Dynamic Programming

Dynamic Programming

Topics

- *Introduction to Dynamic Programming*
- *Optimality Principle*
- *Applications of DP*
- *All Pairs Shortest Paths Problem*

Dynamic Programming

Strategy

Dynamic Programming is a technique for solving *optimization problems*. Generally, such problems have many *feasible solutions,* each having some *associated cost*. The dynamic programming method determines the best solution, that is, the one which has the *maximum* or *minimum cost*.

The algorithm works by splitting a problem into a sequence of optimization subproblems. The subproblems are solved, and then combined to obtain solution to the main problem.

The dynamic programming appears to have same approach as divide-and-conquer algorithm. There are, however, some subtle differences. Unlike the divide-and-conquer method, the dynamic programming splits the problem into *overlapping* subproblems , which are solved in *bottom up* fashion. The optimal solutions to subproblems are stored into a table, which is used repeatedly to build the final or *global optimal* solution.

Dynamic Programming

Solution

In general, solution of problem is obtained by using the dynamic programming algorithm in the following manner:

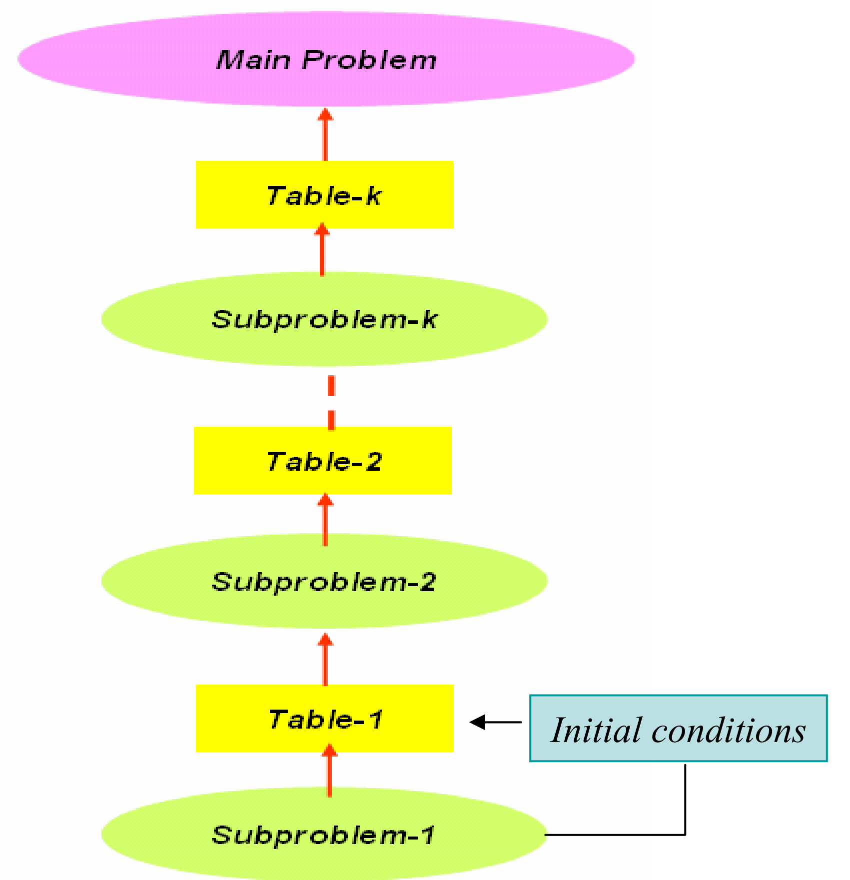
1) A *recursion* is set up to express the running time of the main problem in terms running times of overlapping subproblems

2) The recursion is solved in *bottom-up manner*, by using the *initial conditions* and *previously computed values*. The results are saved in a table

3) The *tabular solutions* to sub-problems are used to construct the solution to the main solution

Dynamic Programming
Building Solution

The steps involved in building a DP solution are depicted in the diagram. First, solution to the smallest subproblem is obtained using the *initial conditions*. The result is saved in a *table*. Using this table, the optimum solution is obtained for the *next level of subproblem*. This procedure is continued until the solution to the main problem is found.



Dynamic Programming procedure

Optimality Principle

Dynamic Programming

Optimality Principle

The dynamic programming approach is based on the principle of optimality. The principle states: "*Optimal solution to a problem can be found, if there exists an optimal solution to a sub-problem*"

Dynamic Programming solution is feasible when *principle of optimality holds true for a given problem.*

The principle may not be applicable to all kinds of optimization problems. This condition places a restriction on the wider applicability of dynamic programming algorithm.

Optimality Principle

Shortest Path

- The principle of optimality holds true for *shortest paths* in a graph

Example: Consider the weighted graph shown in the diagram

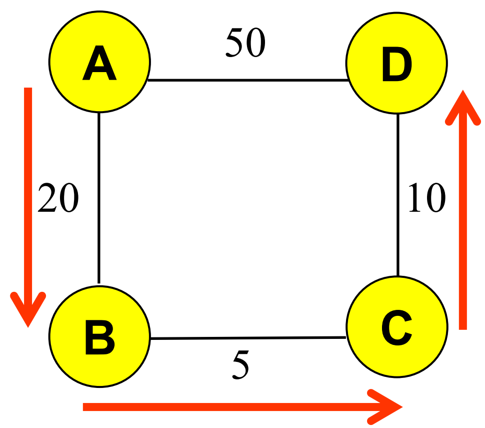
It can be seen that the shortest distance the between the vertices A and D is 35

(i) The shortest path from A to D is $A \to B \to C \to D = 20+5+10=35$

(ii) The shortest path from A to C is $A \to B \to C=20+5=25$

(iii) The shortest path from A to B is $A \to B=20$.

It follows that each of the sub-paths from vertex A to vertex D is also the shortest path between intermediate vertices. Thus, the principle of optimality holds true for shortest distances in the sample graph.



Sample weighted graph. Shortest paths are shown with bold lines

Optimality Principle
Longest Path

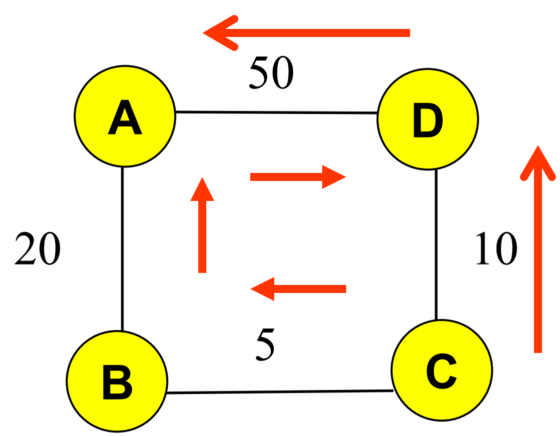
- The principle of optimality does not hold true for ***longest paths in a graph***

Example: Consider the sample weighted graph shown in the diagram

The *longest path* from *C* to *A,* is

$$C \rightarrow D \rightarrow A = 10 + 50 = 60$$

The sub-path $C \rightarrow D = 10$ is *not the longest path.* In fact, the longest path simple path from *C to D* consists of sub-paths $C \rightarrow B \rightarrow A \rightarrow D = 75$



Sample weighted graph. The longest paths are shown with bold lines

Dynamic Programming

Applications

The DP has several applications. Some common useful applications are:

- *Shortest distances in a network*
- *Longest common subsequences in strings*
- *Optimal matrix chain multiplication*
- *Optimal triangulation of a polygon*
- *Optimized binary search trees*
- *Optimal Packaging of Bins*
- *Optimal Assembly line scheduling*

All-Pairs Shortest Paths

All Pairs Shortest Paths

Approaches

In networking problems it is often required to find the shortest distances between all pairs of vertices. There are different algorithms to solve this problem. Some of the methods are as follows.

1) A simple *Brute Force method* can be used to compute all possible distances between different vertices, and then select the smallest computed distance for each pair. This method is not feasible for a large network, because the running time grows exponentially with the size of the graph

2) An efficient algorithm to determines shortest distances from a *start* vertex to all other vertices in a weighted directed or undirected graph, was proposed by *Dijkstra.* The working of *Dijkstra's* algorithm is similar to that of *Prim's*. Like *Prim's* algorithm, it systematically computes the aggregate shortest distances by considering links between two sets of vertices . The *Dijkstra's* algorithm has time complexity of $O(n^2)$ for graph with n vertices. It can be applied to solve the problem of all-pairs shortest distances, by selecting each of the vertices as the start vertex. In this way, shortest distances for the entire graph can be obtained in time $O(n^3)$

3) Another algorithm, known as *Floyd-Warshall* algorithm, provides shortest distances between *all pairs of vertices*. It is based on dynamic programming technique. The algorithm runs in time $O(n^3)$.

Weight Matrix

Definition

The *Floyd-Warshall* algorithm uses a *weight matrix* to compute the intermediate shortest paths
Let $G=(V,E)$ be a weighted graph, with weight $w(i, j)$ associated with the edge (v_i, v_j) . The weight matrix, $W = [\, w_{ij} \,]$ is defined as follows

$$w_{ij} = \begin{cases} 0 \quad \text{if } i = j \quad (i,\ j = 1 \ldots n = |V| \quad) \\ \infty \quad \text{if} \quad (v_i, v_j) \notin E \quad (\text{No link between the pair of vertices}) \\ w(i, j) \text{ if } (v_i, v_j) \in E \quad (\text{Vertices are linked}) \end{cases}$$

➢ The infinity symbol ' ∞' represents missing links in the graph. In actual implementation it may be replaced by some very large number.

Floyd Warshall Algorithm

All-Pairs Shortest Distances

The *Floyd Warshall* algorithm for computing the shortest distances consists of the following steps:

Step #1:Store the weight matrix W into the table $D^{(0)}$. Initialize a table P to store intermediate vertices along the shortest paths between all pair of vertices

Step #2: Using table $D^{(0)}$ compute shortest distances by considering direct paths as well as all other paths that pass through the vertex v_1. Store the shortest distances in table $D^{(1)}$. If the shortest path between vertices v_i and v_j passes through the vertex v_k, then store the intermediate vertex v_k in table P

Step #3: Using table $D^{(1)}$, compute shortest distances between all pairs of vertices, by considering direct paths as well as indirect paths through the vertex v_2. Store these distances in table $D^{(2)}$. By principle of optimality, the table $D^{(2)}$ would contain shortest paths that may be direct or pass through vertices v_1, v_2. If the shortest path between vertices v_i and v_j passes through the vertex v_k, then store the intermediate vertex v_k in table P

Step #4: Repeat the above steps to compute the tables $D^{(3)}$, $D^{(4)}$……$D^{(n)}$. At each stage update the table P by copying the intermediate vertex.

