Grafos com ciclos Negativos

SCC 503 - Alg. Estrut. Dados II

SSSP em grafos com arestas negativas

- O alg. de Dijkstra apresentado funciona para arestas negativas??
- O que acontece se houver ciclos negativos?
- Há uma solução:
  - COmputar o caminho a partir de um vértice origem s (SSSP)
    - Bellman-Ford

- Outra que computa a All-pair shortest Path (APSP) serve para arestas negativas, MAS SEM ciclos.
    - Floyd Warshall

Bellman-Ford

- Seja um vértice source s.
  - -> dist[s] = 0
- Se relaxarmos s → u, então dist[u] já contém o valor correto, certo ?
- Se relaxarmos u →v, então dist[v] estará também correto.
- Se relaxarmos todas arestas E, V-1 vezes então o menor caminho entre s e o vértice mais distante à s será corretamente computado (não é coincidência que o comprimento deste caminho é V-1, certo?)

Bellman-Ford

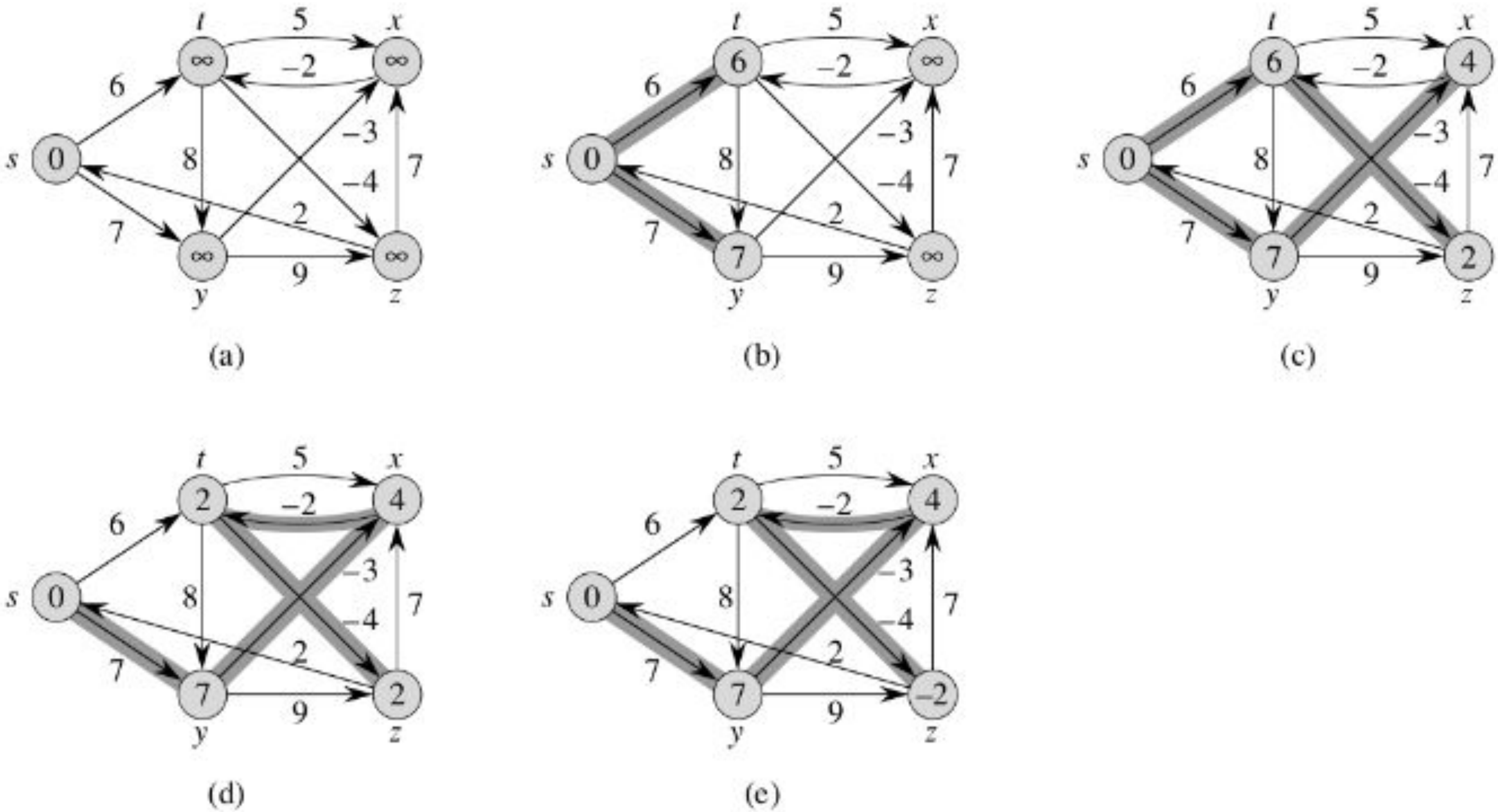
- Percorra o conjunto de arestas V-1 vezes
- Para cada vez, relaxar todas as arestas

