Shortest Path Problem II

SWE2016-44

Floyd Warshall Algorithm

Floyd Warshall Algorithm

In graph theory, the Floyd-Warshall (FW) algorithm is an All-Pairs Shortest Path (APSP) algorithm. This means it can find the shortest path between all pairs of nodes.

The time complexity to run FW is $O(V^3)$ which is ideal for graphs no larger than a couple hundreds nodes.

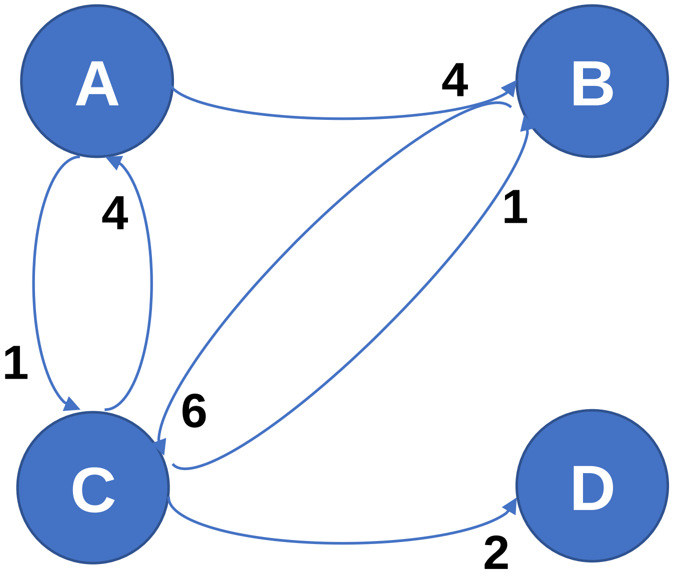
Shortest Path Algorithms

	BFS	Dikstra's	Bellman Ford	Floyd Warshall
Complexity	O(V+E)	O((V+E)logV)	O(VE)	$O(V^3)$
Recommended graph size	Large	Large/Medium	Medium/Small	Small
Good for APSP	Only unweighted graphs	Ok	Bad	Yes
Can detect negative cycles	No	No	Yes	Yes
SP on graph with weighted edges	Incorrect SP answer	Best algorithm	Works	Bad in general
SP on graph with unweighted edges	Best algorithm	Ok	Bad	Bad in general

Competitive Programming 3, P. 161, Steven & Felix Halim

Graph Setup

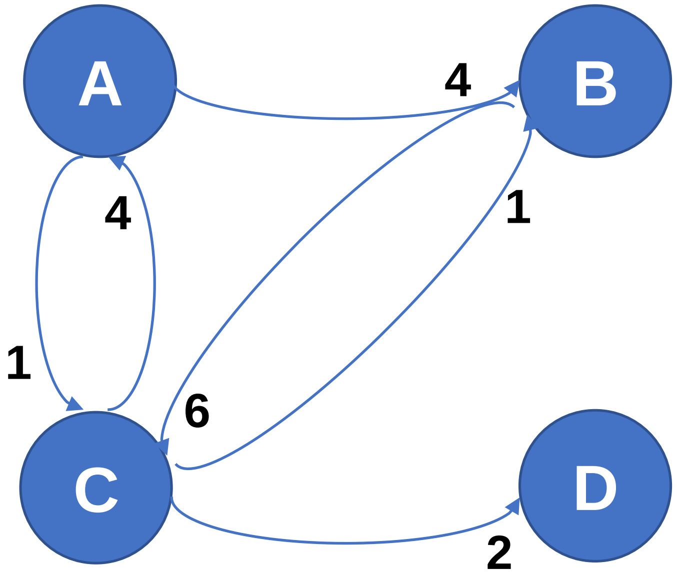
If there is no edge from node i to node j then set the edge value for m[i][j] to be positive infinity.



$$\begin{array}{c c} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{bmatrix} 0 & 4 & 1 & \infty \\ \infty & 0 & 6 & \infty \\ 4 & 1 & 0 & 2 \\ \infty & \infty & \infty & 0 \end{bmatrix} \end{array}$$

Graph Setup

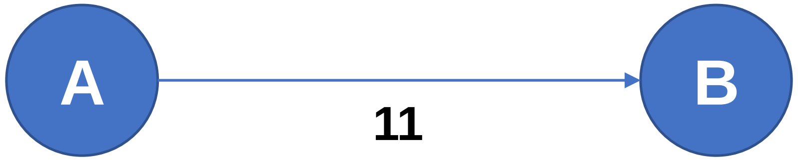
If there is no edge from node i to node j then set the edge value for m[i][j] to be positive infinity.



$$\begin{array}{c|cccc|} & A & B & C & D \\ A & 0 & 4 & 1 & \infty \\ B & \infty & 0 & 6 & \infty \\ C & 4 & 1 & 0 & 2 \\ D & \infty & \infty & \infty & 0 \end{array}$$

Floyd Warshall Algorithm

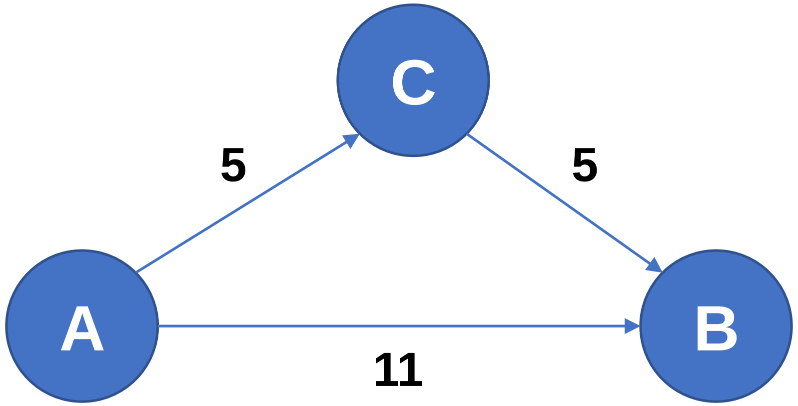
The main idea behind the Floyd-Warshall algorithm is to gradually build up all intermediate routes between nodes i and j to find the optimal path.



Suppose our adjacency matrix tells us that the distance from a to b is: m[a][b] = 11

Floyd Warshall Algorithm

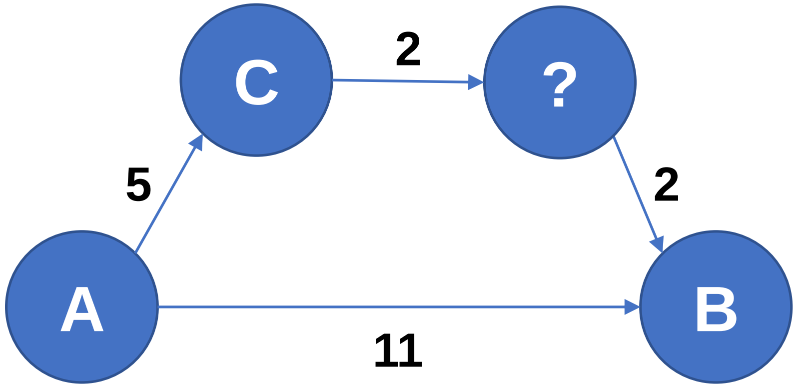
The main idea behind the Floyd-Warshall algorithm is to gradually build up all intermediate routes between nodes i and j to find the optimal path.

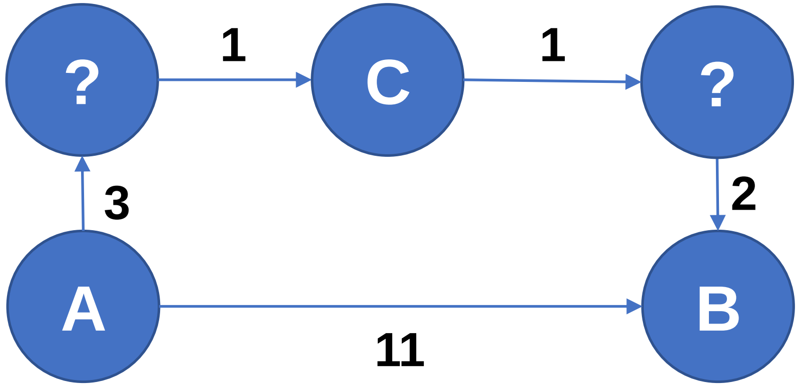


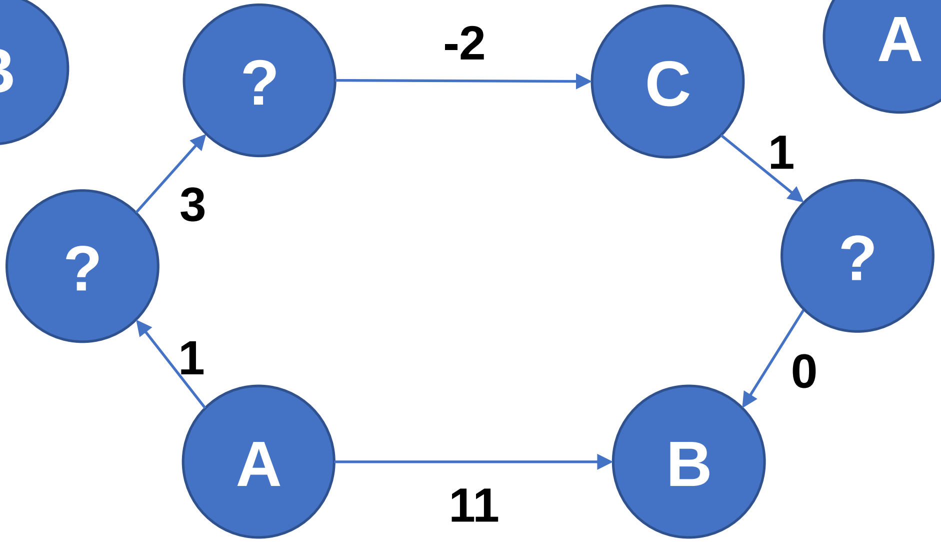
Suppose there exists a third node, c. If m[a][c]+m[c][b]<m[a][b] then it's better to route through c!