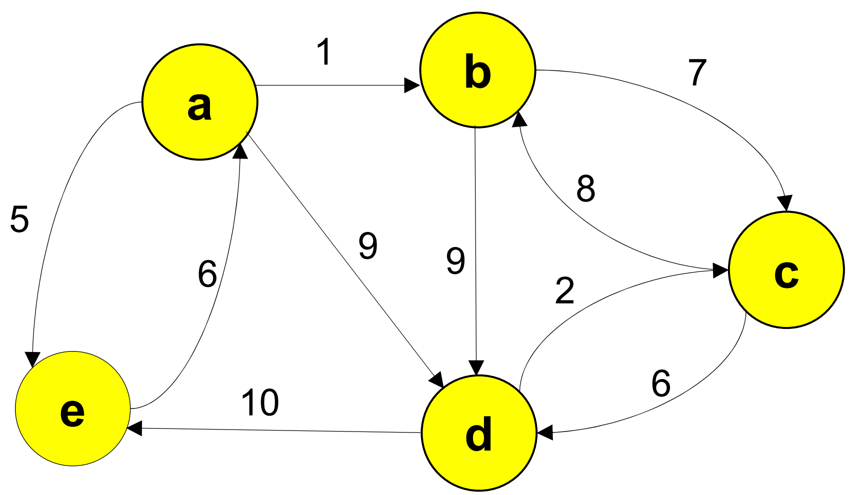
➢ At the end the table $D^{(n)}$ would contain shortest distances between *all* pairs of vertices. The table *P* would contain the intermediate vertices that link the last segments of the shortest paths to the end vertices.

All-Pairs Shortest Paths
Example

Floyd-Warshall Algorithm

Example

Consider the sample weighted graph depicted in figure *(i)*. The weight matrix for the graph is shown in figure *(ii)*. The missing link between a pair of vertices are indicated by the infinity symbol ∞.



(i) Sample weighted graph

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

(ii) Weight Matrix for the sample graph

➢ The use of Floyd-Warshall algorithm is demonstrated in the following diagrams..

Floyd-Warshall Algorithm

Distances from vertex *a* to other vertices, using table $D^{(0)}$

(1) Distance from *a to b*

Direct : $a \rightarrow b=1$

Indirect: $a \rightarrow a \rightarrow b$

$= 0 + 1 = 1$

$D^{(0)}$

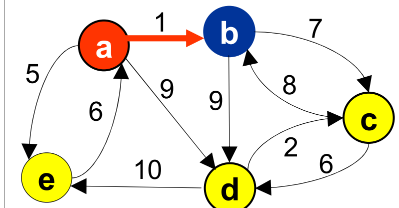
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

$D^{(1)}$

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

P

	a	b	c	d	e
a	-	-			
b					
c					
d					
e					



(2) Distance from *a* to *c*

Direct: $a \rightarrow c = \infty$

Indirect: $a \rightarrow a \rightarrow c$

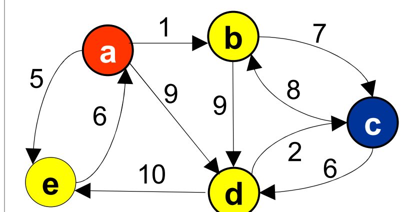
$= 0 + \infty = \infty$

No link, direct or through *a*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞		
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	-		
b					
c					
d					
e					



(3) Distance from *a* to *d*

Direct: $a \rightarrow d=9$

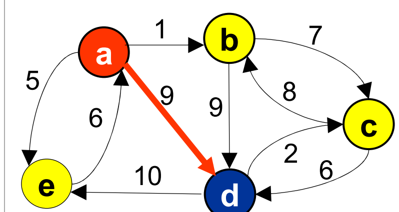
Indirect: $a \rightarrow a \rightarrow d$

$= 0 + 0 = 9$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	-	-	
b					
c					
d					
e					



(4) Distance from *a* to *e*

Direct: $a \rightarrow e=5$

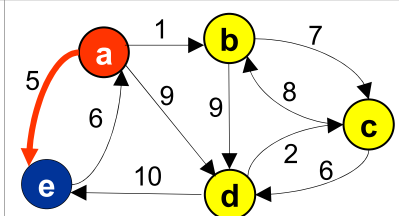
Indirect: $a \rightarrow a \rightarrow e$

$= 0 + 5 = 5$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	-	-	-
b					
c					
d					
e					



Start vertex End vertex

Distances

Direct Indirect

$D^{(0)}$: Distances matrix

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *b* to other vertices, *direct* or *via* vertex *a* , using table $D^{(0)}$

(5) Distance from *b* to *a*

Direct : $b \rightarrow a = \infty$

Indirect: $b \rightarrow a \rightarrow a$

$= \infty + 0 = \infty$

No link, direct or via *a*

$D^{(0)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

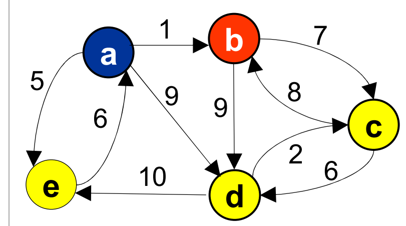
→

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞				
c					
d					
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-				
c					
d					
e					



(6) Distance from *b* to *c*

Direct: $b \rightarrow c = 7$ *(minimum)*

Indirect: $b \rightarrow a \rightarrow c$

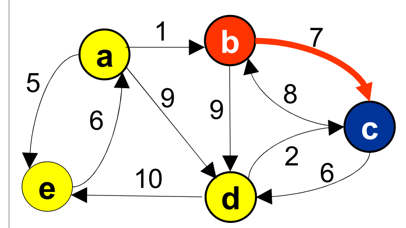
$= \infty + \infty = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

→

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7		
c					
d					
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-		
c					
d					
e					



(7) Distance from *b* to *d*

Direct: $b \rightarrow d = 9$ *(minimum)*

Indirect: $b \rightarrow a \rightarrow d$

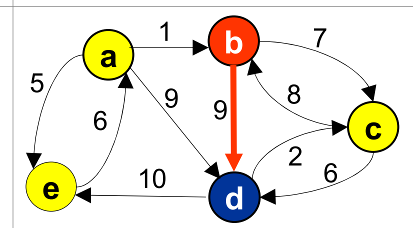
$= \infty + 9 = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

→

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	
c					
d					
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	
c					
d					
e					



(8) Distance from *b* to *e*

Direct: $b \rightarrow e = \infty$

Indirect: $b \rightarrow a \rightarrow e$

$= \infty + 5 = \infty$

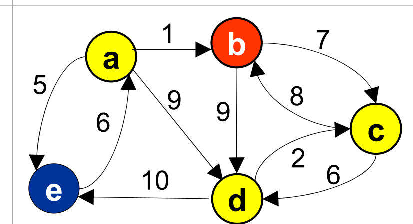
No link direct or through *a*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

→

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c					
d					
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c					
d					
e					



Start vertex End vertex

Distances

Direct Indirect

$D^{(0)}$: Distances matrix

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *c* to other vertices, *direct* or *via* vertex *a* , using table $D^{(0)}$

(9) Distance from *c* to *a*

Direct : $c \rightarrow a = \infty$

Indirect: $c \rightarrow a \rightarrow a$

$= 0 + \infty = \infty$

No link, direct or through a

$D^{(0)}$

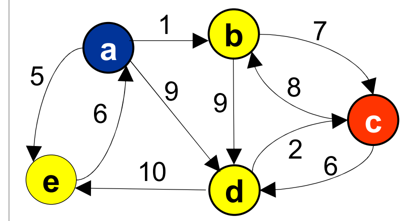
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞				
d					
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-				
d					
e					



(10) Distance from *c* to *b*

Direct: $c \rightarrow b = 8$ *(minimum)*

Indirect: $c \rightarrow a \rightarrow b$

$= \infty + 1 = \infty$

$D^{(0)}$

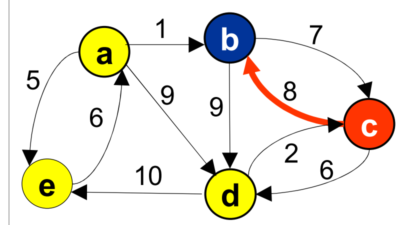
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8			
d					
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-			
d					
e					



(11) Distance from *c* to *d*

Direct: $c \rightarrow d = 6$ *(minimum)*

Indirect: $c \rightarrow a \rightarrow d$

$= \infty + 9 = \infty$

$D^{(0)}$

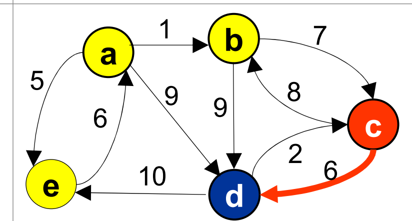
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	a	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	
d					
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	
d					
e					



(12) Distance from *c* to *e*

Direct: $c \rightarrow e = \infty$

Indirect: $c \rightarrow a \rightarrow e$

$= \infty + 5 = \infty$

No link, direct or through *a*

$D^{(0)}$

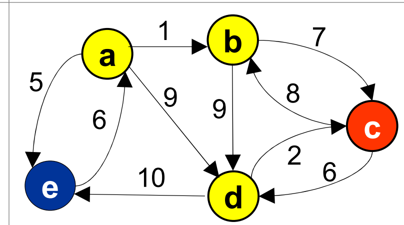
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d					
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d					
e					



Start vertex End vertex

Distances: Direct Indirect

$D^{(0)}$: Distances matrix

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

P: Intermediate vertices for shortest paths

Paths: Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *d* to other vertices, *direct* or *via* vertex *a* , using table $D^{(0)}$

(13) Distance from *d* to *a*
Direct : $d \rightarrow a = \infty$
Indirect: $d \rightarrow a \rightarrow a$
$= \infty + 0 = \infty$
No link, direct or through *a*

$D^{(0)}$

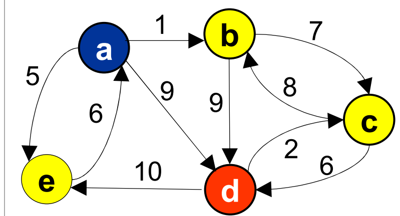
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	∞	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞				
e					

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-				
e					

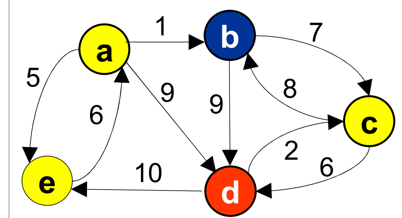


(14) Distance from *d* to *b*
Direct: $d \rightarrow b = \infty$
Indirect: $d \rightarrow a \rightarrow b$
$= \infty + 1 = \infty$
No link, direct or through *a*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞			
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-			
e					

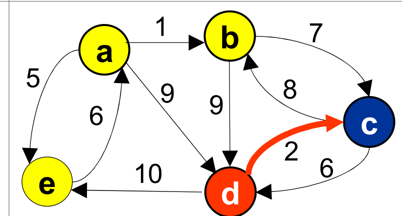


(15) Distance from *d* to *c*
Direct: $d \rightarrow c = 2$ *(minimum)*
Indirect: $d \rightarrow a \rightarrow c$
$= \infty + \infty = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-		
e					

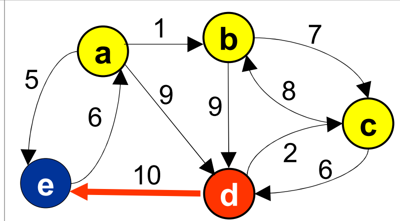


(16) Distance from *d* to *e*
Direct: $d \rightarrow e = 10$ *(minimum)*
Indirect: $d \rightarrow a \rightarrow e$
$= \infty + 5 = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e					

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e					



Start vertex End vertex

Distances
Direct Indirect

$D^{(0)}$: Distances matrix

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *e* to other vertices, *direct* or *via* vertex *a* , using table $D^{(0)}$

(17) Distance from *e* to *a*
Direct : e →a=6
Indirect: e → a →a
= 6+ 0 = 6

$D^{(0)}$

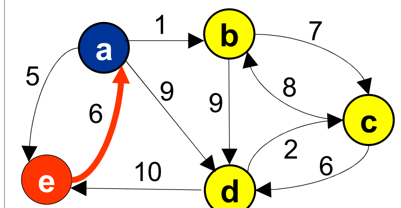
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6				

