All-Pair Shortest Path

Floyd-Warshall's Algorithm

- Negative Weight Edges may be present, But we assume that there are no negative weight cycles.

- It follows **Dynamic Programming** approach.

- For a given directed graph G(V, E) of n vertices
  Input to this algorithm is an n x n matrix **W** representing edge weights.

$$w_{ij} = \begin{cases} 0 & \text{if } i = j \ , \\ \text{the weight of directed edge } (i, j) & \text{if } i \neq j \text{ and } (i, j) \in E \\ \infty & \text{if } i \neq j \text{ and } (i, j) \notin E \end{cases}$$

Floyd-Warshall's Algorithm

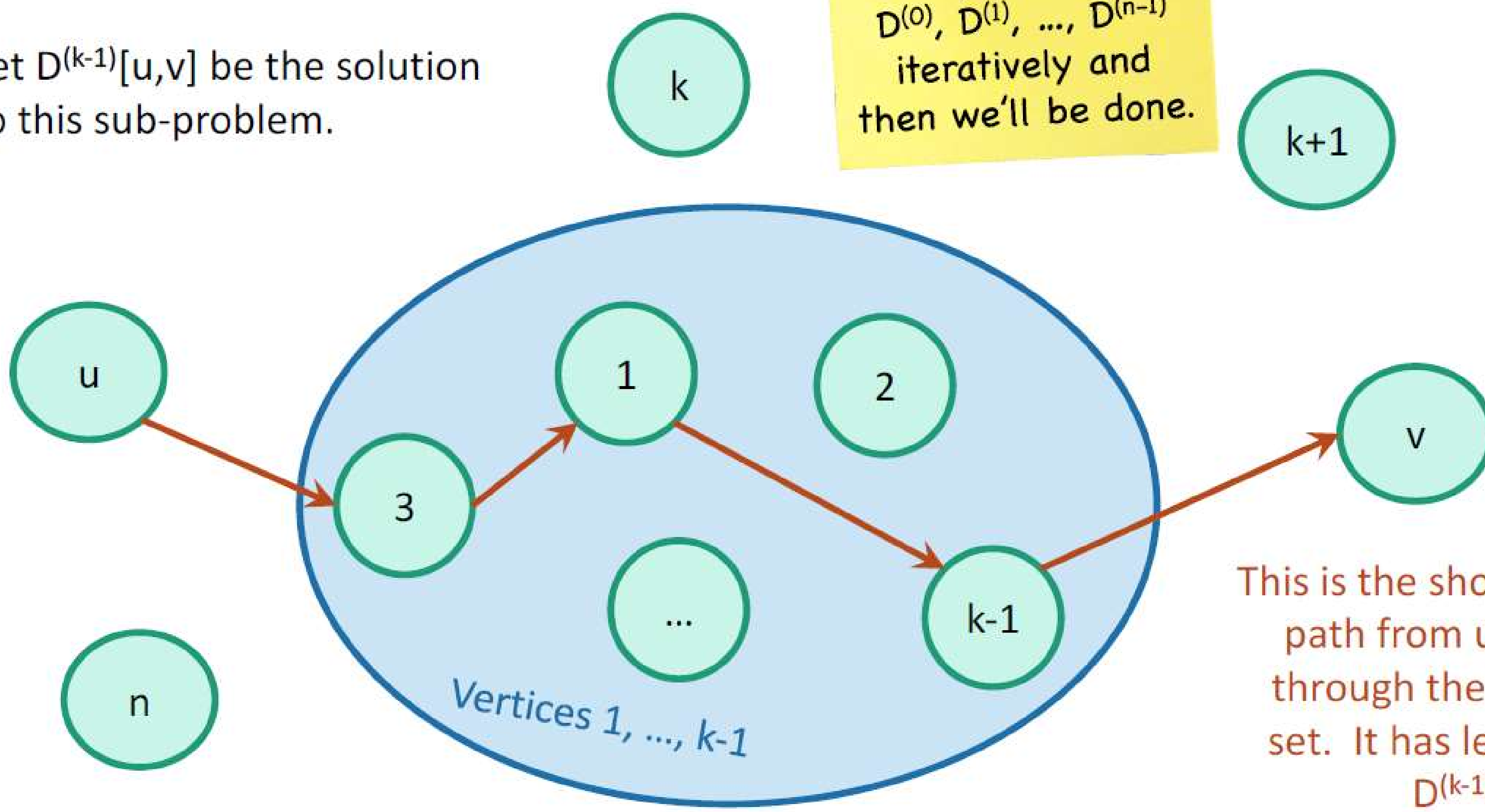
Optimal substructure

Label the vertices 1,2,…,n
(We omit edges in the picture below).

Sub-problem: For all pairs, u,v, find the cost of the shortest path from u to v, so that all the internal vertices on that path are in {1,…,k-1}.

Our DP algorithm will fill in the n-by-n arrays $D^{(0)}, D^{(1)}, \ldots, D^{(n-1)}$ iteratively and then we'll be done.

Let $D^{(k-1)}[u,v]$ be the solution to this sub-problem.