Floyd Warshall Algorithm

The goal of Floyd-Warshall is to eventually consider going through all possible intermediate nodes on paths of different lengths.





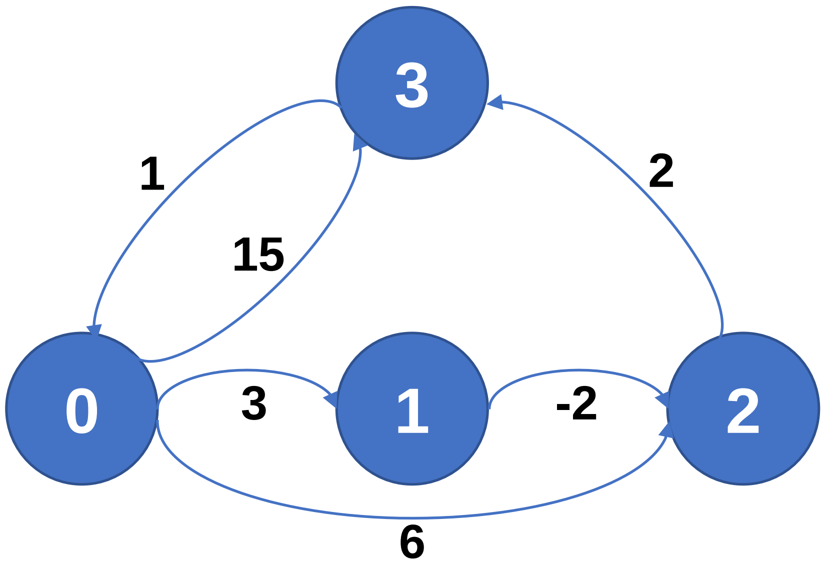


Floyd Warshall Algorithm

```
if d[i][j] > d[i][k] + d[k][j]:
        dp[i][j] = dp[i][k] + dp[k][j]
        path[i][j] = path[k][j]
```

