Faculty of Computer Science
Data Science and Business Analytics (DSBA)

Algorithms and Data Structures

Seminar 11 – Module 3

Bellman-Ford Algorithm
and
Negative Costs

February 2022

J.C. Carrasquel

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

/ Find the minimum path cost from s to every other node in V */*

Bellman-Ford(G = (V,E),**s**):

$d^{(0)}[v] = \infty$ forall v in V

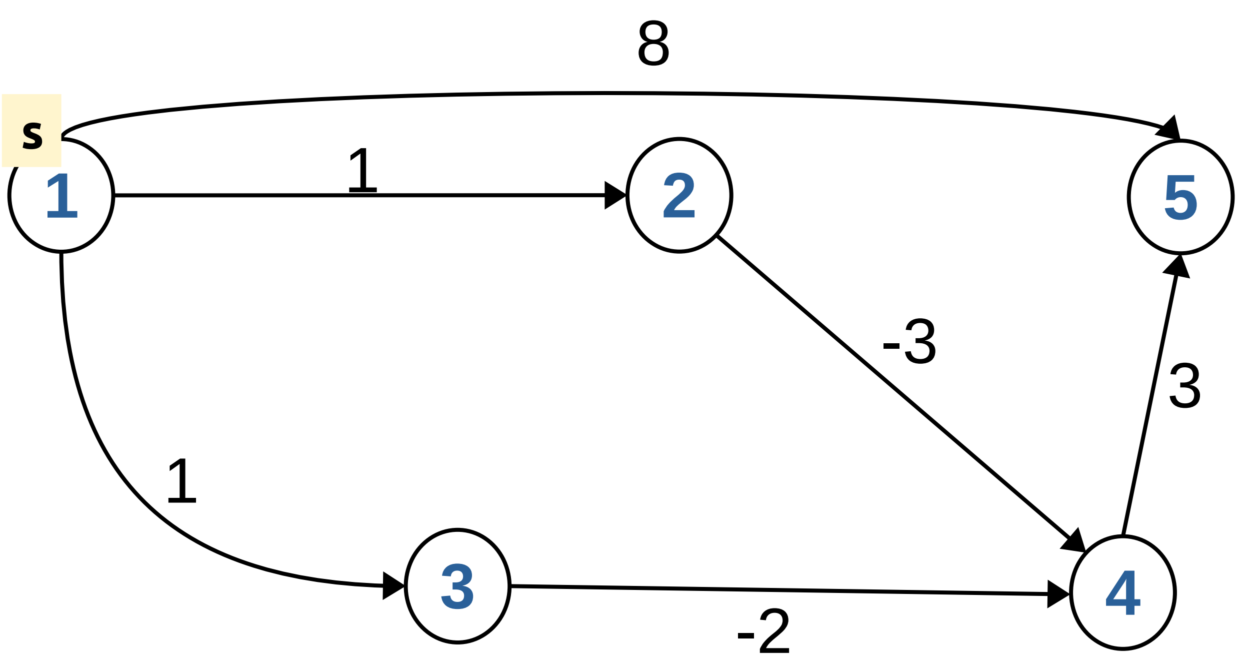
$d^{(0)}[\mathbf{s}] = 0$

For i = 1…n -1: // n nodes

  For each node u in V:

    For node v in neighbors(u):