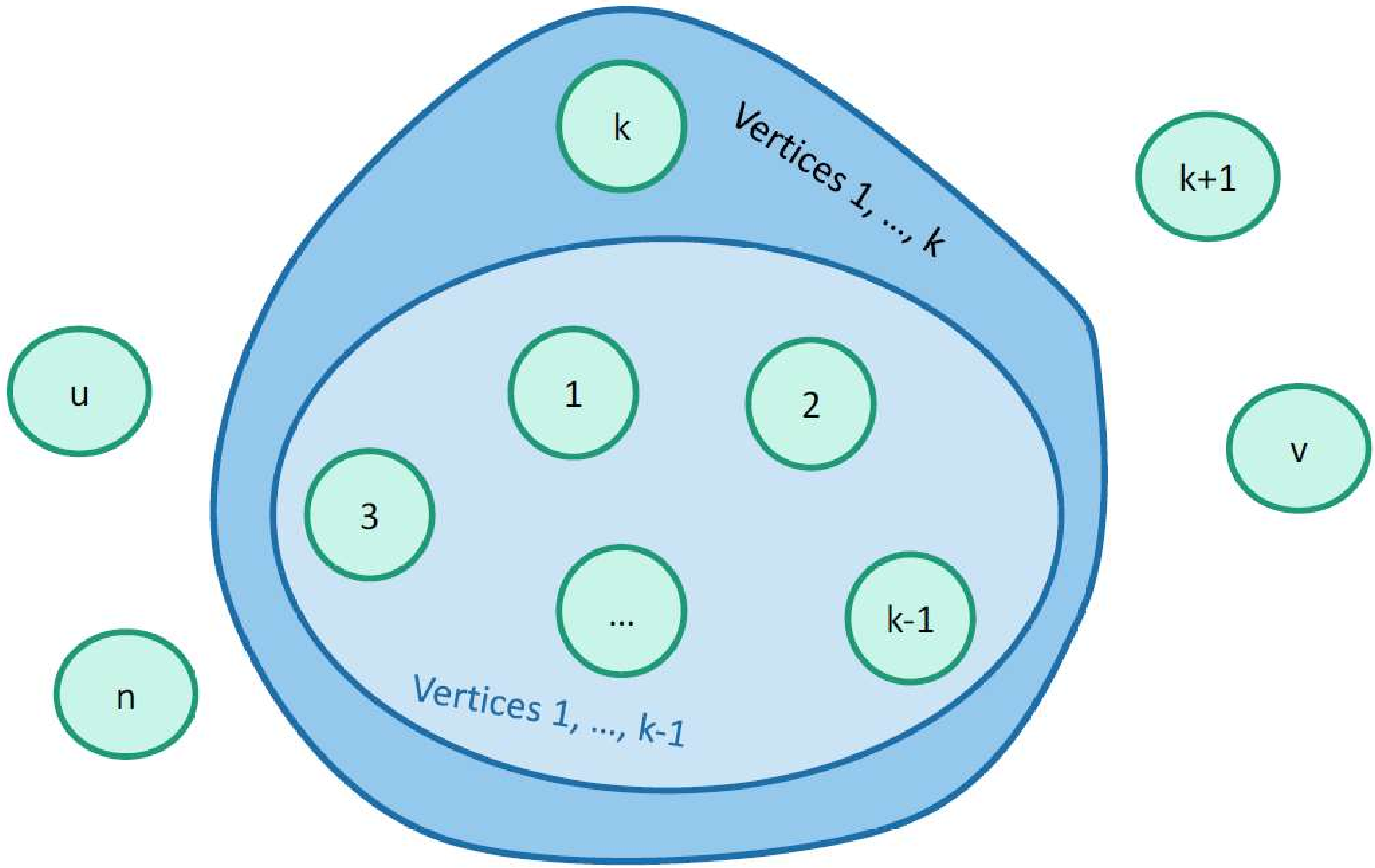


This is the shortest path from u to v through the blue set. It has length $D^{(k-1)}[u,v]$

Floyd-Warshall's Algorithm

How can we find $D^{(k)}[u,v]$ using $D^{(k-1)}$?

$D^{(k)}[u,v]$ is the cost of the shortest path from u to v so that all internal vertices on that path are in {1, …, k}.