Find the best distance from i to j through node k

Floyd Warshall Algorithm




```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Find the best distance from i to j through node k

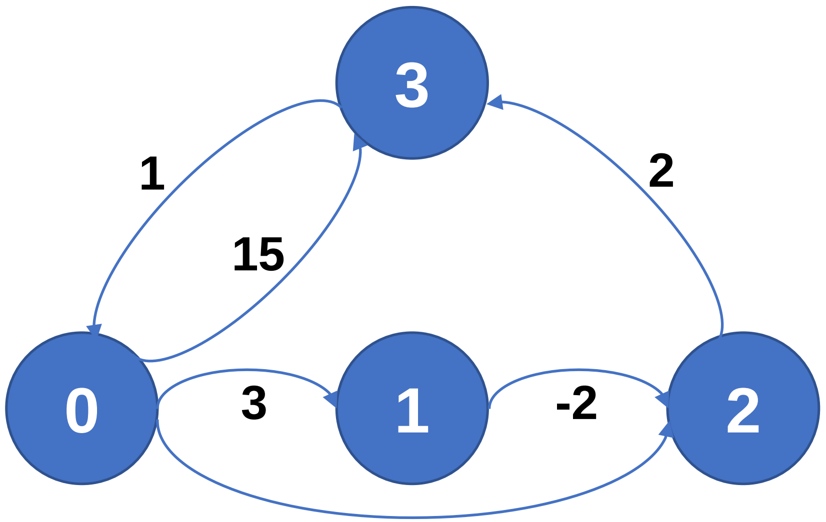
distance

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & \infty & \infty & 0 \end{array}$$

path

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & n & n & n \end{array}$$

Floyd Warshall Algorithm



k=0, i=0

j	0	d[0][0]	d[0][0] + d[0][0]
	1	d[0][1]	d[0][0] + d[0][1]
	2	d[0][2]	d[0][0] + d[0][2]
	3	d[0][3]	d[0][0] + d[0][3]

distance

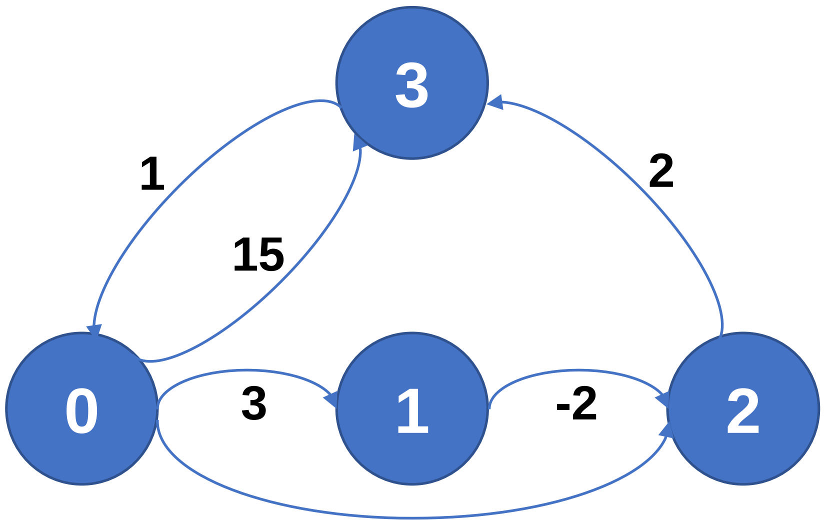
$$
\begin{array}{c|cccc|}
 & 0 & 1 & 2 & 3 \\
\hline
0 & 0 & 3 & 6 & 15 \\
1 & \infty & 0 & -2 & \infty \\
2 & \infty & \infty & 0 & 2 \\
3 & 1 & \infty & \infty & 0 \\
\end{array}
$$

path

$$
\begin{array}{c|cccc|}
 & 0 & 1 & 2 & 3 \\
\hline
0 & n & 0 & 0 & 0 \\
1 & n & n & 1 & n \\
2 & n & n & n & 2 \\
3 & 3 & n & n & n \\
\end{array}
$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

  d[i][j] = d[i][k] + d[k][j]

  path[i][j] = path[k][j]

k=0, i=0

j				
0	d[0][0]	=	d[0][0] + d[0][0]
1	d[0][1]	=	d[0][0] + d[0][1]
2	d[0][2]	=	d[0][0] + d[0][2]
3	d[0][3]	=	d[0][0] + d[0][3]

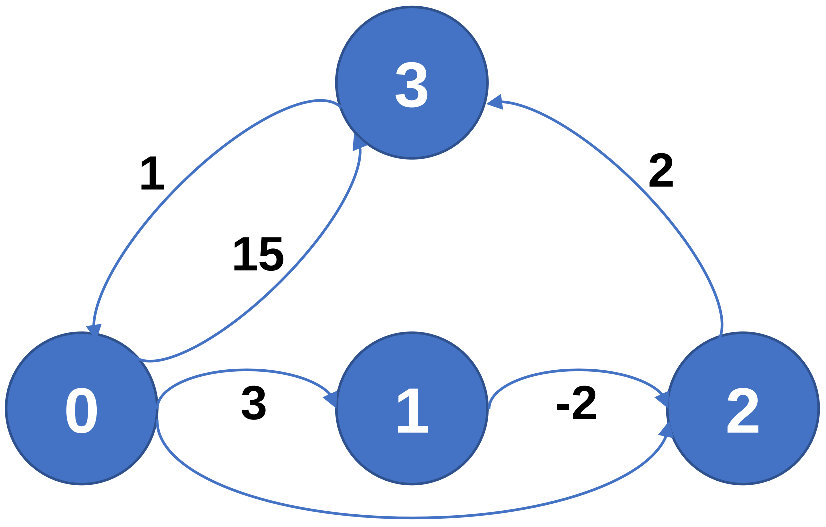
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & \infty & \infty & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & n & n & n \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:
    d[i][j] = d[i][k] + d[k][j]
    path[i][j] = path[k][j]

k=0, i=1

	j		
j	0	d[1][0]	d[1][0] + d[0][0]
	1	d[1][1]	d[1][0] + d[0][1]
	2	d[1][2]	d[1][0] + d[0][2]
	3	d[1][3]	d[1][0] + d[0][3]

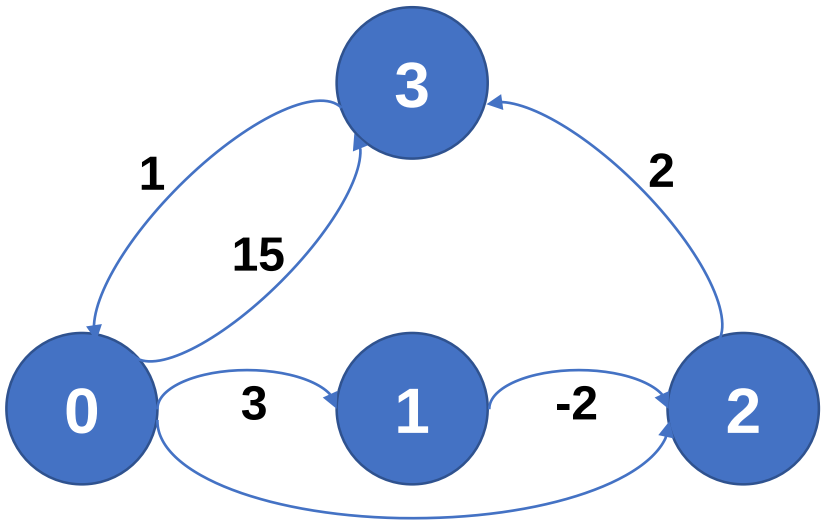
distance

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 6 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 0 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & n & n & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



k=0, i=1

j	0	d[1][0]	=	d[1][0] + d[0][0]
	1	d[1][1]	<	d[1][0] + d[0][1]
	2	d[1][2]	<	d[1][0] + d[0][2]
	3	d[1][3]	=	d[1][0] + d[0][3]

distance

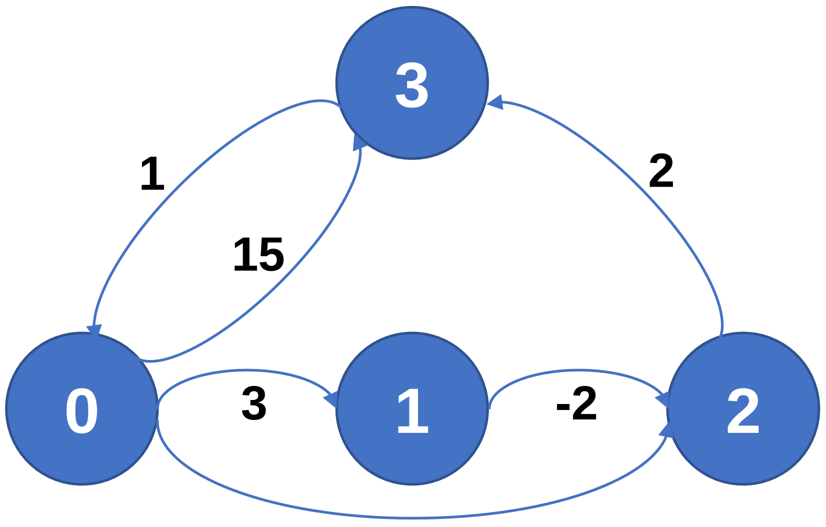
$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 3 & 6 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} n & 0 & 0 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & n & n & n \end{bmatrix} \end{array}$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

k=0, i=2

j	0	d[2][0]	d[2][0] + d[0][0]
	1	d[2][1]	d[2][0] + d[0][1]
	2	d[2][2]	d[2][0] + d[0][2]
	3	d[2][3]	d[2][0] + d[0][3]

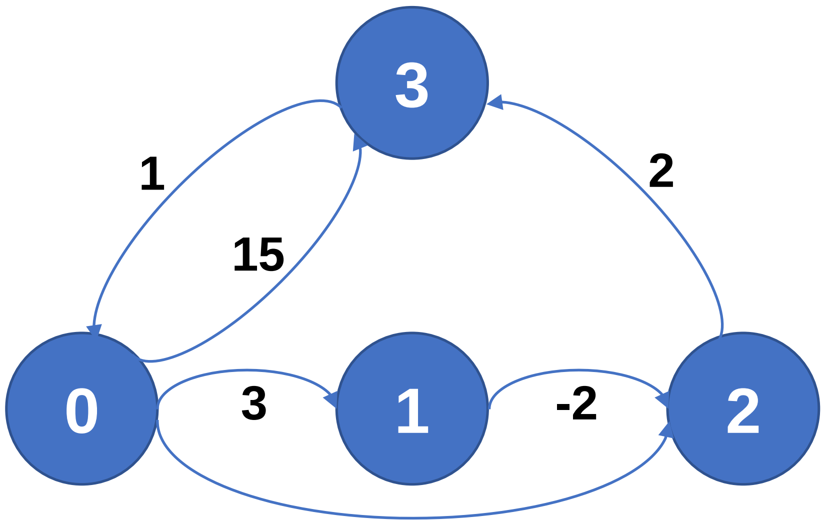
distance

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & \infty & \infty & 0 \end{array}$$

path

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & n & n & n \end{array}$$

Floyd Warshall Algorithm



k=0, i=2

j	0	d[2][0]	=	d[2][0] + d[0][0]
	1	d[2][1]	=	d[2][0] + d[0][1]
	2	d[2][2]	<	d[2][0] + d[0][2]
	3	d[2][3]	<	d[2][0] + d[0][3]

distance

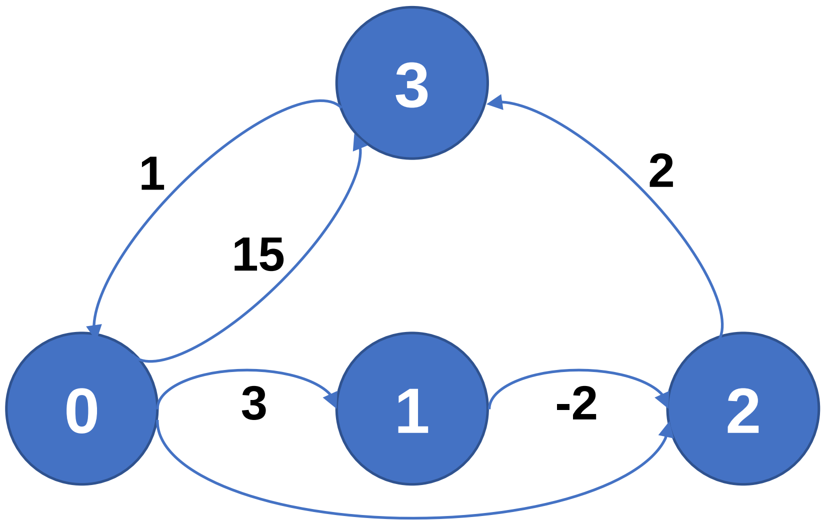
$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & \infty & \infty & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & n & n & n \end{array}$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

  d[i][j] = d[i][k] + d[k][j]

  path[i][j] = path[k][j]

k=0, i=3

	j		
j	0	d[3][0]	d[3][0] + d[0][0]
	1	d[3][1]	d[3][0] + d[0][1]
	2	d[3][2]	d[3][0] + d[0][2]
	3	d[3][3]	d[3][0] + d[0][3]

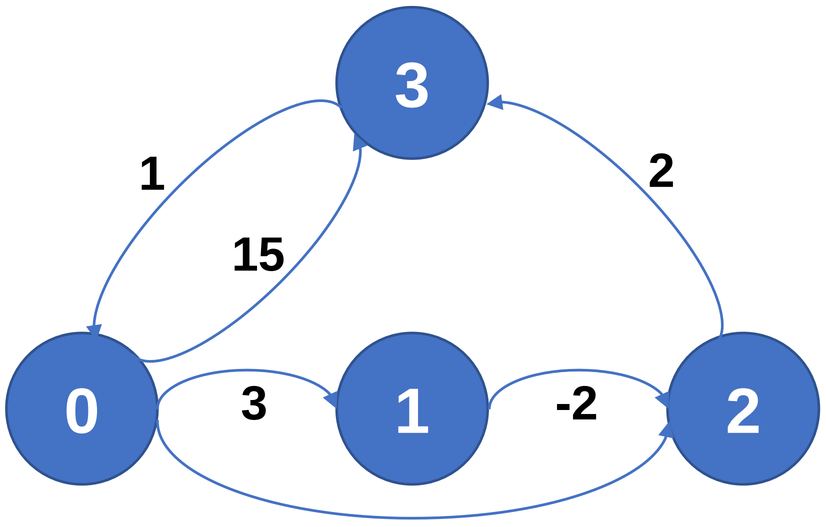
distance

	0	1	2	3
0	0	3	6	15
1	∞	0	-2	∞
2	∞	∞	0	2
3	1	∞	∞	0

path

	0	1	2	3
0	n	0	0	0
1	n	n	1	n
2	n	n	n	2
3	3	n	n	n

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

k=0, i=3

j				
0	d[3][0]	=	d[3][0] + d[0][0]	
1	d[3][1]	>	d[3][0] + d[0][1]	4
2	d[3][2]	>	d[3][0] + d[0][2]	7
3	d[3][3]	<	d[3][0] + d[0][3]	

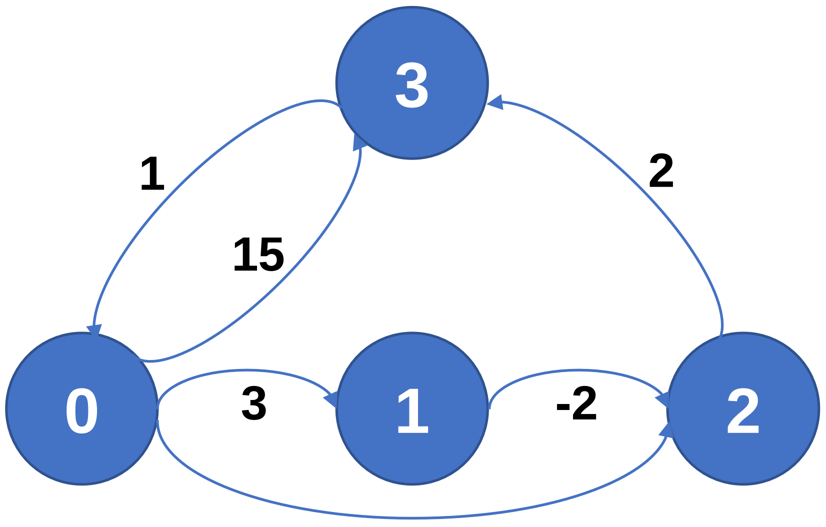
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & \infty & \infty & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & n & n & n \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:
    d[i][j] = d[i][k] + d[k][j]
    path[i][j] = path[k][j]

k=0, i=3

j				
0	d[3][0]	=	d[3][0] + d[0][0]	
1	d[3][1]	>	d[3][0] + d[0][1]	4
2	d[3][2]	>	d[3][0] + d[0][2]	7
3	d[3][3]	<	d[3][0] + d[0][3]	

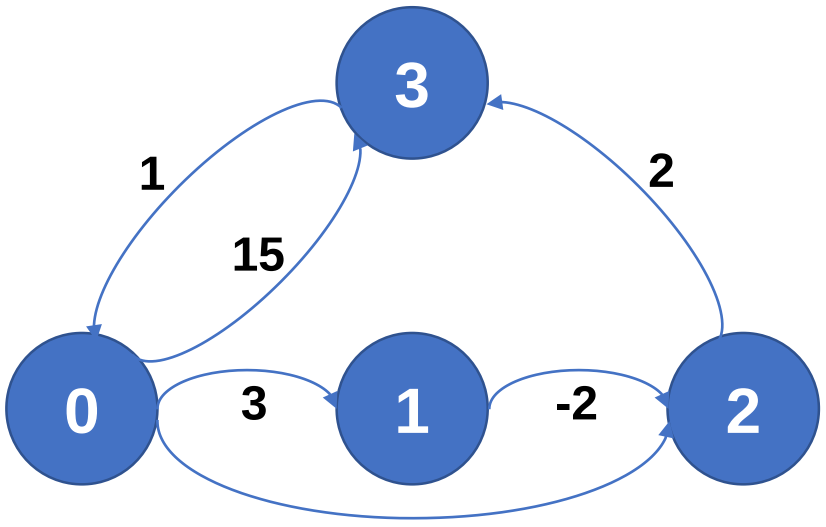
distance

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 6 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & 4 & 7 & 0 \end{array}$$

path

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 0 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & 0 & 0 & n \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

  d[i][j] = d[i][k] + d[k][j]

  path[i][j] = path[k][j]

k=1, i=0

j		
0	d[0][0]	d[0][1] + d[1][0]
1	d[0][1]	d[0][1] + d[1][1]
2	d[0][2]	d[0][1] + d[1][2]
3	d[0][3]	d[0][1] + d[1][3]

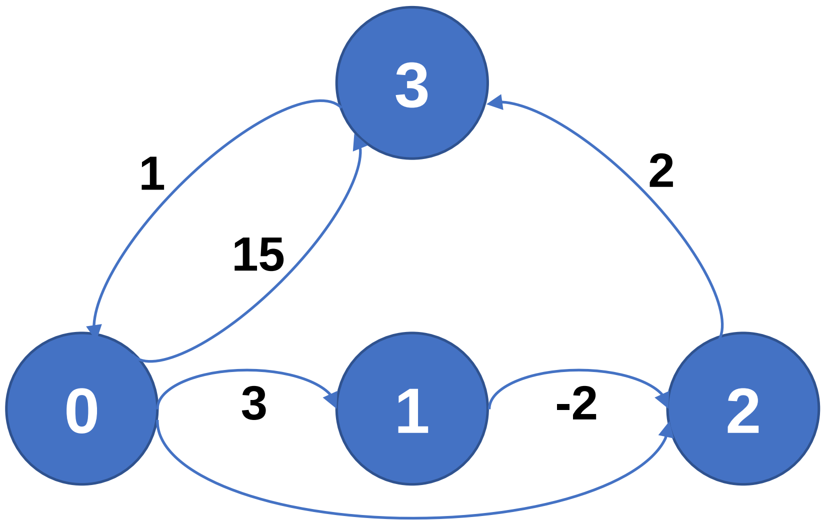
distance

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 6 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 0 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



k=1, i=0

j				
0	d[0][0]	<	d[0][1] + d[1][0]	
1	d[0][1]	=	d[0][1] + d[1][1]	
2	d[0][2]	>	d[0][1] + d[1][2]	1
3	d[0][3]	<	d[0][1] + d[1][3]	

if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

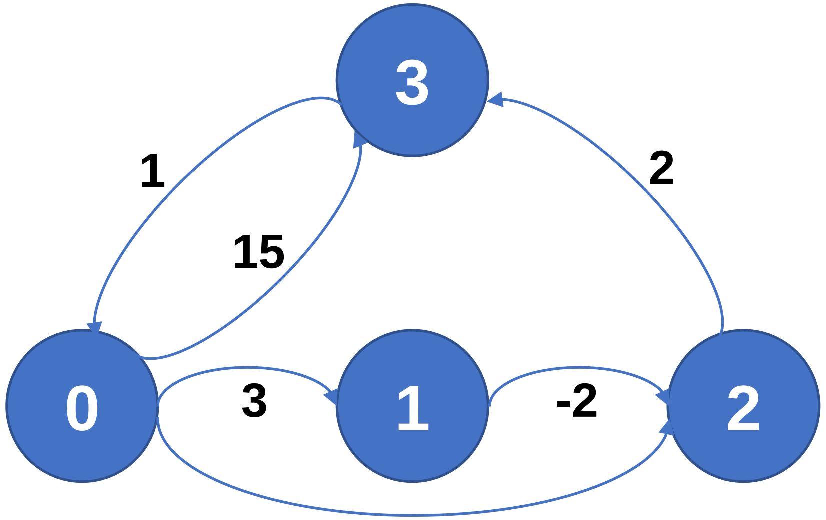
distance

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 6 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 0 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm




```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

k=1, i=0

j				
0	d[0][0]	<	d[0][1] + d[1][0]	
1	d[0][1]	=	d[0][1] + d[1][1]	
2	d[0][2]	>	d[0][1] + d[1][2]	1
3	d[0][3]	<	d[0][1] + d[1][3]	

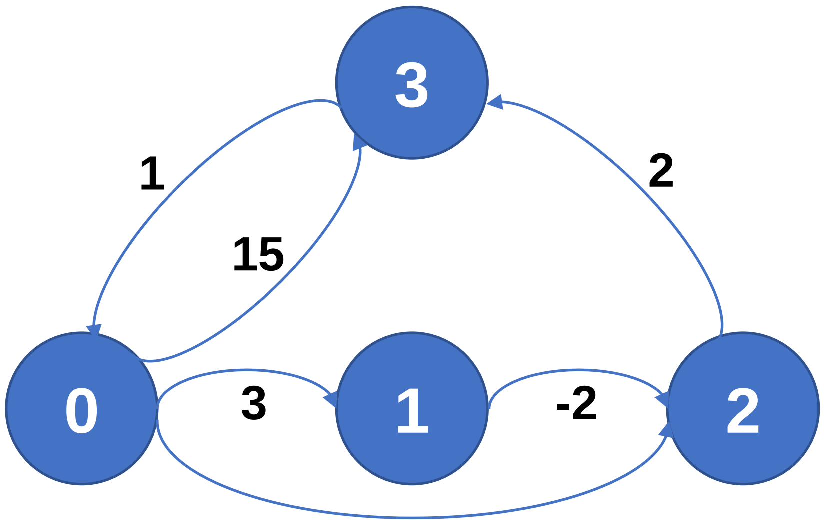
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 1 & 15 \\ 1 & \infty & 0 & -2 & \infty \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & 4 & 7 & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 1 & 0 \\ 1 & n & n & 1 & n \\ 2 & n & n & n & 2 \\ 3 & 3 & 0 & 0 & n \end{array}$$

Floyd Warshall Algorithm



k=1, i=1

	j		
	0	d[1][0]	d[1][1] + d[1][0]
	1	d[1][1]	d[1][1] + d[1][1]
	2	d[1][2]	d[1][1] + d[1][2]
	3	d[1][3]	d[1][1] + d[1][3]

if d[i][j] > d[i][k] + d[k][j]:
    d[i][j] = d[i][k] + d[k][j]
    path[i][j] = path[k][j]

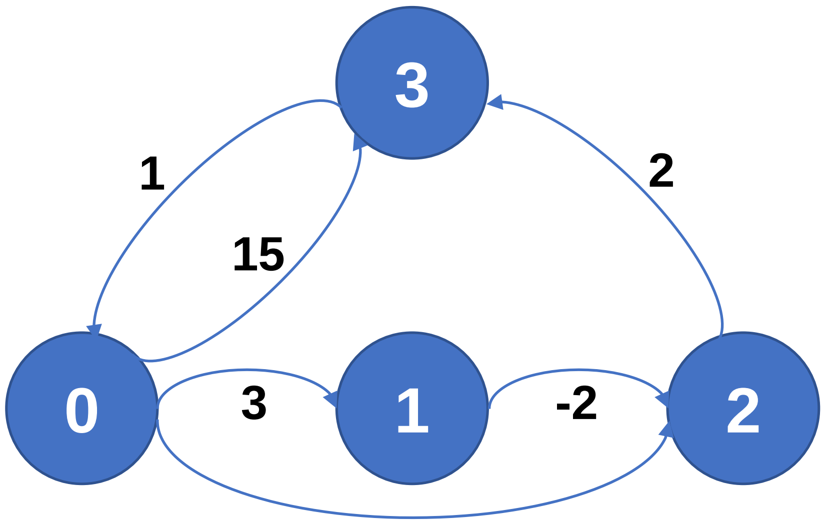
distance

$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix}
\end{array}
$$

path

$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix}
\end{array}
$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

k=1, i=1

j				
0	d[1][0]	=	d[1][1] + d[1][0]
1	d[1][1]	=	d[1][1] + d[1][1]
2	d[1][2]	=	d[1][1] + d[1][2]
3	d[1][3]	=	d[1][1] + d[1][3]