      $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$

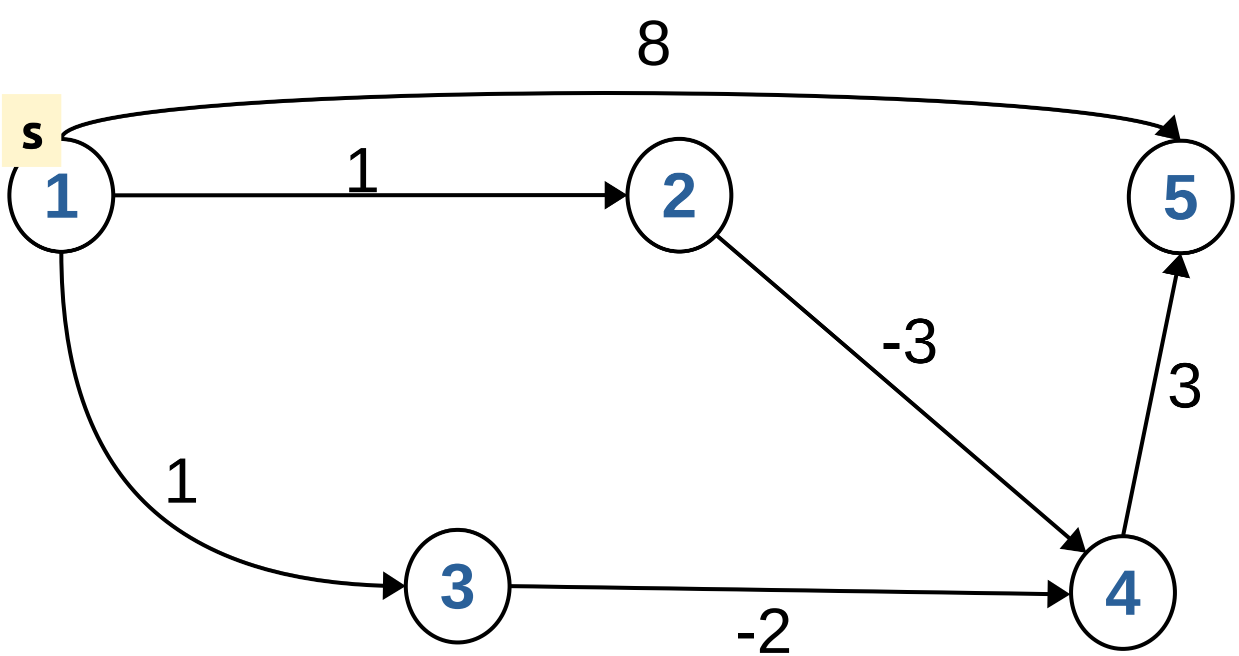


	1	2	3	4	5
$d^{(0)}$:					

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

```
/* Find the minimum path cost from s to every other node in V */
Bellman-Ford(G = (V,E),s):
d(0)[v] = ∞ forall v in V
d(0)[s] = 0
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        d(i)[v] = min(d(i-1)[v],d(i-1)[u] + edgeWeight(u,v))
```

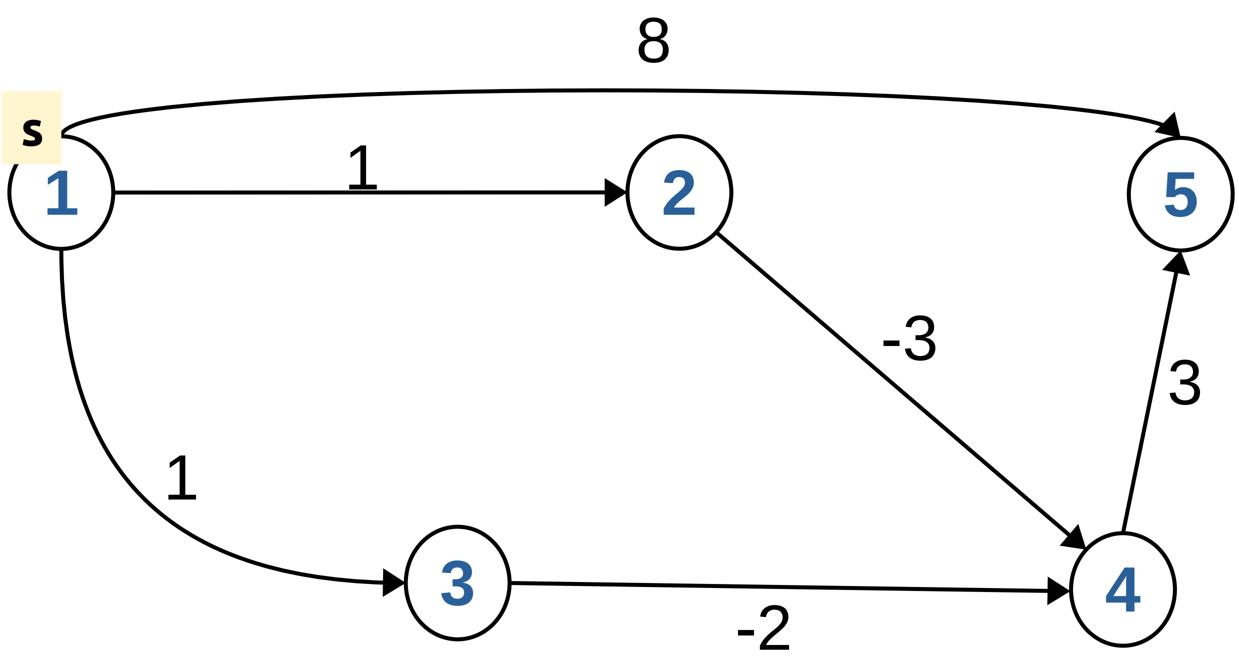


	1	2	3	4	5
$d^{(0)}$:					

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
  For each edge (u,v) in E:
    $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$