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-				

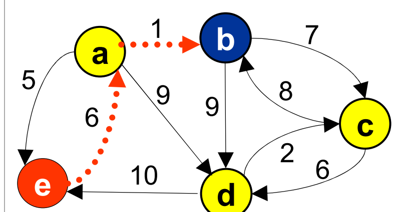


(18) Distance from *e* to *b*
Direct: e → b= ∞
Indirect:e → a → b
= 6 + 1 = 7 (minimum)
Minimum distance via a
Vertex a stored in table *P*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7			

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a			

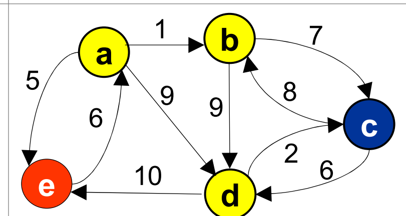


(19) Distance from *e* to *c*
Direct: e → c=∞
Indirect: e → a → c
= 6 + ∞ = ∞
No link, direct or through *a*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞		

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-		

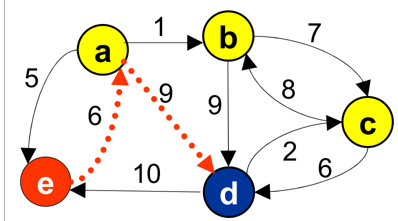


(20) Distance from *e* to *d*
Direct: e → d = ∞
Indirect: e → a → d
= 6 + 9 = 15 (minimum)
Minimum distance via *a*
Vertex *a* stored into table *P*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	∞	∞	∞	0

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	0



Start vertex End vertex

Distances
Direct Indirect

$D^{(0)}$: Distances matrix

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *a* to other vertices, *direct* or *via* vertices *a, b,* using table $D^{(1)}$

(21) Distance from *a* to *b*
Direct : $a \to b = 1$
Indirect: $a \to b \to b$
$= 1 + 0 = 1$

$D^{(1)}$

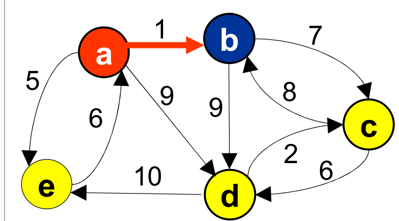
	a	b	c	d	e
a	0	$\boxed{1}$	∞	9	5
b	∞	$\boxed{0}$	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

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

P

	a	b	c	d	e
a	-	-	-	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(22) Distance from *a* to *c*
Direct: $a \to c = \infty$
Indirect: $a \to b \to c$
$= 1 + 7 = 8$ *(minimum)*
Minimum distance via *b*
Vertex *b* stored into table *P*

$D^{(1)}$

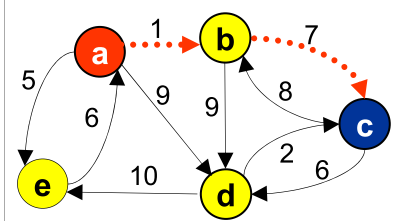
	a	b	c	d	e
a	0	$\boxed{1}$	$\boxed{\infty}$	9	5
b	∞	0	$\boxed{7}$	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8		
b					
c					
d					
e					

P

	a	b	c	d	e
a	-	-	$\boxed{b}$	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(23) Distance from *a* to *d*
Direct: $a \to d = 9$ *(minimum)*
Indirect: $a \to b \to d$
$= 1 + 9 = 10$

$D^{(1)}$

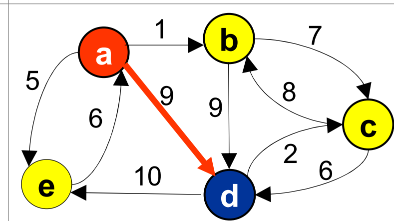
	a	b	c	d	e
a	0	$\boxed{1}$	∞	$\boxed{9}$	5
b	∞	0	7	$\boxed{9}$	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	
b					
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(24) Distance from *a* to *e*
Direct: $a \to e = 5$ *(minimum)*
Indirect: $a \to b \to e$
$= 1 + \infty = \infty$

$D^{(1)}$

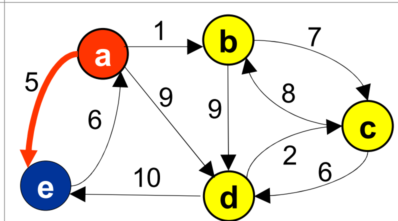
	a	b	c	d	e
a	0	$\boxed{1}$	∞	9	$\boxed{5}$
b	∞	0	7	9	$\boxed{\infty}$
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b					
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *b* to other vertices, *direct* or *via* vertices *a, b* using table $D^{(1)}$

(25) Distance from *b* to *a*
Direct : $b \rightarrow a = \infty$
Indirect: $b \rightarrow b \rightarrow a$
$= 0 + \infty = \infty$
No link, direct or through *a, b*

$D^{(1)}$

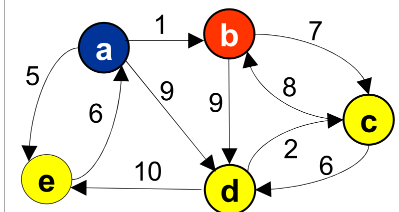
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞				
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(26) Distance from *b* to *c*
Direct: $b \rightarrow c = 7$
Indirect: $b \rightarrow b \rightarrow c$
$= 0 + 7 = 7$

$D^{(1)}$

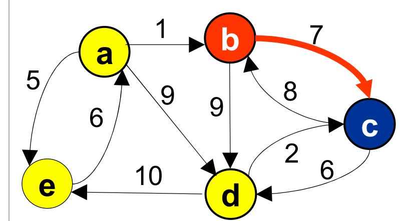
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7		
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(27) Distance from *b* to *d*
Direct: $b \rightarrow d = 9$
Indirect: $b \rightarrow b \rightarrow d$
$= 0 + 9 = 9$

$D^{(1)}$

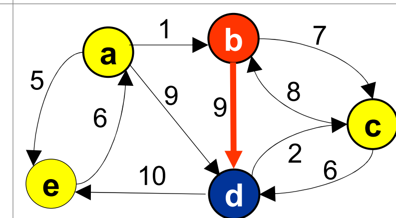
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(28) Distance from *b* to *e*
Direct: $b \rightarrow e = \infty$
Indirect: $b \rightarrow b \rightarrow e$
$= 0 + \infty = \infty$
No link, direct or through *a ,b*

$D^{(1)}$

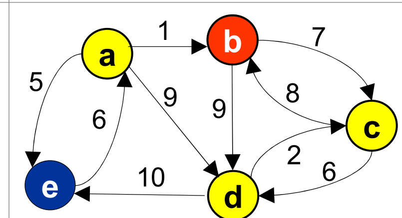
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *c* to other vertices, *direct* or *via* vertices *a, b* using table $D^{(1)}$

(29) Distance from *c* to *a*

Direct : $c \rightarrow a = \infty$

Indirect: $c \rightarrow b \rightarrow a$

$= 8 + \infty = \infty$

No link, direct or through *a, b*

$D^{(1)}$

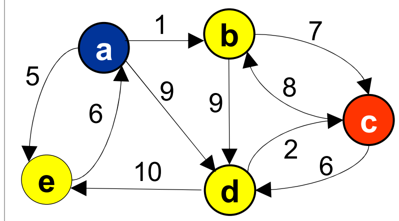
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞				
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(30) Distance from *c* to *b*

Direct: $c \rightarrow b = 8$ *(minimum)*

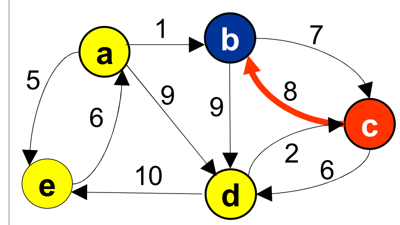
Indirect: $c \rightarrow b \rightarrow b$

$= 8 + 0 = 8$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0		
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(31) Distance from *c* to *d*

Direct: $c \rightarrow d = 6$ *(minimum)*

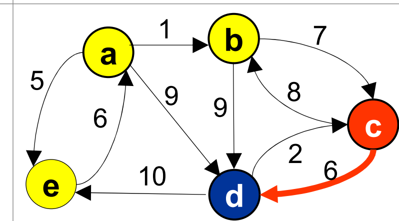
Indirect: $c \rightarrow b \rightarrow d$

$= 8 + 9 = 17$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(32) Distance from *c* to *e*

Direct: $c \rightarrow e = \infty$

Indirect: $c \rightarrow b \rightarrow e$

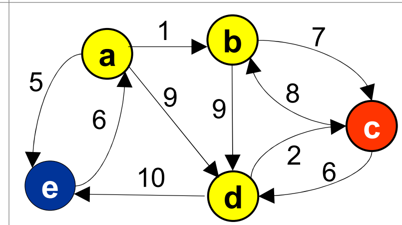
$= 8 + \infty = \infty$

No link, direct or through *a, b*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



Start vertex End vertex

Distances

Direct Indirect

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *d* to other vertices, *direct* or *via* vertices *a, b* using table $D^{(1)}$

(33) Distance from *d* to *a*
Direct : $d \rightarrow a = \infty$
Indirect: $d \rightarrow b \rightarrow a$
$= \infty + \infty = \infty$
No link, direct or via *a, b*

$D^{(1)}$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

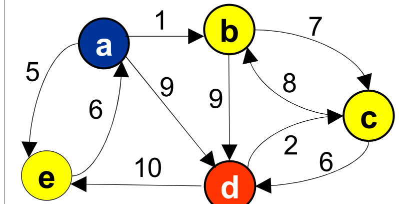
→

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞				
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



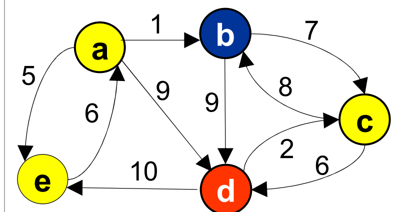
(34) Distance from *d* to *b*
Direct: $d \rightarrow b = \infty$
Indirect: $d \rightarrow b \rightarrow b$
$= \infty + 0 = \infty$
No link, direct or via *a, b*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞			
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



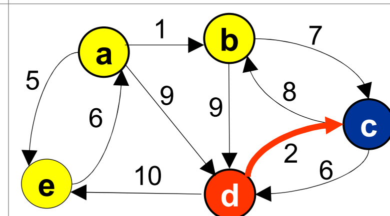
(35) Distance from *d* to *c*
Direct: $d \rightarrow c = 2$ *(minimum)*
Indirect: $d \rightarrow b \rightarrow c$
$= \infty + 7 = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



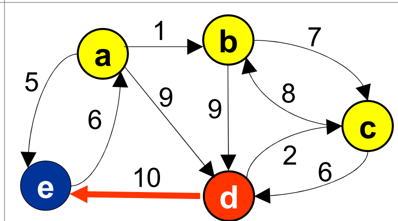
(36) Distance from *d* to *e*
Direct: $d \rightarrow e = 10$ *(minimum)*
Indirect: $d \rightarrow b \rightarrow e$
$= \infty + 5 = \infty$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *e* to other vertices, *direct* or *via* vertices *a, b,* using table $D^{(1)}$

