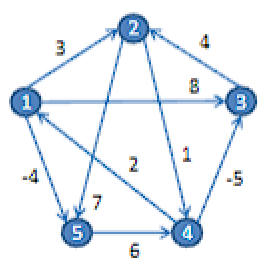
15 - Caminho mínimo - Floyd-Warshall

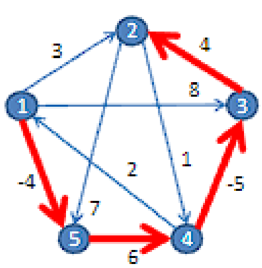
Determinar o caminho mínimo entre todos os pares de vértices do grafo:

- ***Custos positivos:*** para n vértices pode-se utilizar n repetições do método de Dijkstra → cada vez utilizando um vértice como raiz. $O(N^3)$
- ***Custos negativos:*** sem a presença de ciclos negativos, para n vértices pode utilizar n repetições do método de Bellman-Ford (com m arestas). $O(N^4)$

Exemplo:

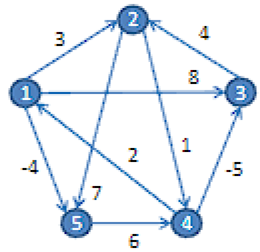


Pode-se observar que o grafo possui custos negativos. Porém ele não possui ciclo negativo? tem que verificar.



Distâncias Mínimas

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



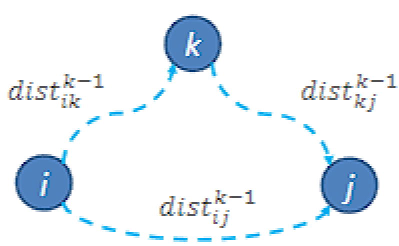
Distâncias Mínimas

	1	2	3	4	5
1	0	1	-3	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

Você faz a repetição o número n de vértices, assim, você vai ter o custo mínimo de cada vértice para cada vértice, por exemplo o custo mínimo de 5 para 1 é 8.

Método de Floyd-Warshall:

Faz o cálculo dos caminhos mais curtos entre todos pares de vértices, consegue lidar com arestas negativas se não tiver ciclo negativo. Utiliza do conceito de relaxação do comprimento dos caminhos, mais curtos de forma incremental aprimorando a estimativa de distância, até que o valor seja alcançado. O algoritmo analisa os caminhos entre os vértices i e j passando por cada um dos k vértices intermediários possíveis.



$$dist_{ik}^{k-1} \quad dist_{kj}^{k-1} \quad dist_{ij}^{k-1}$$

$$dist_{ij}^{k} = \min\{dist_{ij}^{k-1}, dist_{ik}^{k-1} + dist_{kj}^{k-1}\}$$

se dist[i,j] > dist[i,k] + dist[k,j] então
dist[i,j] ← dist[i,k] + dist[k,j]

Para $k = 1, 2, \ldots, n$

Algoritmo:

```
1.  para i = 1, . . ., n faça                              // Inicializar distâncias
        para j = 1, . . ., n | j ≠ i faça  dist[i, j] ← ∞;
        dist[i, i] ← 0;
2.  para toda aresta (i, j) ∈ E(G) faça  dist[i, j] ← d_ij;
3.  para  k = 1, . . ., n  faça                            // Para cada possível intermediário
        para i = 1, . . ., n faça
            para j = 1, . . ., n faça
                se dist[i, j] > dist[i, k] + dist[k, j] então   // Testar caminho de i para j
                    dist[i, j] ← dist[i, k] + dist[k, j];       // Atualizar a distância
```

Exemplo:

$Dist^0$

	1	2	3	4	5
1	$\boxed{0}$	3	8	∞	-4
2	$\boxed{\infty}$	0	∞	1	7
3	$\boxed{\infty}$	4	0	∞	∞
4	$\boxed{2}$	∞	-5	0	∞
5	$\boxed{\infty}$	∞	∞	6	0

$$dist_{5j} = \min\{dist_{5j}, dist_{51} + dist_{1j}\}$$

$k = 1$

$Dist^1$

	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

$Dist^1$

	1	2	3	4	5
1	0	$\boxed{3}$	8	∞	-4
2	∞	$\boxed{0}$	∞	1	7
3	∞	$\boxed{4}$	0	∞	∞
4	2	$\boxed{5}$	-5	0	-2
5	∞	$\boxed{\infty}$	∞	6	0

$$dist_{5j} = \min\{dist_{5j}, dist_{52} + dist_{2j}\}$$

$k = 2$

$Dist^2$

	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

Dist²

	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

$$dist_{5j} = \min\{dist_{5j}, dist_{53} + dist_{3j}\}$$

$k = 3$

Dist³

	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

Dist³

	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

$$dist_{5j} = \min\{dist_{5j}, dist_{54} + dist_{4j}\}$$

$k = 4$

Dist⁴

	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

Dist⁴

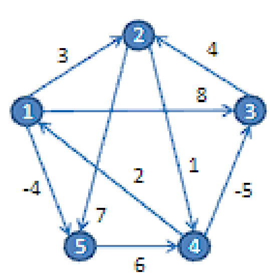
	1	2	3	4	5
1	0	3	-1	4	-4
2	3	0	-4	1	-1
3	7	4	0	5	3
4	2	-1	-5	0	-2
5	8	5	1	6	0

$$dist_{5j} = \min\{dist_{5j}, dist_{55} + dist_{5j}\}$$

$k = 5$

Dist⁵

	1	2	3	4	5
1	0	1	-3	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



Distâncias Mínimas

	1	2	3	4	5
1	0	1	-3	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