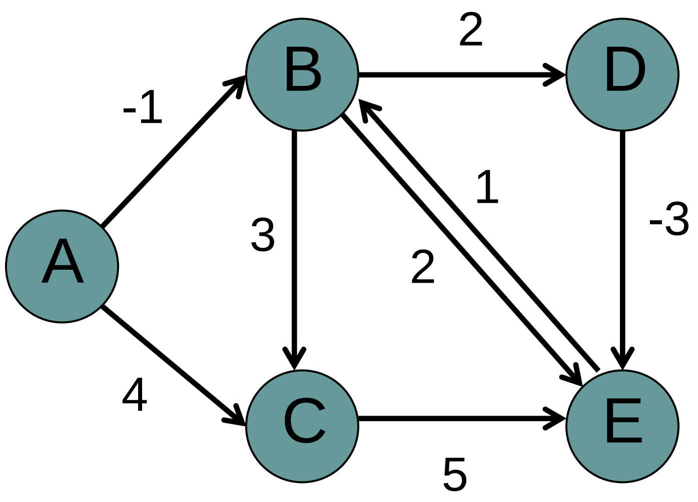
All Pairs Shortest Path

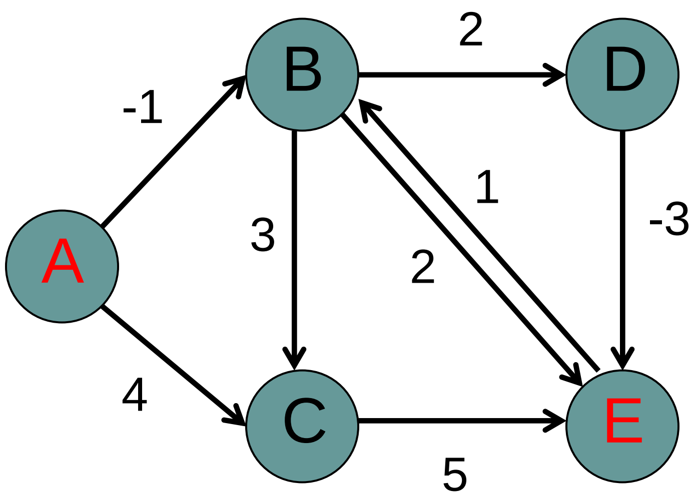
Dr. Navjot Singh

Design and Analysis of Algorithms

Shortest Paths

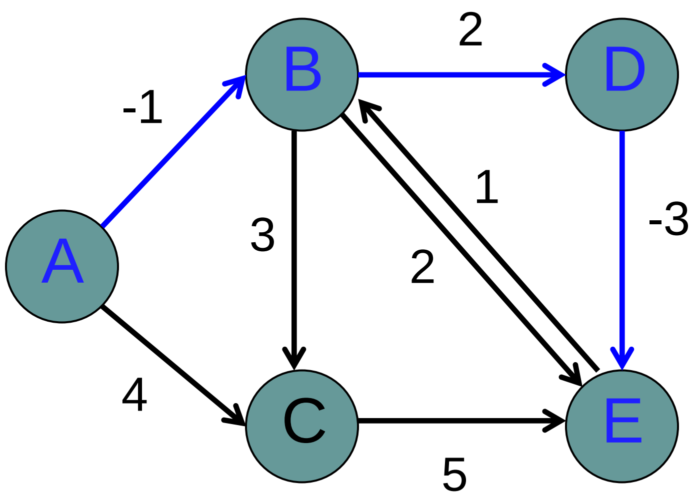


Shortest Paths



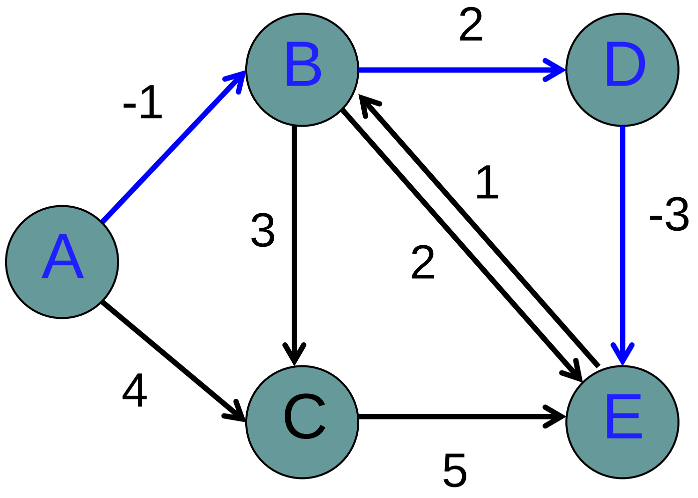
What is the shortest path from A to E?

Shortest Paths



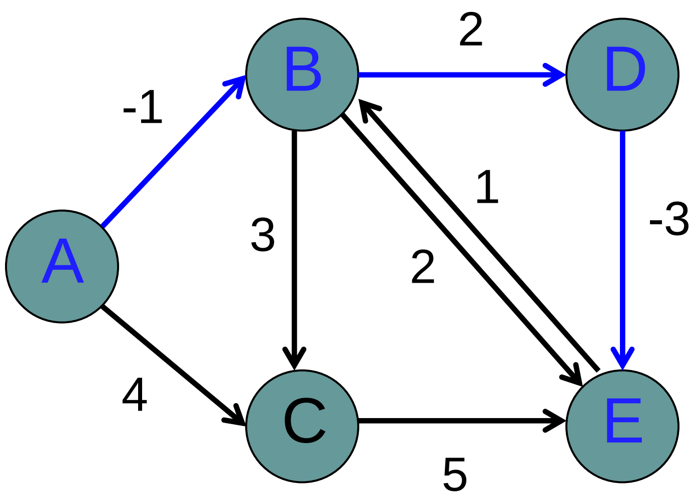
-2

Shortest Paths



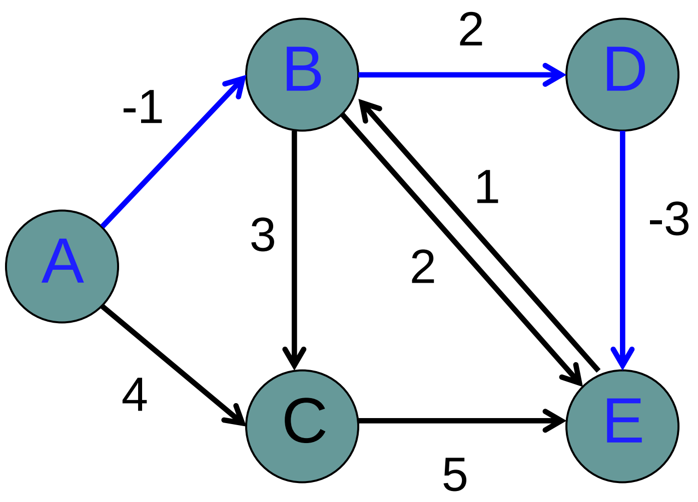
What algorithm would we use to calculate this?

Shortest Paths



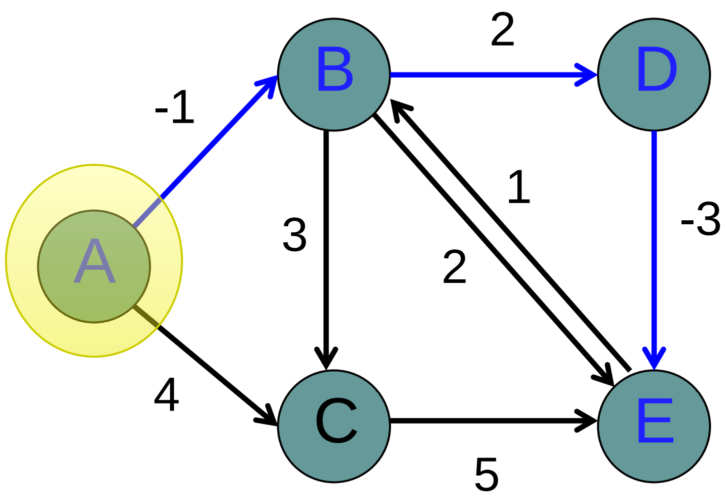
- Bellman-Ford (since the graph has negative edges)
- O(VE)

Shortest Paths



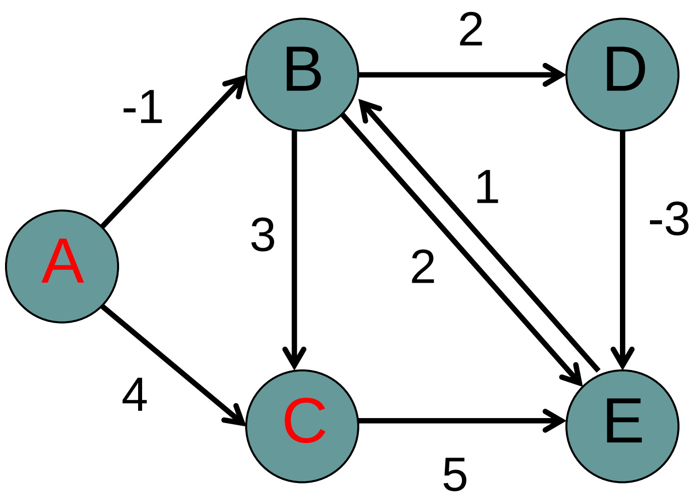
- Bellman-Ford (since the graph has negative edges)
- O(VE)
- Called a single-source shortest path algorithm. Why?

Shortest Paths



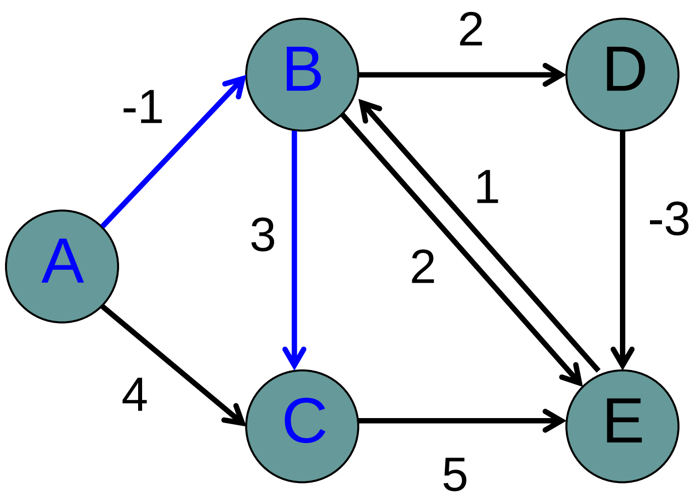
- Bellman-Ford (since the graph has negative edges)
- O(VE)
- Calculate all paths from a **single vertex.**

Shortest Paths



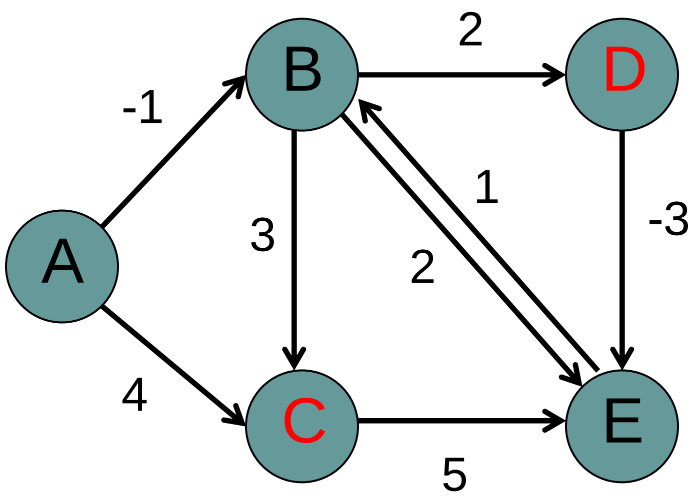
What is the shortest path from A to C?
If we already calculated A to E using Bellman-Ford do we need to do any work?

Shortest Paths



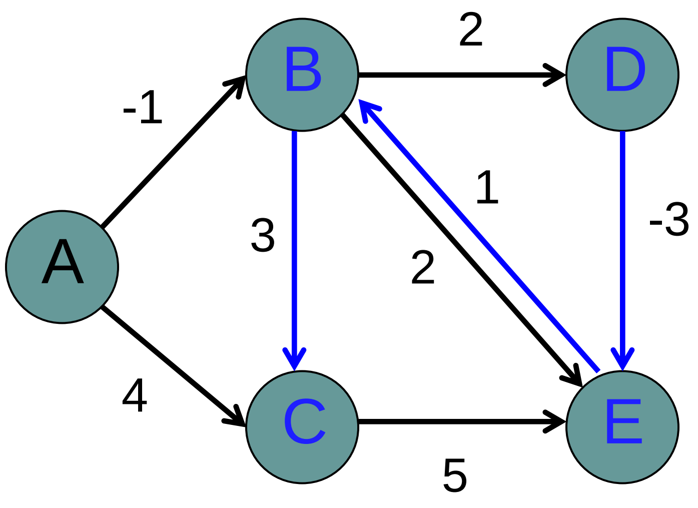
No new calculations!
Bellman-Ford calculates all shortest paths starting at A.

Shortest Paths



What is the shortest path from D to C?
If we already calculated A to E using
Bellman-Ford do we need to do any work?

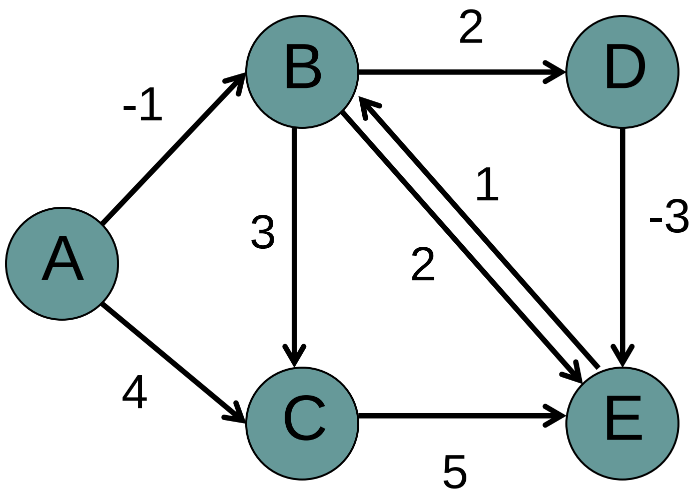
Shortest Paths



Different source.
Have to run Bellman-Ford again!

All pairs shortest paths

All pairs shortest paths: calculate the shortest paths between *all* vertices



All pairs shortest paths

All pairs shortest paths: calculate the shortest paths between *all* vertices

Easy solution?

All pairs shortest paths

All pairs shortest paths: calculate the shortest paths between *all* vertices

Run Bellman-Ford from each vertex!

