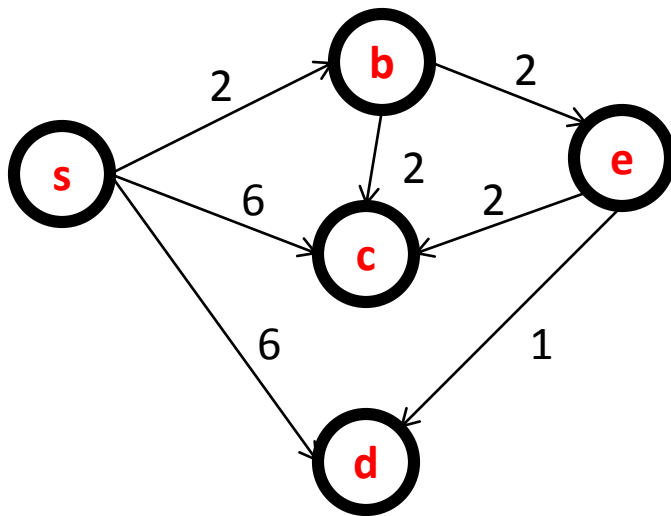


Dijkstra's Algorithm

SSSP, non-neg

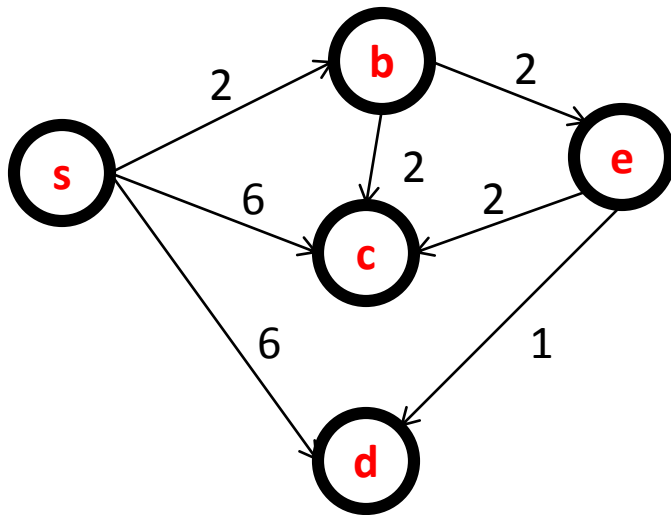


Edge weights = $w(x,y)$

Final distances = $d(x,y) = d_w(x,y)$

Dijkstra's Algorithm

SSSP, non-neg



	s	b	c	d	e
L(x)	0	∞	∞	∞	∞

x <- extractmin

// L(x) is the distance of s to x

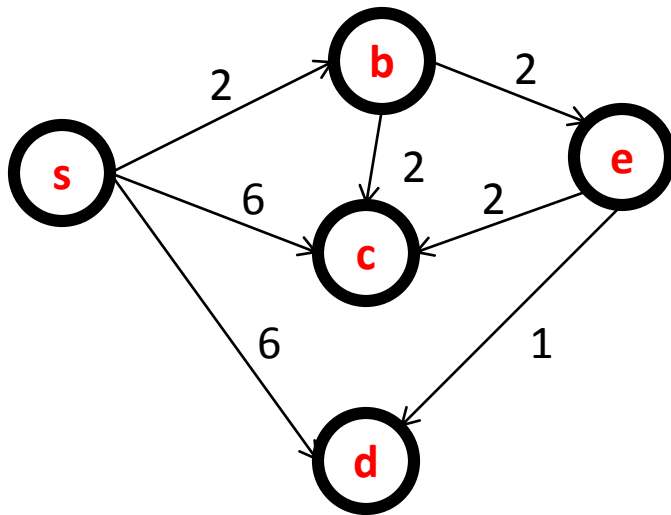
// mark x as final

"relax" all the edges out of x

$L(y) <- \min (L(y), L(x) + w(x,y))$

Dijkstra's Algorithm

SSSP, non-neg



	s	b	c	d	e
L(x)	0	2	6	6	∞

x <- extractmin

// L(x) is the distance of s to x

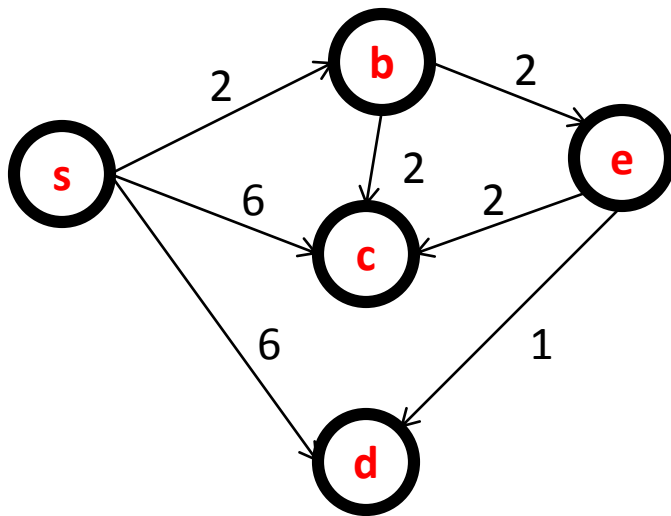
// mark x as final

"relax" all the edges out of x

$L(y) <- \min (L(y), L(x) + w(x,y))$

Dijkstra's Algorithm