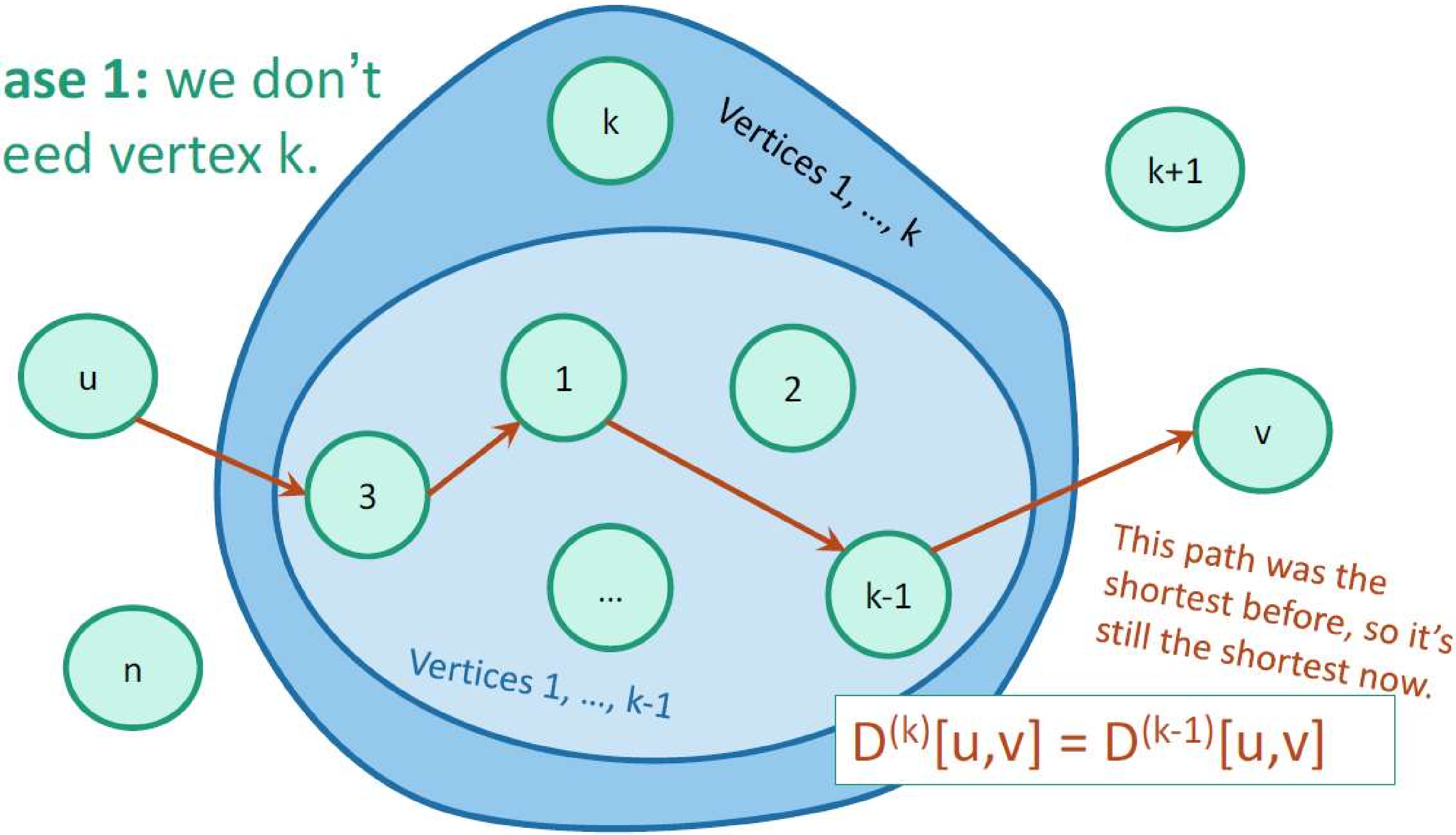


Floyd-Warshall's Algorithm

How can we find $D^{(k)}[u,v]$ using $D^{(k-1)}$?

$D^{(k)}[u,v]$ is the cost of the shortest path from u to v so that all internal vertices on that path are in {1, …, k}.

Case 1: we don't need vertex k.