Running time (in terms of E and V)?

All pairs shortest paths

All pairs shortest paths: calculate the shortest paths between *all* vertices

Run Bellman-Ford from each vertex!

$O(V^2E)$

- Bellman-Ford: $O(VE)$
- V calls, one for each vertex

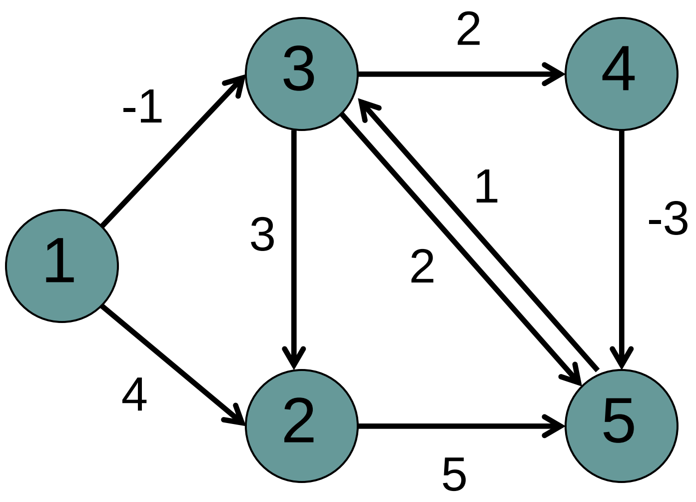
Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Floyd-Warshall: key idea

d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



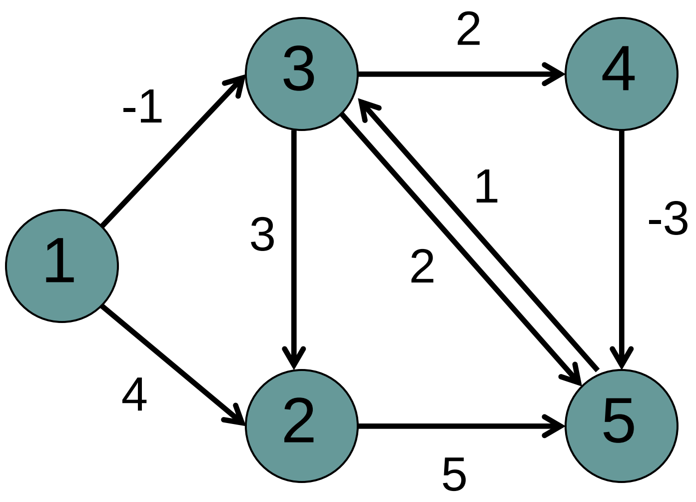
What is d_{15}^{2}?

What is d_{15}^{3}?

What is d_{41}^{4}?

Floyd-Warshall: key idea

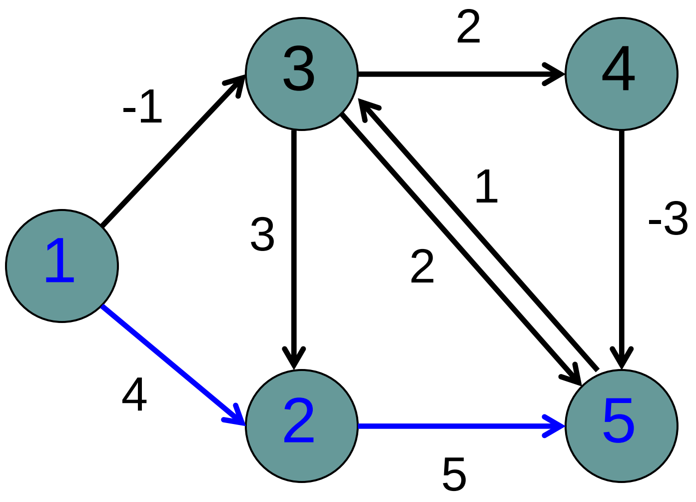
$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j using only vertices $\{1, 2, \ldots, k\}$



What is $d_{15}{}^{2}$?

Floyd-Warshall: key idea

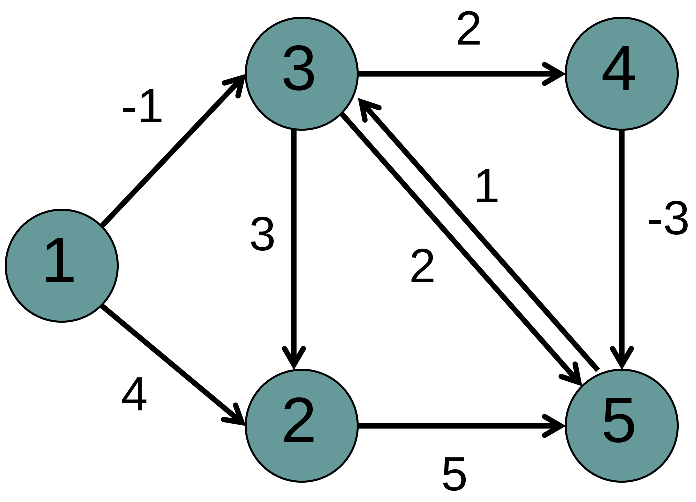
$d_{ij}^{\,k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



$d_{15}^{\,2} = 9$. Can only use 2.

Floyd-Warshall: key idea

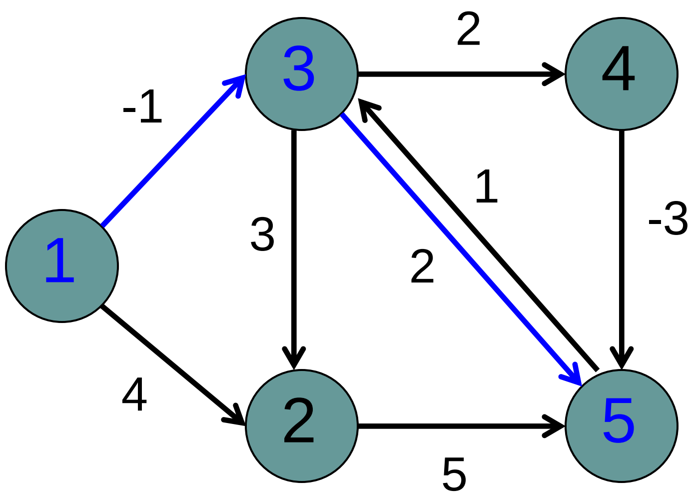
d_{ij}^{k} = shortest path from vertex i to vertex j using only vertices $\{1, 2, \ldots, k\}$



What is d_{15}^{3}?

Floyd-Warshall: key idea

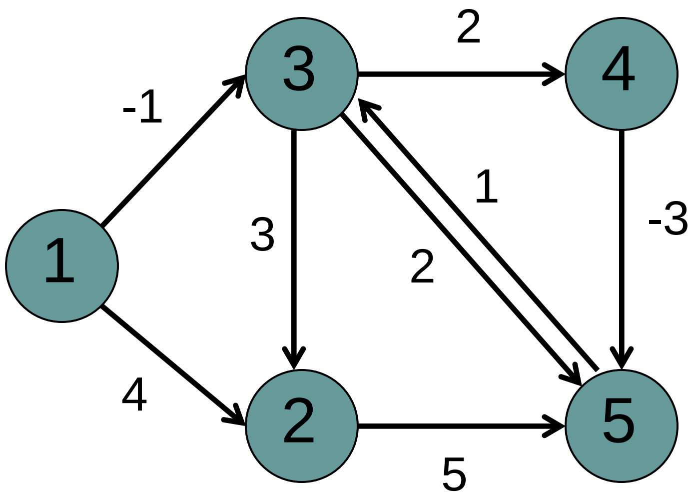
d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



d_{15}^{3} = 1. Can't use vertex 4.

Floyd-Warshall: key idea

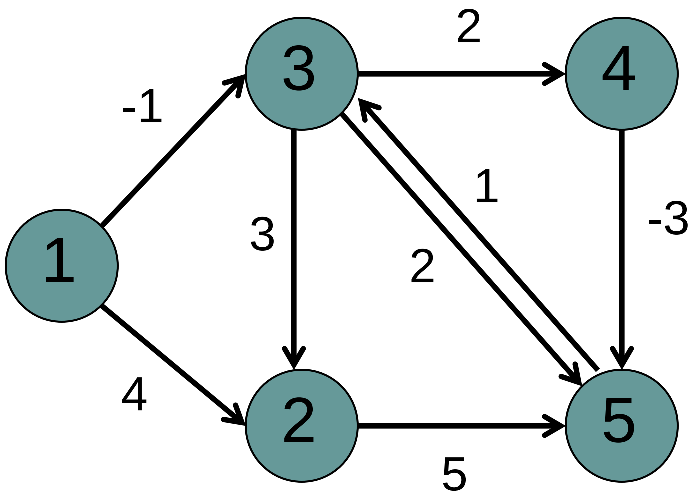
d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



What is d_{41}^{4}?

Floyd-Warshall: key idea

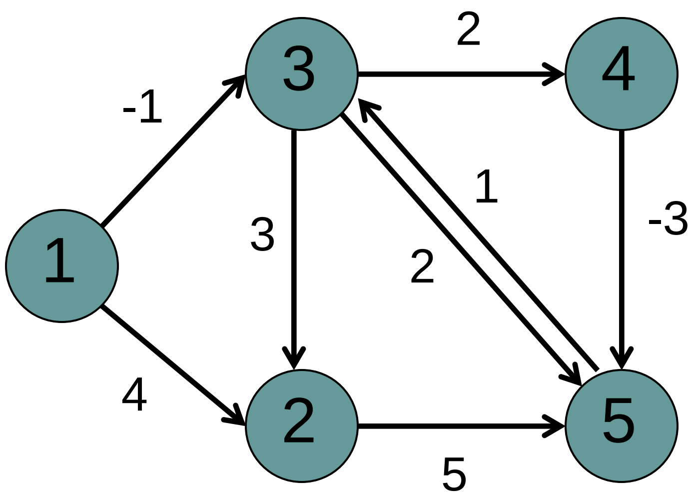
d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



$d_{41}^{4} = \infty$. No possible path.

Floyd-Warshall: key idea

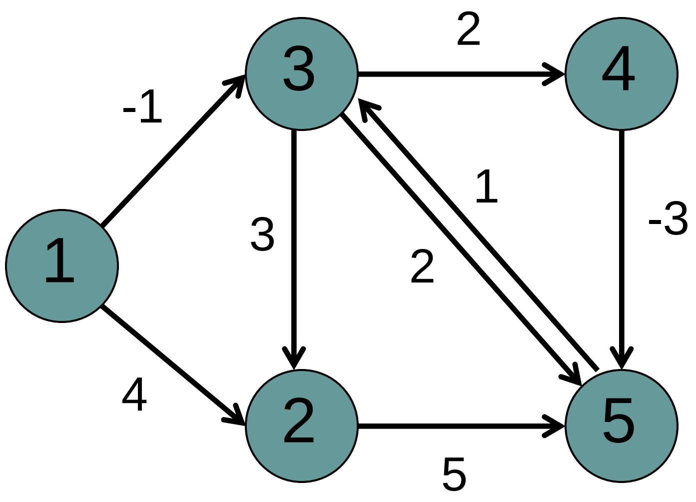
d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$



What is d_{33}^{5}?

Floyd-Warshall: key idea

$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \dots, k\}$



$d_{33}{}^{5}$ = 0. $d_{ii}{}^{k}$ = 0 for all i.

Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

If we want all possibilities, how many values are there (i.e. what is the size of $d_{ij}{}^{k}$)? (Poll)

Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

V^3

- i: all vertices
- j: all vertices
- k: all vertices

Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \dots, k\}$

What is d_{ij}^{V}?

- Distance of the shortest path from i to j
- If we can calculate this, for all (i, j), we're done!

Recursive relationship

$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Assume we know $d_{ij}{}^{k}$

How can we calculate $d_{ij}{}^{k+1}$, i.e. shortest path now including vertex k+1? (Hint: in terms of $d_{ij}{}^{k}$)

Two options:
1) Vertex k+1 doesn't give us a shorter path
2) Vertex k+1 does give us a shorter path

Recursive relationship

d_{ij}^{k} = shortest path from vertex i to vertex j using only vertices $\{1, 2, \dots, k\}$

Two options:

1) Vertex k+1 doesn't give us a shorter path

2) Vertex k+1 does give us a shorter path

d_{ij}^{k+1} = ?

Recursive relationship

$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Two options:

1) Vertex k+1 doesn't give us a shorter path
2) Vertex k+1 does give us a shorter path

$$d_{ij}{}^{k+1} = d_{ij}{}^{k}$$

Recursive relationship

d_{ij}^{k} = shortest path from vertex i to vertex j using only vertices $\{1, 2, \ldots, k\}$

Two options:

1) Vertex k+1 doesn't give us a shorter path

2) Vertex k+1 does give us a shorter path

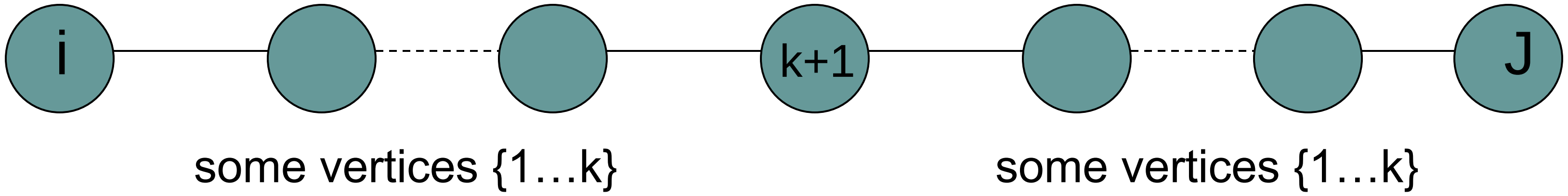
d_{ij}^{k+1} = ?

Recursive relationship

d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Two options:
1) Vertex k+1 doesn't give us a shorter path
2) Vertex k+1 does give us a shorter path

$d_{ij}^{k+1} = ?$



What is the cost of this path?