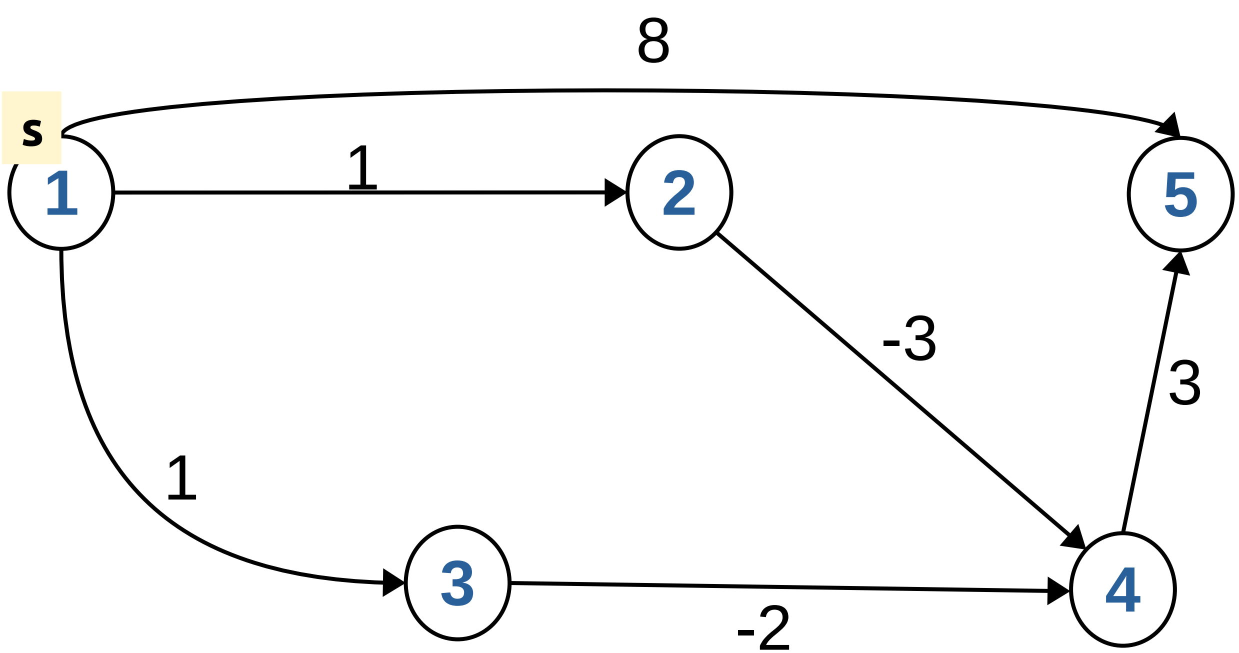
	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0				

E = { ((**1,2**) , 1), ((**1,3**), 1)), ((**1,5**), 8), ((**2,4**) , -3), ((**3,4**), -2), ((**4,5**), 3) }

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-2	1

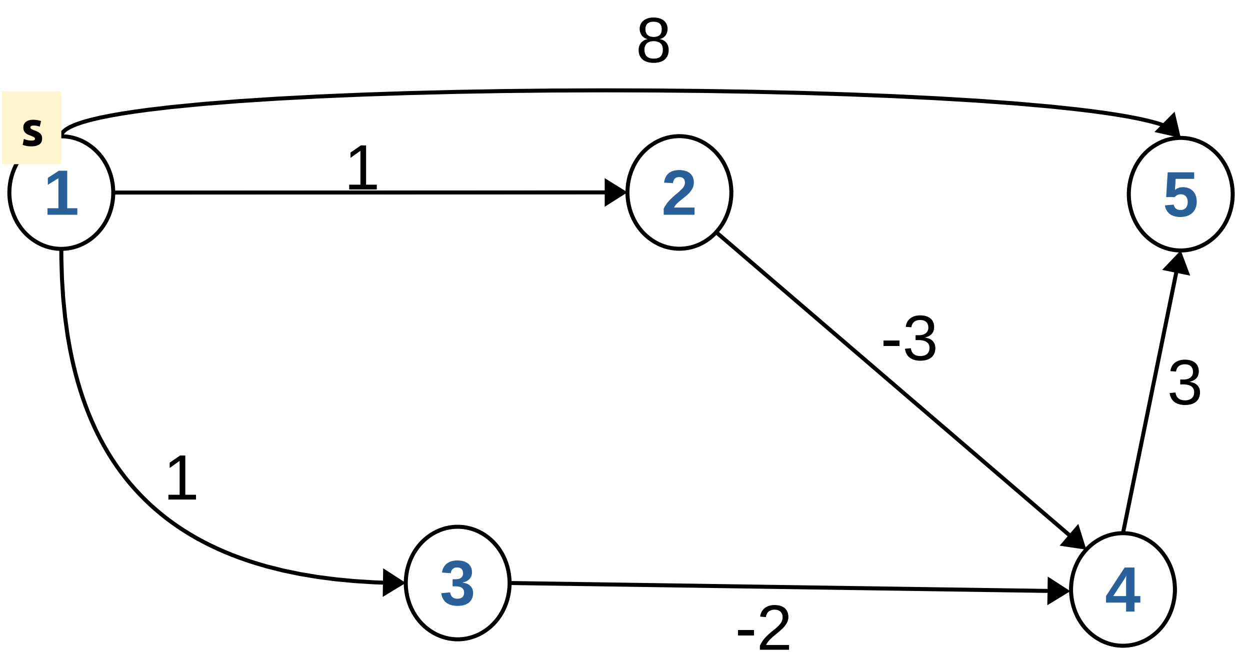
…

E = { ((**1,2**) , 1), ((**1,3**), 1)), ((**1,5**), 8), ((**2,4**) , -3), ((**3,4**), -2), ((**4,5**), 3) }

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-2	1

...