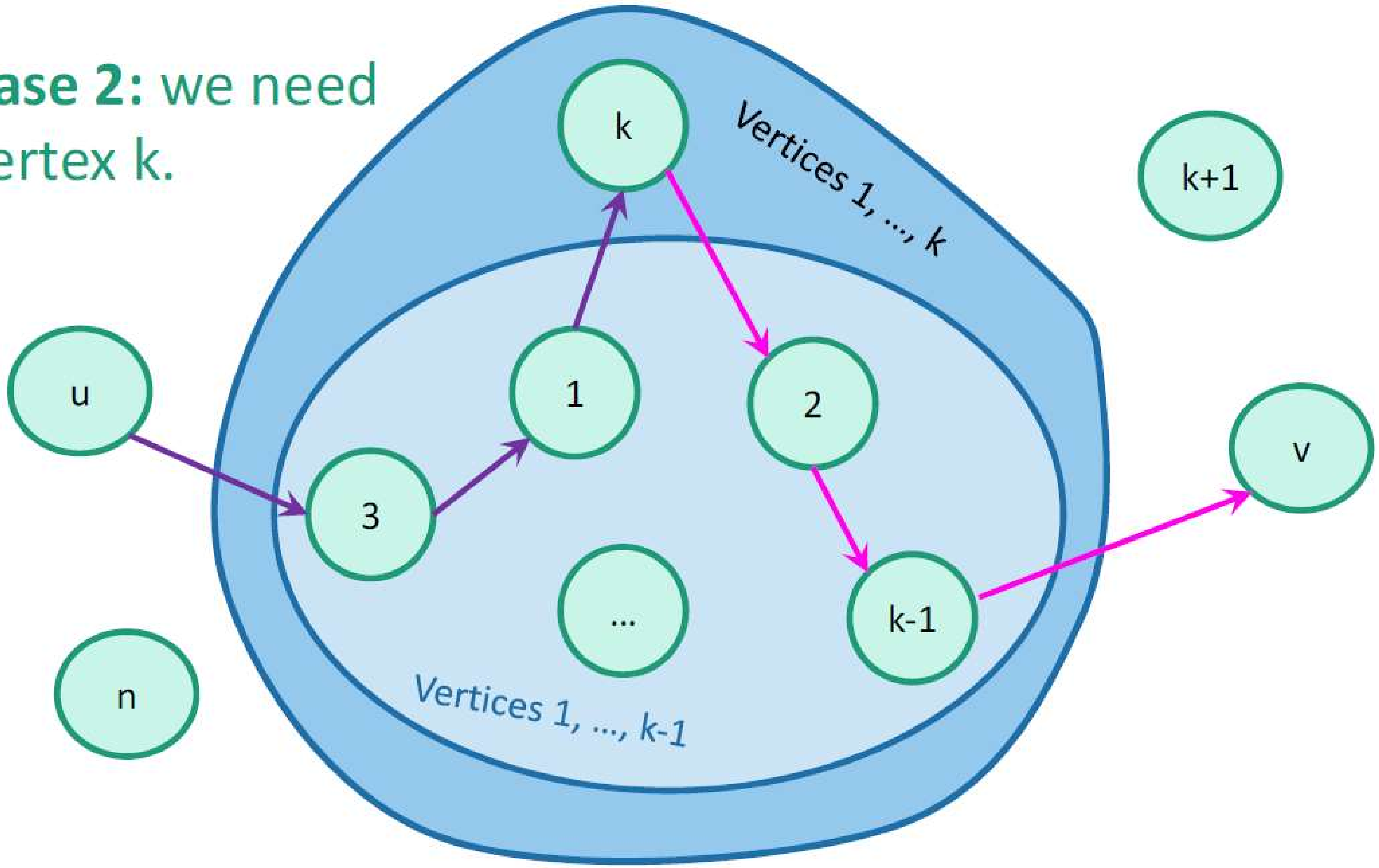
$$D^{(k)}[u,v] = D^{(k-1)}[u,v]$$

Floyd-Warshall's Algorithm

How can we find $D^{(k)}[u,v]$ using $D^{(k-1)}$?

$D^{(k)}[u,v]$ is the cost of the shortest path from u to v so that all internal vertices on that path are in {1, …, k}.

Case 2: we need vertex k.



Floyd-Warshall's Algorithm

How can we find $D^{(k)}[u,v]$ using $D^{(k-1)}$?

- $D^{(k)}[u,v] = \min\{ D^{(k-1)}[u,v], D^{(k-1)}[u,k] + D^{(k-1)}[k,v] \}$

Case 1: Cost of shortest path through {1,…,k-1}

Case 2: Cost of shortest path from u to k and then from k to v through {1,…,k-1}

- Optimal substructure:
  - We can solve the big problem using smaller problems.
- Overlapping sub-problems:
  - $D^{(k-1)}[k,v]$ can be used to help compute $D^{(k)}[u,v]$ for lots of different u's.

Floyd-Warshall's Algorithm

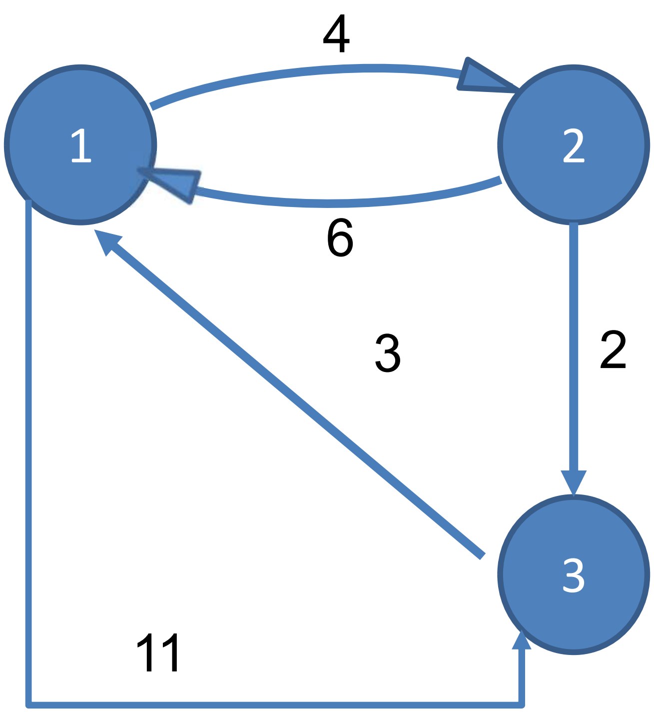
$d_{ij}^{(k)}$ – weight of shortest path from vertex i to j for which intermediate vertices are $v_1, ..., v_k$

Floyd-Warshall's Algorithm

FloydWarshall(**matrix** W)
n ← rows[W]
$D^{(0)} \leftarrow W$
for $k \leftarrow 1$ **to** n **do**
for $i \leftarrow 1$ **to** n **do**
for $j \leftarrow 1$ **to** n **do**
$d_{ij}^{(k)} \leftarrow \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})$
return $D^{(n)}$

Running Time: O (n^3)