Recursive relationship

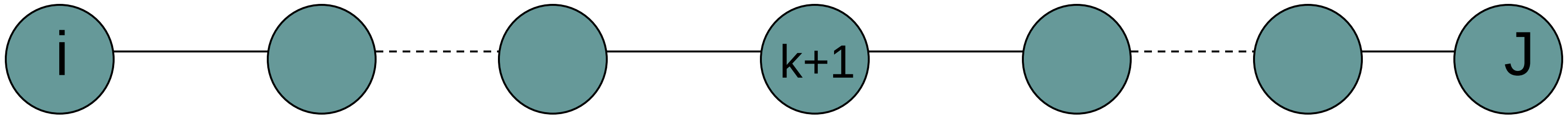
d_{ij}^{k} = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Two options:

1) Vertex k+1 doesn't give us a shorter path

2) Vertex k+1 does give us a shorter path

$$d_{ij}^{k+1} = d_{i(k+1)}^{k} + d_{(k+1)j}^{k}$$



some vertices {1…k} some vertices {1…k}

$d_{i(k+1)}^{k}$ + $d_{(k+1)j}^{k}$

Recursive relationship

$d_{ij}{}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Two options:
1) Vertex k+1 doesn't give us a shorter path
2) Vertex k+1 does give us a shorter path

$d_{ij}{}^{k+1} = ?$

How do we combine these two options?

Recursive relationship

${d_{ij}}^{k}$ = shortest path from vertex i to vertex j
using only vertices $\{1, 2, \ldots, k\}$

Two options:
1) Vertex k+1 doesn't give us a shorter path
2) Vertex k+1 does give us a shorter path

$${d_{ij}}^{k+1} = \min(dijk, {d_{i(k+1)}}^{k} + {d_{(k+1)j}}^{k})$$

Pick whichever is shorter

Floyd-Warshall

- Calculate d_{ij}^{k} for increasing k, i.e. k = 1 to V

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

for k = 1 to V

for i = 1 to V

for j = 1 to V

$$\text{dijk} = \min(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1})$$

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

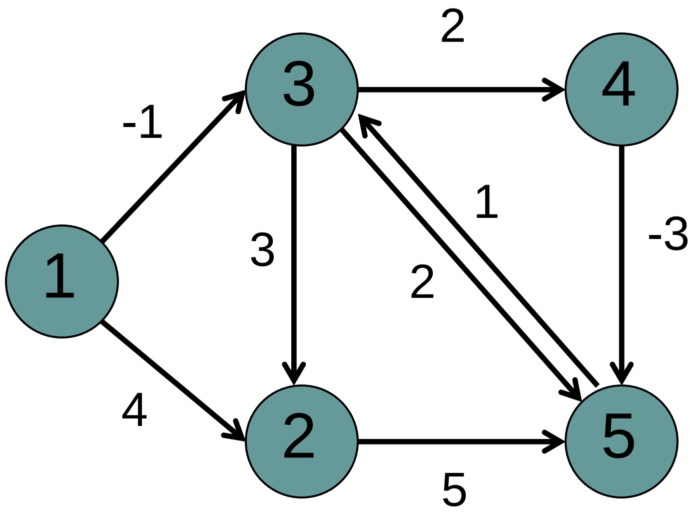
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1})$$

return d^V



k = 0

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 4 & -1 & \infty & \infty \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

adjacency matrix

k = 1

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 4 & -1 & \infty & \infty \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

no change

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

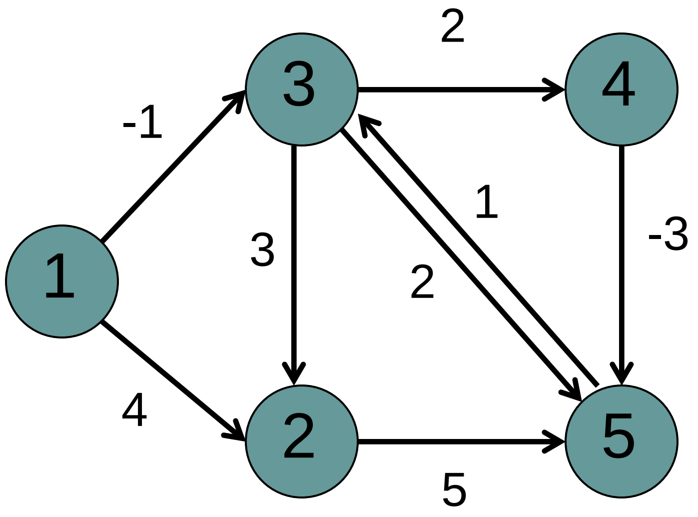
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 1

$$\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & \infty \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix}
\end{array}$$

k = 2

$$\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & ? \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}
\end{array}$$

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

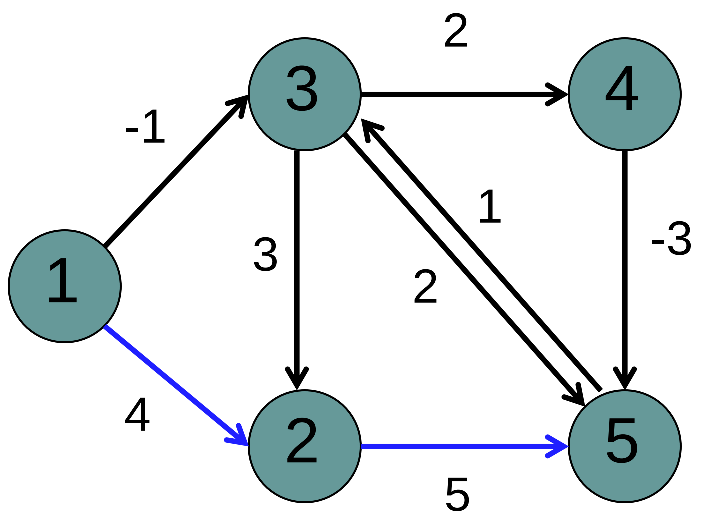
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 1

$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & \infty \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 2

$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

minimum

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

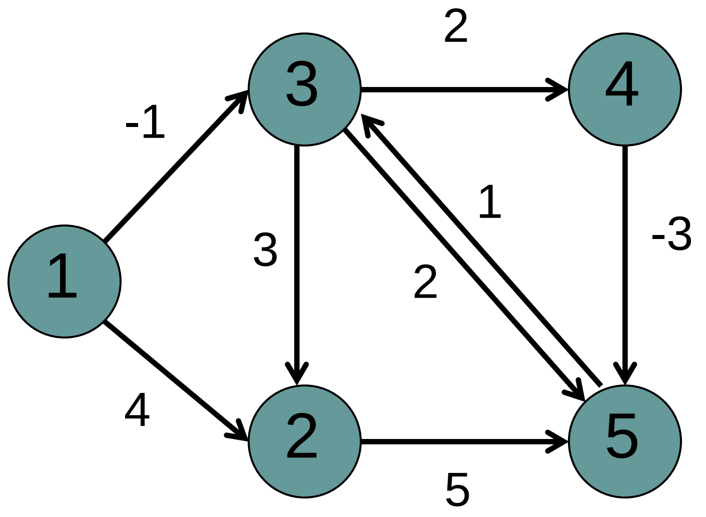
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

	1	2	3	4	5
1	0	4	-1	¥	9
2	¥	0	¥	¥	5
3	¥	3	0	2	2
4	¥	¥	¥	0	-3
5	¥	¥	1	¥	0

k = 3

	1	2	3	4	5
1	0	?			
2					
3					
4					
5					

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

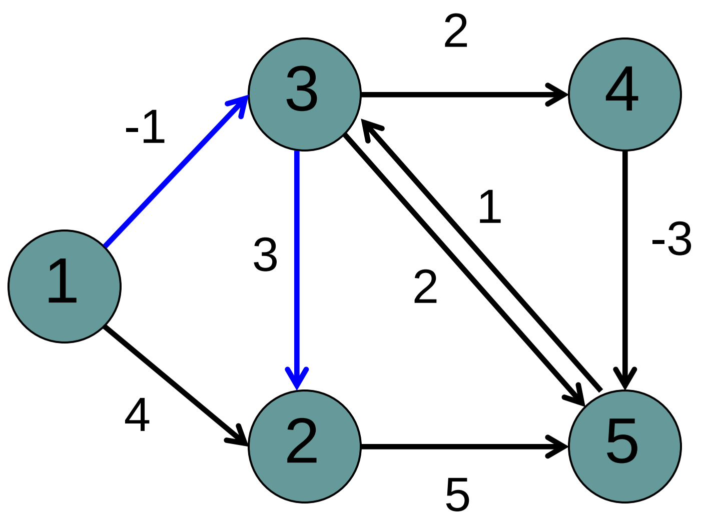
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

minimum

k = 3

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix} 0 & 2 & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Found a shorter path!

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

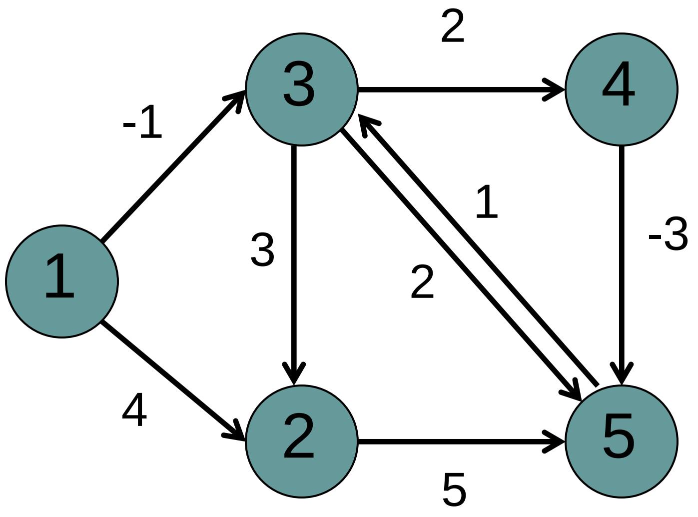
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 3

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

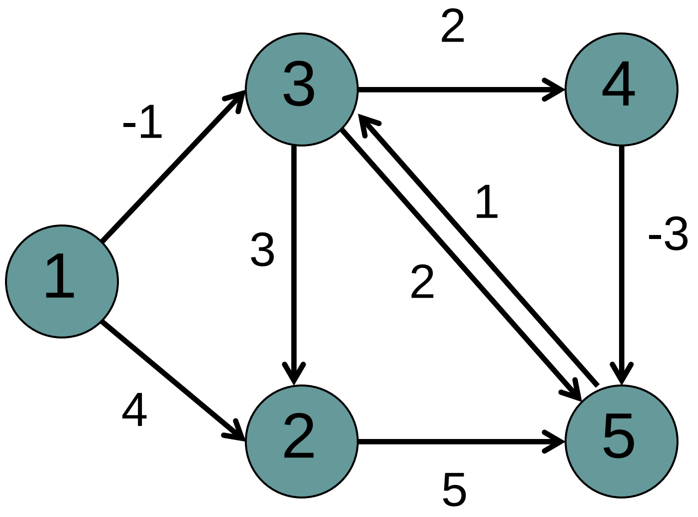
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 3

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix} 0 & 2 & -1 & ? & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

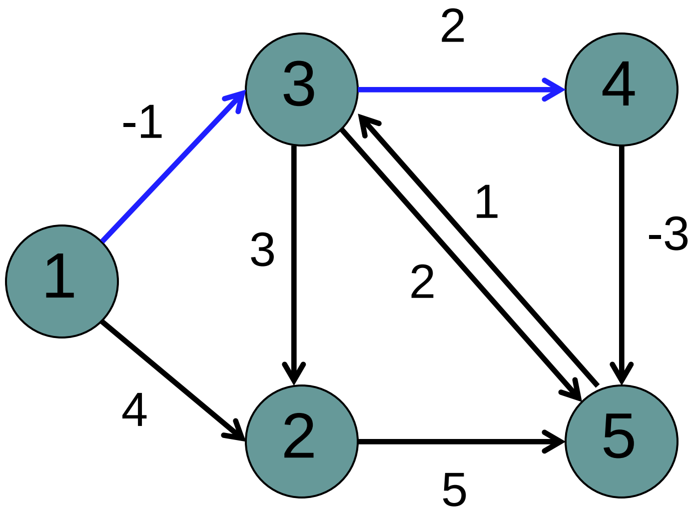
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$$

return d^V



k = 2

	1	2	3	4	5
1	0	4	-1	¥	9
2	¥	0	¥	¥	5
3	¥	3	0	2	2
4	¥	¥	¥	0	-3
5	¥	¥	1	¥	0

minimum

k = 3

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

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

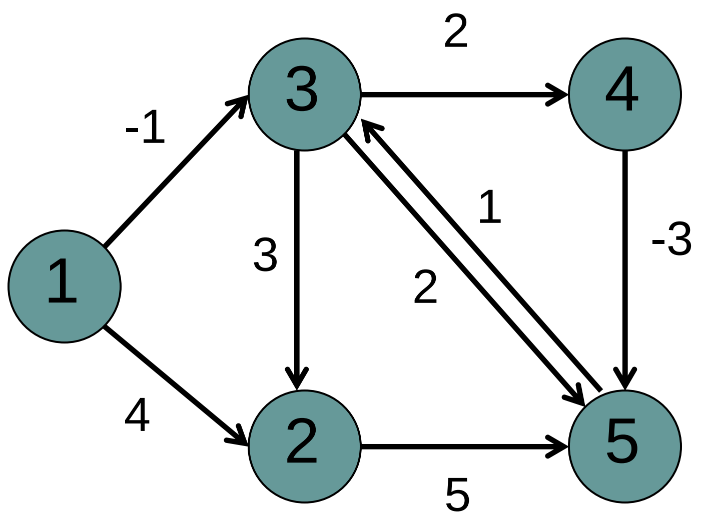
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$$

return d^V



k = 2

$$\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5
\end{array}
\begin{array}{c}
\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 4 & -1 & \infty & 9 \\
\infty & 0 & \infty & \infty & 5 \\
\infty & 3 & 0 & 2 & 2 \\
\infty & \infty & \infty & 0 & -3 \\
\infty & \infty & 1 & \infty & 0
\end{bmatrix}
\end{array}$$

k = 3

$$\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5
\end{array}
\begin{array}{c}
\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 2 & -1 & 1 & \\
 & & & & \\
 & & & & \\
 & & & & \\
 & & & &
\end{bmatrix}
\end{array}$$

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

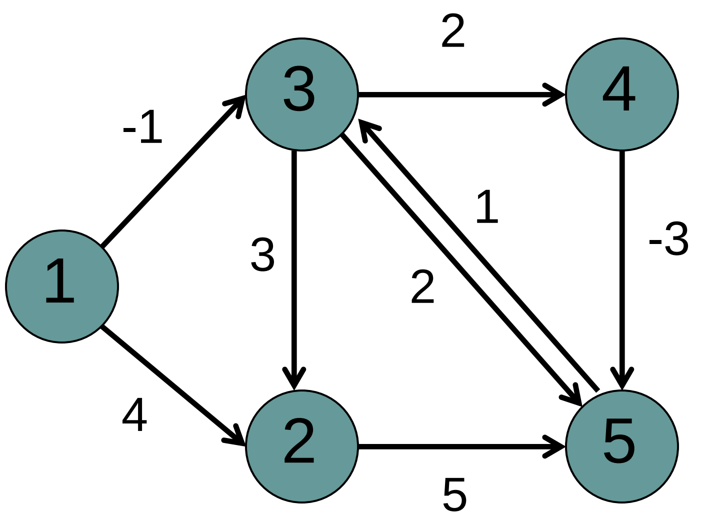
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

      $\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$

return d^V



k = 2

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 3

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & -1 & 1 & ? \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):

$d^0 = W$ // initialize with edge weights

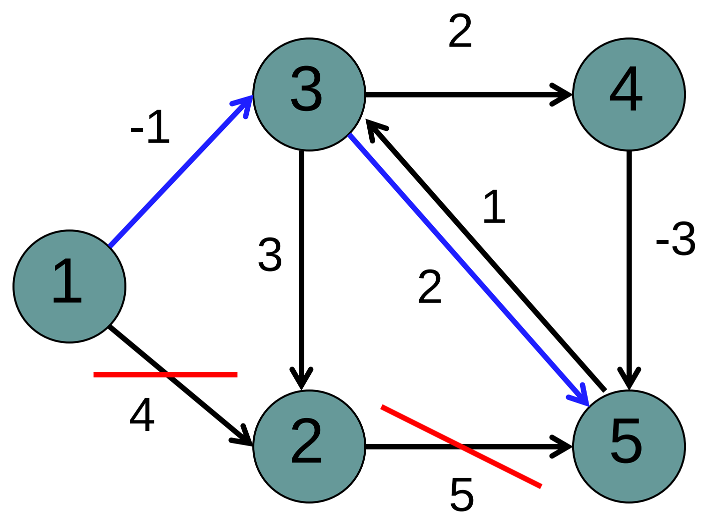
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

minimum

k = 3

$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Found a shorter path!