If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

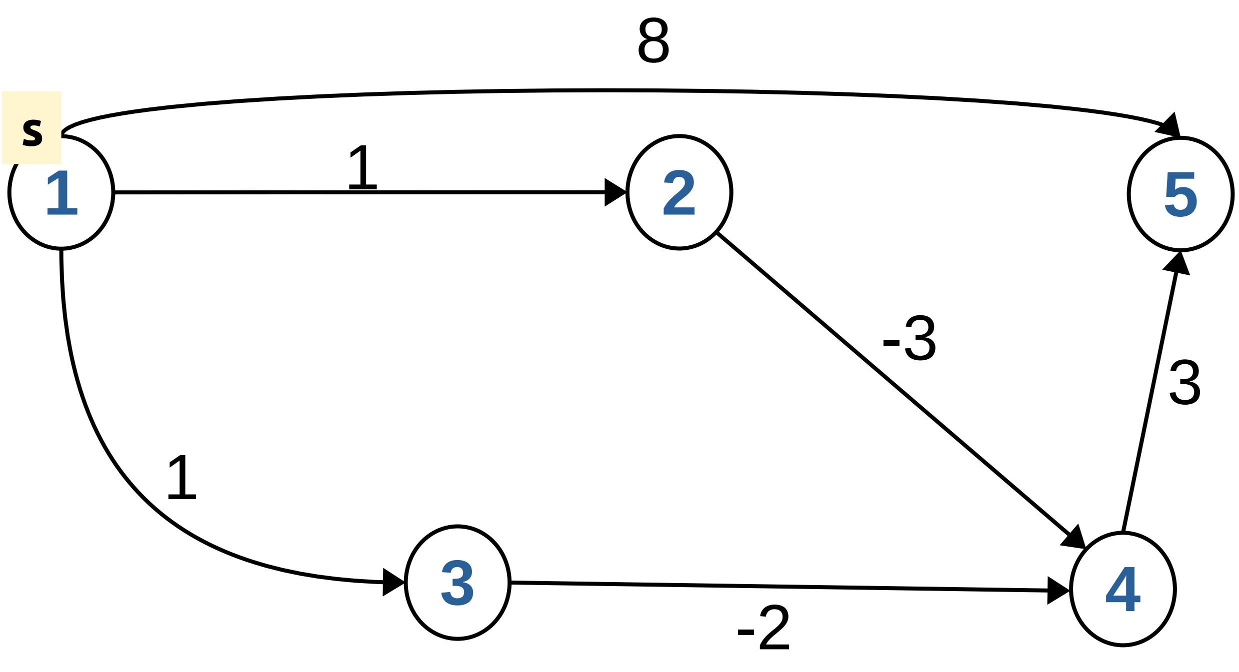
E = { ((**1,2**) , 1), ((**1,3**), 1)), ((**1,5**), 8), ((**2,4**) , -3), ((**3,4**), -2), ((**4,5**), 3) }

Bellman-Ford Algorithm

Can handle edges with negative costs (weights)

```
/* Find the minimum path cost from s to every other node in V */
Bellman-Ford(G = (V,E),s):
d(0)[v] = ∞ forall v in V
d(0)[s] = 0
For i = 1...n -1: // n nodes
      For each edge (u,v) in E:
            d(i)[v] = min(d(i-1)[v],d(i-1)[u] + edgeWeight(u,v))
```

Generally, we do not assume graphs to be *topologically sorted*.
So, in the worst case may take n-1 iterations
To get the correct min. path cost
n-1: maximum length of a path.



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-2	1

...