(37) Distance from *e* to *a*

Direct : $e \rightarrow a = 6$ *(minimum)*

Indirect: $e \rightarrow b \rightarrow a$

$= 7 + \infty \quad = \infty$

$D^{(1)}$

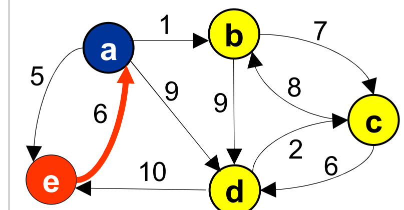
	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6				

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(38) Distance from *e* to *b*

Direct: $e \rightarrow b = 7$

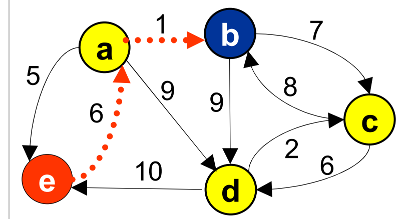
Indirect: $e \rightarrow b \rightarrow b$

$= 7 + 0 \quad = 7$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7			

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	-	a	-



(39) Distance from *e* to *c*

Direct: $e \rightarrow c = \infty$

Indirect: $e \rightarrow b \rightarrow c$

$= 7 + 7 = 14$ *(minimum)*

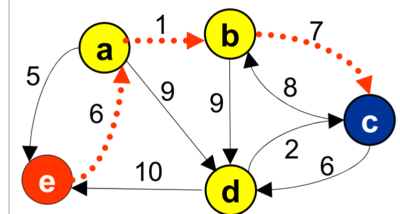
Minimum distance is via *b*

Vertex *b stored into table P*

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14		

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



(40) Distance from *e* to *d*

Direct: $e \rightarrow d = 15$ *(minimum)*

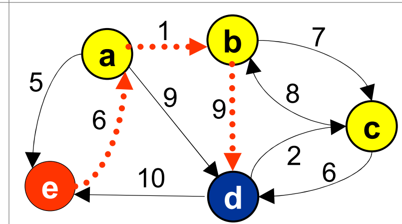
Indirect: $e \rightarrow b \rightarrow d$

$= 7 + 9 \quad = 16$

	a	b	c	d	e
a	0	1	∞	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	∞	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances

Direct Indirect

$D^{(1)}$: Shortest distances, direct, or via vertex *a*

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *a* to other vertices, *direct* or *via* vertices *a, b, c* , using table $D^{(2)}$

(41) Distance from *a* to *b*
Direct : $a \rightarrow b = 1$ *(minimum)*
Indirect: $a \rightarrow c \rightarrow b$
$= 8 + 8 = 16$

$D^{(2)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

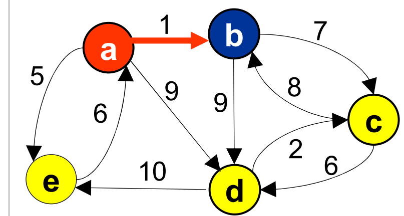
→

$D^{(3)}$

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

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



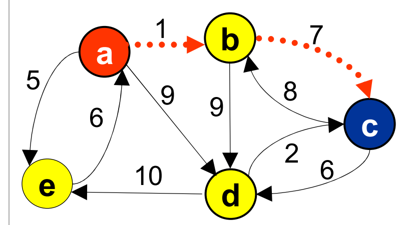
(42) Distance from *a* to *c*
Direct: $a \rightarrow c = 8$
Indirect: $a \rightarrow c \rightarrow c$
$= 8 + 0 = 8$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8		
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



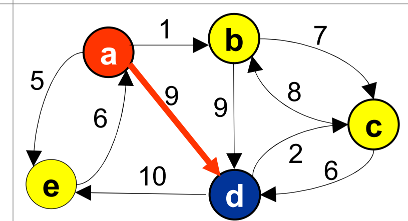
(43) Distance from *a* to *d*
Direct: $a \rightarrow d = 9$ *(minimum)*
Indirect: $a \rightarrow c \rightarrow d$
$= 8 + 6 = 14$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



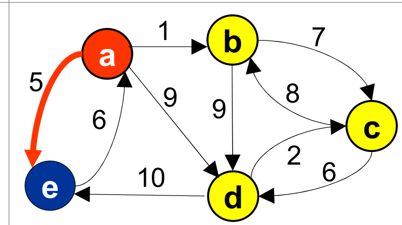
(44) Distance from *a* to *e*
Direct: $a \rightarrow e = 5$ *(minimum)*
Indirect: $a \rightarrow c \rightarrow e$
$= 9 + \infty = \infty$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

$D^{(3)}$: Shortest distances, direct, or via vertices *a, b, c*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *b* to other vertices, *direct* or *via* vertices *a, b, c* , using table $D^{(2)}$

(45) Distance from *b* to *a*
Direct : b → a=∞
Indirect: b → c → a
= 7+∞=∞
No link, direct or via *a, b, c*

$D^{(2)}$

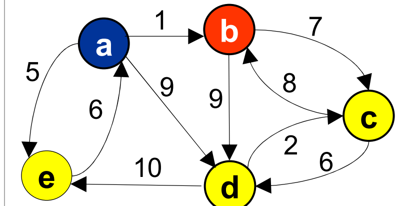
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0			
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-

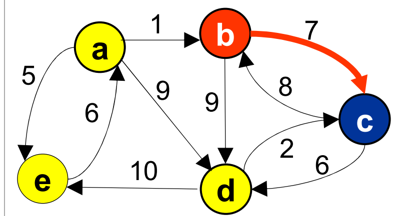


(46) Distance from *b* to *c*
Direct: b → c= 7
Indirect: b → c → c
= 7 +0 = 7

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7		
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-

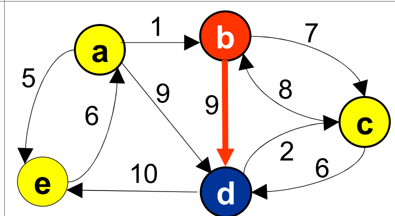


(47) Distance from *b* to *d*
Direct: b → d=9 (minimum)
Indirect: b → c → d
=7 + 6 = 13

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-

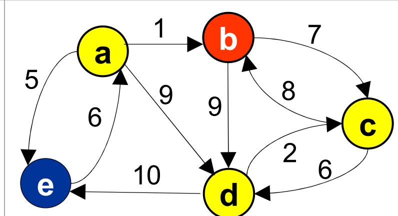


(48) Distance from *b* to *e*
Direct: b → e=∞
Indirect: b → c → e
= 7+ ∞ = ∞
No link, direct or via *a, b, c*

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(2)}$: Shortest distances, direct, or via vertices *a*, *b*

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, *c*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *c* to other vertices, *direct* or *via* vertices *a, b, c* , using table $D^{(2)}$

(49) Distance from *c* to *a*
Direct : $c \rightarrow a = \infty$
Indirect: $c \rightarrow c \rightarrow a$
$= 0 + \infty = \infty$
No link, direct or via *a, b, c*

$D^{(2)}$

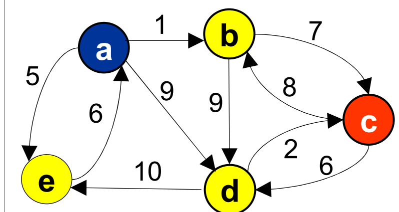
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞				
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



(50) Distance from *c* to *b*
Direct: $c \rightarrow b = 8$
Indirect: $c \rightarrow c \rightarrow b$
$= 0 + 8 = 8$

$D^{(2)}$

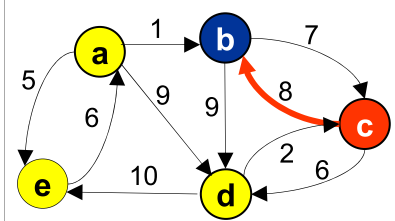
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0		
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



(51) Distance from *c* to *d*
Direct: $c \rightarrow d = 6$
Indirect: $c \rightarrow c \rightarrow d$
$= 0 + 6 = 6$

$D^{(2)}$

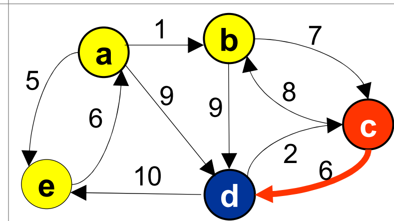
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



(52) Distance from *c* to *e*
Direct: $c \rightarrow e = \infty$
Indirect: $c \rightarrow c \rightarrow e$
$= 0 + \infty = \infty$
No link, direct or via *a, b, c*

$D^{(2)}$

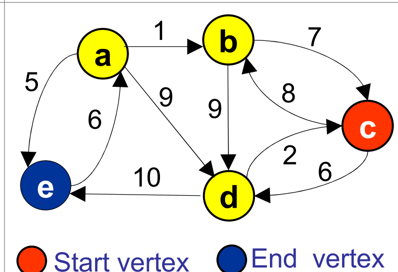
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



Distances
Direct Indirect

$D^{(2)}$: Shortest distances, direct, or via vertices *a*, *b*

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, *c*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *d* to other vertices, *direct* or *via* vertices *a, b, c* using table $D^{(2)}$

(53) Distance from *d* to *a*

Direct : $d \rightarrow a = \infty$

Indirect: $d \rightarrow c \rightarrow a$

$= 2 + \infty = \infty$

No link, direct or via *a, b, c*

$D^{(2)}$

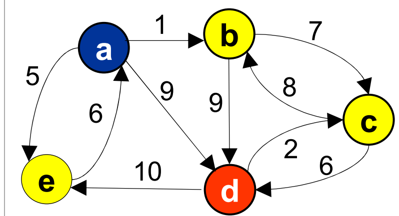
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞				
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	-	-	-	-
e	-	a	b	a	-



(54) Distance from *d* to *b*

Direct: $d \rightarrow b = \infty$

Indirect: $d \rightarrow c \rightarrow b$

$= 2 + 8 = 10$ *(minimum)*

Minimum distance via *c*

Vertex *c* stored into table *P*

$D^{(2)}$

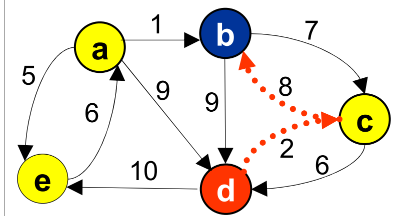
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10			
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(55) Distance from *d* to *c*

Direct: $d \rightarrow c = 2$

Indirect: $d \rightarrow c \rightarrow c$

$= 2 + 0 = 2$

$D^{(2)}$

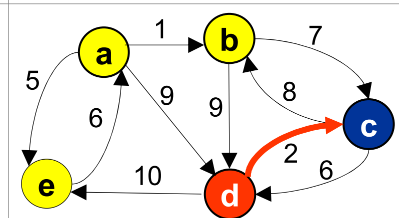
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(56) Distance from *d* to *e*

Direct: $d \rightarrow e = 10$ *(minimum)*

Indirect: $d \rightarrow c \rightarrow e$

$= 2 + \infty = \infty$

$D^{(2)}$

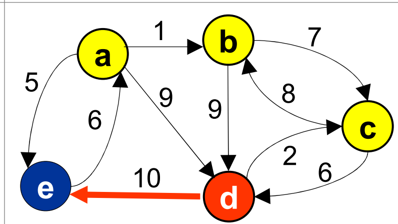
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	∞	2	0	10
e	6	7	14	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances

Direct Indirect

$D^{(2)}$: Shortest distances, direct, or via vertices *a*, *b*

$D^{(3)}$: Shortest distances, direct, or via vertices *a, b, c*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *e* to other vertices, *direct* or *via* vertices *a, b, c,* using table $D^{(2)}$