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

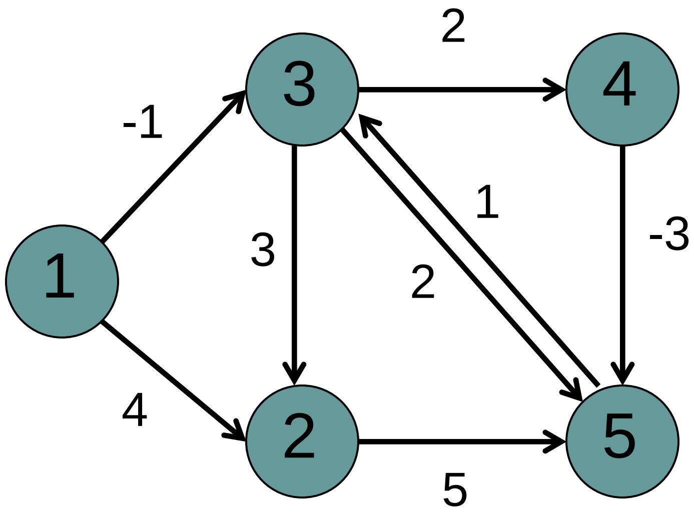
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 2

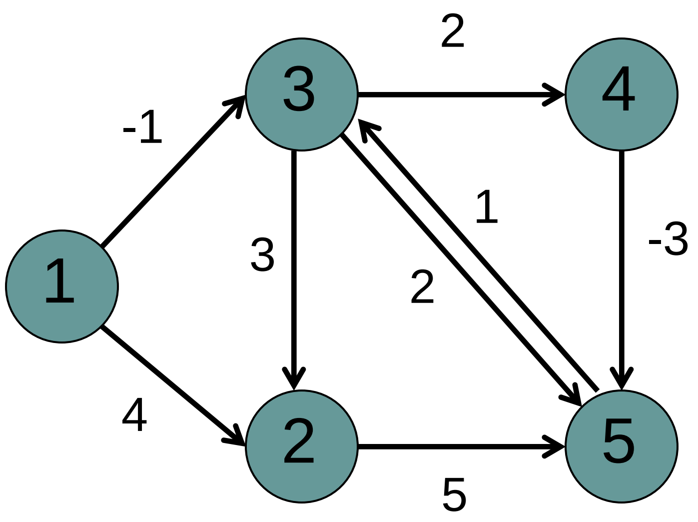
$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 4 & -1 & \infty & 9 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 3

$$\begin{array}{c|c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):
d^0 = W   // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $$\text{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$$

return d^V



k = 3

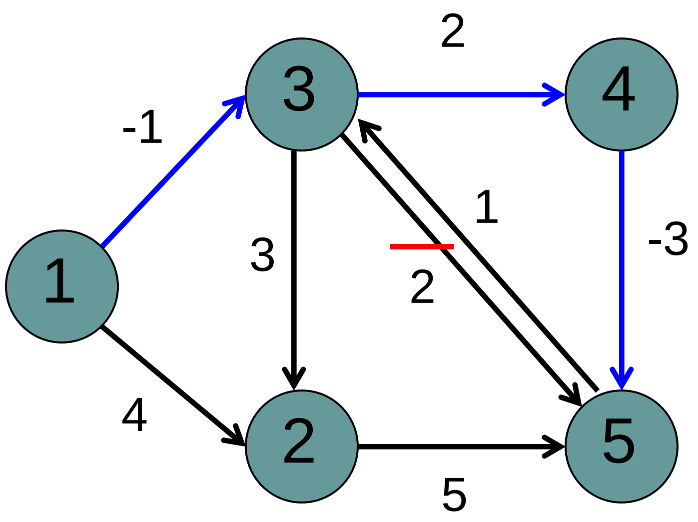
$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 4

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & ? \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):
d^0 = W // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$

return d^V



k = 3

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

minimum

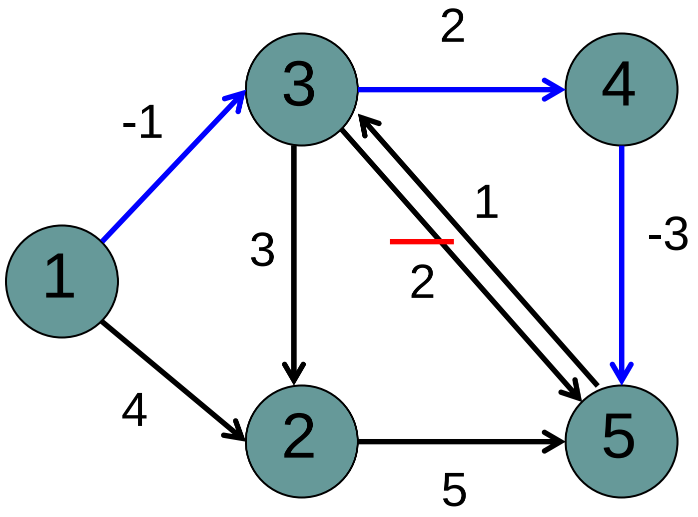
k = 4

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & -1 & 1 & -2 \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Found a shorter path!

Floyd-Warshall(G = (V,E,W)):
d^0 = W // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 3

$$\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix}
\end{array}$$

k = 4

$$\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & -2 \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}
\end{array}$$

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

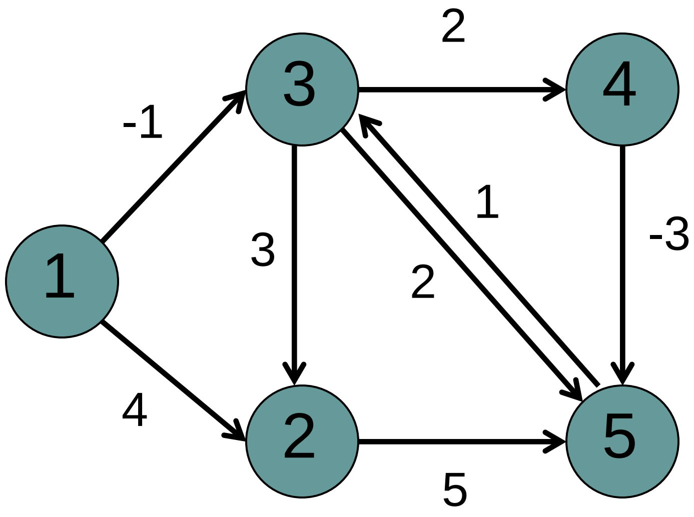
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

      $\text{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$

return d^V



k = 3

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

k = 4

	1	2	3	4	5
1	0	2	-1	1	-2
2	¥	0	¥	¥	5
3	¥	3	0	2	?
4					
5					

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

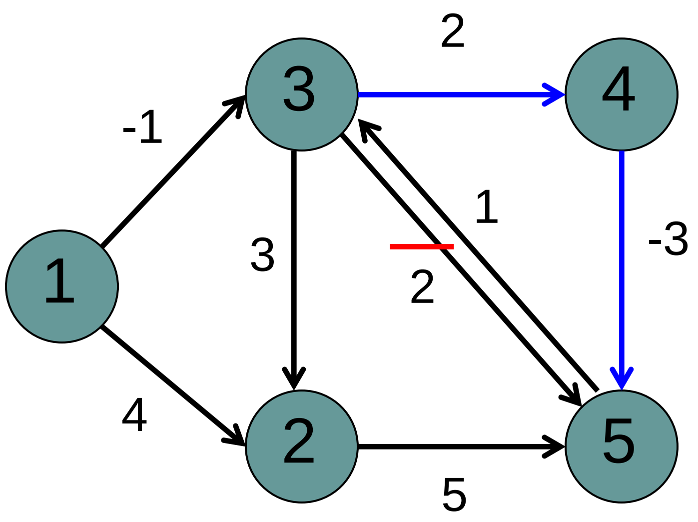
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\mathrm{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 3

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

minimum

k = 4

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 2 & -1 & 1 & -2 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & -1 \\ & & & & \\ & & & & \end{bmatrix} \end{array}$$

Found a shorter path!

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

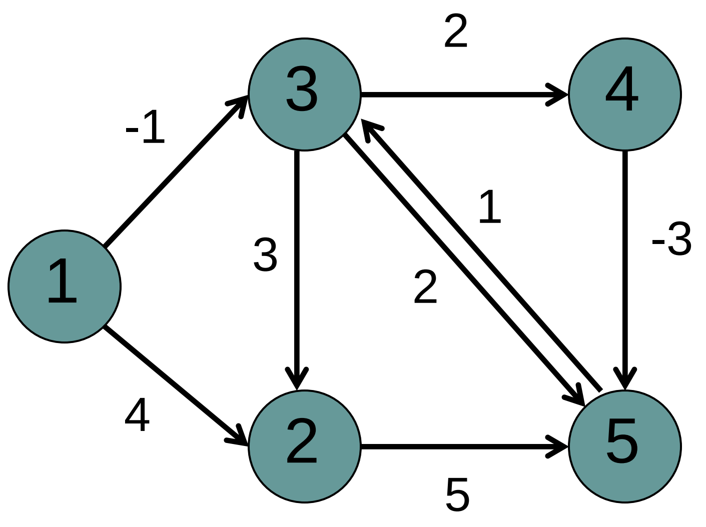
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$$

return d^V



k = 3

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & 1 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & 2 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

k = 4

$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 2 & -1 & 1 & -2 \\ \infty & 0 & \infty & \infty & 5 \\ \infty & 3 & 0 & 2 & -1 \\ \infty & \infty & \infty & 0 & -3 \\ \infty & \infty & 1 & \infty & 0 \end{bmatrix} \end{array}$$

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

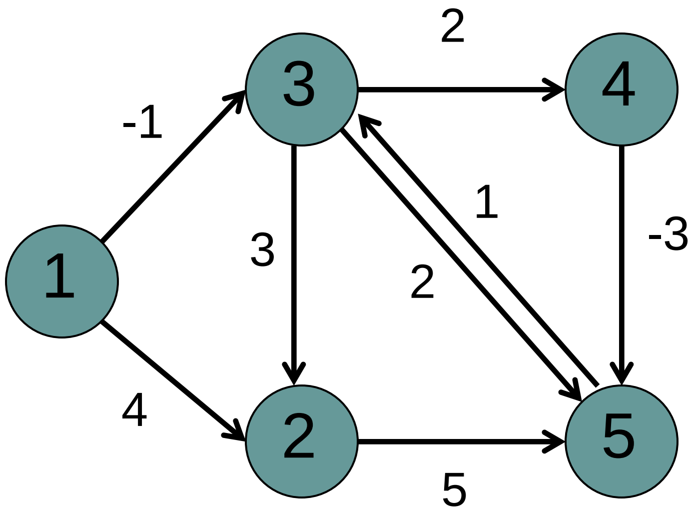
for k = 1 to V

  for i = 1 to V

    for j = 1 to V

$$\text{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$$

return d^V



k = 4

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

k = 5

	1	2	3	4	5
1	0	2	-1	1	-2
2	¥	0	7	9	5
3	¥	3	0	2	-1
4	¥	1	-2	0	-3
5	¥	¥	1	¥	0

Done!

Floyd-Warshall analysis

Is it correct?

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

for k = 1 to V

  for i = 1 to V

    for j = 1 to V

      $\boxed{\mathrm{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})}$

return d^V

Floyd-Warshall analysis

Is it correct?

Any assumptions?

Floyd-Warshall(G = (V,E,W)):
d^0 = W // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $\boxed{\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})}$

return d^V

Floyd-Warshall analysis

Is it correct?

Assuming the graph has no negative cycles!

What happens if there is a negative cycle?

Floyd-Warshall(G = (V,E,W)):
d^0 = W // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $\boxed{\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})}$

return d^V

Floyd-Warshall analysis

If the graph has a negative weight cycle, at the end, at least one of the diagonal entries will be a negative number, i.e., we there’s a way to get back to a vertex using all of the vertices that results in a negative weight

$$
\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} &
\begin{bmatrix}
0 & 2 & -1 & 1 & -2 \\
\infty & 0 & 7 & 9 & 5 \\
\infty & 3 & 0 & 2 & -1 \\
\infty & 1 & -2 & 0 & -3 \\
\infty & \infty & 1 & \infty & 0
\end{bmatrix}
\end{array}
$$

Floyd-Warshall analysis

Run-time?

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

for k = 1 to V

  for i = 1 to V

    for j = 1 to V

      $\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$

return d^V

Floyd-Warshall analysis

Run-time: $\theta(V^3)$

Floyd-Warshall(G = (V,E,W)):
d^0 = W // initialize with edge weights
for k = 1 to V
  for i = 1 to V
    for j = 1 to V
      $\text{dijk} = \min(d_{ij}{}^{k-1}, d_{ik}{}^{k-1} + d_{kj}{}^{k-1})$

return d^V

Floyd-Warshall analysis

Space usage?