- Qual a complexidade??

- É mais lento que Dijkstra..

- E se houver ciclo negativo?? existe solução de caminho mínimo???

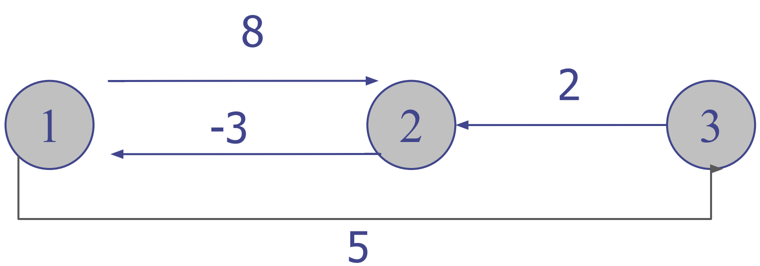
Bellman-Ford



Floyd Warshall

- Análogo ao Bellman-Ford, mas calcula o caminho mínimo para todos os pares de vértices..
- Utiliza Matriz de adjacência, por eficiência
- Complexidade alta: V^3

Floyd-Warshall

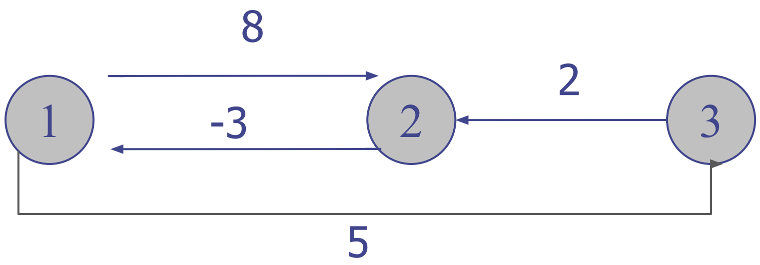


	1	2	3
1			
2			
3			

A

- Inicialmente os custos entre vértices adjacentes são inseridos na tabela *A*
- Pesos de *self-loops* não são considerados

Floyd-Warshall

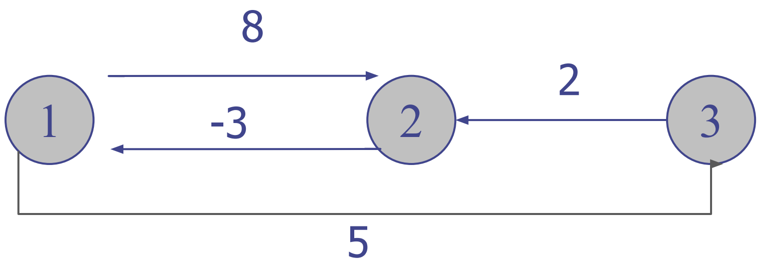


	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

- Inicialmente os custos entre vértices adjacentes são inseridos na tabela *A*
- Pesos de *self-loops* não são considerados

Floyd-Warshall

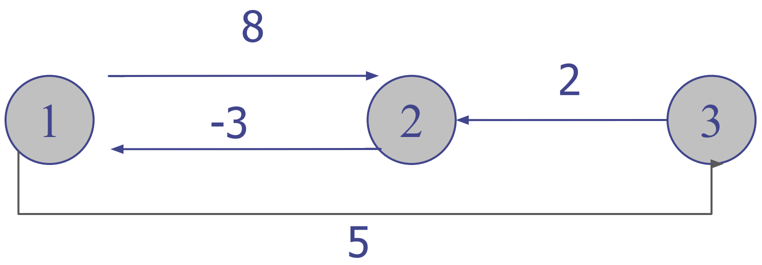


	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

- A matriz *A* é percorrida *|V|* vezes
- A cada iteração *k*, verifica-se se um caminho entre dois vértices *(v, w)* que passa também pelo vértice *k* é mais curto do que o caminho mais curto conhecido

