

The resultant table after Step 3 is

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12	22	22	22
3	0					
4	0					

Step 4:
When i=3 weight(i)=3 and value(i)=20
We have to compute the results for various values of j ->(1,2,3,4,5)
F(3,1)=F(2,1)=10 because j<weight(i)
F(3,2)=F(2,2)=12 because j<weight(i)
F(3,3)=max(F(2,3),F(2,0)+20)=max(22,0+20)=22
F(3,4)=max(F(2,4),F(2,1)+20)=max(22,10+20)=30
F(3,5)=max(F(2,5),F(2,2)+20)=max(22,12+20)=32

The resultant table after Step 4 is

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12	22	22	22

3	0	10	12	22	30	32
4	0					

Step 5:
When i=4 weight(i)=2 and value(i)=15
We have to compute the results for various values of j ->(1,2,3,4,5)
F(4,1)=F(3,1)=10 because j<weight(i)
F(4,2)=max(F(3,2),F(3,0)+15)=max(12,0+15)=15
F(4,3)=max(F(3,3),F(3,1)+15)=max(22,10+15)=25
F(4,4)=max(F(3,4),F(3,2)+15)=max(30,12+15)=30
F(4,5)=max(F(3,5),F(3,3)+15)=max(32,22+15)=37

The final table is

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	12	12	12	12
2	0	10	12	22	22	22
3	0	10	12	22	30	32
4	0	10	15	25	30	37

Since F(4,5)>F(3,5) item 4 has to be included in the Knapsack.
Remaining capacity of Knapsack is 5-2=3
Since F(3,3)=F(2,3) item 3 need not be in the optimal subset.
Since F(2,3)>F(1,3) item 2 is included in the Knapsack
Remaining Capacity of Knapsack is 3-1=2
Since F(1,2)>F(0,2) item 1 is in the Knapack

Thus optimal solution is {item1, item2, item4} with value 37.

REFERENCE: ANANY LEVITIN "INTRODUCTION TO DESIGN AND ANALYSIS OF ALGORITHMS" 3RD EDITION, PEARSON PUBLISHER