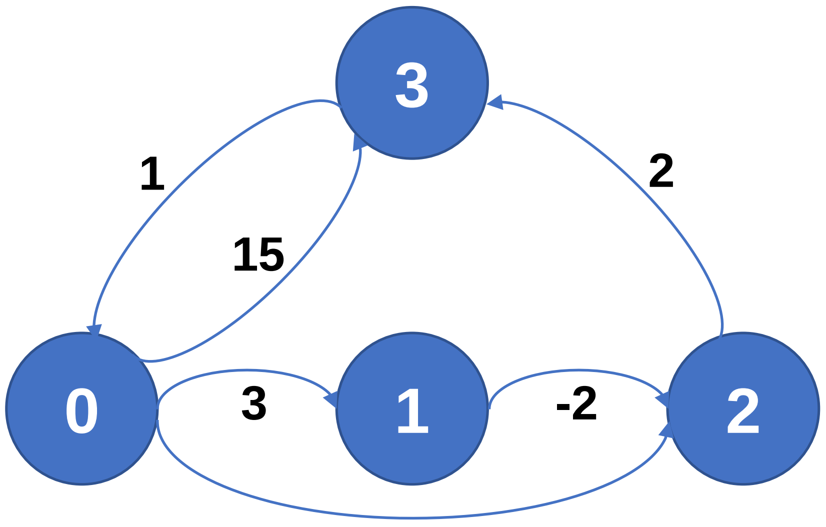
distance

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

k=1, i=2

j	0	d[2][0]	d[2][1] + d[1][0]
	1	d[2][1]	d[2][1] + d[1][1]
	2	d[2][2]	d[2][1] + d[1][2]
	3	d[2][3]	d[2][1] + d[1][3]

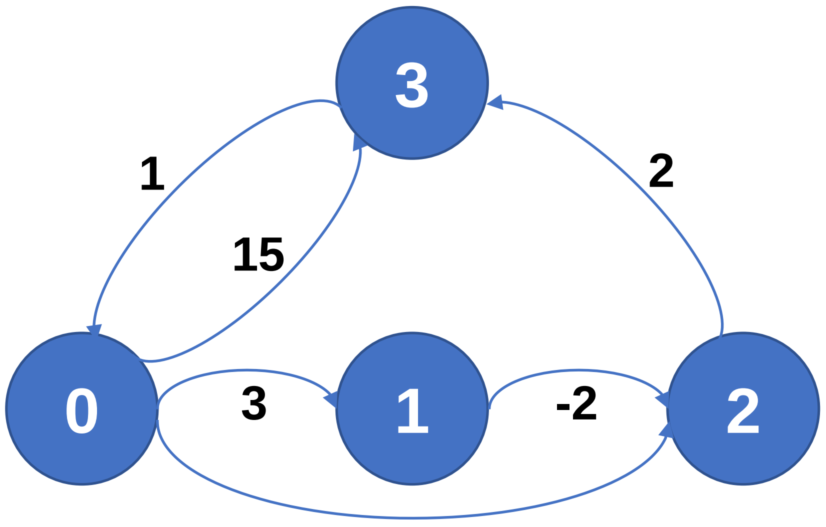
distance

$$\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix}
\end{array}$$

path

$$\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix}
\end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

  d[i][j] = d[i][k] + d[k][j]

  path[i][j] = path[k][j]

k=1, i=2

j			
0	d[2][0]	=	d[2][1] + d[1][0]
1	d[2][1]	=	d[2][1] + d[1][1]
2	d[2][2]	<	d[2][1] + d[1][2]
3	d[2][3]	<	d[2][1] + d[1][3]

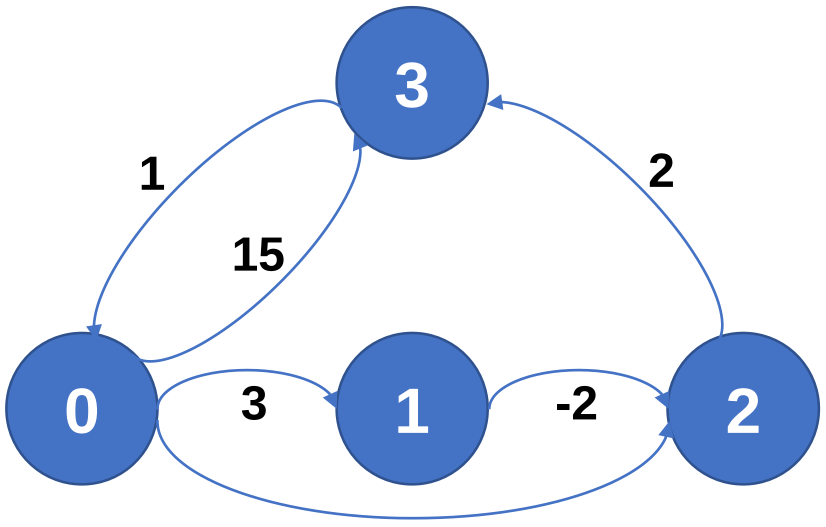
distance

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

  d[i][j] = d[i][k] + d[k][j]

  path[i][j] = path[k][j]

k=1, i=3

j	0	d[3][0]	d[3][1] + d[1][0]
	1	d[3][1]	d[3][1] + d[1][1]
	2	d[3][2]	d[3][1] + d[1][2]
	3	d[3][3]	d[3][1] + d[1][3]

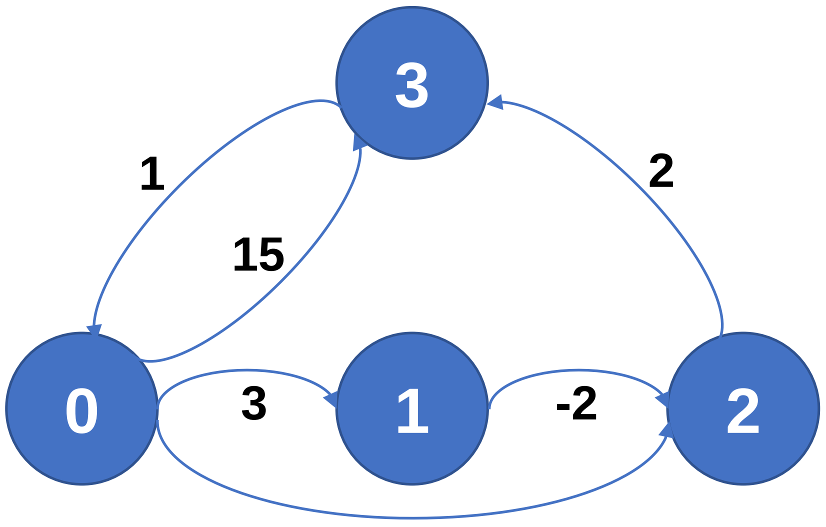
distance

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



if d[i][j] > d[i][k] + d[k][j]:

d[i][j] = d[i][k] + d[k][j]

path[i][j] = path[k][j]

k=1, i=3

j					
	0	d[3][0]	<	d[3][1] + d[1][0]	
	1	d[3][1]	=	d[3][1] + d[1][1]	
	2	d[3][2]	>	d[3][1] + d[1][2]	2
	3	d[3][3]	<	d[3][1] + d[1][3]	

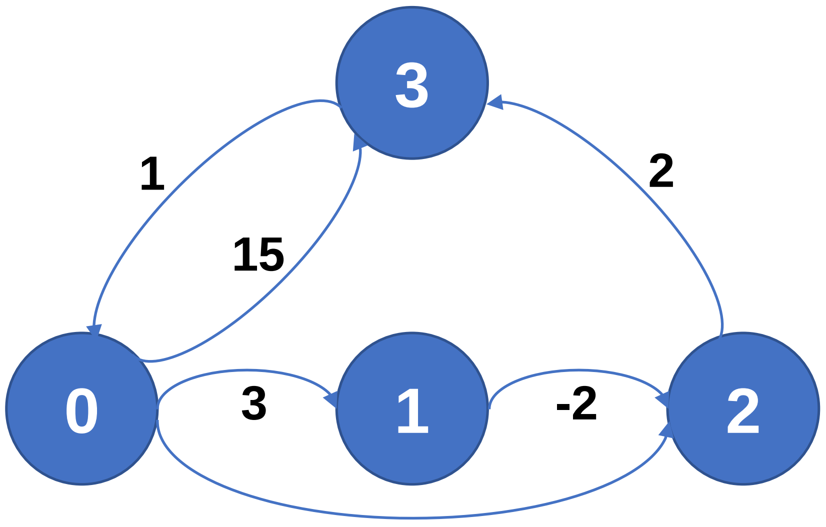
distance

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 7 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 0 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



k=1, i=3

j	0	d[3][0]	<	d[3][1] + d[1][0]	
	1	d[3][1]	=	d[3][1] + d[1][1]	
	2	d[3][2]	>	d[3][1] + d[1][2]	2
	3	d[3][3]	<	d[3][1] + d[1][3]	

distance

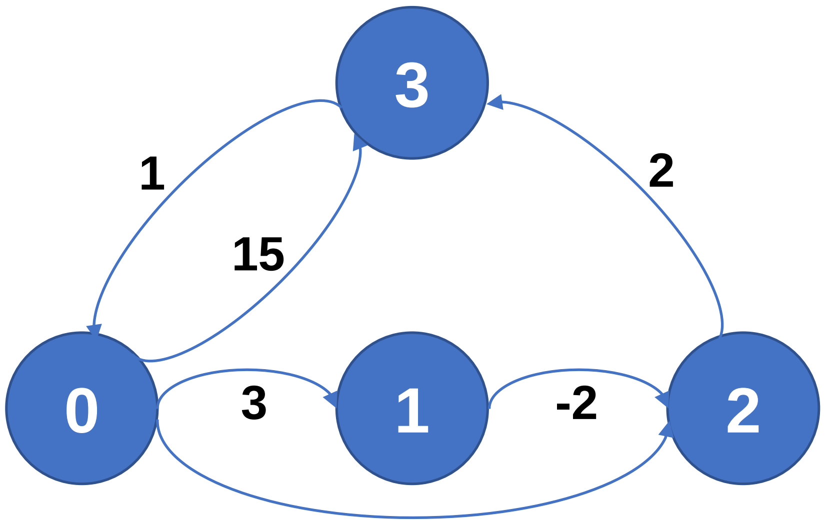
$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 15 \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 2 & 0 \end{bmatrix}
\end{array}
$$

path

$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 0 \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 1 & n \end{bmatrix}
\end{array}
$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



k=2, i=0

j				
0	d[0][0]	<	d[0][2] + d[2][0]	
1	d[0][1]	<	d[0][2] + d[2][1]	
2	d[0][2]	=	d[0][2] + d[2][2]	
3	d[0][3]	>	d[0][2] + d[2][3]	