SSSP, non-neg



	s	b	c	d	e
L(x)	0	2	4	6	4

x <- extractmin

// L(x) is the distance of s to x

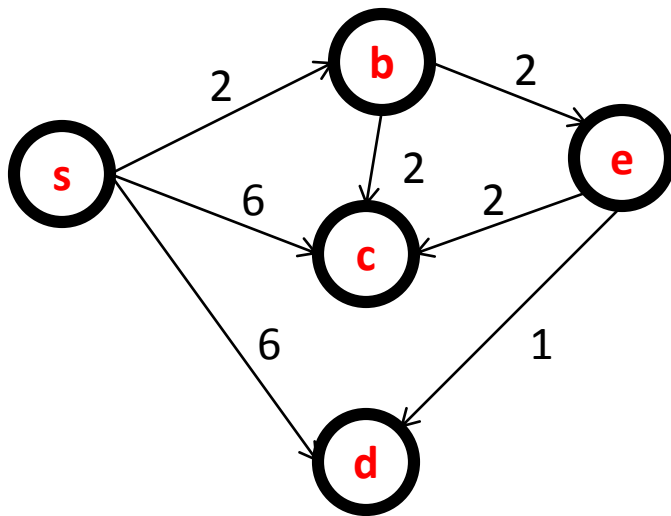
// mark x as final

"relax" all the edges out of x

$L(y) \leftarrow \min (L(y), L(x) + w(x,y))$

Dijkstra's Algorithm

SSSP, non-neg



↓

	s	b	c	d	e
L(x)	0	2	4	6	4

`x <- extractmin`

`// L(x) is the distance of s to x`

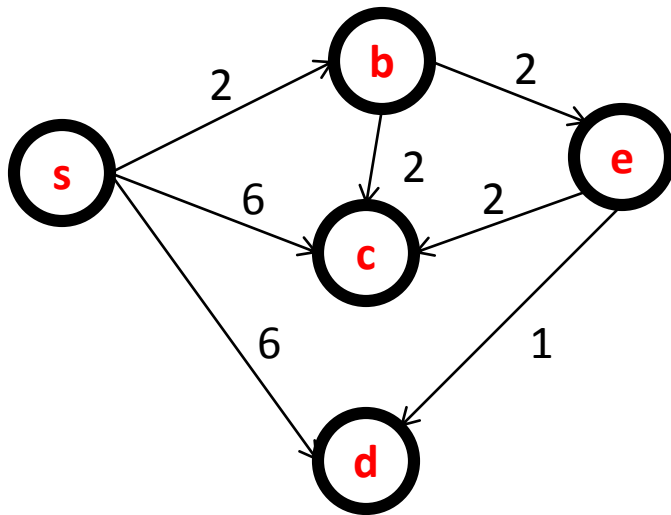
`// mark x as final`

"relax" all the edges out of x

`L(y) <- min (L(y), L(x) + w(x,y))`

Dijkstra's Algorithm

SSSP, non-neg



↓

	s	b	c	d	e
L(x)	0	2	4	5	4

```
x <- extractmin
```

```
// L(x) is the distance of s to x
```

```
// mark x as final
```

```
"relax" all the edges out of x
```