Example 1

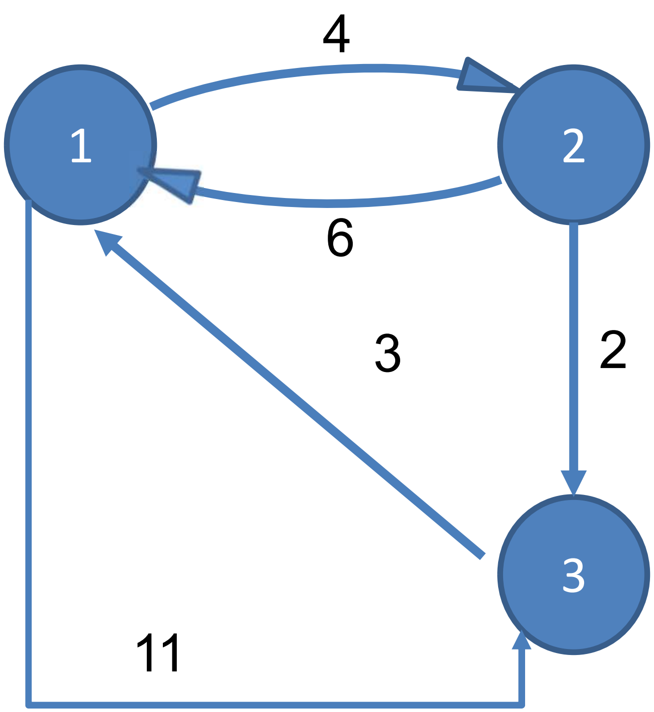


$W =$

	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

D^0	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

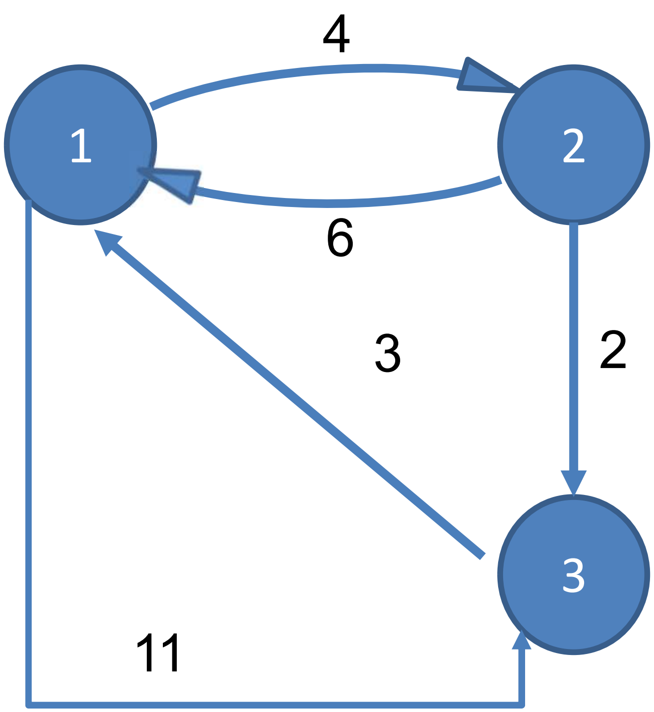
Example 1



D^0	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

D^1	1	2	3
1	0	4	11
2	6	0	2
3	3	7	0

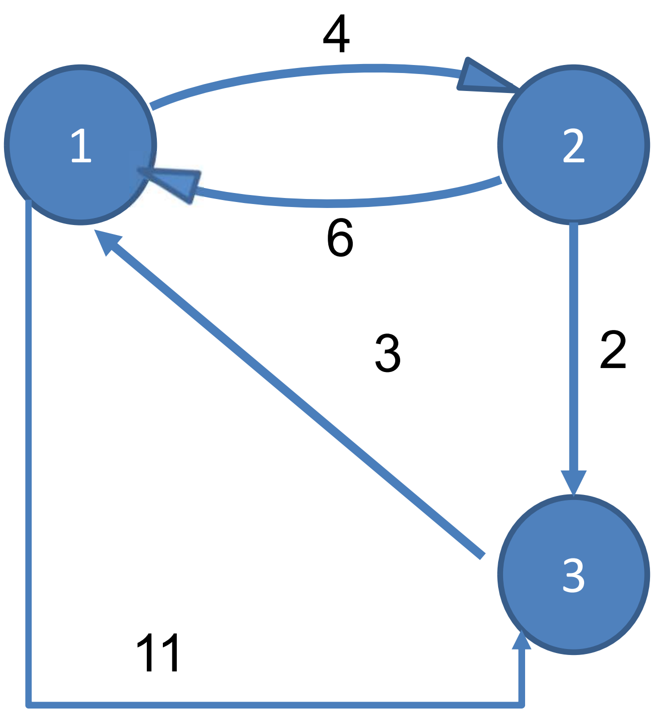
Example 1



D^{1}_{ϕ}	1	2	3
1	0	4	11
2	6	0	2
3	3	7	0

D^2	1	2	3
1	0	4	6
2	6	0	2
3	3	7	0

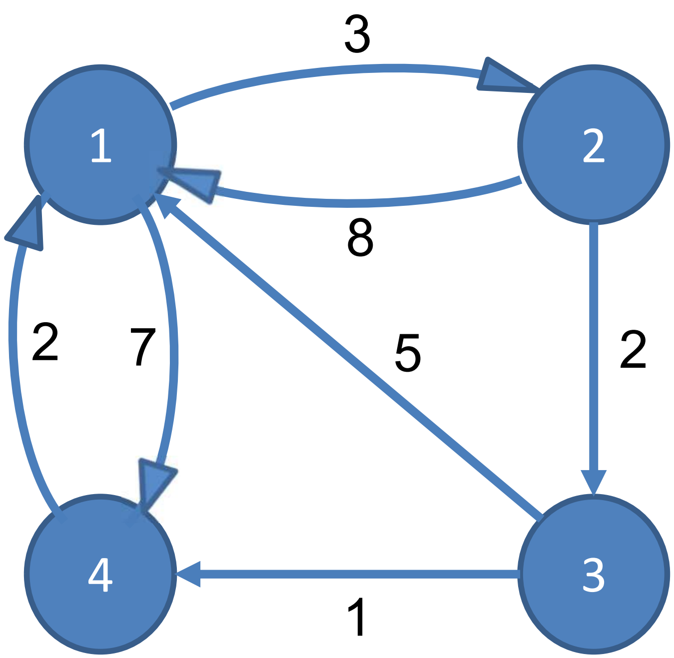
Example 1



D^2	1	2	3
1	0	4	6
2	6	0	2
3	3	7	0

D^3	1	2	3
1	0	4	6
2	5	0	2
3	3	7	0

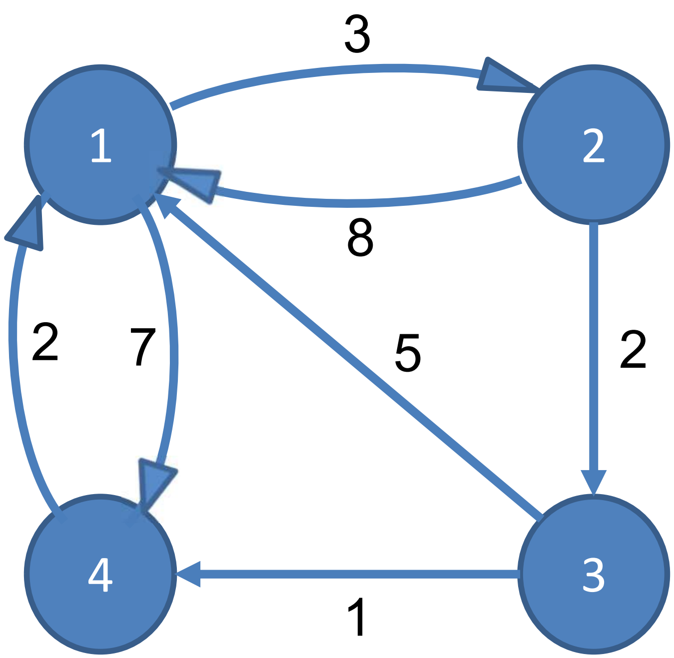
Example 2