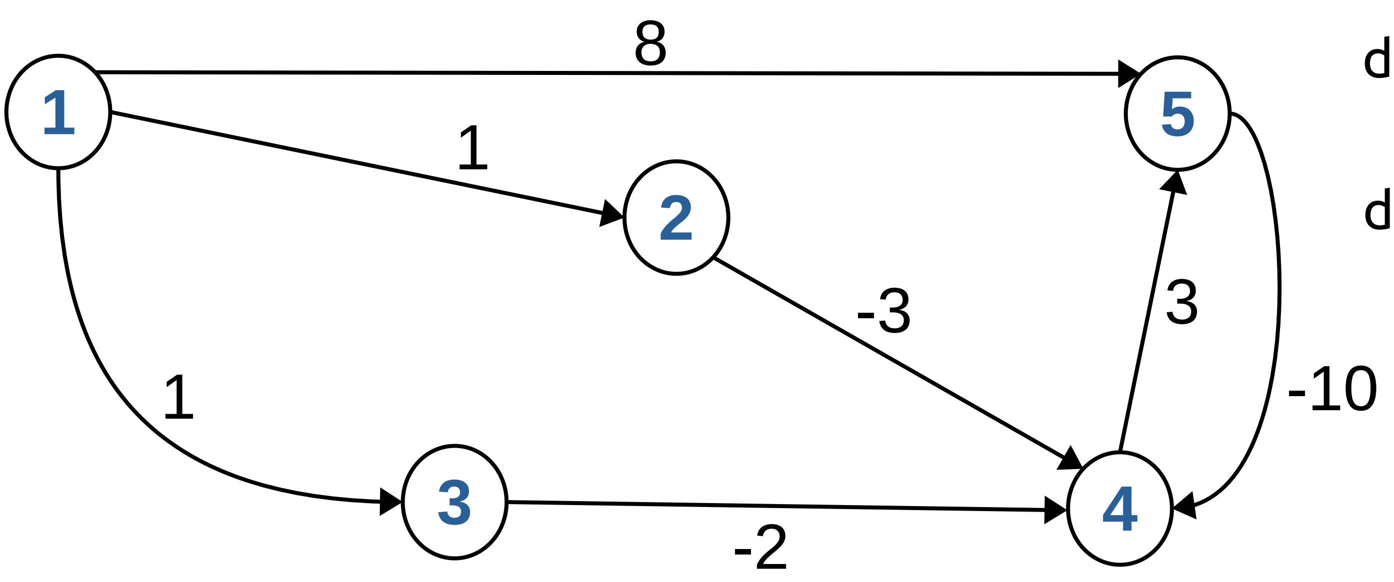
If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

E = { ((**1**,**2**) , 1), ((**1**,**3**), 1)), ((**1**,**5**), 8), ((**2**,**4**) , -3), ((**3**,**4**), -2), ((**4**,**5**), 3) }

Bellman-Ford Algorithm

What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

```
/* Find the minimum path cost from s to every other node in V */
Bellman-Ford(G = (V,E),s):
d(0)[v] = ∞ forall v in V
d(0)[s] = 0
For i = 1…n -1: // n nodes
      For each edge (u,v) in E:
            d(i)[v] = min(d(i-1)[v],d(i-1)[u] + edgeWeight(u,v))
```



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-2	1

…

E = { ((1,2) , 1), ((1,3), 1)), ((1,5), 8), ((2,4) , -3), ((3,4), -2), ((4,5), 3), ((5,4), -10) }

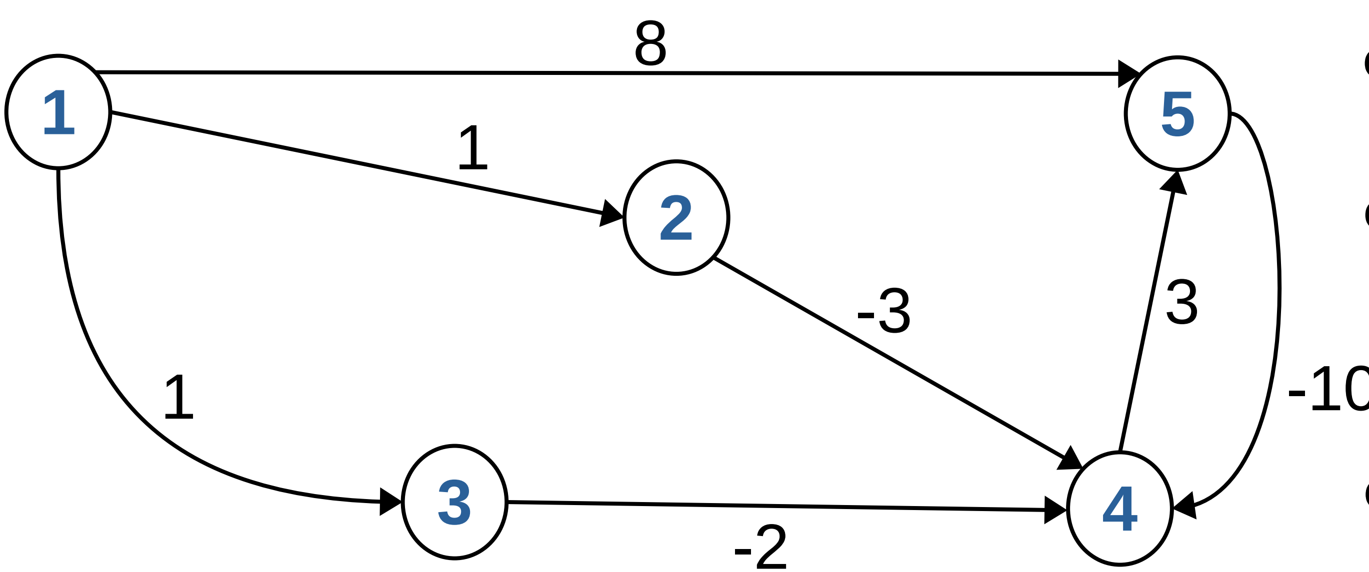
Bellman-Ford Algorithm

What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

```
/* Find the minimum path cost from s to every other node in V */
Bellman-Ford(G = (V,E),s):
d^(0)[v] = ∞ forall v in V
d^(0)[s] = 0
For i = 1…n -1: // n nodes
      For each edge (u,v) in E:
            d^(i)[v] = min(d^(i-1)[v],d^(i-1)[u] + edgeWeight(u,v))
```

If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

What if we do one more iteration?