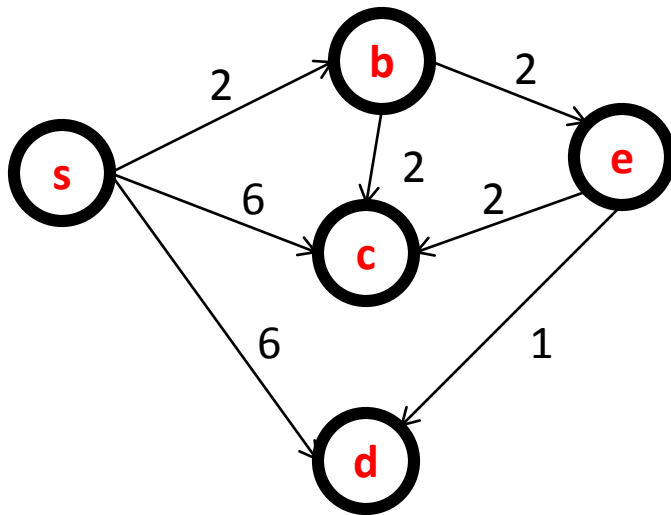
```
L(y) <- min ( L(y), L(x) + w(x,y) )
```

Dijkstra's Algorithm

SSSP, non-neg

m decreasekeys
n extractmins

Fib heap: $O(m + n \log n)$



	s	b	c	d	e
L(x)	0	2	4	5	4

x <- extractmin

// L(x) is the distance of s to x

// mark x as final

"relax" all the edges out of x

$L(y) <- \min (L(y), L(x) + w(x,y))$

Bellman-Ford-Moore Algorithm

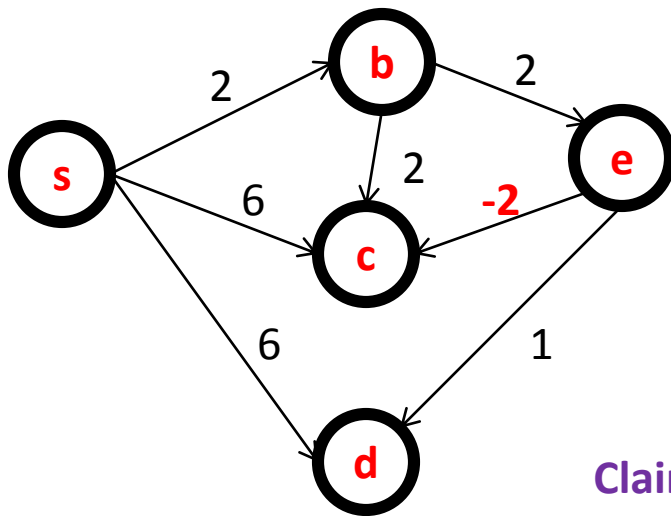
SSSP, neg wts OK

Bellman-Ford-Moore Algorithm

SSSP, neg wts OK

n rounds
 m time per round
 O(mn) time

	s	b	c	d	e
L(x)	0	∞	∞	∞	∞



// L(x) is an upper bound on d(s,x)

for t = 1 to n-1

for all vertices x

// "relax" all the edges out of x

$L(y) \leftarrow \min (L(y), L(x) + w(x,y))$

Claim: if graph has non negative cycles, B-F-M is OK.

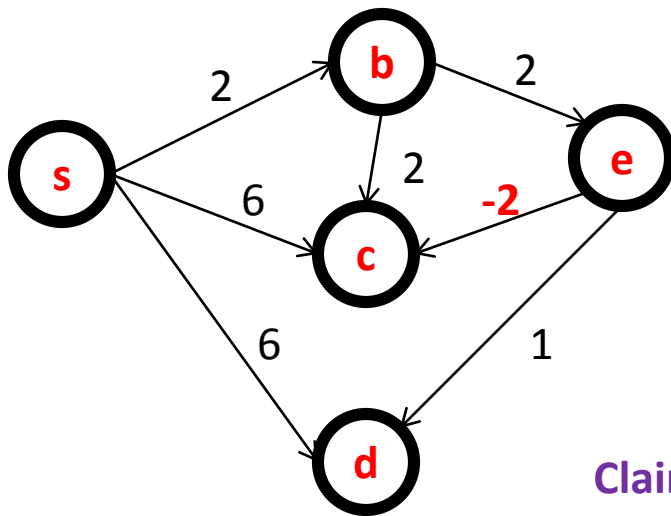
Proof: induction. At end of round t, L(y) is shortest path using at most t edges.