$$W = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

$$D^0 = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

Example 2

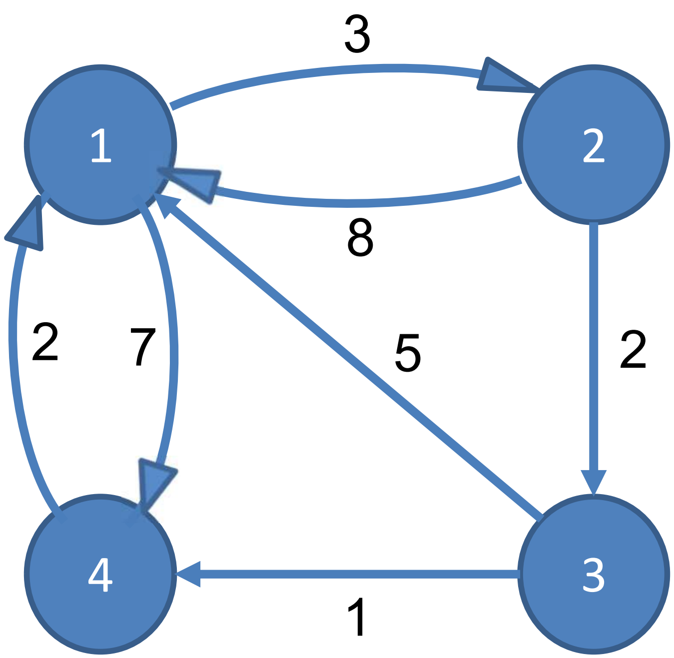


$$D^0 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & & \\ 2 & & & & \\ 3 & & & & \\ 4 & & & & \end{array} \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix}$$

$$D^1 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & 15 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & \infty & 0 \end{bmatrix} \end{array}$$

$$D^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 7 \\ 8 & 0 & 2 & 15 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