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

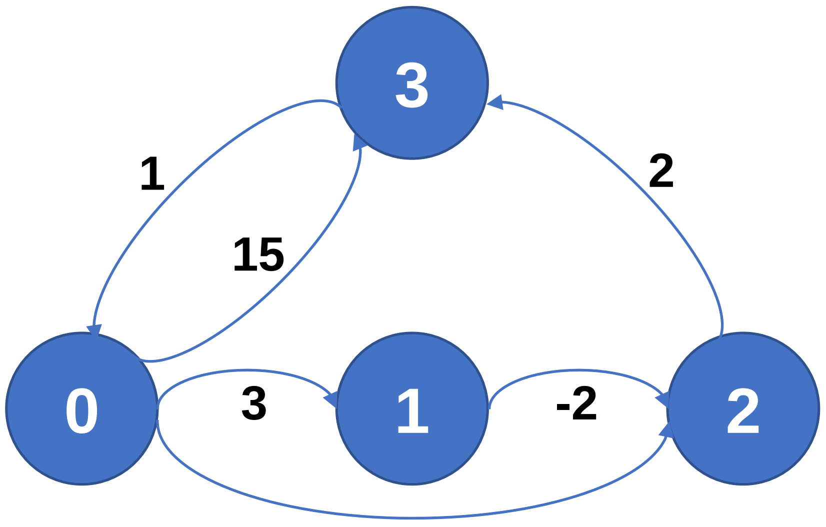
distance

$$\begin{array}{c|c} & \begin{matrix} 0 & 1 & 2 & 3 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \end{matrix} & \begin{bmatrix} 0 & 3 & 1 & \color{red}{3} \\ \infty & 0 & -2 & \infty \\ \infty & \infty & 0 & 2 \\ 1 & 4 & 2 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c|c} & \begin{matrix} 0 & 1 & 2 & 3 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \end{matrix} & \begin{bmatrix} n & 0 & 1 & \color{red}{2} \\ n & n & 1 & n \\ n & n & n & 2 \\ 3 & 0 & 1 & n \end{bmatrix} \end{array}$$

Floyd Warshall Algorithm



k=2, i=1

	j			
j	0	d[1][0]	=	d[1][2] + d[2][0]
	1	d[1][1]	<	d[1][2] + d[2][1]
	2	d[1][2]	=	d[1][2] + d[2][2]
	3	d[1][3]	>	d[1][2] + d[2][3]

distance

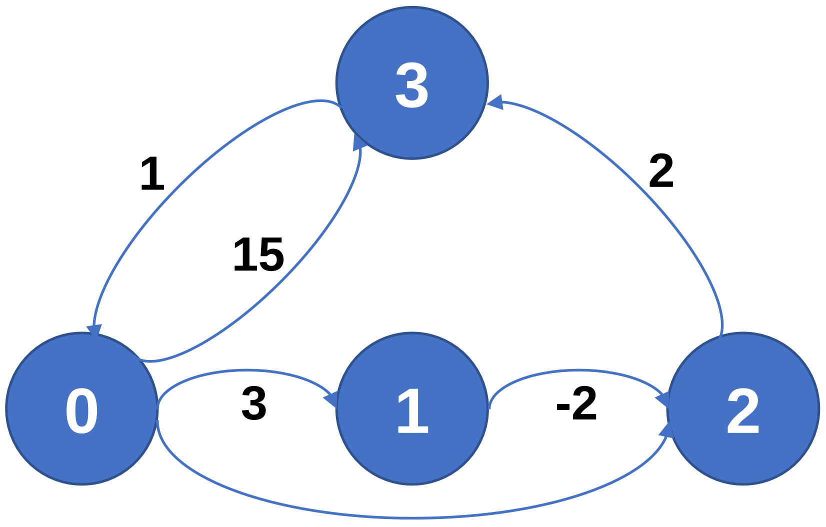
$$
\begin{array}{c|cccc|}
 & 0 & 1 & 2 & 3 \\
\hline
0 & 0 & 3 & 1 & 3 \\
1 & \infty & 0 & -2 & 0 \\
2 & \infty & \infty & 0 & 2 \\
3 & 1 & 4 & 2 & 0
\end{array}
$$

path

$$
\begin{array}{c|cccc|}
 & 0 & 1 & 2 & 3 \\
\hline
0 & n & 0 & 1 & 2 \\
1 & n & n & 1 & 2 \\
2 & n & n & n & 2 \\
3 & 3 & 0 & 1 & n
\end{array}
$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



k=2, i=2

j				
0	d[2][0]	=	d[2][2] + d[2][0]
1	d[2][1]	=	d[2][2] + d[2][1]
2	d[2][2]	=	d[2][2] + d[2][2]
3	d[2][3]	=	d[2][2] + d[2][3]

if d[i][j] > d[i][k] + d[k][j]:

d[i][j] = d[i][k] + d[k][j]

path[i][j] = path[k][j]

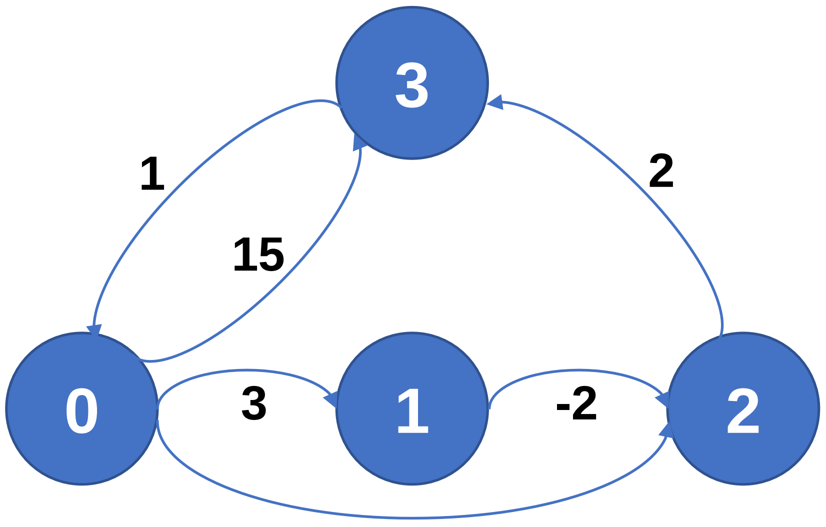
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 1 & 3 \\ 1 & \infty & 0 & -2 & 0 \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & 4 & 2 & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 1 & 2 \\ 1 & n & n & 1 & 2 \\ 2 & n & n & n & 2 \\ 3 & 3 & 0 & 1 & n \end{array}$$

Floyd Warshall Algorithm



k=2, i=3

j				
0	d[3][0]	<	d[3][2] + d[2][0]	
1	d[3][1]	<	d[3][2] + d[2][1]	
2	d[3][2]	=	d[3][2] + d[2][2]	
3	d[3][3]	<	d[3][2] + d[2][3]	

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

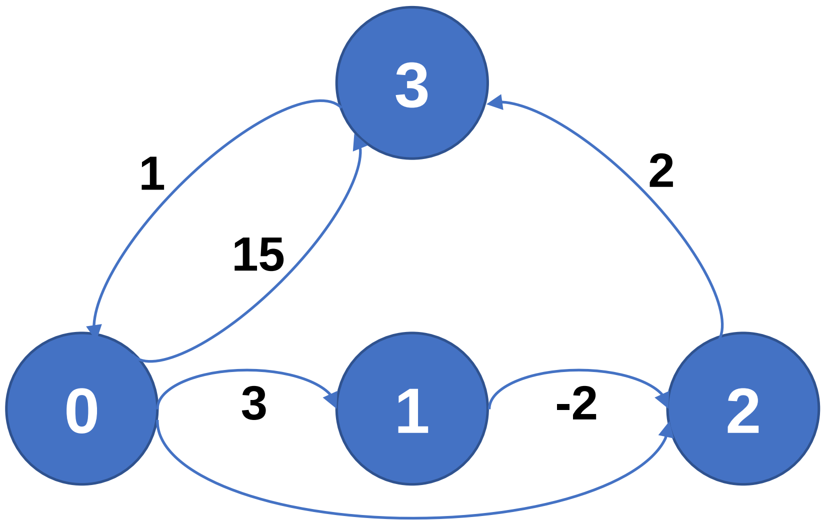
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 1 & 3 \\ 1 & \infty & 0 & -2 & 0 \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & 4 & 2 & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 1 & 2 \\ 1 & n & n & 1 & 2 \\ 2 & n & n & n & 2 \\ 3 & 3 & 0 & 1 & n \end{array}$$