(57) Distance from *e* to *a*
Direct : $e \rightarrow a = 6$ *(minimum)*
Indirect: $e \rightarrow c \rightarrow a$
$= 14 + \infty = \infty$

$D^{(2)}$

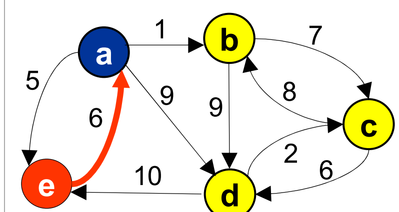
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	$\boxed{\infty}$	8	0	6	∞
d	∞	∞	2	0	10
e	$\boxed{6}$	7	$\boxed{14}$	15	0

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6				

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-

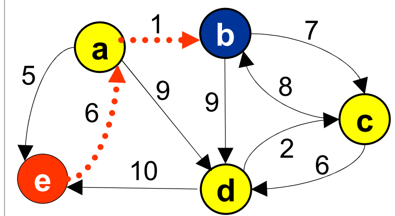


(58) Distance from *e* to *b*
Direct: $e \rightarrow b = 7$ *(minimum*)*
Indirect: $e \rightarrow c \rightarrow b$
$= 14 + 2 = 16$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	$\boxed{8}$	0	6	∞
d	∞	∞	2	0	10
e	6	$\boxed{7}$	$\boxed{14}$	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7			

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-

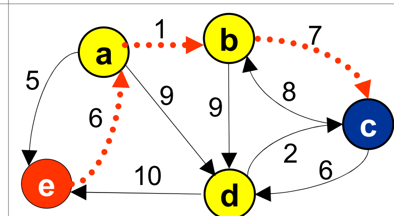


(59) Distance from *e* to *c*
Direct: $e \rightarrow c = 14$
Indirect: $e \rightarrow c \rightarrow c$
$= 14 + 0 = 14$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	$\boxed{0}$	6	∞
d	∞	∞	2	0	10
e	6	7	$\boxed{14}$	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14		

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-

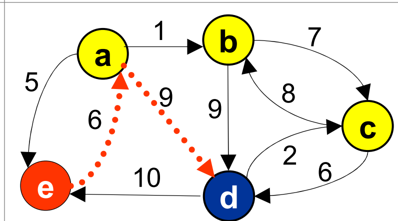


(60) Distance from *e* to *d*
Direct: $e \rightarrow d = 15$ *(minimum)*
Indirect: $e \rightarrow c \rightarrow d$
$= 14 + 6 = 20$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	$\boxed{6}$	∞
d	∞	∞	2	0	10
e	6	7	$\boxed{14}$	$\boxed{15}$	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(2)}$: Shortest distances, direct, or via vertices *a, b*

$D^{(3)}$: Shortest distances, direct, or via vertices *a, b, c*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *a* to other vertices, *direct* or *via* vertices *a, b, c, d*, using table $D^{(3)}$

(61) Distance from *a* to *b*
Direct : $a \to b = 1$ *(minimum)*
Indirect: $a \to d \to b$
$= 9 + 10 = 19$

$D^{(3)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

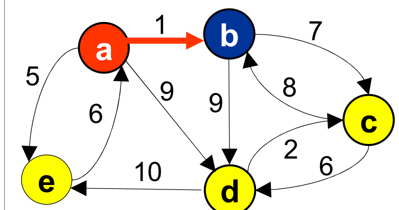
→

$D^{(4)}$

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

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



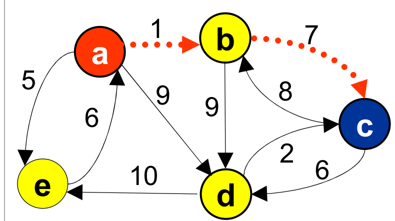
(62) Distance from *a* to *c*
Direct: $a \to c = 8$ *(minimum)*
Indirect: $a \to d \to c$
$= 9 + 2 = 11$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8		
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



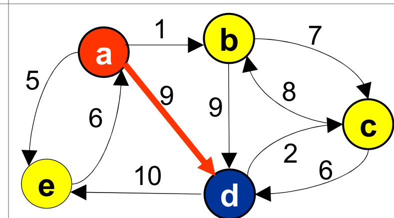
(63) Distance from *a* to *d*
Direct: $a \to d = 9$
Indirect: $a \to d \to d$
$= 9 + 0 = 9$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



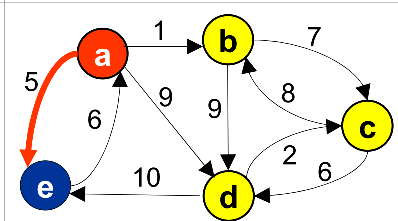
(64) Distance from *a* to *e*
Direct: $a \to e = 5$ *(minimum)*
Indirect: $a \to d \to e$
$= 9 + 10 = 19$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, c

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *b* to other vertices, *direct* or *via* vertices *a, b, c, d* , using table $D^{(3)}$

(65) Distance from *b* to *a*

Direct : $b \rightarrow a = \infty$

Indirect: $b \rightarrow d \rightarrow a$

$= 9 + \infty = \infty$

$D^{(3)}$

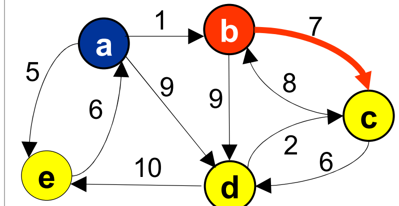
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0			
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(66) Distance from *b* to *c*

Direct: $b \rightarrow c = 7$ *(minimum)*

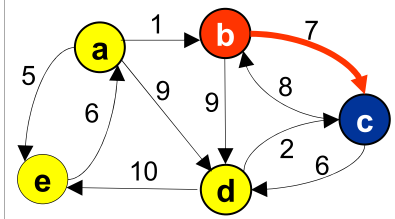
Indirect: $b \rightarrow d \rightarrow c$

$= 9 + 2 = 11$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7		
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(67) Distance from *b* to *d*

Direct: $b \rightarrow d = 9$

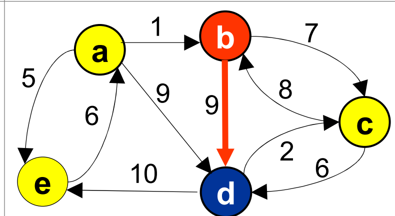
Indirect: $b \rightarrow d \rightarrow d$

$= 9 + 0 = 9$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	-
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(68) Distance from *b* to *e*

Direct: $b \rightarrow e = \infty$

Indirect: $b \rightarrow d \rightarrow e$

$= 9 + 10 = 19$ *(minimum)*

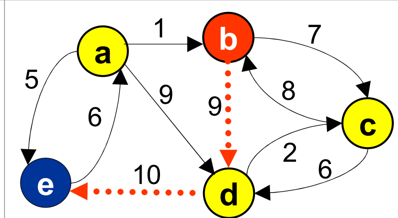
Minimum distance is via *d*

Vertex *d* stored into table *P*

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances

Direct Indirect

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, c

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *c* to other vertices, *direct* or *via* vertices *a, b, c, d*, using table $D^{(3)}$

(69) Distance from *c* to *a*
Direct : $c \to a = \infty$
Indirect: $c \to d \to a$
$= 6 + \infty = \infty$
No link, direct or via *a, b, c, d*

$D^{(3)}$

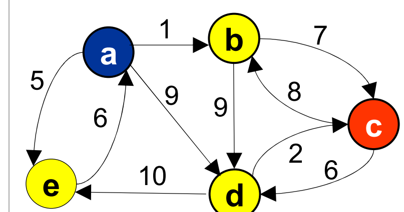
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞				
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(70) Distance from *c* to *b*
Direct: $c \to b = 8$ *(minimum)*
Indirect: $c \to d \to b$
$= 6 + 10 = 16$

$D^{(3)}$

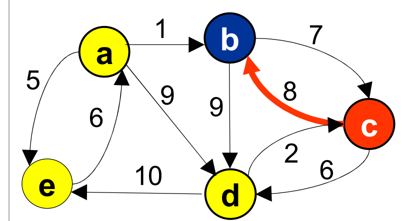
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0		
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(71) Distance from *c* to *d*
Direct: $c \to d = 6$ *(minimum)*
Indirect: $c \to d \to d$
$= 6 + 0 = 6$

$D^{(3)}$

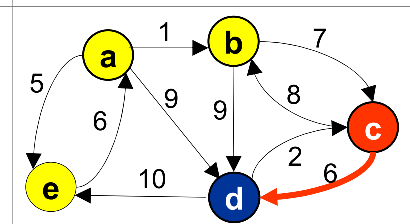
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	-
d	-	c	-	-	-
e	-	a	b	a	-



(72) Distance from *c* to *e*
Direct: $c \to e = \infty$
Indirect: $c \to d \to e$
$= 6 + 10 = 16$ *(minimum)*
Minimum distance is via *d*
Vertex *d* stored into table *P*

$D^{(3)}$

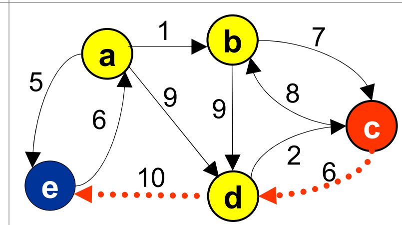
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(3)}$: Shortest distances, direct, or via vertices *a, b, c*

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *d* to other vertices, *direct* or *via* vertices *a, b, c, d*, using table $D^{(3)}$

(73) Distance from *d* to *a*

Direct : $d \rightarrow a = \infty$

Indirect: $d \rightarrow d \rightarrow a$

$= 0 + \infty = \infty$

No link, direct or via *a, b, c, d*

$D^{(3)}$

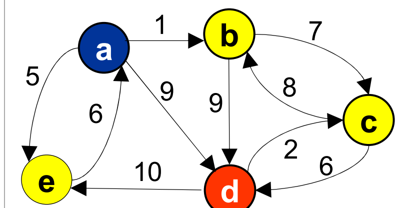
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞				
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



(74) Distance from *d* to *b*

Direct: $d \rightarrow b = 10$

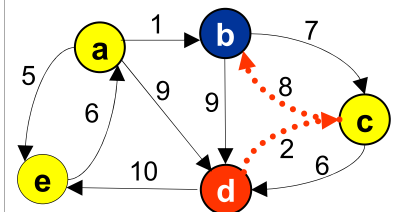
Indirect: $d \rightarrow d \rightarrow b$

$= 0 + 10 = 10$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10			
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



(75) Distance from *d* to *c*

Direct: $d \rightarrow c = 2$ *(minimum)*

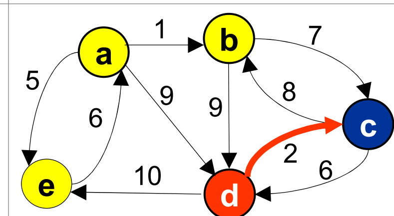
Indirect: $d \rightarrow d \rightarrow c$

$= 0 + 2 = 2$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



(76) Distance from *d* to *e*

Direct: $d \rightarrow e = 10$

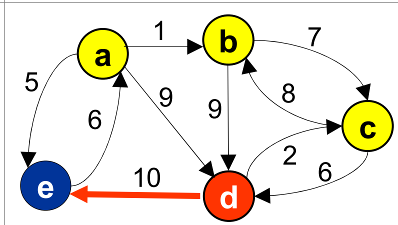
Indirect: $d \rightarrow d \rightarrow e$