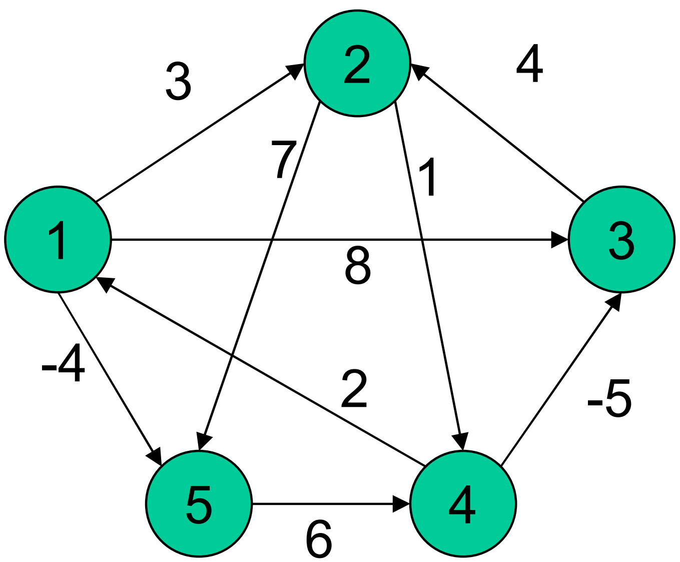
Example 2



$$D^3 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 6 \\ 7 & 0 & 2 & 3 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

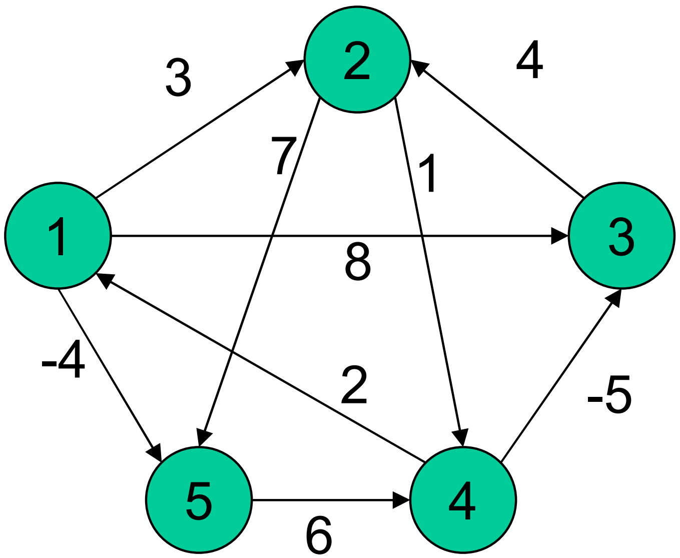
$$D^4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 6 \\ 5 & 0 & 2 & 3 \\ 3 & 6 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

Floyd-Warshall's Algorithm(Example 3)