Floyd Warshall Algorithm



k=3, i=0

j				
0	d[0][0]	<	d[0][3] + d[3][0]
1	d[0][1]	<	d[0][3] + d[3][1]
2	d[0][2]	<	d[0][3] + d[3][2]
3	d[0][3]	=	d[0][3] + d[3][3]

distance

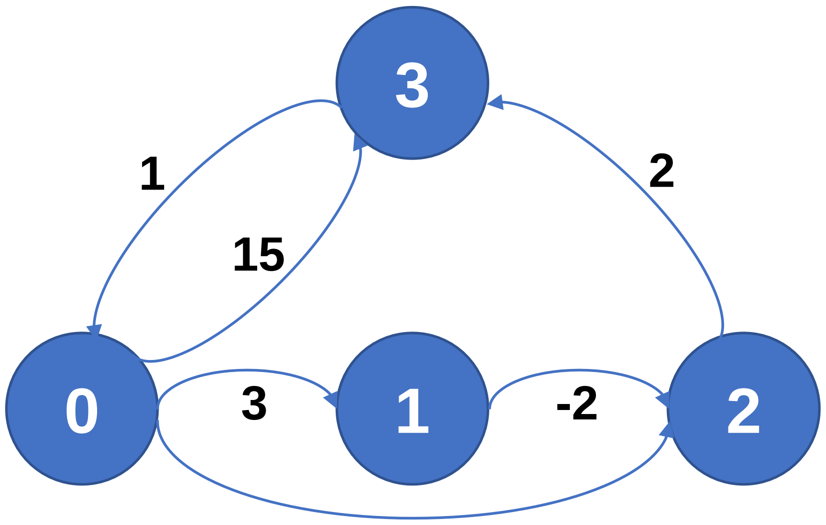
	0	1	2	3
0	0	3	1	3
1	∞	0	-2	0
2	∞	∞	0	2
3	1	4	2	0

path

	0	1	2	3
0	n	0	1	2
1	n	n	1	2
2	n	n	n	2
3	3	0	1	n

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



k=3, i=1

	j			
j	0	d[1][0]	>	d[1][3] + d[3][0]
	1	d[1][1]	<	d[1][3] + d[3][1]
	2	d[1][2]	<	d[1][3] + d[3][2]
	3	d[1][3]	=	d[1][3] + d[3][3]

distance

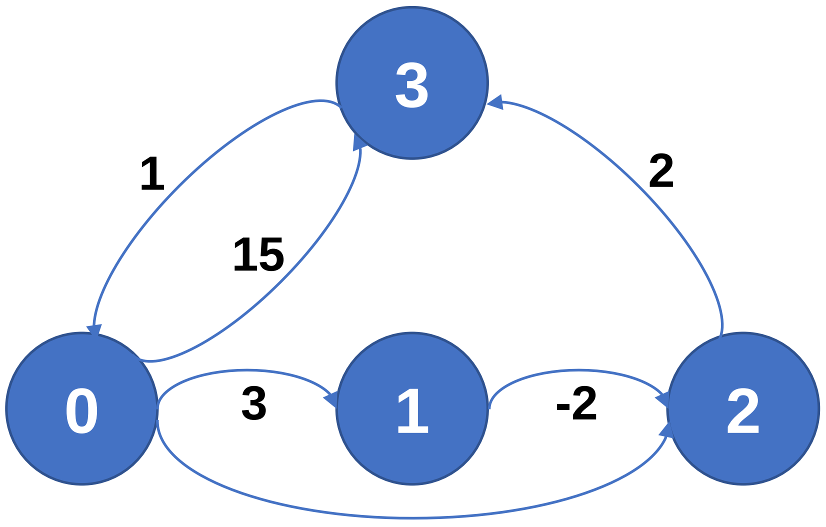
$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 1 & 3 \\ 1 & 1 & 0 & -2 & 0 \\ 2 & \infty & \infty & 0 & 2 \\ 3 & 1 & 4 & 2 & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 1 & 2 \\ 1 & 3 & n & 1 & 2 \\ 2 & n & n & n & 2 \\ 3 & 3 & 0 & 1 & n \end{array}$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



k=3, i=2

j				
0	d[2][0]	>	d[2][3] + d[3][0]
1	d[2][1]	>	d[2][3] + d[3][1]
2	d[2][2]	<	d[2][3] + d[3][2]
3	d[2][3]	=	d[2][3] + d[3][3]

if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]
    path[i][j] = path[k][j]

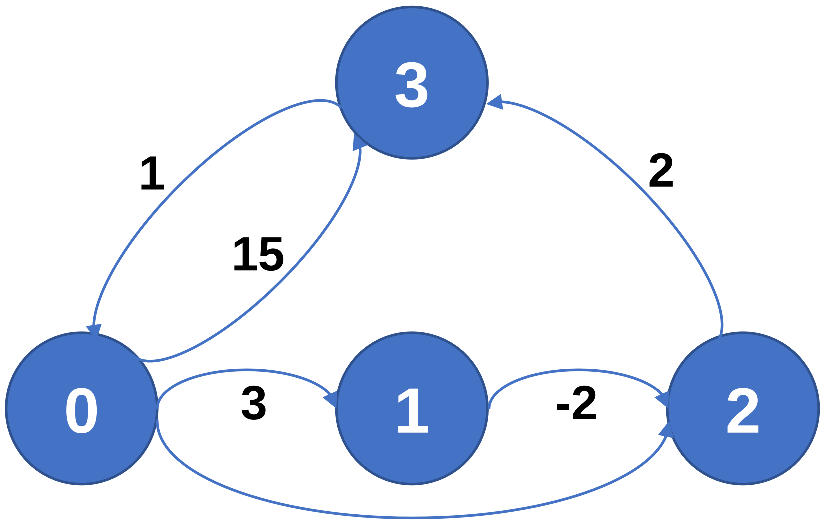
distance

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 3 & 1 & 3 \\ 1 & 1 & 0 & -2 & 0 \\ 2 & 3 & 6 & 0 & 2 \\ 3 & 1 & 4 & 2 & 0 \end{array}$$

path

$$\begin{array}{c|cccc|} & 0 & 1 & 2 & 3 \\ \hline 0 & n & 0 & 1 & 2 \\ 1 & 3 & n & 1 & 2 \\ 2 & 3 & 0 & n & 2 \\ 3 & 3 & 0 & 1 & n \end{array}$$

Floyd Warshall Algorithm



k=3, i=3

j				
0	d[3][0]	=	d[3][3] + d[3][0]
1	d[3][1]	=	d[3][3] + d[3][1]
2	d[3][2]	=	d[3][3] + d[3][2]
3	d[3][3]	=	d[3][3] + d[3][3]

distance

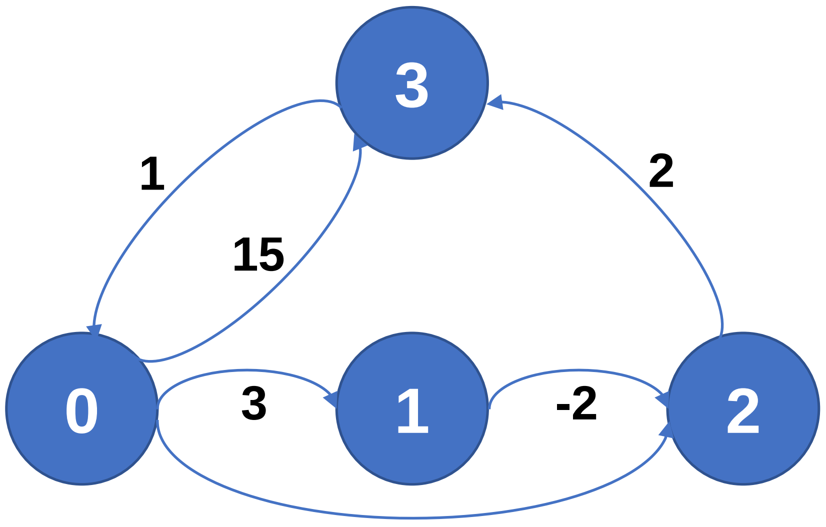
$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 3 \\ 1 & 0 & -2 & 0 \\ 3 & 6 & 0 & 2 \\ 1 & 4 & 2 & 0 \end{bmatrix} \end{array}$$

path

$$\begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 2 \\ 3 & n & 1 & 2 \\ 3 & 0 & n & 2 \\ 3 & 0 & 1 & n \end{bmatrix} \end{array}$$

```
if d[i][j] > d[i][k] + d[k][j]:
        d[i][j] = d[i][k] + d[k][j]
        path[i][j] = path[k][j]
```

Floyd Warshall Algorithm



0 → 3?

0 → 1 → 2 → 3

3 → 2?

3 → 0 → 1 → 2

if d[i][j] > d[i][k] + d[k][j]:

    d[i][j] = d[i][k] + d[k][j]

    path[i][j] = path[k][j]

distance

$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 3 & 1 & 3 \\ 1 & 0 & -2 & 0 \\ 3 & 6 & 0 & 2 \\ 1 & 4 & 2 & 0 \end{bmatrix}
\end{array}
$$

path

$$
\begin{array}{c c}
 & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} n & 0 & 1 & 2 \\ 3 & n & 1 & 2 \\ 3 & 0 & n & 2 \\ 3 & 0 & 1 & n \end{bmatrix}
\end{array}
$$

Complexity

Time Complexity: $O(V^3)$

Space Complexity: $O(V^2)$

Implemetation

```
floydWarshall (graph[][], V){

initialize path[V][V], d[V][V]
for i=0:V-1
        for j=0:V-1
                d[i][j] = graph[i][j]
                if (graph[i][j]!=INF and i!=j)  path[i][j] = i
                else                            path[i][j] = -1
for k=0:V-1
        for i=0:V-1
                for j=0:V-1
                        if (d[i][k]!=INF and d[k][j]!=INF and d[i][j] > d[i][k] + d[k][j])
                                d[i][j] = d[i][k] + d[k][j]
                                path[i][j] = path[k][j]
}
```

Johnson’s algorithm

Johnson’s algorithm

Find shortest paths between every pair of vertices in a given weighted directed Graph and weights may be negative.

Time complexity of Floyd Warshall Algorithm is $O(V^3)$. Using Johnson’s algorithm, we can find all pair shortest paths in $O(V^2 \log V + VE)$ time.

Johnson’s algorithm uses both Dijkstra and Bellman-Ford as subroutines.

Johnson’s algorithm

If we apply Dijkstra’s Single Source shortest path algorithm for every vertex, considering every vertex as source, we can find all pair shortest paths in $O(V^2 \log V)$ time.