$= 0 + 10 = 10$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances

Direct Indirect

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, c

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

P: Intermediate vertices for shortest paths

Paths

Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *e* to other vertices, *direct* or *via* vertices *a, b, c, d*, using table $D^{(3)}$

(77) Distance from *e* to *a*
Direct : $e \rightarrow a=6$ *(minimum)*
Indirect: $e \rightarrow d \rightarrow a$
$= 15+ \infty = \infty$

$D^{(3)}$

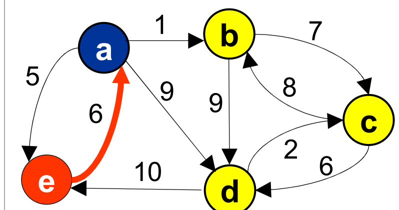
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6				

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-

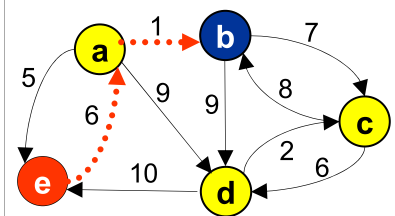


(78) Distance from *e* to *b*
Direct: $e \rightarrow b= 7$ *(minimum)*
Indirect: $e \rightarrow d \rightarrow b$
$= 15 + 10 = 25$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7			

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-

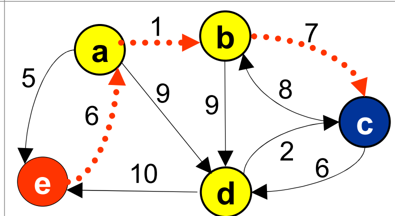


(79) Distance from *e* to *c*
Direct: $e \rightarrow c=14$ *(minimum)*
Indirect: $e \rightarrow d \rightarrow c$
$= 15 + 2 = 17$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14		

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-

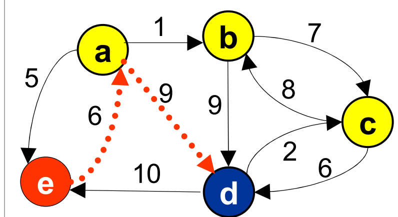


(80) Distance from *e* to *d*
Direct: $e \rightarrow d=15$
Indirect: $e \rightarrow d \rightarrow d$
$=15 + 0 = 15$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	∞
c	∞	8	0	6	∞
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(3)}$: Shortest distances, direct, or via vertices *a*, *b*, *c*

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *a* to other vertices, *direct* or *via* vertices *a, b, c, d, e* , using table $D^{(4)}$

(81) Distance from *a* to *b*
Direct : $a \to b=1$ *(minimum)*
Indirect: $a \to e \to b$
$=5 + 7 = 12$

$D^{(4)}$

	a	b	c	d	e
a	0	$\boxed{1}$	8	9	$\boxed{5}$
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	$\boxed{7}$	14	15	0

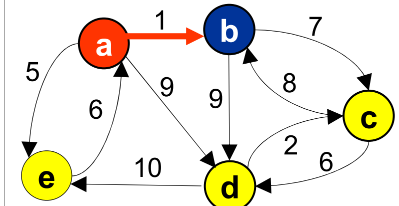
→

$D^{(5)}$

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

P

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



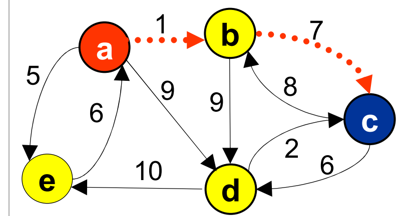
(82) Distance from *a* to *c*
Direct: $a \to c= 8$ *(minimum)*
Indirect: $a \to e \to c$
$= 5 + 14 = 19$

	a	b	c	d	e
a	0	1	$\boxed{8}$	9	$\boxed{5}$
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	$\boxed{14}$	15	0

→

	a	b	c	d	e
a	0	1	8		
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



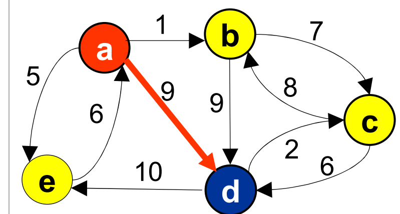
(83) Distance from *a* to *d*
Direct: $a \to d=9$ *(minimum)*
Indirect: $a \to e \to d$
$= 5 + 15 = 20$

	a	b	c	d	e
a	0	1	8	$\boxed{9}$	$\boxed{5}$
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	$\boxed{15}$	0

→

	a	b	c	d	e
a	0	1	8	9	
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



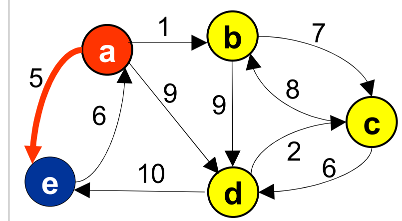
(84) Distance from *a* to *e*
Direct: $a \to e=5$
Indirect: $a \to e \to e$
$= 5 + 0 = 5$

	a	b	c	d	e
a	0	1	8	9	$\boxed{5}$
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	$\boxed{0}$

→

	a	b	c	d	e
a	0	1	8	9	5
b					
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	-	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

$D^{(5)}$: Shortest distances, direct, or via vertices *a, b, c, d* , e

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *b* to other vertices, *direct* or *via* vertices *a, b, c, d, e* , using table $D^{(4)}$

(85) Distance from *b* to *a*
Direct : *b* → *a*=∞
Indirect: *b* → *e* → *a*
= *19+ 6= 25 (minimum)*
Minimum distance is via *e*
Vertex *e* stored into table *P*

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

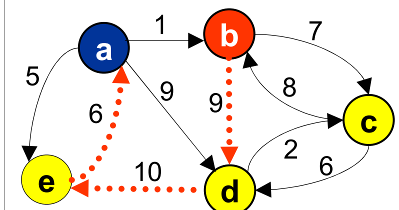
→

$D^{(5)}$

	a	b	c	d	e
a	0	1	8	9	5
b	25				
c					
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



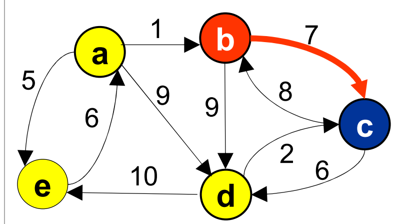
(86) Distance from *b* to *c*
Direct: *b* → *c*= *7 (minimum)*
Indirect: *b* →*e* → *c*
=*19 + 14 = 3 3*

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7		
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



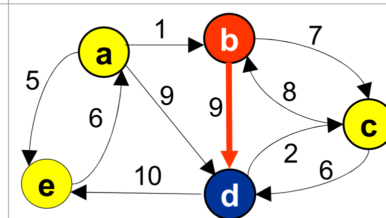
(87) Distance from *b* to *d*
Direct: *b* → *d=9 (minimum)*
Indirect: *b* →*e* → *d*
= *19 + 15 = 34*

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



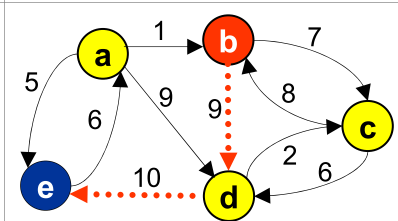
(88) Distance from *b* to *e*
Direct: *b* → *e=19*
Indirect: *b* → *e* → *e*
=*19 + 0 = 19*

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c					
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	-	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

$D^{(5)}$: Shortest distances, direct, or via vertices *a, b, c, d* , e

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *c* to other vertices, *direct* or *via* vertices *a, b, c, d, e* , using table $D^{(4)}$

(89) Distance from *c* to *a*
Direct : $c \rightarrow a = \infty$
Indirect: $c \rightarrow e \rightarrow a$
$= 16 + 6 = 22$
Minimum distance is via *e*
Vertex *e* stored into table *P*

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

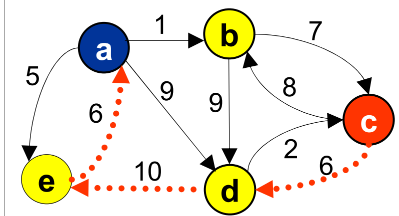
→

$D^{(5)}$

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22				
d					
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



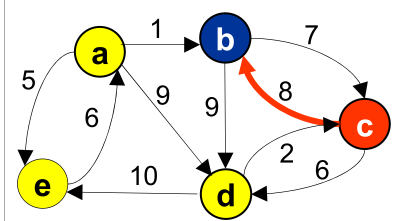
(90) Distance from *c* to *b*
Direct: $c \rightarrow b = 8$ *(minimum)*
Indirect: $c \rightarrow e \rightarrow b$
$= 16 + \mathbf{7} = 23$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0		
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



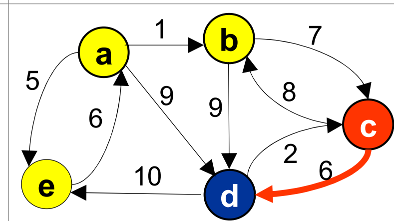
(91) Distance from *c* to *d*
Direct : $c \rightarrow d = 6$ *(minimum)*
Indirect: $c \rightarrow e \rightarrow d$
$= 16 + 14 = 30$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



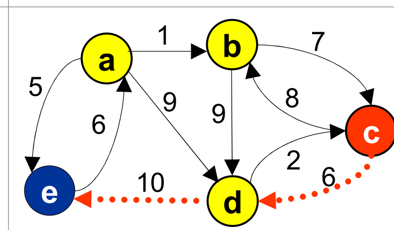
(92) Distance from *c* to *e*
Direct : $c \rightarrow e = 16$
Indirect: $c \rightarrow e \rightarrow e$
$= 16 + 0 = 16$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d					
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	-	c	-	-	-
e	-	a	b	a	-



Start vertex End vertex

Distances
Direct Indirect

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

$D^{(5)}$: Shortest distances, direct, or via vertices *a, b, c, d* , e

P: Intermediate vertices for shortest paths

Paths
Direct Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *d* to other vertices, *direct* or *via* vertices *a, b, c, d, e* , using table $D^{(4)}$

(93) Distance from *d* to *a*
Direct : $d \to a = \infty$
Indirect: $d \to e \to a$
$= 10 + 6 = 16$
Minimum distance is via *e*
Vertex *e* stored into table *P*

$D^{(4)}$

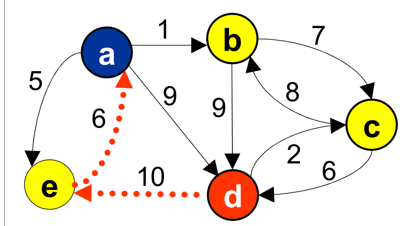
	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

$D^{(5)}$

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16				
e					

P

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-

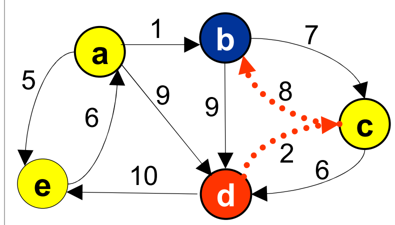


(94) Distance from *d* to *b*
Direct: $d \to b = 10$ *(minimum)*
Indirect: $d \to e \to b$
$= 10 + 7 = 17$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10			
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-