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-9	1
…					
$d^{(2)}$:					

E = { ((1,2) , 1), ((1,3), 1)), ((1,5), 8), ((2,4) , -3), ((3,4), -2), ((4,5), 3), ((5,4), -10) }

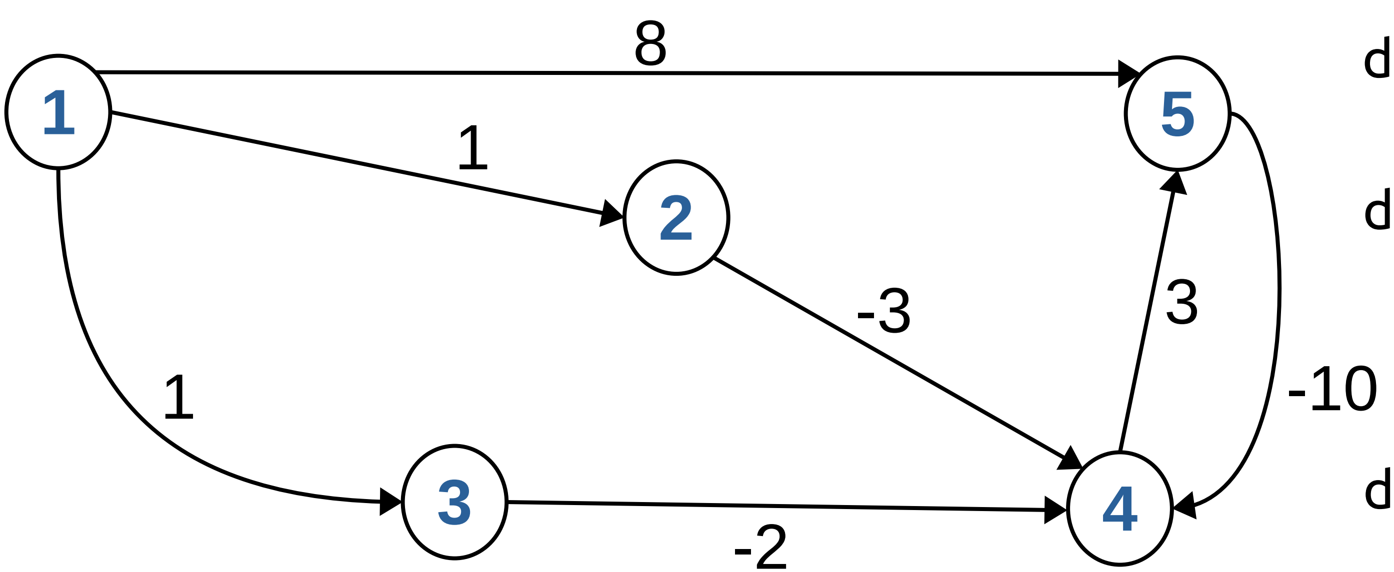
Bellman-Ford Algorithm

What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$

If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

What if we do one more iteration?



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-9	1
…					
$d^{(2)}$:	0	1	1		

E = { ((**1**,**2**) , 1), ((**1**,**3**), 1)), ((**1**,**5**), 8), ((**2**,**4**) , -3), ((**3**,**4**), -2), ((**4**,**5**), 3), ((**5**,**4**), -10) }