Floyd-Warshall



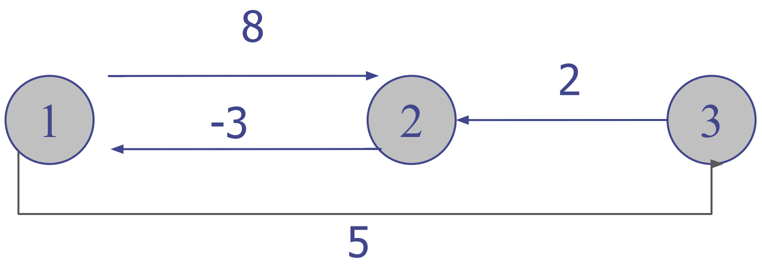
Ou seja:

$$A[v, w] = \min(A[v, w], A[v, k] + A[k, w])$$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

Floyd-Warshall



Ou seja:

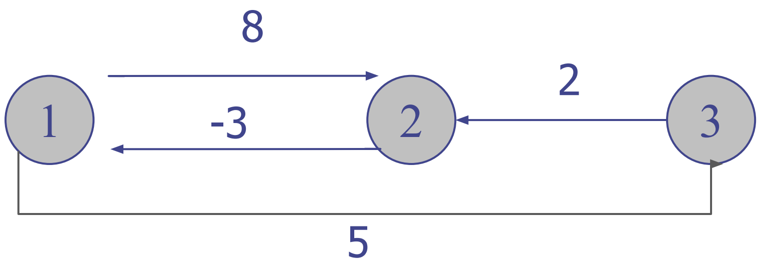
$$A[1, 1] = \min(A[1, 1], A[1, 1] + A[1, 1])$$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

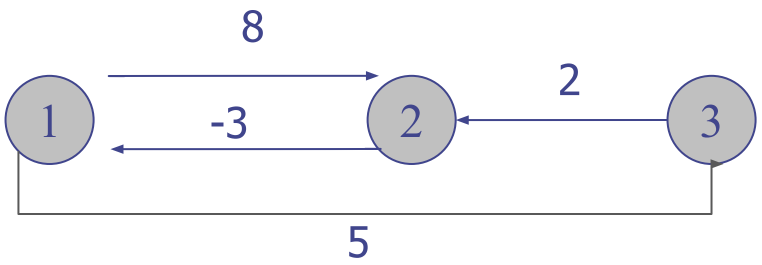
$A[1, 2] = \min(A[1, 2], A[1, 1] + A[1, 2])$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

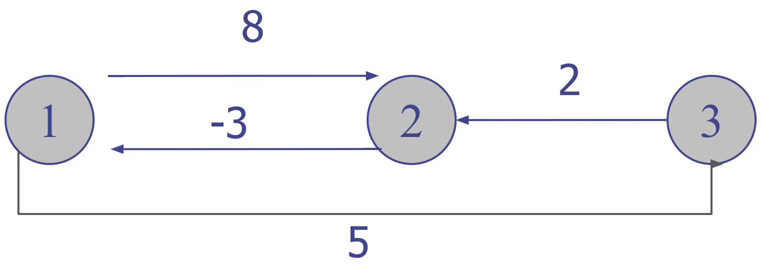
$$A[1,3] = \min(A[1,3],\ A[1,1] + A[1,3])$$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

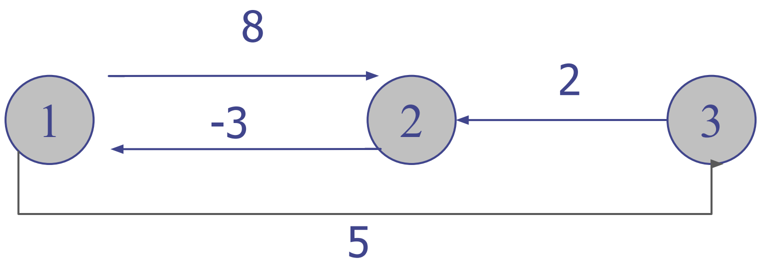
$A[2, 1] = \min(A[2, 1], A[2, 1] + A[1, 1])$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