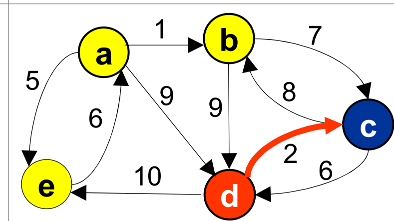


(95) Distance from *d* to *c*
Direct: $d \to c = 2$ *(minimum)*
Indirect: $d \to e \to c$
$= 10 + 14 = 24$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-

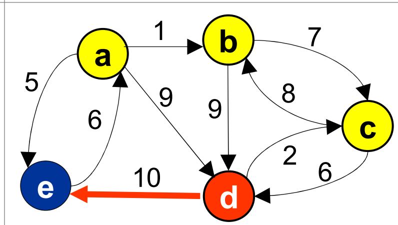


(96) Distance from *d* to *e*
Direct : $d \to e = 10$
Indirect: $d \to e \to e$
$= 10 + 0 = 10$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e					

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Start vertex   End vertex

Distances
Direct   Indirect

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

$D^{(5)}$: Shortest distances, direct, or via vertices *a, b, c, d* , e

P: Intermediate vertices for shortest paths

Paths
Direct   Indirect

Floyd-Warshall Algorithm

Shortest distances from vertex *e* to other vertices, *direct* or *via* vertices *a, b, c, d, e* , using table $D^{(4)}$

(97) Distance from *e* to *a*

Direct :

e →a=6

Indirect:

e → e →a =0 + 60 = 6

$D^{(4)}$

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

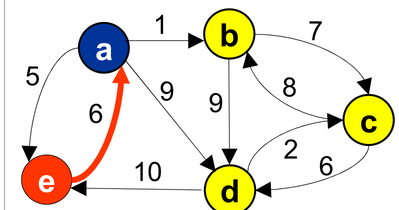
→

$D^{(5)}$

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e	6				

P

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



(98) Distance from *e* to *b*

Direct : e → b= 7

Indirect: e →e → b

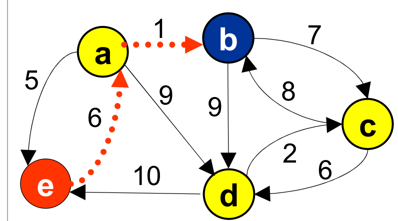
= 0 + 7 = 7

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e	6	7			

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



(99) Distance from *e* to *c*

Direct : e → c=14

Indirect: e → e → c

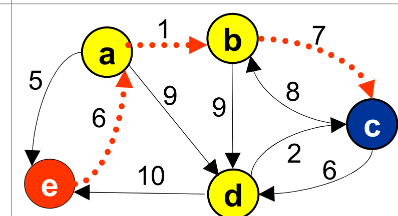
= 0+ 14 =14

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e	6	7	14		

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



(100) Distance from *e* to *d*

Direct : e → d=15

Indirect: e → e → d

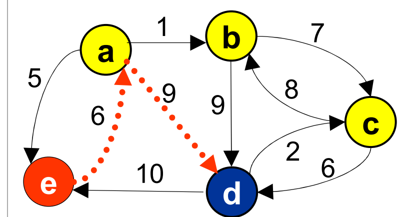
=0 + 15 = 15

	a	b	c	d	e
a	0	1	8	9	5
b	∞	0	7	9	19
c	∞	8	0	6	16
d	∞	10	2	0	10
e	6	7	14	15	0

→

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e	6	7	14	15	0

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Start vertex   End vertex

Distances

Direct   Indirect

$D^{(4)}$: Shortest distances, direct, or via vertices *a, b, c, d*

$D^{(5)}$: Shortest distances, direct, or via vertices *a, b, c, d* , e

P: Intermediate vertices for shortest paths

Paths

Direct   Indirect

Floyd-Warshall Algorithm

All Pairs Shortest Distances

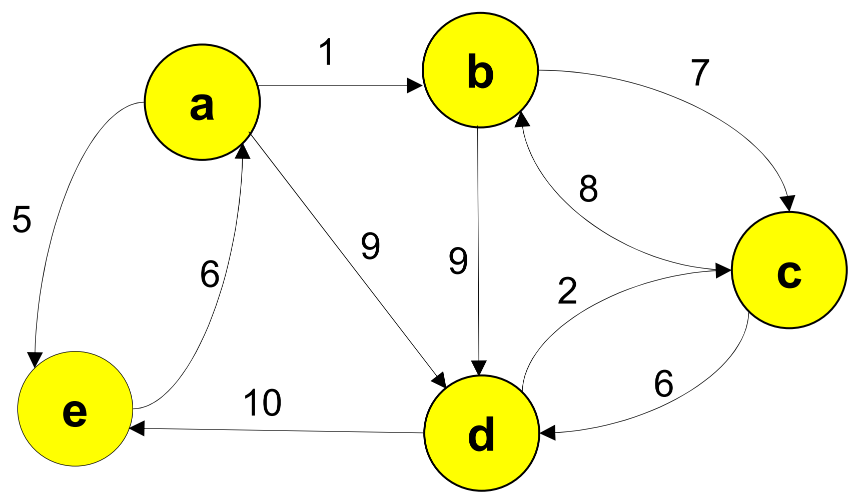
The table in figure *(i)*, generated by the *Floyd Warshall* algorithm, stores the shortest distances between all pairs of vertices *V={a, b, c, d, e }* of the sample graph depicted in figure *(ii)*. The first column of the table contains the start vertices. The top row contains the end vertices of the shortest paths. For example, the shortest distance from vertex *a* to vertex *c* is *8*, whereas, shortest distance from vertex *c* to *a* is *22*.

End vertices

Start vertices

	a	b	c	d	e
a	0	1	8	9	5
b	25	0	7	9	19
c	22	8	0	6	16
d	16	10	2	0	10
e	6	7	14	15	0

(i) Table of all pairs shortest distances



(ii) Sample weighted graph

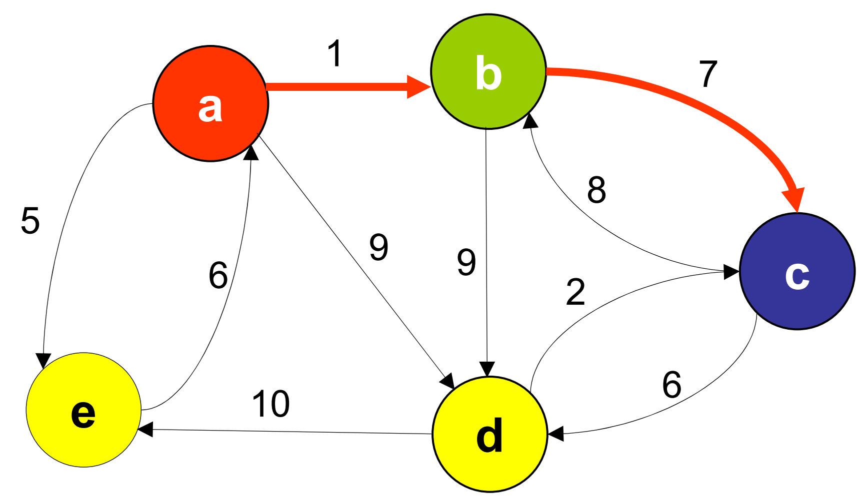
Floyd-Warshall Algorithm

All Pairs Shortest Paths

The table *(i)*, stores the information about the intermediate vertices on a shortest path between a pair of vertices *V={a, b, c, d, e }* of the sample graph depicted in figure *(ii)*. The first column of the table contains the start vertices. The top row contains the end vertices of the shortest paths. A dash (-) means that there is direct shortest path between the corresponding vertices For example, there is a direct shortest path from vertex *a* to vertex *b.* A vertex label in the table indicates, the vertex that links the ***last segment*** of the shortest path from a given vertex. There are ten such paths. Some pass through more than one vertex. For example, the last part of the shortest path from *a* to *c* passes through the vertex *b*. The table can be used to map all other nine shortest paths that pass through the intermediate vertices, as illustrated in the next set of diagrams.

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-

(i) Table of intermediate vertices in a shortest path



(ii) Sample weighted graph The highlighted Path is shortest path from a to c through b

Floyd-Warshall Algorithm

Shortest paths through intermediate vertices

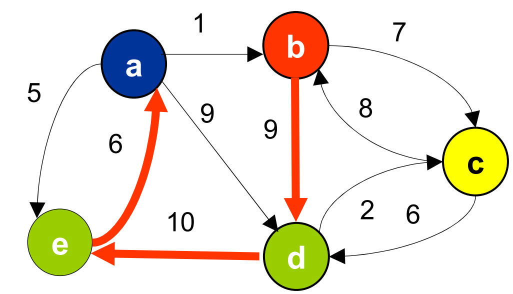
***Path #2*:** From table P it follows that the last segment of the shortest path, from vertex b to vertex a, passes through the vertex e. Thus,

$$b \rightarrow e \rightarrow a$$

Again, the shortest path from b to e passes through d. Therefore, the shortest path is

$$b \rightarrow d \rightarrow e \rightarrow a$$

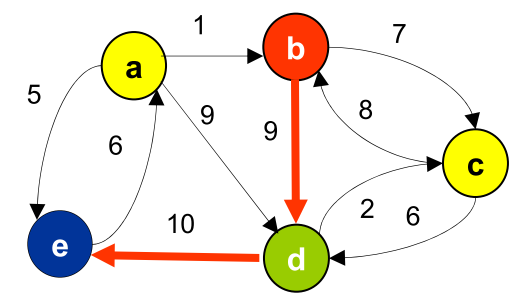
	a	b	c	d	e
a	-	-	b	-	-
b	$\boxed{e}$	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Path #3: The shortest path from vertex b to vertex e passes through the vertex d. The shortest path from b to d is direct. Thus shortest path from b to e is

$$b \rightarrow d \rightarrow e$$

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	$\boxed{d}$
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



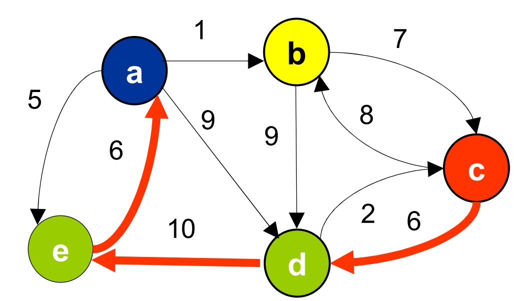
Path #4 : The shortest path from vertex c to vertex a passes through the vertex e. Thus,

$$c \rightarrow e \rightarrow a$$

However, shortest path from c to e passes through d. Therefore, the shortest path from c to a, through the intermediate vertices d, e , is

$$c \rightarrow d \rightarrow e \rightarrow a$$

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	$\boxed{e}$	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Intermediate vertex