Floyd-Warshall(G = (V,E,W)):

d^0 = W // initialize with edge weights

for k = 1 to V

for i = 1 to V

for j = 1 to V

$$\mathrm{dijk} = \min({d_{ij}}^{k-1}, {d_{ik}}^{k-1} + {d_{kj}}^{k-1})$$

return d^V

Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

d_{ij}^{k} = shortest path from vertex i to vertex j using only vertices $\{1, 2, \ldots, k\}$

If we want all possibilities, how many values are there (i.e. what is the size of d_{ij}^{k})?

Floyd-Warshall: key idea

Label all vertices with a number from 1 to V

d_{ij}^{k} = shortest path from vertex i to vertex j using only vertices $\{1, 2, \ldots, k\}$

V^3

- i: all vertices
- j: all vertices
- k: all vertices

Can we do better?

Floyd-Warshall analysis

Space usage: θ(V^2)

Only need the current value and the previous

```
Floyd-Warshall(G = (V,E,W)):
d^0 = W          // initialize with edge weights
for k = 1 to V
   for i = 1 to V
      for j = 1 to V
         dijk = min(d_ij^{k-1}, d_ik^{k-1} + d_kj^{k-1})

return d^V
```

All pairs shortest paths

V * Bellman-Ford: $O(V^2E)$

Floyd-Warshall: $\theta(V^3)$

All pairs shortest paths

All pairs shortest paths for positive weight graphs: calculate the shortest paths between *all* points

Easy solution?

All pairs shortest paths

All pairs shortest paths for positive weight graphs: calculate the shortest paths between *all* points

Run Dijsktras from each vertex!

Running time (in terms of E and V)?

All pairs shortest paths

All pairs shortest paths for positive weight graphs: calculate the shortest paths between *all* points

Run Dijsktras from each vertex!

$O(V^2 \log V + V E)$

- V calls do Dijkstras
- Dijkstras: $O(V \log V + E)$

All pairs shortest paths

V * Bellman-Ford: $O(V^2E)$

Floyd-Warshall: $\theta(V^3)$

V * Dijkstras: $O(V^2 \log V + V E)$

Is this any better?

All pairs shortest paths

V * Bellman-Ford: $O(V^2E)$

Floyd-Warshall: $\theta(V^3)$

V * Dijkstras: $O(V^2 \log V + V E)$

If the graph is sparse!

All pairs shortest paths

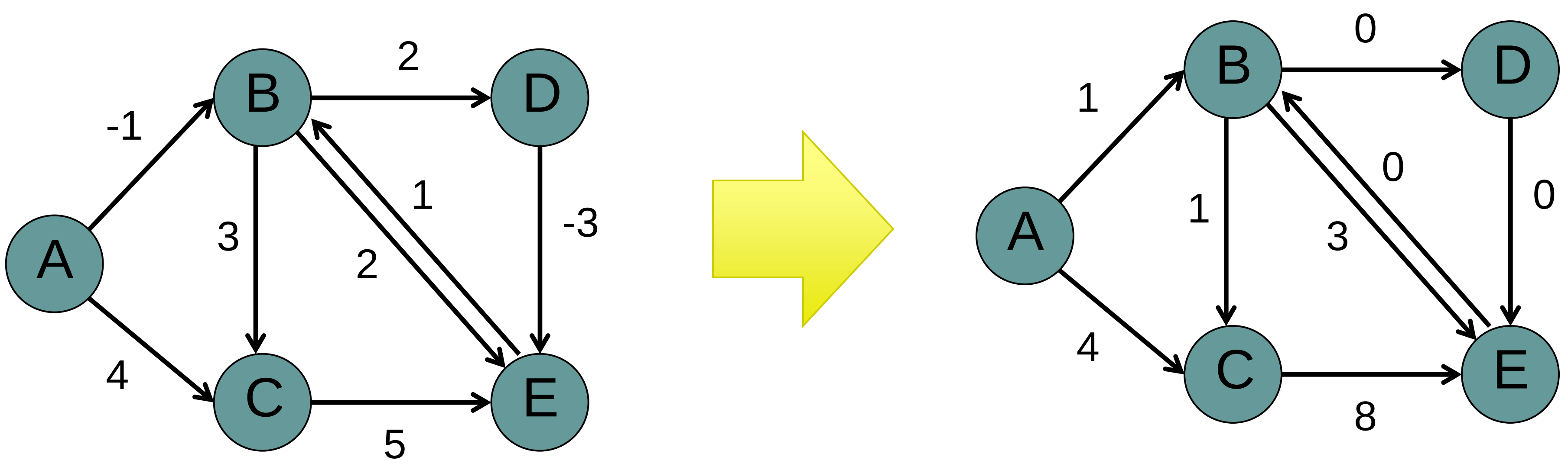
All pairs shortest paths for positive weight graphs: calculate the shortest paths between *all* points

Run Dijsktras from each vertex!

Challenge: Dijkstras assumes positive weights

Johnson's: key idea

Reweight the graph to make all edges positive
such that shortest paths are preserved



Lemma

let *h* be *any* function mapping a vertex to a real value

If we change the graph weights as:

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

The shortest paths are preserved

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

Lemma: proof

Let *s*, v_1, v_2, …, v_k, *t* be a path from *s* to *t*

The weight in the reweighted graph is:

$$\hat{w}(s,v_1,...,v_k,t) = w(s,v_1) + h(s) - h(v_1) + \hat{w}(v_1,...,v_k,t)$$

$$= w(s,v_1) + h(s) - h(v_1) + w(v_1,v_2) + h(v_1) - h(v_2) + \hat{w}(v_2,...,v_k,t)$$

$$= w(s,v_1) + h(s) + w(v_1,v_2) - h(v_2) + \hat{w}(v_2,...,v_k,t)$$

$$= w(s,v_1) + h(s) + w(v_1,v_2) - h(v_2) + w(v_2,v_3) + h(v_2) - h(v_3) + \hat{w}(v_3,...,v_k,t)$$

$$= w(s,v_1) + h(s) + w(v_1,v_2) + w(v_2,v_3) - h(v_3) + \hat{w}(v_3,...,v_k,t)$$

…

$$= w(s,v_1,...,v_k,t) + h(s) - h(t)$$

Lemma: proof

$$\hat{w}(s, v_1, \ldots, v_k, t) = w(s, v_1, \ldots, v_k, t) + h(s) - h(t)$$

Claim: the weight change preserves shortest paths, i.e. if a path was the shortest from s to t in the original graph it will still be the shortest path from s to t in the new graph.

Justification?

Lemma: proof

$$\hat{w}(s, v_1, \ldots, v_k, t) = w(s, v_1, \ldots, v_k, t) + h(s) - h(t)$$

Claim: the weight change preserves shortest paths, i.e. if a path was the shortest from s to t in the original graph it will still be the shortest path from s to t in the new graph.

h(s) – h(t) is a constant and will be the same for all paths from s to t, so the absolute ordering of all paths from s to t will not change.

Lemma

let *h* be *any* function mapping a vertex to a real value

If we change the graph weights as:

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

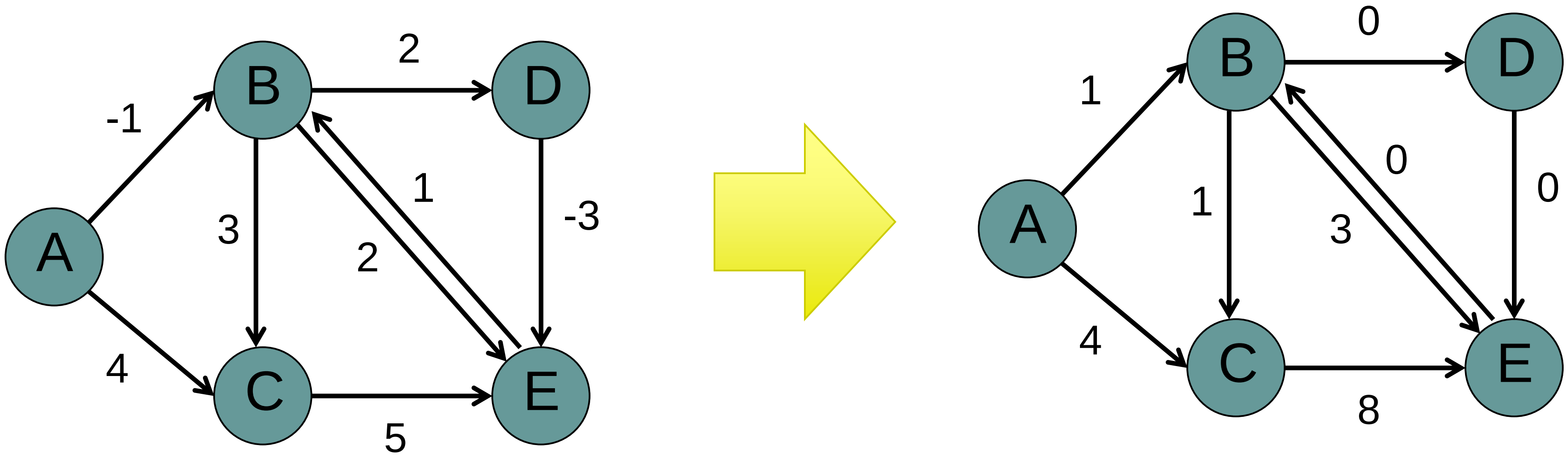
The shortest paths are preserved

Big question: how do we pick h?

Selecting h

Need to pick h such that the resulting graph has all weights as positive

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$



Johnson’s algorithm

Create G’ with one extra node s with 0 weight edges to all nodes
run Bellman-Ford(G’,s)

if no negative-weight cycle

    reweight edges in G with h(v)=shortest path from s to v
    run Dijkstra’s from every vertex
    reweight shortest paths based on G

Create G'

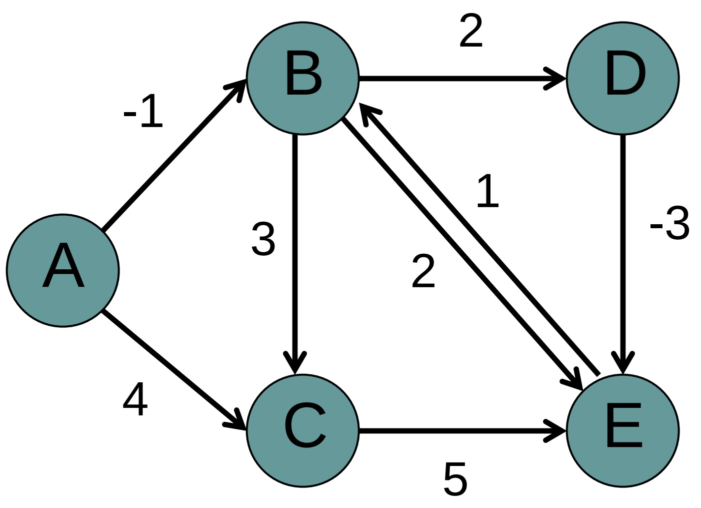
run Bellman-Ford(G',s)

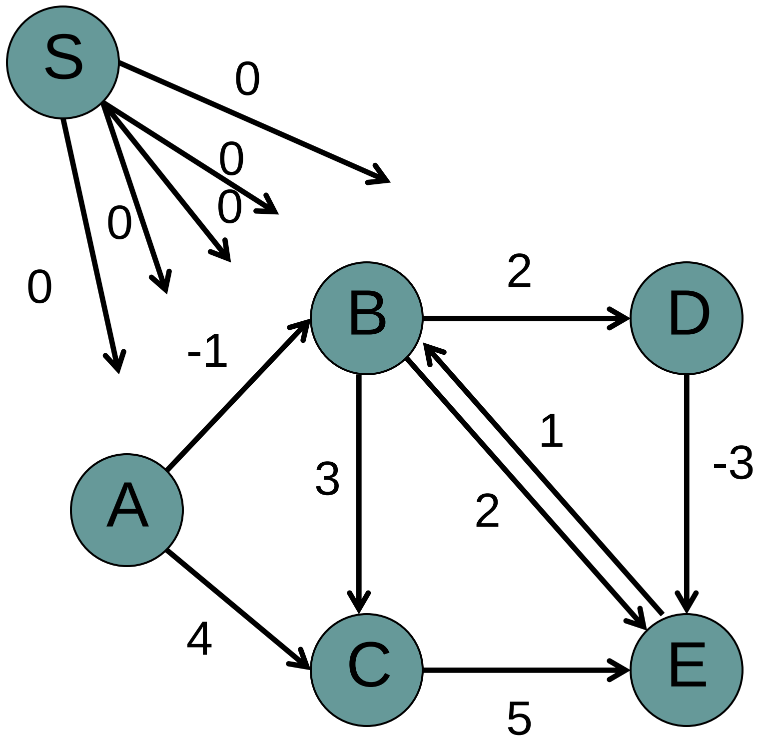
if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G





Create G’

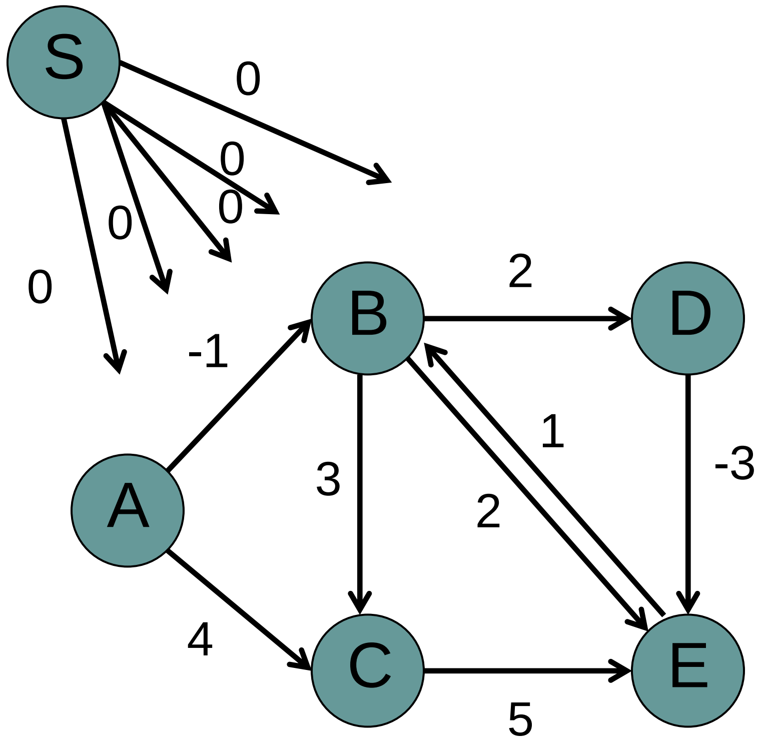
run Bellman-Ford(G’,s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G



S➔A: ?