$k = 1$

Floyd-Warshall



	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

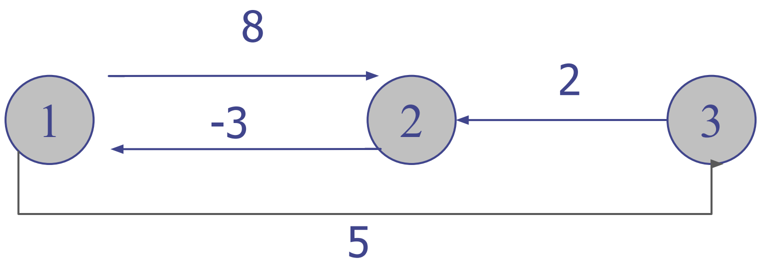
A

Ou seja:

$$A[2, 2] = \min(A[2, 2], A[2, 1] + A[1, 2])$$

$k = 1$

Floyd-Warshall



Ou seja:

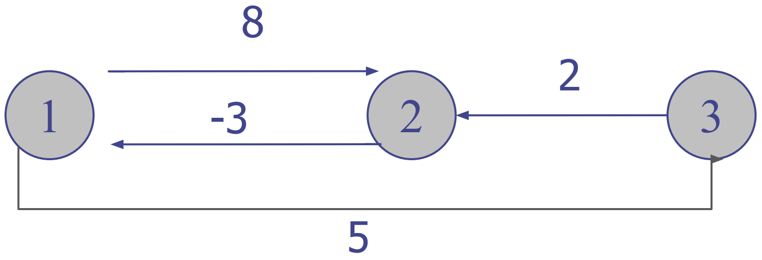
$A[2, 3] = \min(A[2, 3], A[2, 1] + A[1, 3])$

	1	2	3
1	0	8	5
2	-3	0	∞
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

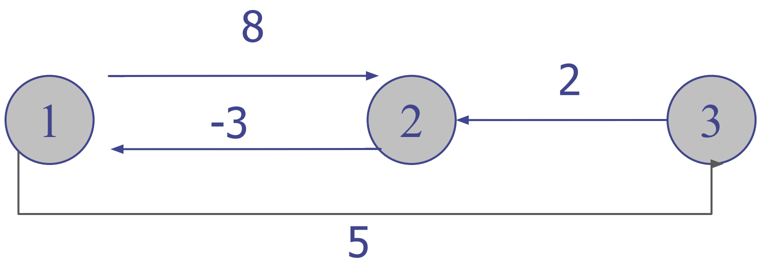
$$A[2, 3] = \min(A[2, 3], A[2, 1] + A[1, 3])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

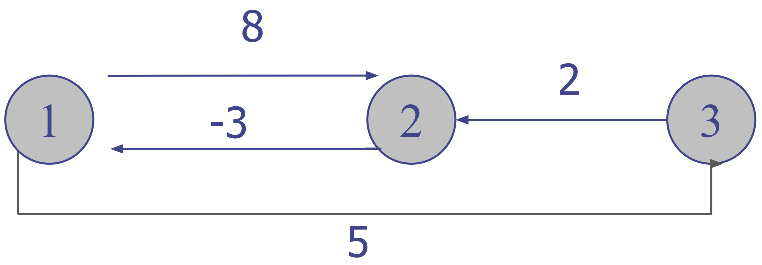
$$A[3,1] = \min(A[3,1],\ A[3,1] + A[1,3])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

k = 1

Floyd-Warshall



Ou seja:

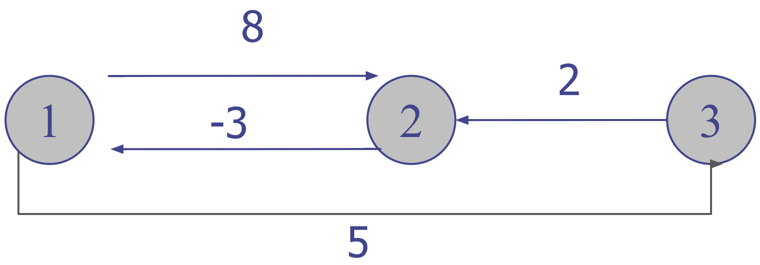
$$A[3, 2] = \min(A[3, 2], A[3, 1] + A[1, 2])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