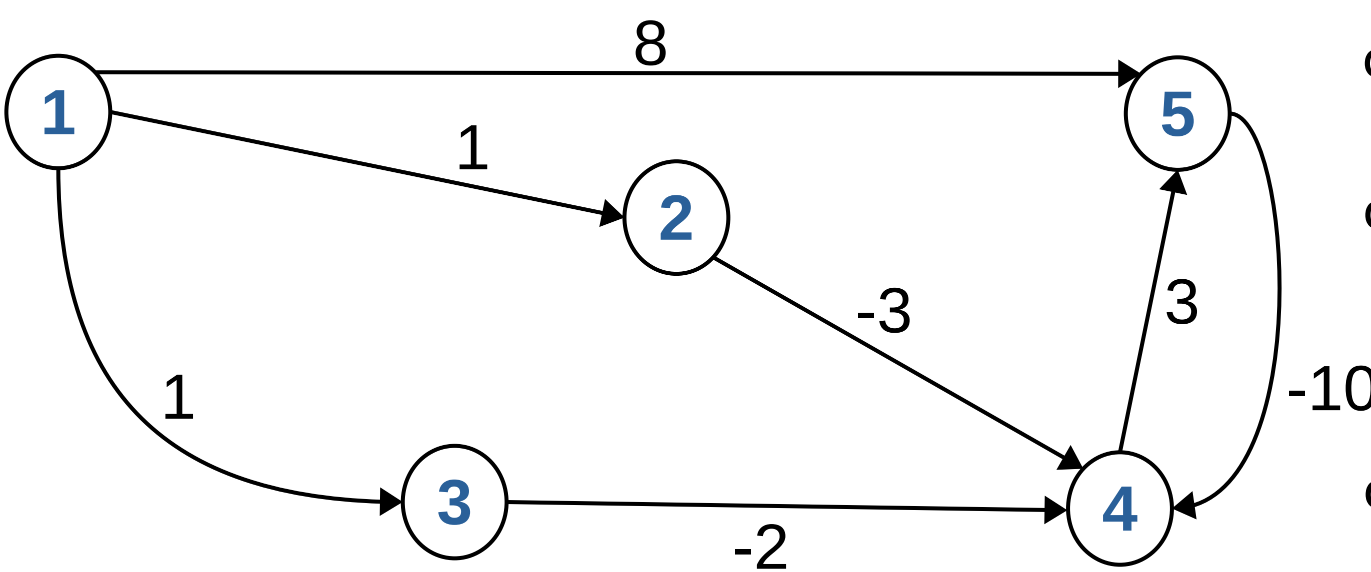
Bellman-Ford Algorithm

What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$

If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

What if we do one more iteration?



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-9	1

…

	1	2	3	4	5
$d^{(2)}$:	0	1	1		-6

E = { ((1,2) , 1), ((1,3), 1)), ((1,5), 8), ((2,4) , -3), ((3,4), -2), ((4,5), 3), ((5,4), -10) }

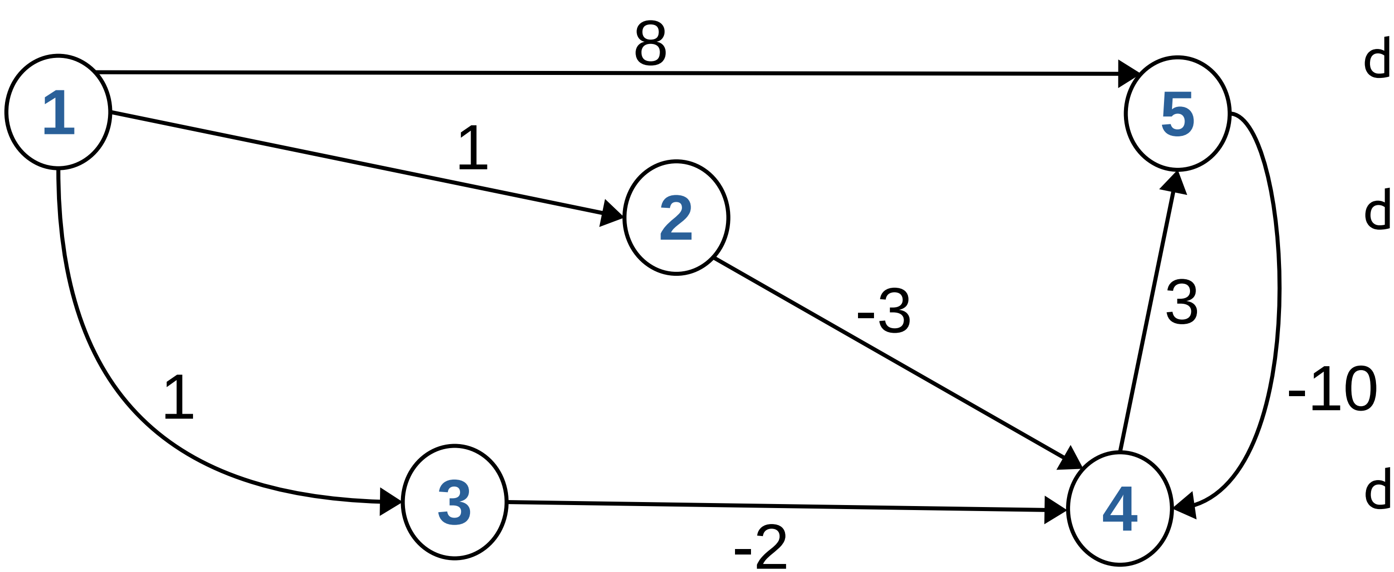
Bellman-Ford Algorithm

What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
  For each edge (u,v) in E:
    $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$

If the the graph is *topologically sorted*
Then Bellman-Ford finds the minimum
Path costs in 1 iteration.

What if we do one more iteration?