S➔B:

S➔C:

S➔D:

S➔E:

Create G'

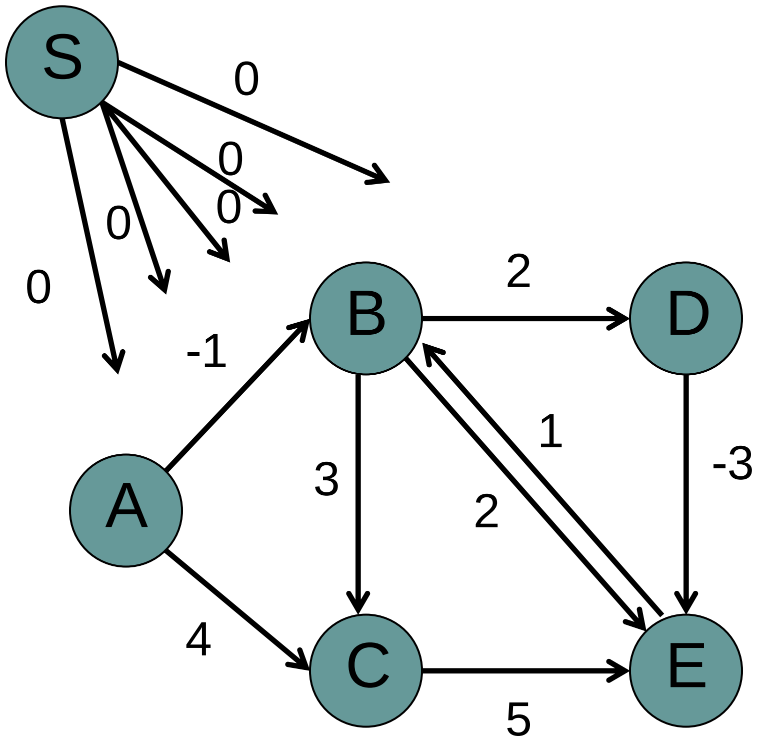
run Bellman-Ford(G',s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G



S➔A: 0

S➔B:

S➔C:

S➔D:

S➔E:

Create G’

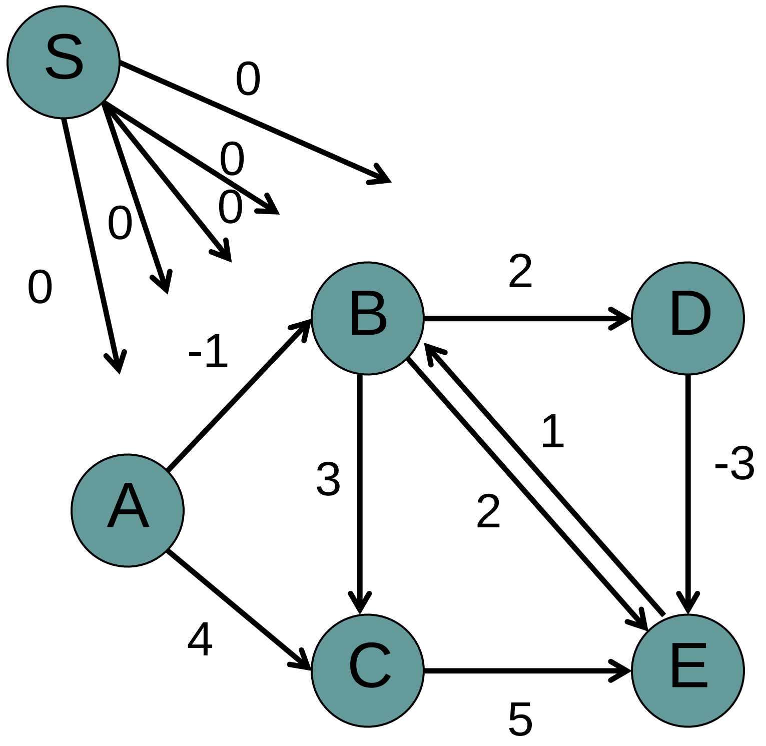
run Bellman-Ford(G’,s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G



S➔A: 0

S➔B: ?

S➔C:

S➔D:

S➔E:

Create G’

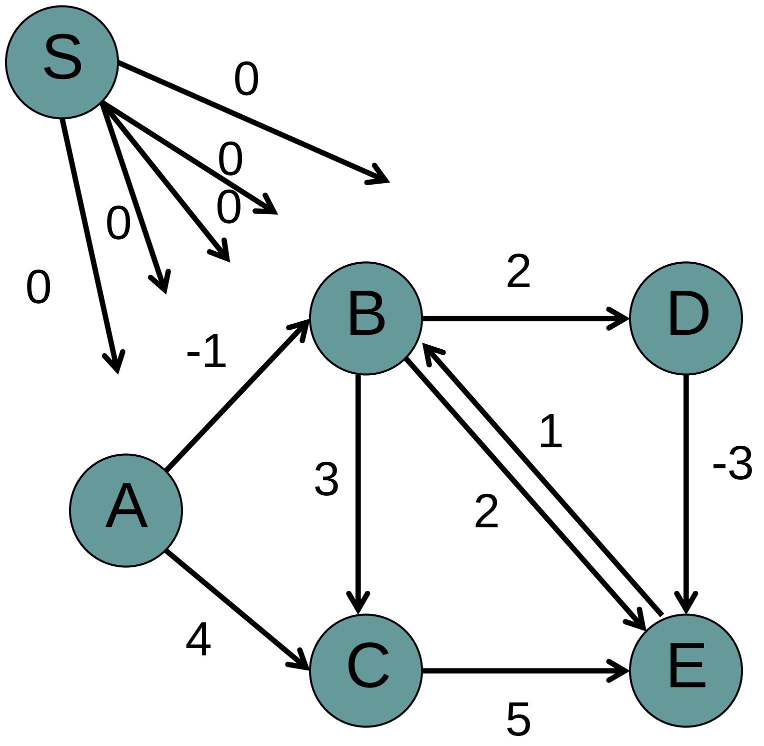
run Bellman-Ford(G’,s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G



S➔A: 0
S➔B: -2
S➔C:
S➔D:
S➔E:

Create G’

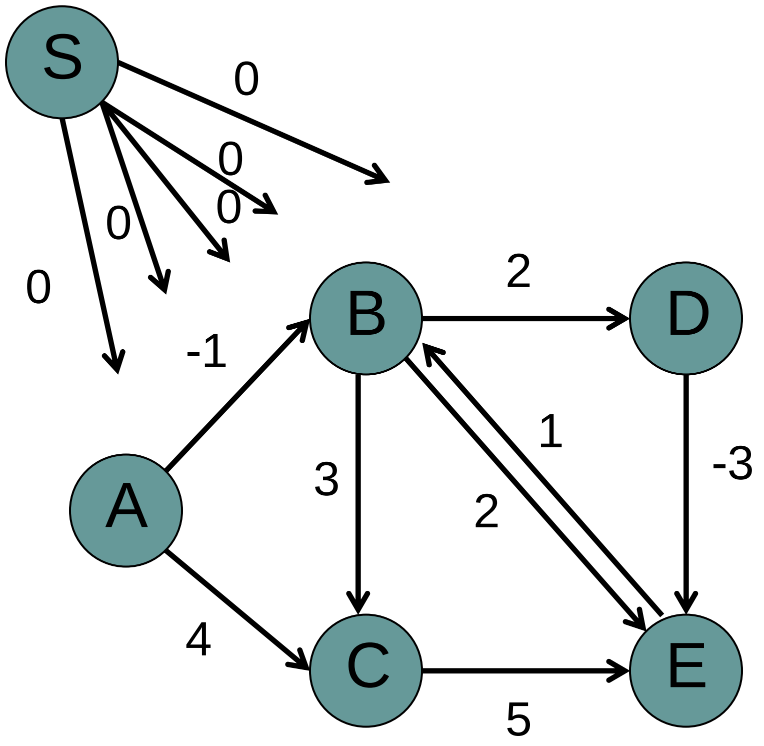
run Bellman-Ford(G’,s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G



S➔A: 0

S➔B: -2

S➔C: 0

S➔D: 0

S➔E: -3

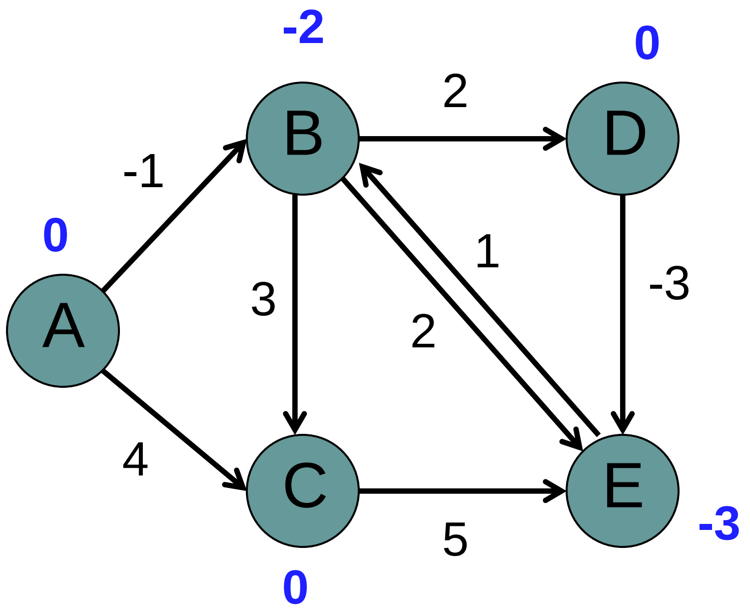
S➔A: 0

S➔B: -2

S➔C: 0

S➔D: 0

S➔E: -3



Create G’

run Bellman-Ford(G’,s)

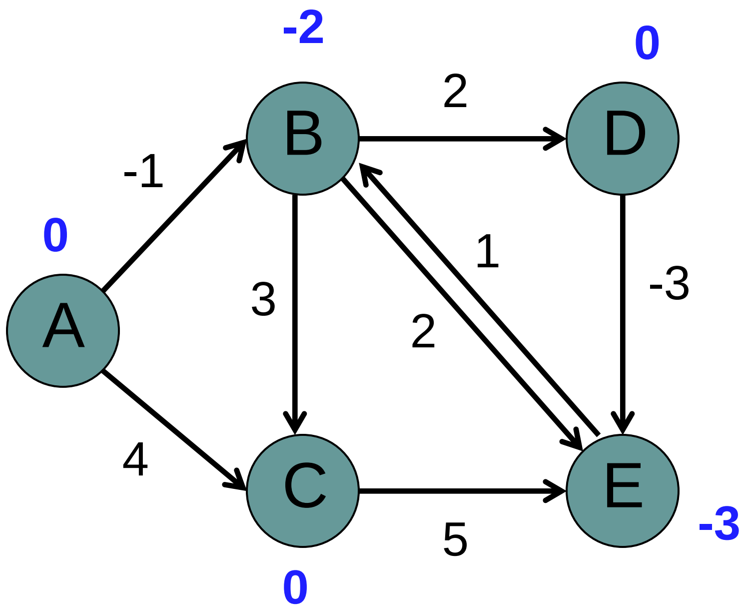
if no negative-weight cycle

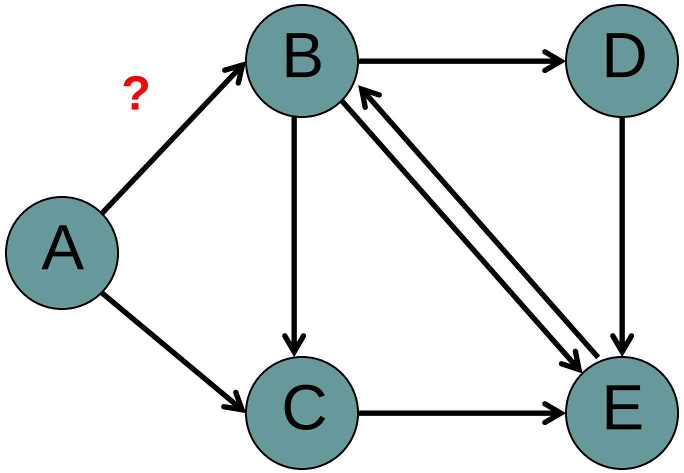
reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$





Create G’

run Bellman-Ford(G’,s)

if no negative-weight cycle

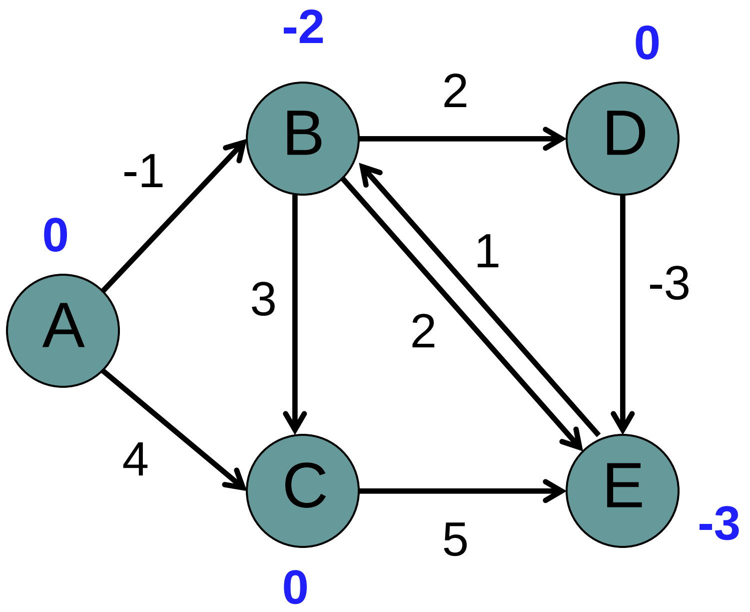
    reweight edges in G with h(v)=shortest path from s to v

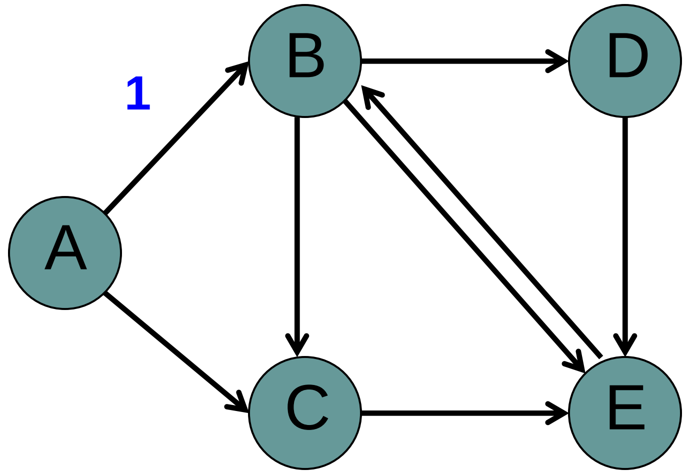
    run Dijkstra’s from every vertex

    reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

-1 + 0 - -2





Create G'

run Bellman-Ford(G',s)