$k = 1$

Floyd-Warshall



Ou seja:

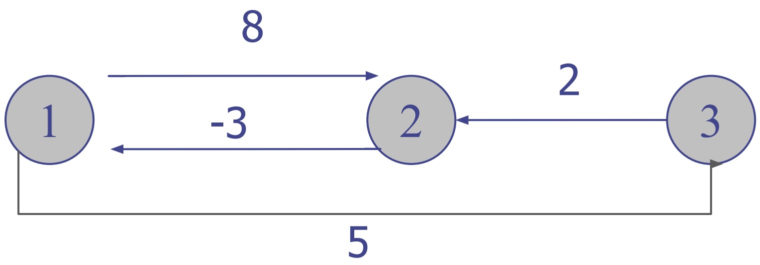
$A[3, 3] = \min(A[3, 3], A[3, 1] + A[1, 3])$

	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

$k = 1$

Floyd-Warshall

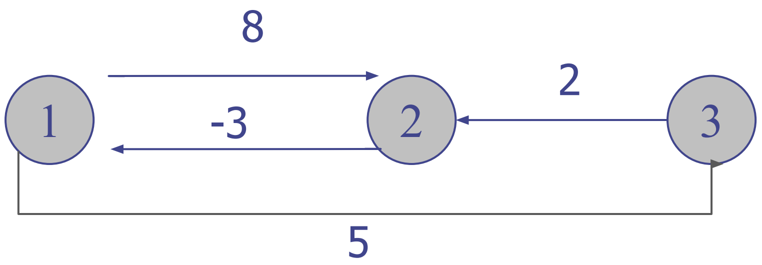


	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

- Ao final da iteração *k=1* tem-se todos os caminhos mais curtos entre *v* e *w* que podem passar pelo vértice 1.
- O processo se repete para *k=2* e *k=3*.

Floyd-Warshall



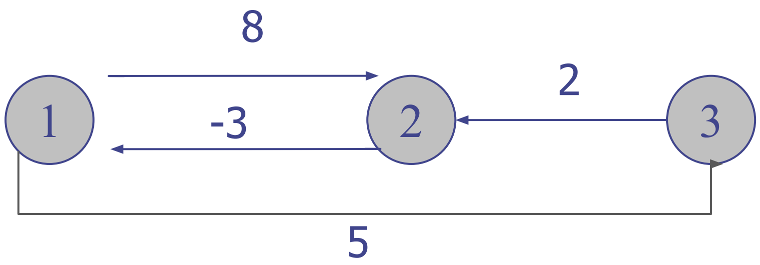
$$A[3, 1] = \min(A[3, 1], A[3, 2] + A[2, 1])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	∞	2	0

A

$k = 2$

Floyd-Warshall



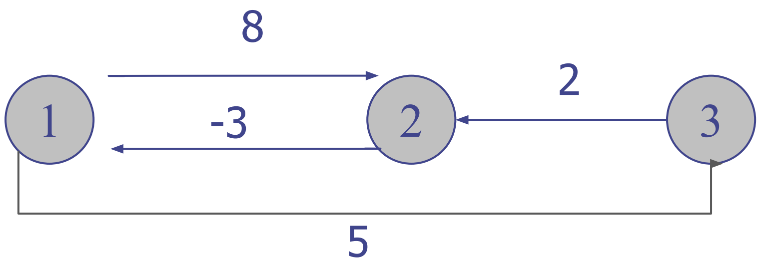
$$A[3, 1] = \min(A[3, 1], A[3, 2] + A[2, 1])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	-1	2	0