Start vertex   Mid vertex   End vertex

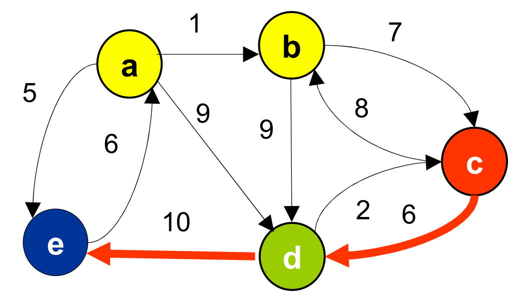
Floyd-Warshall Algorithm

Shortest paths through intermediate vertices

Path #5: The shortest path from vertex *c* to vertex *e* passes through the vertex *d*. The shortest path from *c* to *d* is *direct*. Thus, the shortest path from *c* to *e* is

$$c \rightarrow d \rightarrow e$$

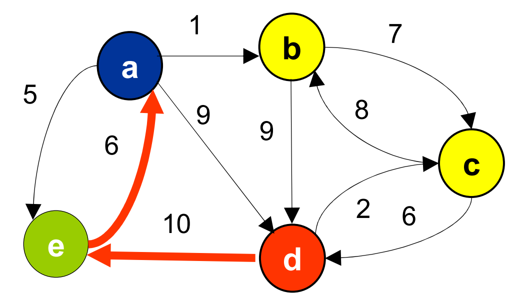
	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Path #6: The shortest path from vertex *d* to vertex *a* passes through the vertex *e*. The shortest path from *d* to *e* is *direct*. Thus, the shortest path from *d* to *a* is

$$d \rightarrow e \rightarrow a$$

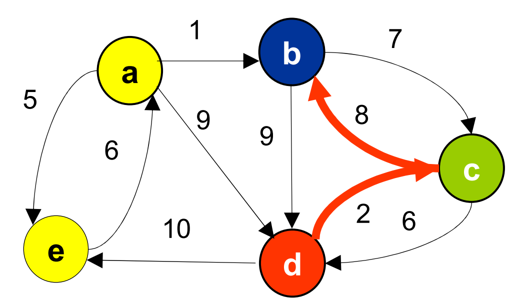
	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Path #7: The shortest path from vertex *d* to vertex *b* passes through the vertex *c*. Again, the shortest path from *d* to *b* is *direct*. Thus, the shortest path from *d* to *c* is

$$d \rightarrow b \rightarrow c$$

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Intermediate vertex   Start vertex   Mid vertex   End vertex

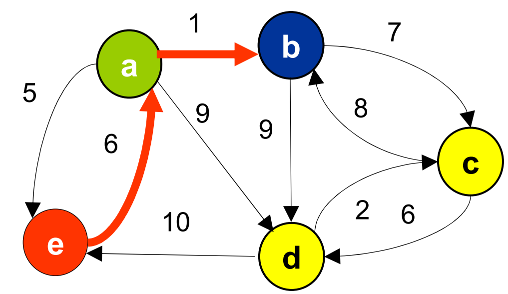
Floyd-Warshall Algorithm

Shortest paths through intermediate vertices

Path #8: The shortest path from vertex *e* to vertex *b* passes through the vertex *a*. The shortest path from *e* to *a* is *direct*. Thus, the shortest path from *e* to *b* is

$$e \rightarrow a \rightarrow b$$

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



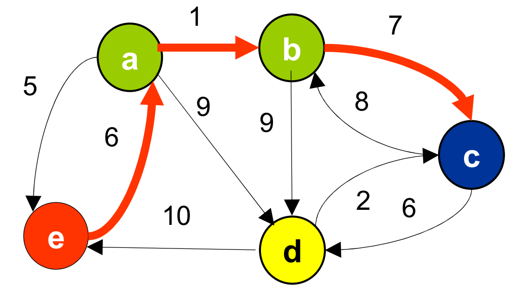
Path #9: The last segment of the shortest path from vertex *e* to vertex *c* passes through the vertex *b*. Thus,

$$e \rightarrow b \rightarrow c$$

Again, the shortest path from *e* to *b* passes through *a*. Thus, the shortest path from *e* to *c* is

$$e \rightarrow a \rightarrow b \rightarrow c$$

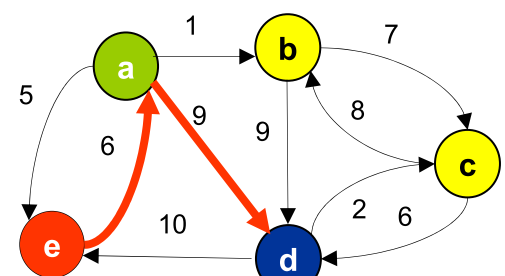
	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Path # 10: The shortest path from vertex *e* to vertex *d* passes through the vertex *a*. The shortest path from *e* to *a* is *direct*. Thus, the shortest path from *e* to *d* is

$$e \rightarrow a \rightarrow d$$

	a	b	c	d	e
a	-	-	b	-	-
b	e	-	-	-	d
c	e	-	-	-	d
d	e	c	-	-	-
e	-	a	b	a	-



Intermediate vertex

Start vertex   Mid vertex   End vertex

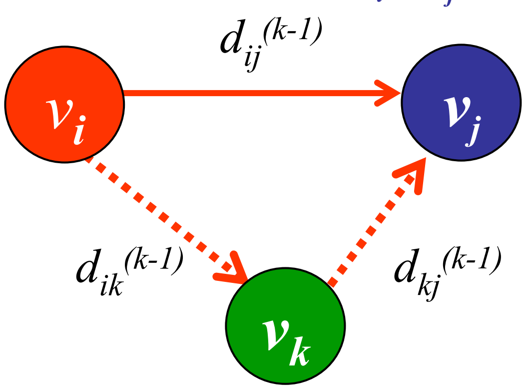
All Pairs Shortest Distances

Recurrence

Let $d_{ij}^{(k)}$ be an element *of* $D^{(k)}$. In particular, $d_{ij}^{(0)}$ is equal to w_{ij} of *weight matrix*

The element $d_{ij}^{(k)}$ is computed by using the entries in the matrix $D^{(k-1)}$. If the shortest distance is along the *direct path*, then $d_{ij}^{(k)}=d_{ij}^{(k-1)}$. However, if the shortest path is through an *intermediate vertex*, then $d_{ij}^{(k)}=d_{ik}^{(k-1)}+d_{kj}^{(k-1)}$.

Direct path from v_i to v_j



Indirect path through vertex v_k

- It follows that the distance $d_{ij}^{(k)}$ satisfies the following recurrence:

$$d_{ij}^{(0)} = w_{ij}$$

$$d_{ij}^{(k)} = \text{minimum}\left(d_{ij}^{(k-1)},\ d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \right)$$

Floyd-Warshall Algorithm
Implementation

The following code implements the algorithm. It computes the shortest paths between the vertices of a weighted graph. The vertices are numbered *1,2, ...n*. *W[i,j]* is the weigh matrix. The outer-most loop identifies the intermediate vertices through which the shortest paths pass. The indexes of the intermediate vertices are saved in table *P*, which stores the highest index of vertex along the shortest path. In other words, *P* stores the vertex of the *last shortest sub-path* to the *end vertex*

```
SHORTEST-DISTANCE(W, n)
 for i ←1 to n do
 for j←1 to n do
    D[i,j] ←W[i,j]            ► Copy weight matrix to table D
 for i ←1 to n do
 for j←1 to n do
        P[i,j] ←0             ►  Initialize table P to hold intermediate vertices
for k ←1 to n do              ►Intermediate vertices, numbered 1,2,3…n, are referenced by index k.
   for i ←1 to n do           ►Start  vertices, numbered 1,2,3…n, are referenced by index i
      for  j ←1 to n do       ►End  vertices, numbered 1,2,3…n, are referenced by index j
       if D[i, k]+D[k ,j] <D[i, j]         ►Shortest path from vertex i to vertex j  is via  vertex k
       then D[i, j] ←D[i, k] + D[k, j]  ►Update the shortest distance  and copy to original table
            P[i, j] ←k         ►Store index of intermediate vertex along the shortest path in table P
 return D,P
```

Visualization

Visualization Floyd Warshall Algorithm

Exit

Execution Speed

○ Slow ○ Medium ◉ Fast

Start Algorithm | Trace Algorithm | Draw Graph

End of execution

All Pairs Shortest Distances

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	65	56	86	81	90	70	37	69	61	49	41	91	13
B	65	0	49	79	47	58	38	56	37	28	17	38	35	62
C	56	49	0	30	47	73	53	71	52	52	32	53	35	43
D	86	79	30	0	49	81	66	101	82	80	62	83	61	73
E	81	47	47	49	0	32	17	59	53	31	38	51	12	68
F	90	58	73	81	32	0	49	80	21	61	41	62	44	86
G	70	38	53	66	17	49	0	42	41	14	21	34	29	66
H	37	56	71	101	59	80	42	0	59	54	39	52	71	50
I	69	37	52	82	53	21	41	59	0	40	20	41	64	65
J	61	28	52	80	31	61	14	54	40	0	20	20	43	65
K	49	17	32	62	38	41	21	39	20	20	0	21	50	45
L	41	38	53	83	51	62	34	52	41	20	21	0	63	54
M	91	35	35	61	12	44	29	71	64	43	50	63	0	78
N	13	62	43	73	68	86	66	50	65	65	45	54	78	0

Table of Intermediate Vertices

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	-	-	N	N	N	K	K	-	K	L	-	-	N	-
B	-	-	K	K	M	K	K	K	K	-	-	K	-	K
C	N	K	-	-	M	K	K	K	K	K	-	K	-	-
D	N	K	-	-	-	E	E	K	K	G	C	K	E	C
E	N	M	M	-	-	-	-	G	F	G	G	J	-	-
F	K	K	K	E	-	-	E	K	-	K	I	K	E	K
G	K	K	K	E	-	E	-	-	K	-	-	J	E	K
H	-	K	K	K	G	K	-	-	K	-	-	-	G	A
I	K	K	K	K	F	-	K	K	-	K	-	K	-	K
J	L	-	K	G	G	K	-	-	K	-	-	-	G	K
K	-	-	-	C	G	I	-	-	-	-	-	-	G	-
L	-	K	K	K	J	K	J	-	K	-	-	-	J	A
M	N	-	-	E	-	E	E	G	-	G	G	J	-	C
N	-	K	-	C	-	K	K	A	K	K	-	A	C	-

Printing Shortest Paths

Implementation

The following code makes recursive calls to print the vertices along the shortest path between vertices with indexes *i* and *j*. The intermediate vertices are listed using the table *P*

```
LIST-PATH( i, j )                        ► Print shortest path between vertices v_i and v_j
   if P[i,j]≠0
      then LIST-PATH ( i, P[ i, j ])      ► Intermediate vertex
         Print "v" + P[ i, j ]            ► Print the vertex on the shortest path
         LIST-PATH ( P[ i, j ], j )       ► Make recursive call to list other intermediate vertices
 return
```

Analysis of Floyd-Warshall Algorithm

Time and Space Complexity

The code for the implementation of Floyd-Warshall algorithm consists of the following *three nested loops*, each *executing n times*

```
for k ←1 to n do                          ► Intermediate vertices numbered 1, 2, 3,…n
    for i ←1 to n do                      ► Start vertices numbered 1, 2, 3,…n
        for j ←1 to n do                  ► End vertices numbered 1, 2, 3,…n
         if D[i, k]+D[k ,j] <D[i, j]      ► Shortest paths via vertices numbered 1,2,..,k
          then D[i, j] ←D[i, k]+D[k, j]
```

- Therefore, the *time complexity,* of the algorithm is

$$T(n)=\theta(n \times n \times n)=\theta(n^3)$$

- The algorithm has *space complexity* of $S(n) =\theta(n^2)$, because in each iteration the matrices $D^{(0)}, D^{(1)}, \; \ldots D^{(n)}$ the are obtained by *updating* and *overwriting* the preceding distances in the source matrix. Further, each matrix stores *n x n* distances between all pairs of graph vertices

Dynamic Programming

Limitations

The disadvantages associated with dynamic programming are:

- *There is an overhead of memory requirement to store tables for the solution of sub problems*

- *The formulation of sub problem structure is difficult*

- *The principle of optimality must hold*