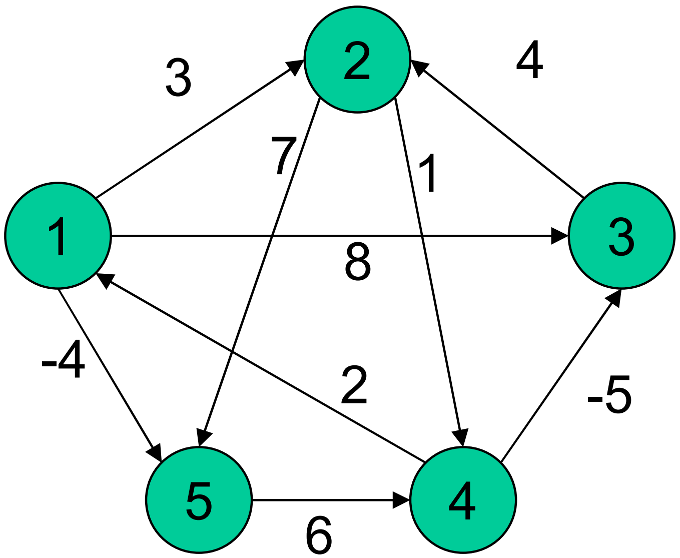
W

	1	2	3	4	5
1	0	3	8	∞	-4
2	∞	0	∞	1	7
3	∞	4	0	∞	∞
4	2	∞	-5	0	∞
5	∞	∞	∞	6	0



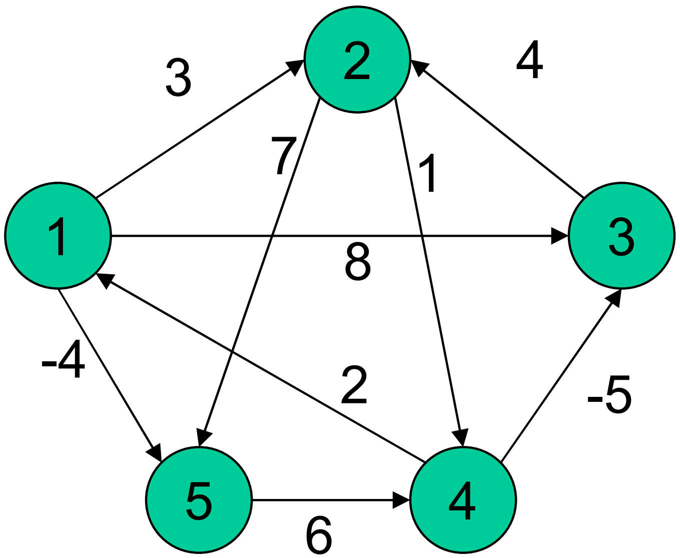
$D^{(0)}$

	1	2	3	4	5
1	0	3	8	∞	-4
2	∞	0	∞	1	7
3	∞	4	0	∞	∞
4	2	∞	-5	0	∞
5	∞	∞	∞	6	0



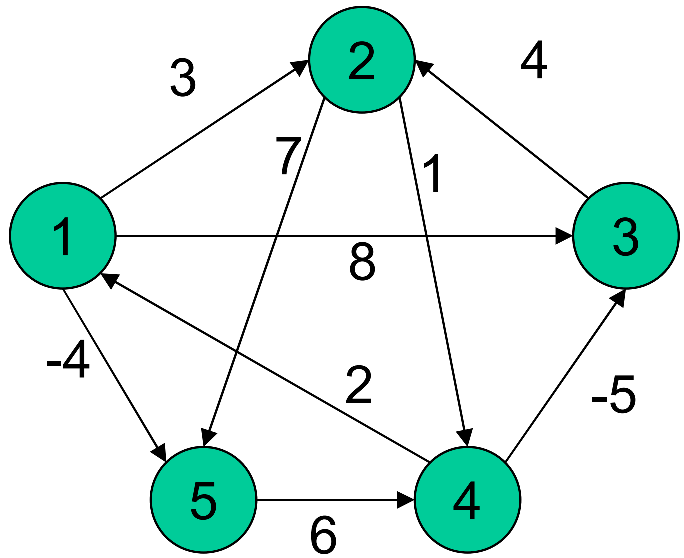
$D^{(1)}$

	1	2	3	4	5
1	0	3	8	∞	-4
2	∞	0	∞	1	7
3	∞	4	0	∞	∞
4	2	5	-5	0	-2
5	∞	∞	∞	6	0



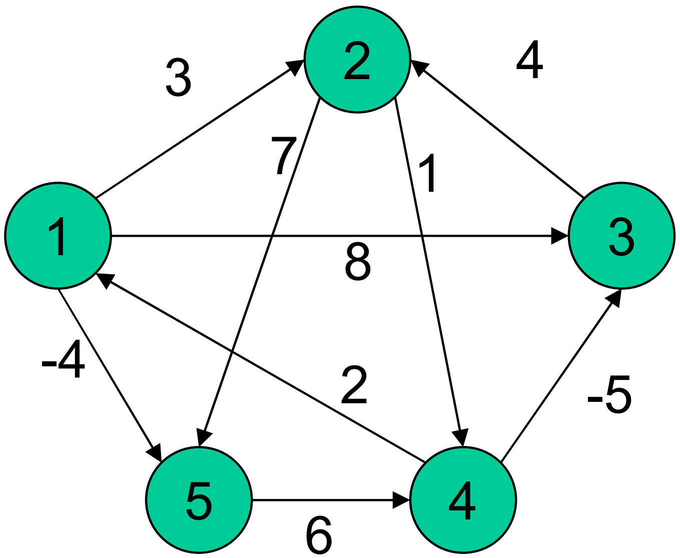
D^(2)

	1	2	3	4	5
1	0	3	8	4	-4
2	∞	0	∞	1	7
3	∞	4	0	5	11
4	2	5	-5	0	-2
5	∞	∞	∞	6	0



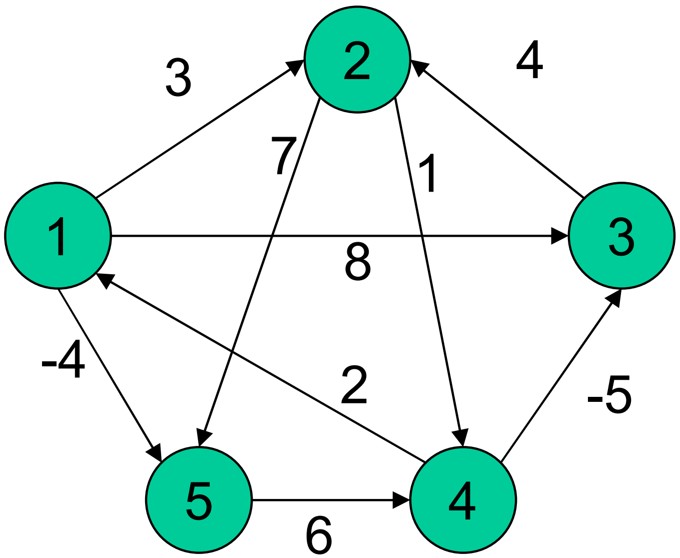
$D^{(3)}$

	1	2	3	4	5
1	0	3	8	4	-4
2	∞	0	∞	1	7
3	∞	4	0	5	11
4	2	-1	-5	0	-2
5	∞	∞	∞	6	0



D$^{(4)}$

	1	2	3	4	5
1	0	3	-1	4	-4
2	3	0	-4	1	-1
3	7	4	0	5	3
4	2	-1	-5	0	-2
5	8	5	1	6	0



$D^{(5)}$

	1	2	3	4	5
1	0	3	-1	2	-4
2	3	0	-4	1	-1
3	7	4	0	5	3
4	2	-1	-5	0	-2
5	8	5	1	6	0