A

$k = 2$

Floyd-Warshall



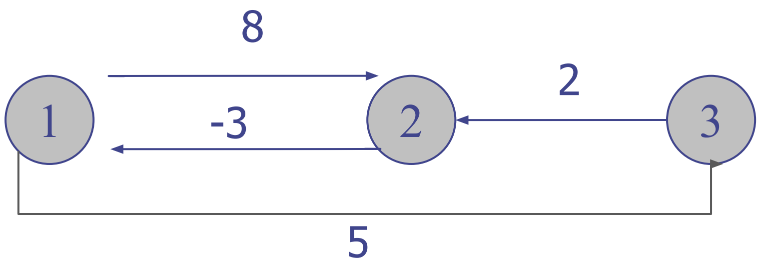
$$A[1, 2] = \min(A[1, 2], A[1, 3] + A[3, 2])$$

	1	2	3
1	0	8	5
2	-3	0	2
3	5	2	0

A

$k = 3$

Floyd-Warshall



$$A[1, 2] = \min(A[1, 2], A[1, 3] + A[3, 2])$$

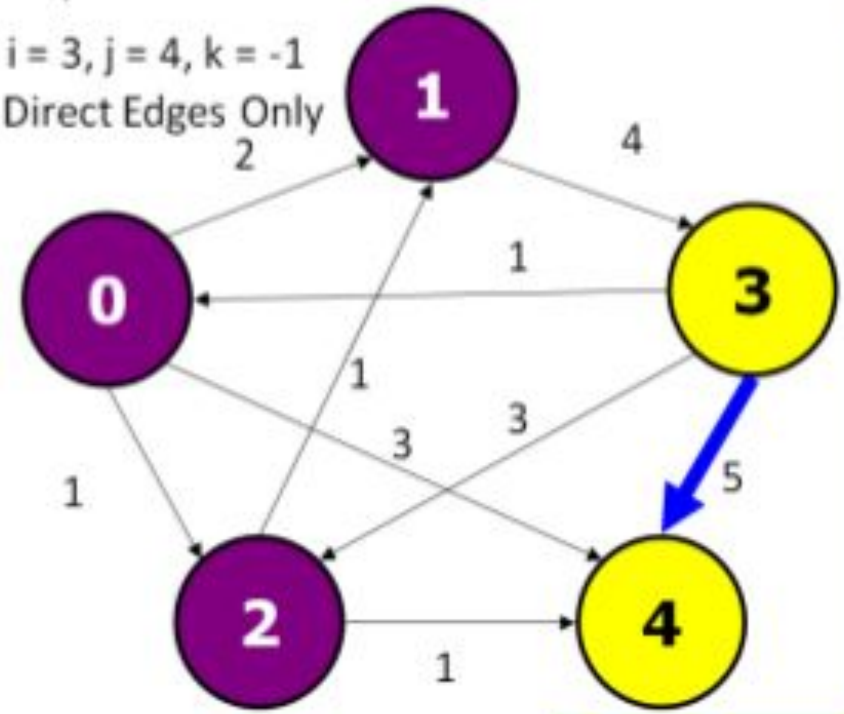
	1	2	3
1	0	7	5
2	-3	0	2
3	5	2	0

A

$k = 3$

Floyd Warshall - 2

i = 3, j = 4, k = -1
Direct Edges Only



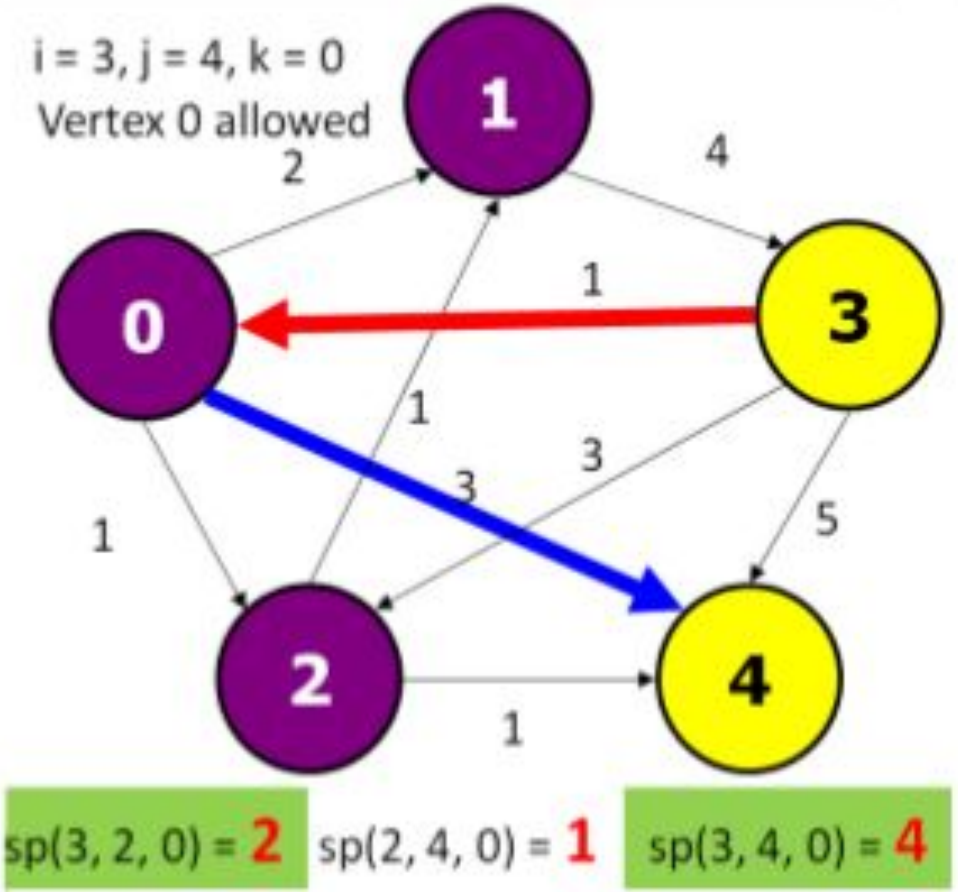
sp(3, 2, -1) = **3** sp(2, 4, -1) = **1** sp(3, 4, -1) = **5**