Bellman-Ford-Moore Algorithm

SSSP, neg wts OK

n rounds
 m time per round
 O(mn) time

	s	b	c	d	e
L(x)	0	2	2	5	4



// L(x) is an upper bound on d(s,x)

for t = 1 to n-1

for all vertices x

// "relax" all the edges out of x

$$L(y) \leftarrow \min (L(y), L(x) + w(x,y))$$

Claim: if graph has non negative cycles, B-F-M is OK.

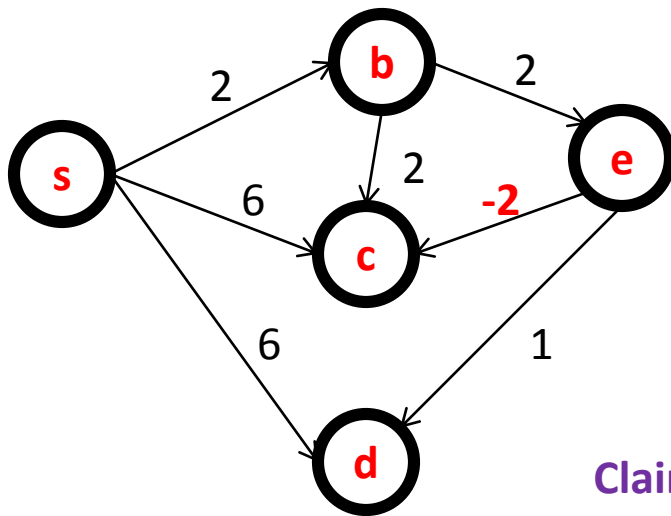
Proof: induction. At end of round t, L(y) is shortest path using at most t edges.

Bellman-Ford-Moore Algorithm

SSSP, neg wts OK

n rounds
 m time per round
 O(mn) time

	s	b	c	d	e
L(x)	0	2	2	5	4



// L(x) is an upper bound on w(s,x)

for t = 1 to n-1

for all vertices x

// "relax" all the edges out of x

$L(y) \leftarrow \min (L(y), L(x) + w(x,y))$

Claim: If at end, some edge xy is "over-tight"
 (it has $L(y) > L(x) + w(x,y)$)
 then graph has negative cycle.

Lots more work!

E.g., extensions and implementations of (just) Dijkstra's algorithm:

$O(m + n^2)$	Dijkstra'59
$O(m \log n)$	William'64
$O(m + n \log n)$	Fredman and Tarjan'87
$O(m\sqrt{\log n})$	Fredman and Willard'93
$O(m + n \frac{\log n}{\log \log n})$	Fredman and Willard'94
$O(m \log \log n)$	Thorup'96
$O(m + n\sqrt{\log n}^{1+\varepsilon})$	Thorup'96
$O(m + n \sqrt[3]{\log n}^{1+\varepsilon})$	Raman'97
$O(m + n \sqrt[3]{\log n}^{1+\varepsilon})$	Raman'97
$O(m\sqrt{\log \log n})$	Han and Thorup'02
$O(m + n \log \log n)$	Thorup'03
$O(m \log \log C)$	van Emde Boas'77
$O(m + n\sqrt{\log C})$	Ahuja et.al.'90
$O(m + n \sqrt[3]{\log C \log \log C})$	Cherkassky et.al.'97
$O(m + n \sqrt[4]{\log C}^{1+\varepsilon})$	Raman'97
$O(m + n \log \log C)$	Thorup'03

All Pairs SP

Non-neg weights:

n Dijkstras

$$O(mn + n^2 \log n)$$

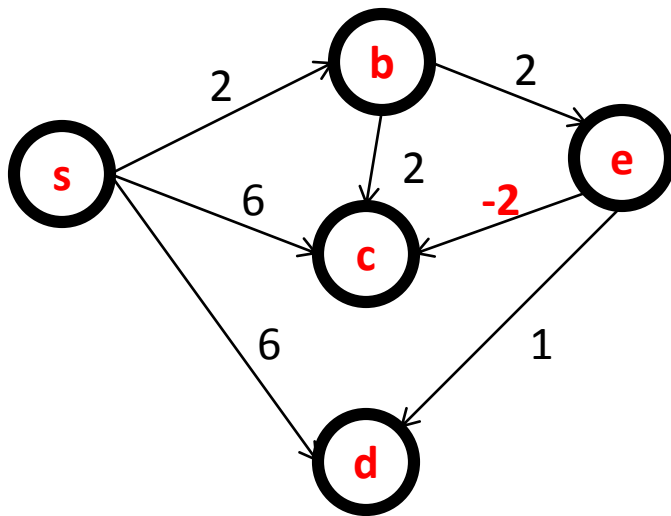
Neg weights:

n B-F-M

$$O(mn^2)$$

Johnson's Algorithm

APSP, neg wts OK



Find “feasible potentials” $\Phi(x)$ such that the “reduced weights”

$$\hat{w}(x,y) := \Phi(x) + w(x,y) - \Phi(y) \geq 0$$

Fact: reduced weights \hat{w} non-neg

Claim: $d_w(x,y) = d_{\hat{w}}(x,y) + \Phi(y) - \Phi(x)$

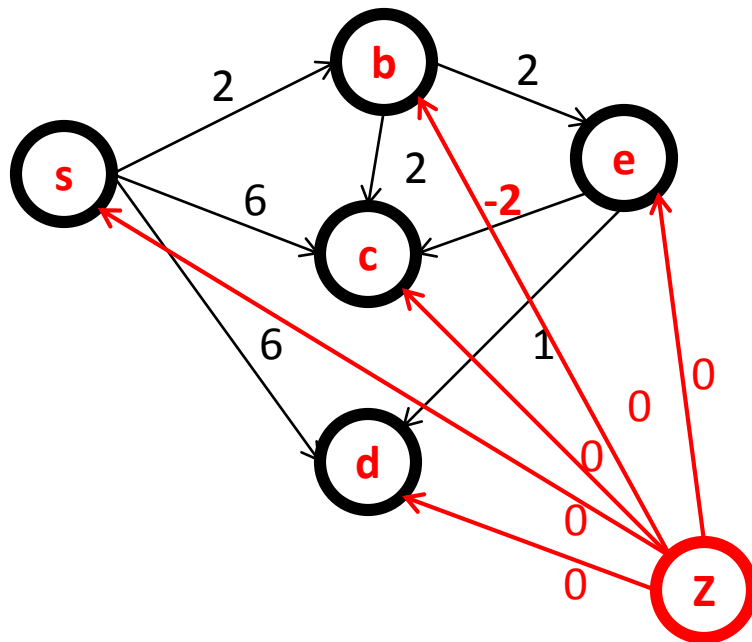
So shortest paths don't change, though their weights might.

\Rightarrow suffices to find APSP in this non-neg weights graph!

How to find these “feasible potentials”?

Johnson's Algorithm

APSP, neg wts OK



Find “feasible potentials” $\Phi(x)$ such that the “reduced weights”

$$\hat{w}(x,y) := \Phi(x) + w(x,y) - \Phi(y) \geq 0$$

Fact: reduced weights \hat{w} non-neg

Claim: $d_w(x,y) = d_{\hat{w}}(x,y) + \Phi(y) - \Phi(x)$

So shortest paths don't change, though their weights might.

\Rightarrow suffices to find APSP in this non-neg weights graph!

How to find these “feasible potentials”?

Shortest paths from Z.

No negative cycles created, so B-F-M works.

All Pairs SP

Non-neg weights:

n Dijkstras

$O(mn + n^2 \log n)$

Neg weights:

n B-F-M

$O(mn^2)$

Neg weights:

B-F-M + n Dijkstras

$O(mn + n^2 \log n)$

Neg weights:

Floyd-Warshall

$O(n^3)$

for all vertices z

for all vertices x,y

$d(x,y) \leftarrow \min\{d(x,y), d(x,z) + d(z,y)\}$

All Pairs SP

Non-neg weights:

n Dijkstras

$$O(mn + n^2 \log n)$$

Neg weights:

n B-F-M

$$O(mn^2)$$

Neg weights:

B-F-M + n Dijkstras

$$O(mn + n^2 \log n)$$

Neg weights:

Floyd-Warshall

$$O(n^3)$$

Neg weights:

Naïve Min-Sum-Product

$$O(n^3 \log n)$$