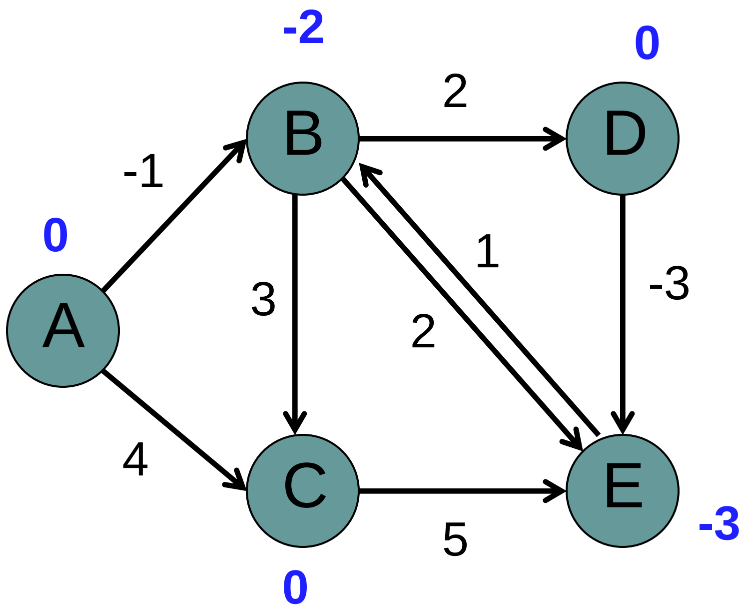
if no negative-weight cycle

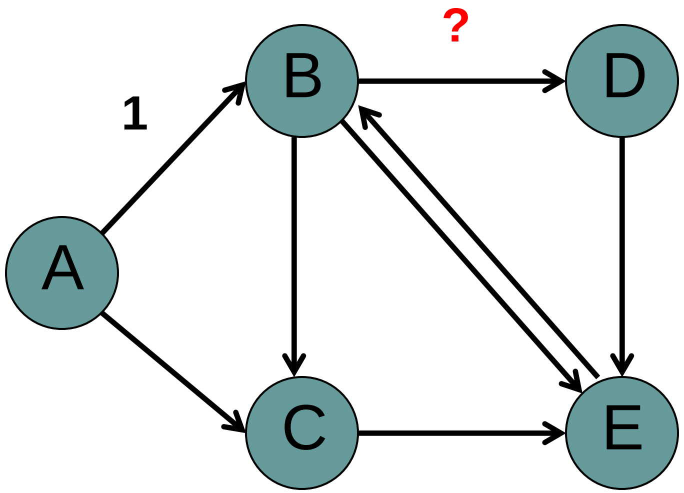
reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$





Create G'

run Bellman-Ford(G',s)

if no negative-weight cycle

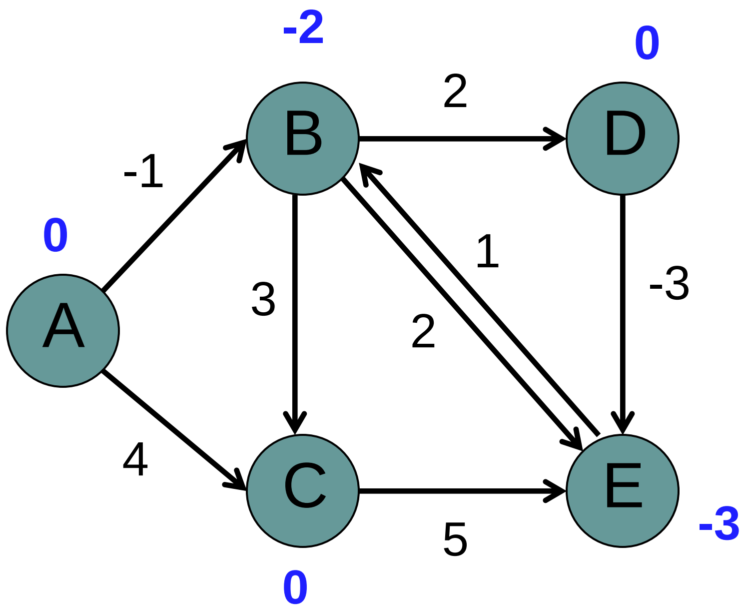
reweight edges in G with h(v)=shortest path from s to v

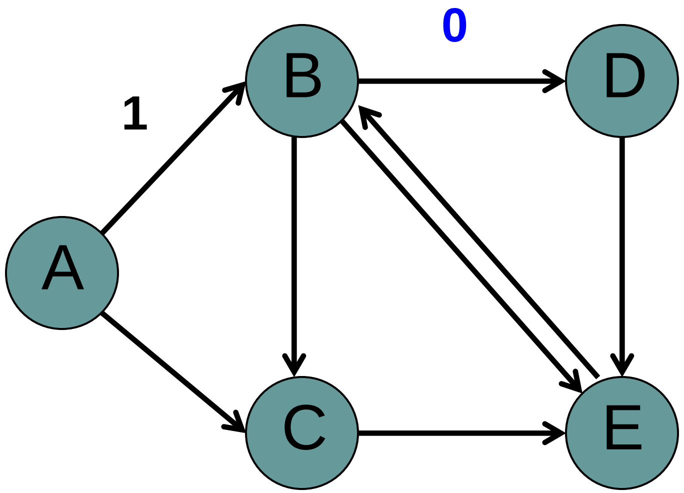
run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

2 + -2 - 0





Create G'

run Bellman-Ford(G',s)

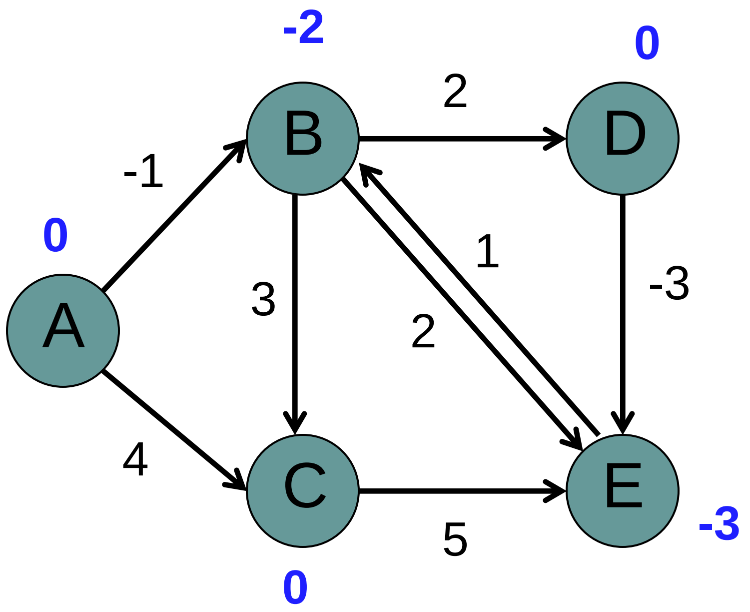
if no negative-weight cycle

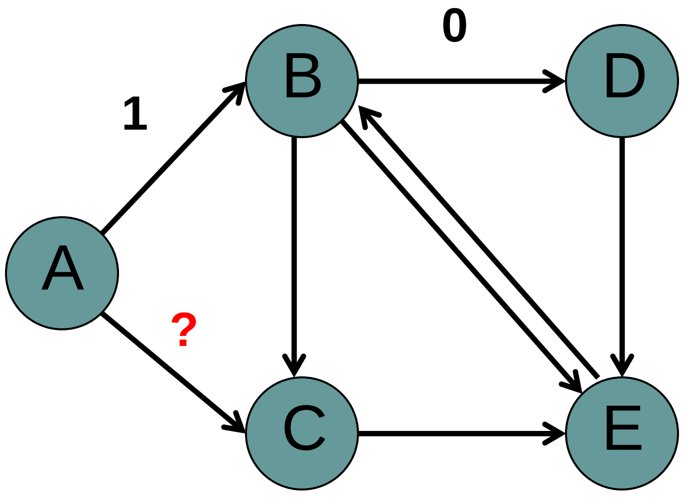
reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$





Create G'

run Bellman-Ford(G',s)

if no negative-weight cycle

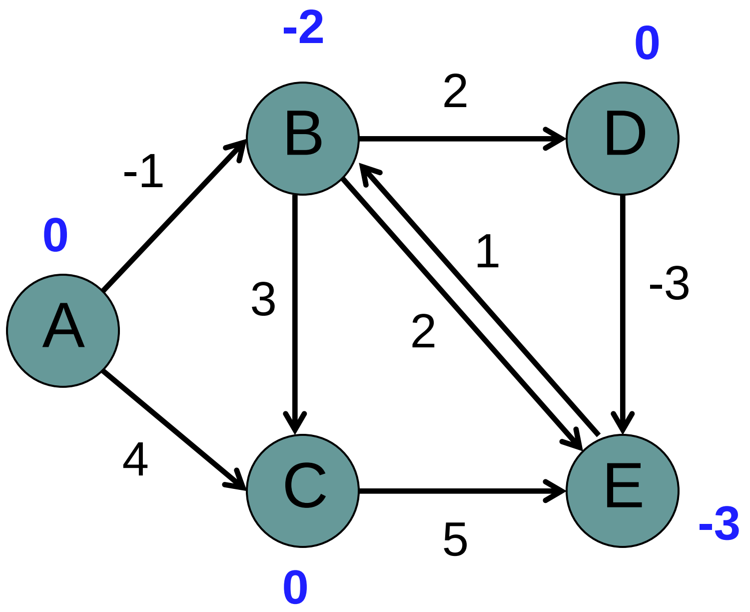
reweight edges in G with h(v)=shortest path from s to v

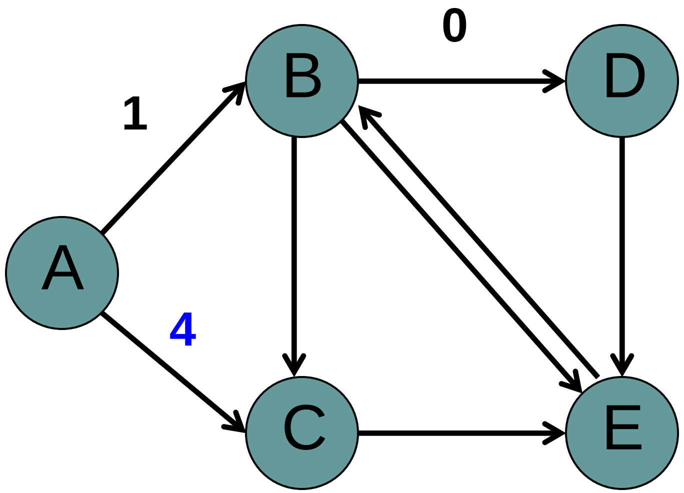
run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

$$4 \quad + \; 0 \; - \; 0$$





Create G'

run Bellman-Ford(G',s)

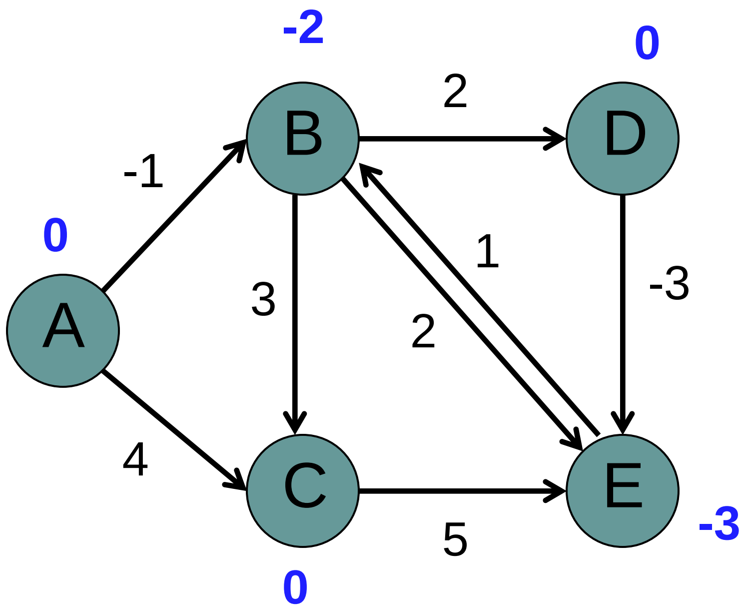
if no negative-weight cycle

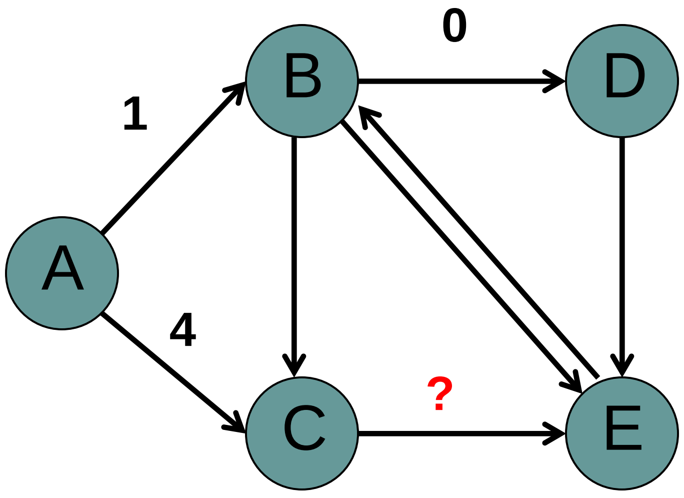
reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$





Create G'

run Bellman-Ford(G',s)

if no negative-weight cycle

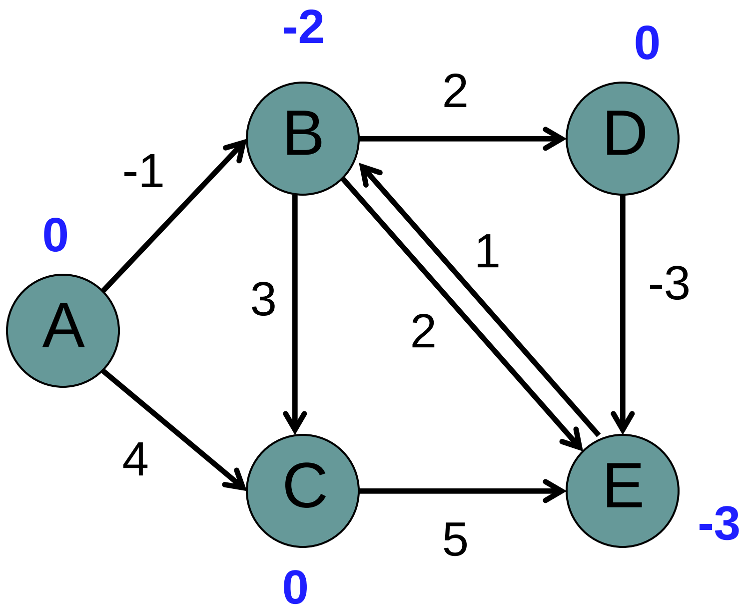
reweight edges in G with h(v)=shortest path from s to v