We will monitor these two values

The current content of Adjacency Matrix D
at **k = -1**

k = -1	0	1	2	3	4
0	0	2	1	∞	3
1	∞	0	∞	4	∞
2	∞	1	0	∞	1
3	1	∞	3	0	5
4	∞	∞	∞	∞	0

Floyd Warshall - 2

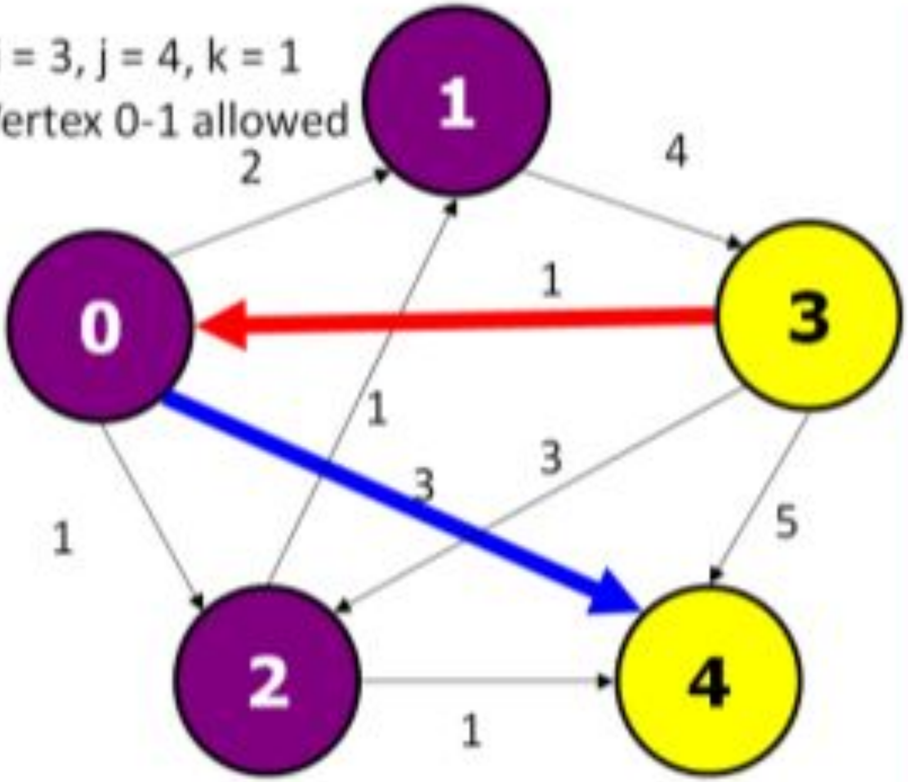


The current content of Adjacency Matrix D at **k = 0**

k = 0	0	1	2	3	4
0	0	2	1	∞	3
1	∞	0	∞	4	∞
2	∞	1	0	∞	1
3	1	3	2	0	4
4	∞	∞	∞	∞	0

Floyd Warshall - 2

i = 3, j = 4, k = 1
Vertex 0-1 allowed

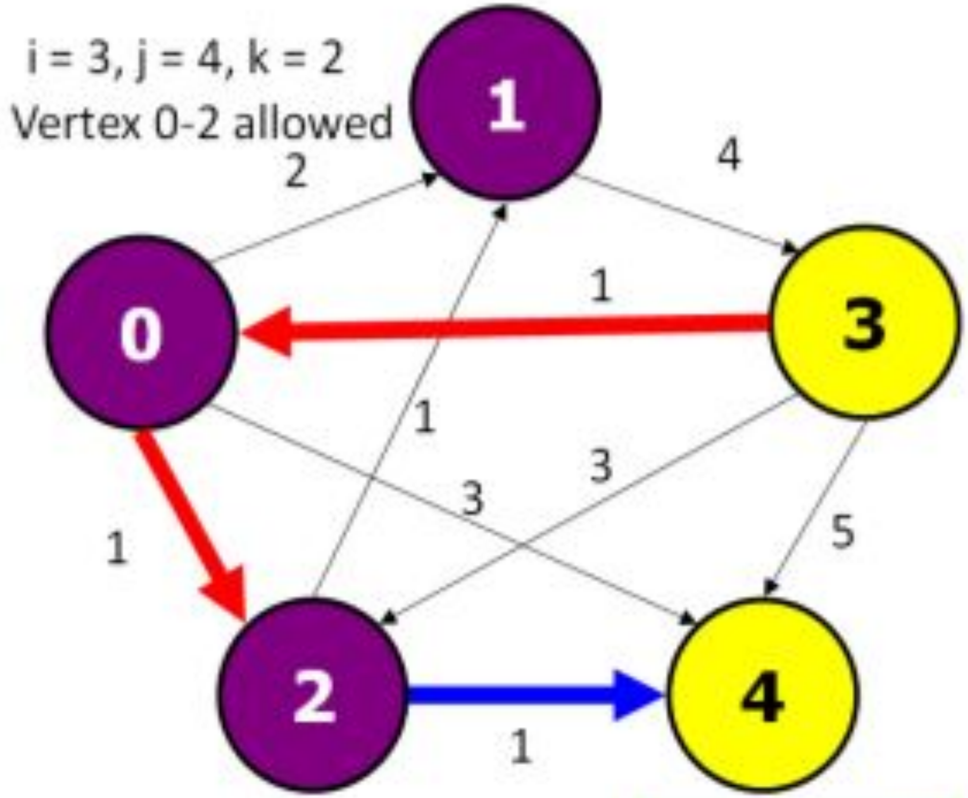


sp(3, 2, 1) = **2** sp(2, 4, 1) = **1** sp(3, 4, 1) = **4**

The current content of Adjacency Matrix D at **k = 1**

k = 1	0	1	2	3	4
0	0	2	1	6	3
1	∞	0	∞	4	∞
2	∞	1	0	5	1
3	1	3	2	0	**4**
4	∞	∞	∞	∞	0

Floyd Warshall - 2



sp(3, 2, 2) = **2** sp(2, 4, 2) = **1** sp(3, 4, 2) = **3**

The current content of Adjacency Matrix D at **k = 2**

k = 2	**0**	**1**	**2**	**3**	**4**
0	0	2	1	6	**2**
1	∞	0	∞	4	∞
2	∞	1	0	5	1
3	1	3	2	0	**3**
4	∞	∞	∞	∞	0

Floyd Warshall - 2

k = 1	0	1	2	3	4
0	0	2	1	6	**3**
1	∞	0	∞	4	∞
2	∞	1	0	5	1
3	1	3	2	0	4
4	∞	∞	∞	∞	0

k=1

k = 2	0	1	2	3	4
0	0	2	1	6	**2**
1	∞	0	∞	4	∞
2	∞	1	0	5	1
3	1	3	2	0	3
4	∞	∞	∞	∞	0

k=2