→ Using Dijkstra’s single source shortest path seems to be a better option than Floyd Warshall, but the problem with Dijkstra’s algorithm is that it doesn’t work for negative weight edge.

→ The idea of Johnson’s algorithm is to re-weight all edges and make them all positive, then apply Dijkstra’s algorithm for every vertex.

Johnson’s algorithm

1) Form G’ by adding a new vertex s and a new edge (s, v) with length 0 for each $v \in G$.
2) Run Bellman-Ford algorithm on G’ with source s. If B-F detects a negative weight cycle in G’, then return.
3) For each edge (u, v), assign the new weight as “original weight + h[u] – h[v]”.
4) Remove the added vertex s and run Dijkstra’s algorithm for every vertex.

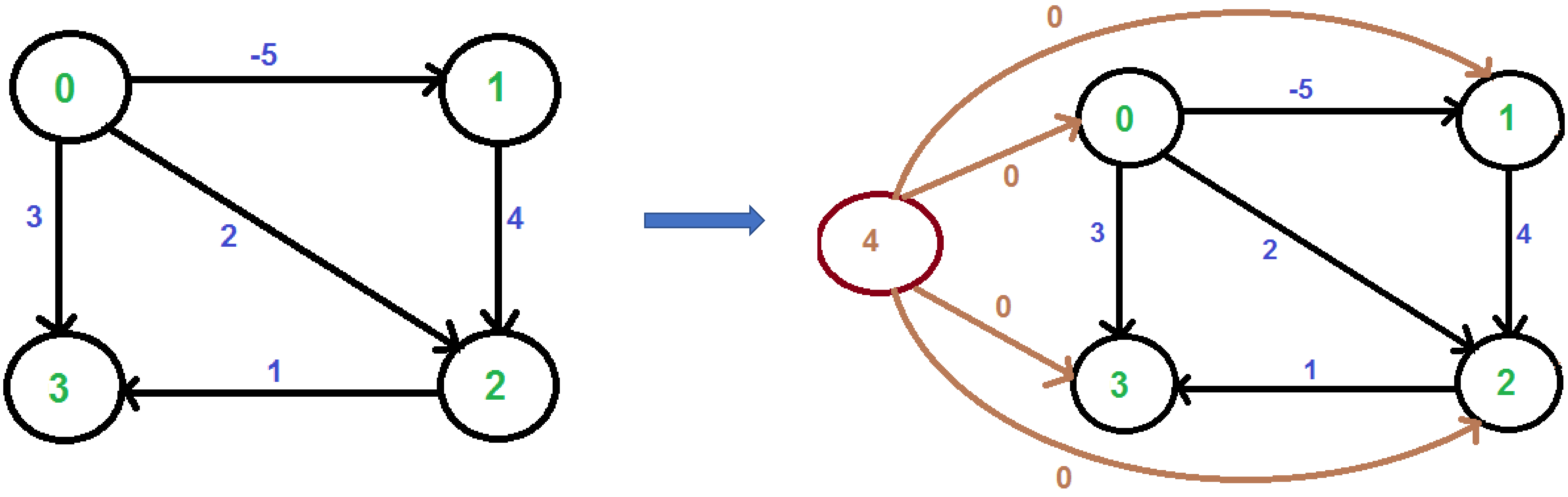
Johnson’s algorithm

How does the transformation ensure nonnegative weight edges?

→ h[v] <= h[u] + w(u, v)

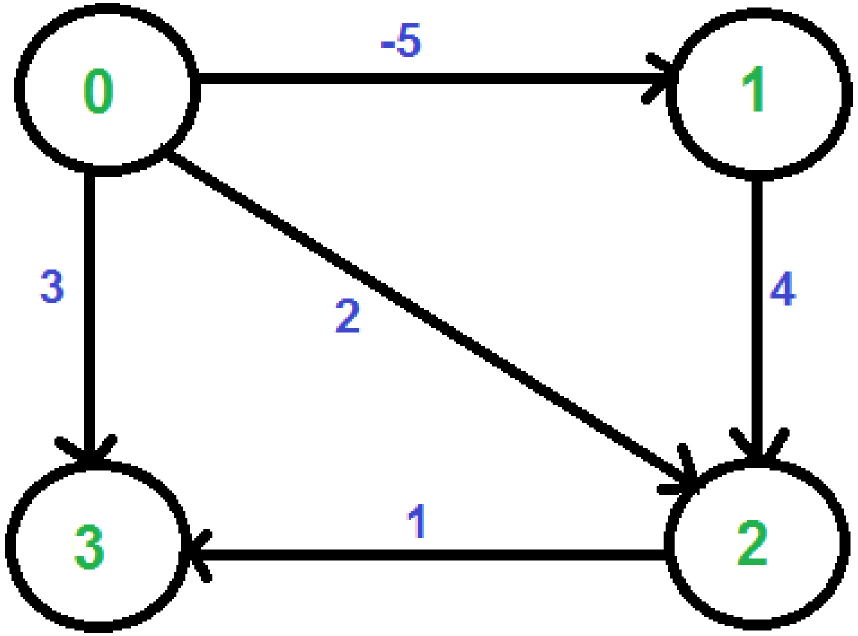
→ The new weights are w(u, v) + h[u] - h[v]. The value of the new weights must be greater than or equal to zero because of the inequality "h[v] <= h[u] + w(u, v)".

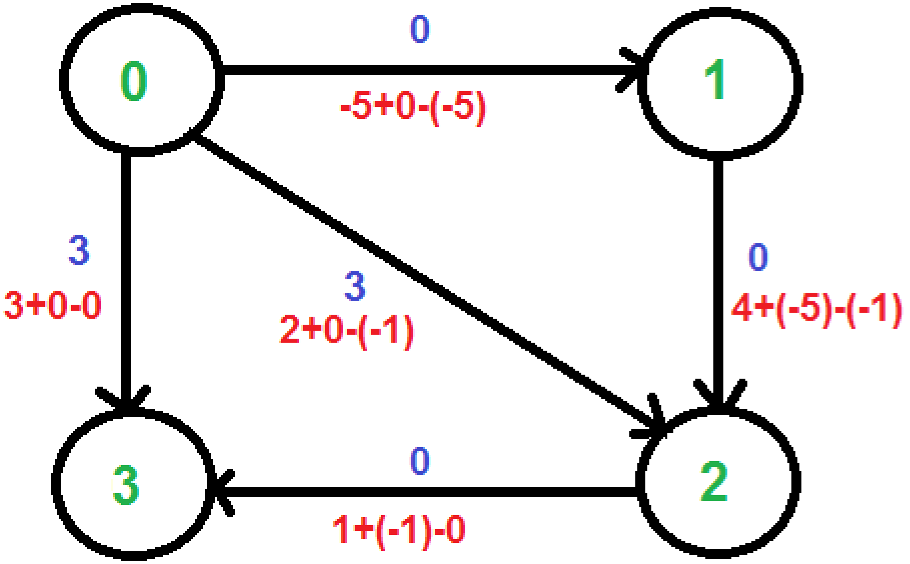
Johnson's algorithm



h[] = {0, -5, -1, 0}

Johnson's algorithm

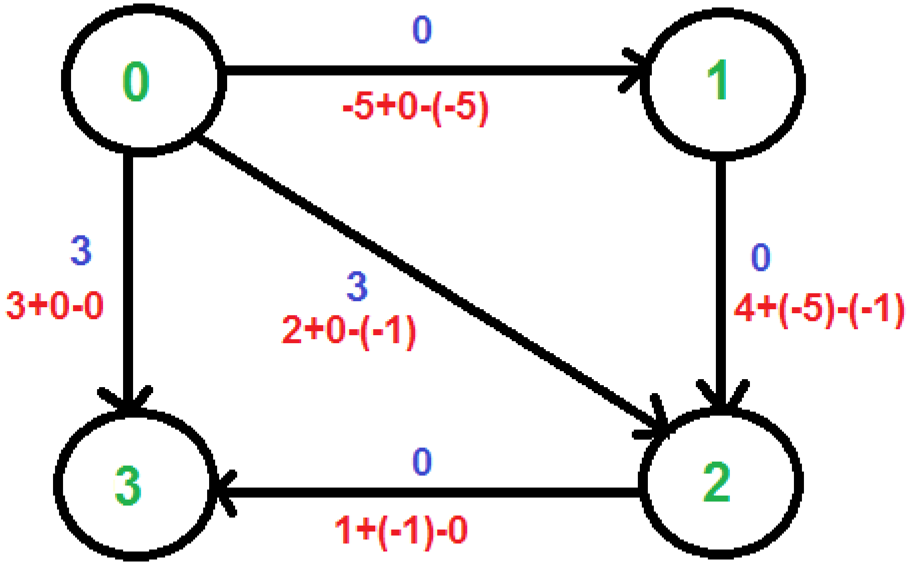




Distances from 4 to 0, 1, 2 and 3 are 0, -5, -1 and 0 respectievely.

h[] = {0, -5, -1, 0}

Johnson's algorithm



Distances from 4 to 0, 1, 2 and 3 are 0, -5, -1 and 0 respectievely.

Run Dijkstra's shortest path algorithm for every vertex as source

Complexity

Time complexity of Bellman Ford: O(VE)

Time complexity of Dijkstra: O(V LogV)

→ Time Complexity: $O(V^2 \text{ Log}V + VE)$

The time complexity of Johnson's algorithm becomes same as Floyd Warshell when the graphs is complete (For a complete graph $E = O(V^2)$.

→ for sparse graphs, the algorithm performs much better than Floyd Warshell.

Reference

- Charles Leiserson and Piotr Indyk, “*Introduction to Algorithms*”, September 29, 2004

- https://www.geeksforgeeks.org

- https://en.wikipedia.org/wiki