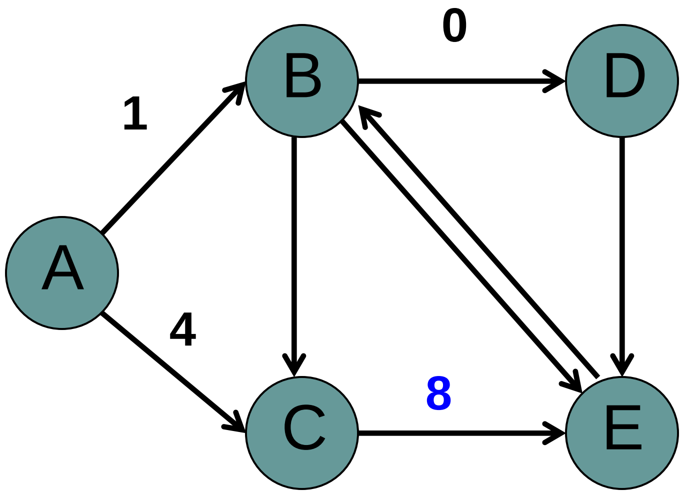
run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

$$5 \quad + \; 0 \; - \; -3$$





Create G'

run Bellman-Ford(G',s)

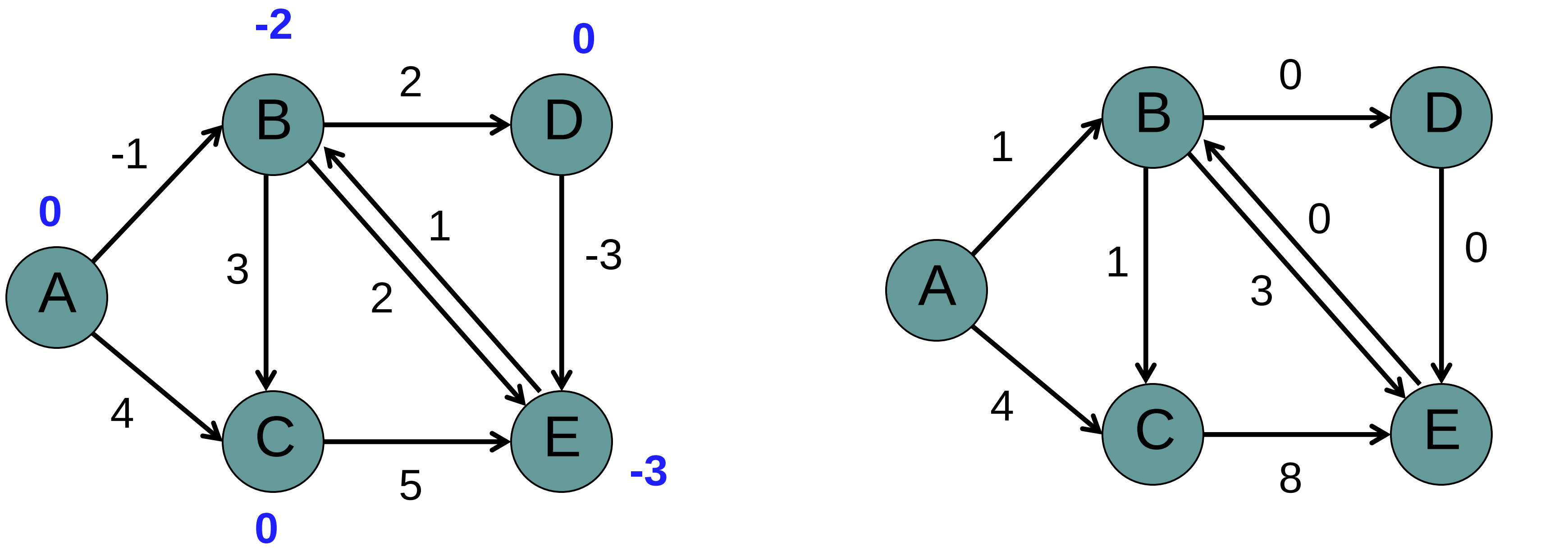
if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$



Create G’

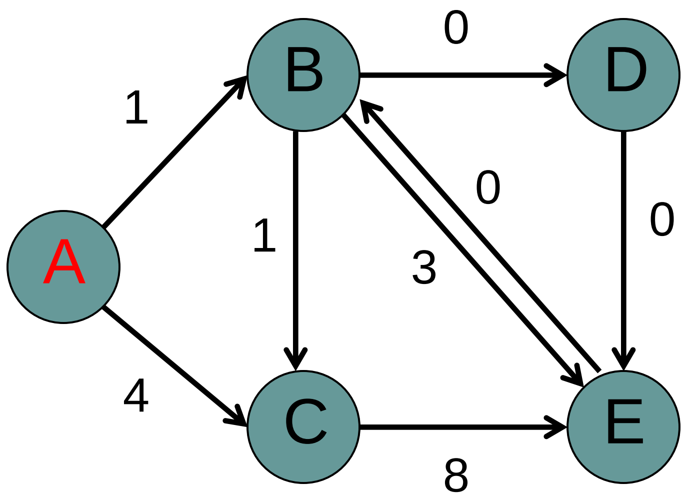
run Bellman-Ford(G’,s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra’s from every vertex

reweight shortest paths based on G



Create G'

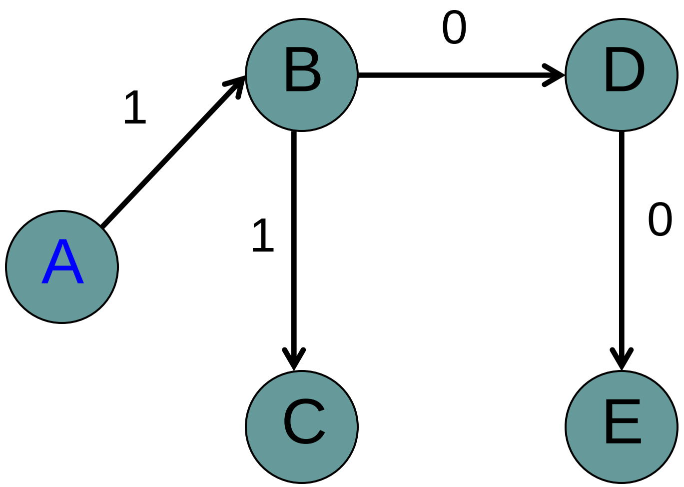
run Bellman-Ford(G',s)

if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G



Create G'

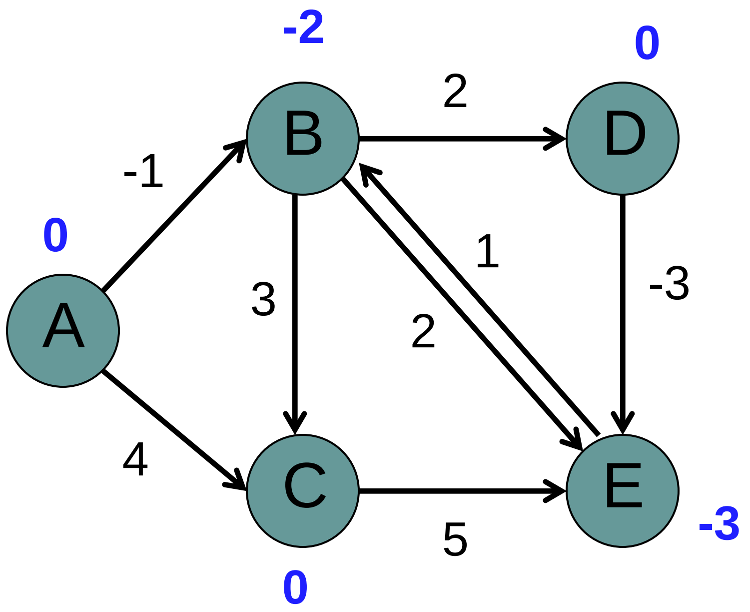
run Bellman-Ford(G',s)

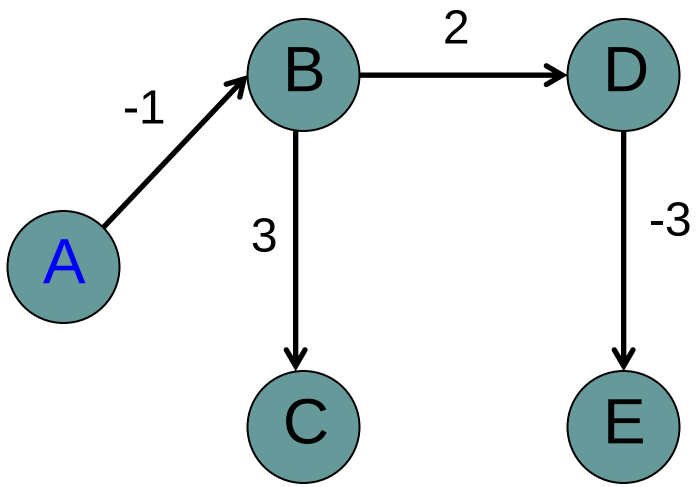
if no negative-weight cycle

reweight edges in G with h(v)=shortest path from s to v

run Dijkstra's from every vertex

reweight shortest paths based on G



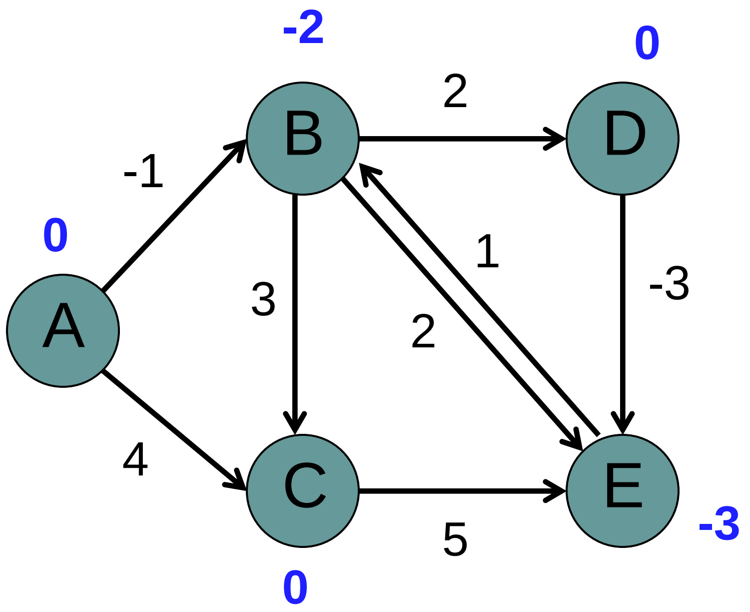


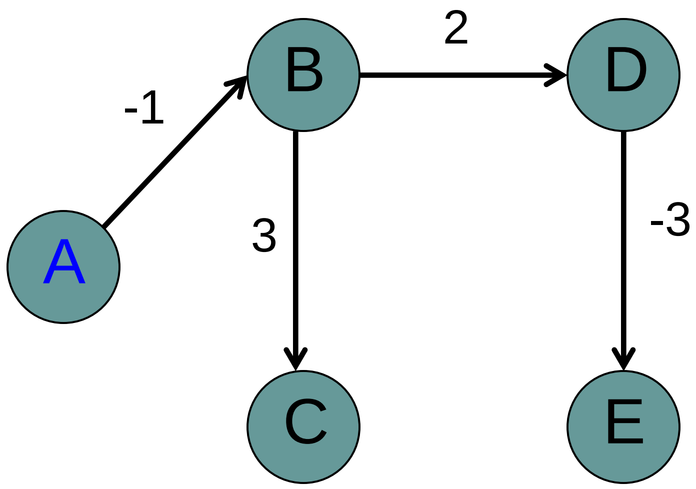
A→B: -1

A→C: 2

A→D: 1

A→E: -2





Selecting h

Need to pick h such that the resulting graph has all weights as positive

Create G' with one extra node s with 0 weight edges to all nodes
run Bellman-Ford(G',s)
if no negative-weight cycle
    reweight edges in G with h(v)=shortest path from s to v
    run Dijkstra's from every vertex
    reweight shortest paths based on G

Why does this work (i.e. how do we guarantee that reweighted graph has only positive edges)?

Reweighted graph is positive

Take two nodes u and v

h(u) shortest distance from s to u
h(v) shortest distance from s to v

Claim: $h(v) \leq h(u) + w(u,v)$

Why?

Reweighted graph is positive

Take two nodes u and v

h(u) shortest distance from s to u
h(v) shortest distance from s to v

Claim: $h(v) \leq h(u) + w(u, v)$

If this weren’t true, we could have made a shorter path s to v using u

… but this is in contradiction with how we defined h(v)

Reweighted graph is positive

Take two nodes u and v

h(u) shortest distance from s to u
h(v) shortest distance from s to v

$$h(v) \le h(u) + w(u,v)$$

$$\underbrace{w(u,v) + h(u) - h(v)}_{\text{What is this?}} \ge 0$$

Reweighted graph is positive

Take two nodes u and v

h(u) shortest distance from s to u
h(v) shortest distance from s to v

$$h(v) \le h(u) + w(u,v)$$

$$\underbrace{w(u,v) + h(u) - h(v)} \ge 0$$

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v) \ge 0$$

All edge weights in reweighted graph
are non-negative

Johnson’s algorithm

Create G’

run Bellman-Ford(G’,s)

if no negative-weight cycle

    reweight edges in G

    run Dijkstra’s from every vertex

    reweight shortest paths based on G

Run-time?

Johnson's algorithm

Create G' $\theta(V)$

run Bellman-Ford(G',s) $O(V^2)$

if no negative-weight cycle

reweight edges in G $\theta(E)$

run Dijkstra's from every vertex $O(V^2 \log V + VE)$

reweight shortest paths based on G $\theta(E)$

Run-time?

All pairs shortest paths

V * Bellman-Ford: $O(V^2E)$

Floyd-Warshall: $\theta(V^3)$

Johnson's: $O(V^2 \log V + V E)$

Acknowledgements

- Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C., Introduction to algorithms. MIT press, 2009
- Dr. David Kauchak, Pomona College
- Prof. David Plaisted, The University of North Carolina at Chapel Hill