	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-9	1
…					
$d^{(2)}$:	0	1	1	-16	-6

E = { ((**1**,**2**) , 1), ((**1**,**3**), 1)), ((**1**,**5**), 8), ((**2**,**4**) , -3), ((**3**,**4**), -2), ((**4**,**5**), 3), ((**5**,**4**), -10) }

Bellman-Ford Algorithm

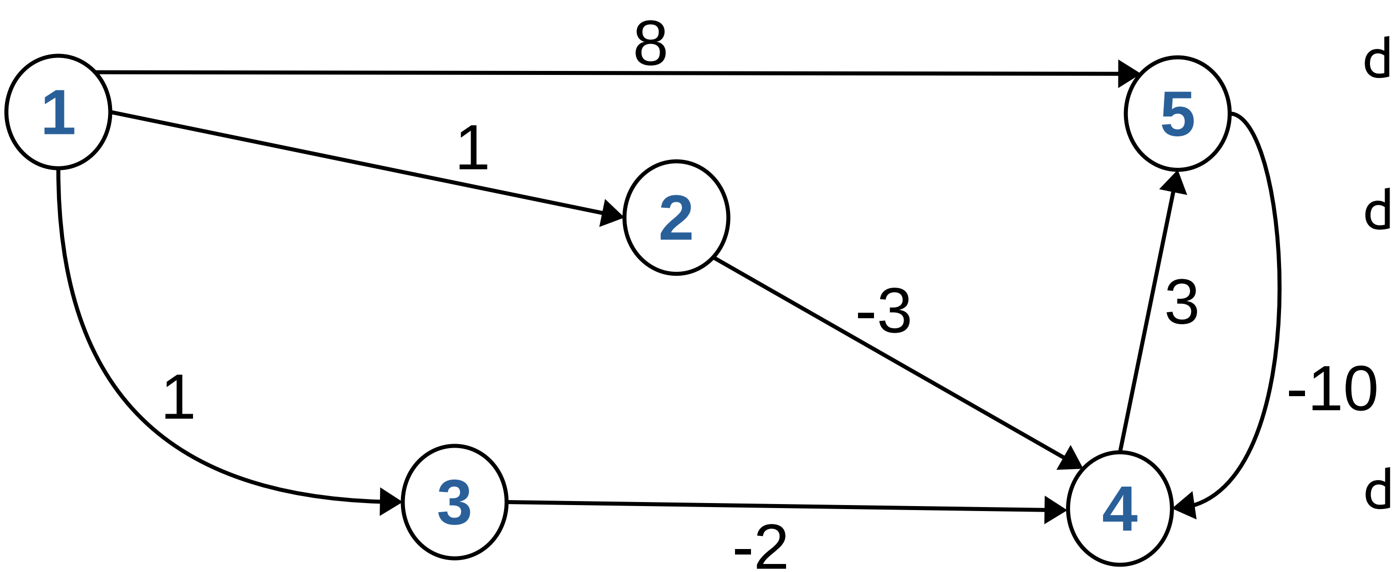
What if the graph has a negative cycle?
How to detect the negative cycle with Bellman-Ford?

/ Find the minimum path cost from s to every other node in V */*
Bellman-Ford(G = (V,E),**s**):
$d^{(0)}[v] = \infty$ forall v in V
$d^{(0)}[\mathbf{s}] = 0$
For i = 1…n -1: // n nodes
    For each edge (u,v) in E:
        $d^{(i)}[v] = \min(d^{(i-1)}[v], d^{(i-1)}[u] + \text{edgeWeight}(u,v))$

Calculate $d^{(n)}$. if $d^{(n)}$!= $d^{(n-1)}$ then neg. cycle detected.

If the graph has a negative cycle,
The shortest path problem become *meaningless*

Your min path cost will be simply $-\infty$

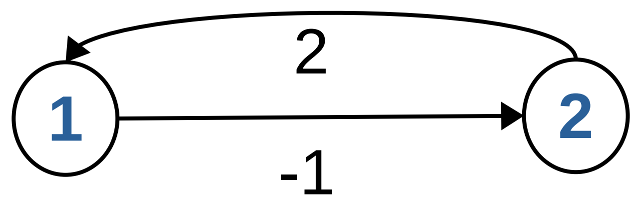


	1	2	3	4	5
$d^{(0)}$:	0	∞	∞	∞	∞
$d^{(1)}$:	0	1	1	-9	1
…					
$d^{(n)}$:	0	1	1	**-16**	**-6**

E = { ((**1,2**) , 1), ((**1,3**), 1)), ((**1,5**), 8), ((**2,4**) , -3), ((**3,4**), -2), ((**4,5**), 3), ((**5,4**), -10) }

Floyd-Warshall Algorithm and Negative Cycles

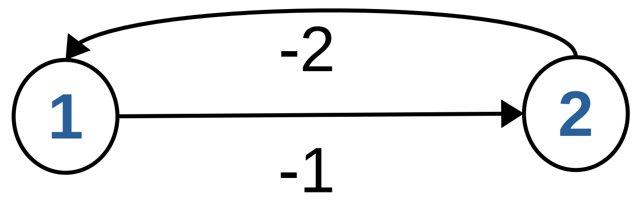
Also can handle edges with negative weights
and detects negative cycles



	1	2
1	0	-1
2	2	0

Floyd-Warshall Algorithm and Negative Cycles

Also can handle edges with negative weights
and detects negative cycles



	1	2
1	-3	
2		-3

How to detect a negative cycle with Floyd-Warshall?

If there is at least one negative value in the diagonal of the matrix
then it implies a negative cycle is in the graph.