Shortest paths: dynamic programming & Bellman Ford

Shortest paths: dynamic programming, O(mn) time and O(mn) space
Computing the shortest paths to a target t

```
For v ∈ V
  M[0, v] = ∞
M[0, t] = 0

For i = 1 : |V|-1
  For v ∈ V
    M[i, v] = M [i – 1, v]
    For (v, w) ∈ E
      M [i, v] = min{M [i, v]
                     M [i–1, w] + c vw }
```

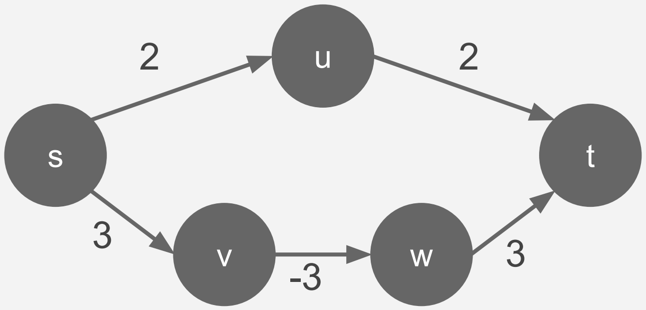
Initial distances from target

The longest simple path would include only |V|-1 edges

Value copied so M[i, v] will track the min for the iteration

After iterating through every edge our choice for M [i, v] is optimal

Shortest paths: dynamic programming



min{M[2, w], M[1, t] + c_{wt}}
min{3, 3}

min{M[1, s], M[0, u] + c_{su}}
min{∞, ∞ + 2}

min{M[2, v], M[1, w] + c_{vw}}
min{∞, 0}

min{M[3, s], M[2, v] + c_{sw}}
min{4, 3}

M	**s**	**u**	**v**	**w**	**t**
0	∞	∞	∞	∞	0
1	∞	2	∞	3	0
2	4	2	0	3	0
3	3	2	0	3	0
4	3	2	0	3	0

Shortest paths: **Bellman Ford**

Shortest paths: Bellman Ford, O(mn) time and O(n) space

Direction: shortest paths to target `t`

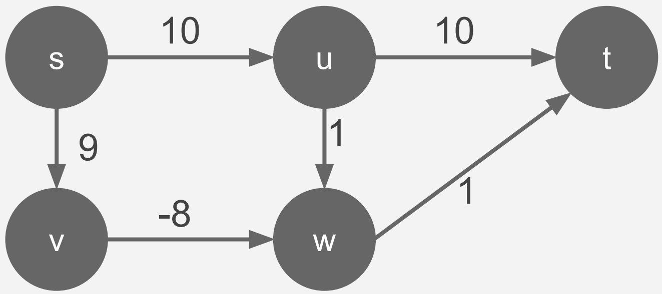
```
For v ∈ V
  dist[v] = ∞
  edge[v] = null
dist[t] = 0

For i = 1 : |V|-1
  For (v, w) ∈ E
    If dist[v] > dist[w] + c_vw
      dist[v] = dist[w] + c_vw
      edge[v] = w
```

Initial distances

dist[t] = 0: SP's to t
dist[s] = 0: SP's from s

The longest simple path would include only |V|-1 edges

Shortest paths: dynamic programming



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist[s] > dist[u] + c_{su}
∞ > ∞ + 10 = FALSE

dist[s] > dist[v] + c_{sv}
∞ > ∞ + 9 = FALSE

dist[u] > dist[w] + c_{uw}
∞ > ∞ + 1 = FALSE

dist[v] > dist[w] + c_{su}
∞ > ∞ + -8 = FALSE

dist[u] > dist[t] + c_{ut}
∞ > 0 + 10 = TRUE
dist[u] = 10, edge[u] = t

dist[w] > dist[t] + c_{wt}
10 > 0 + 1 = TRUE
dist[w] = 1, edge[w] = t

dist, edge	**s**	**u**	**v**	**w**	**t**
i=0	∞,-	∞,-	∞,-	∞,-	0,t
i=1	∞,-	10,t	∞,-	1,t	0,t
i=2					
i=3					
i=4					

Shortest paths: dynamic programming

dist[s] > dist[u] + c_{su}
∞ > 10 + 10 = TRUE
dist[s] = 20, edge[s] = u

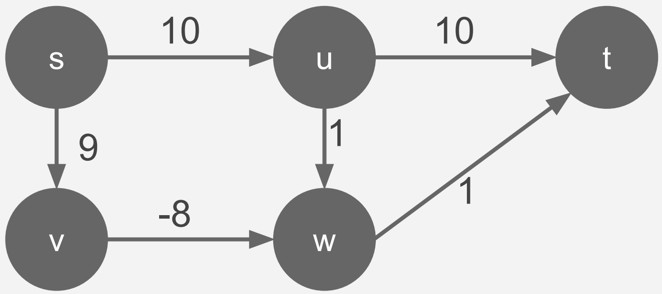
dist[s] > dist[v] + c_{sv}
20 > ∞ + 9 = FALSE

dist[u] > dist[w] + c_{uw}
10 > 1 + 1 = TRUE
dist[u] = 2, edge[u] = w

dist[v] > dist[w] + c_{su}
∞ > 1 + -8 = TRUE
dist[v] = -7, edge[v] = w

dist[u] > dist[t] + c_{ut}
2 > 0 + 10 = FALSE

dist[w] > dist[t] + c_{wt}
1 > 0 + 1 = FALSE



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist, edge	**s**	**u**	**v**	**w**	**t**
i=0	∞,-	∞,-	∞,-	∞,-	0,t
i=1	∞,-	10,t	∞,-	1,t	0,t
i=2	20,u	2,w	-7,w	1,t	0,t
i=3					
i=4					

Shortest paths: dynamic programming

```
dist[s] > dist[u] + c_su
  20    >   10    + 10     = FALSE
```

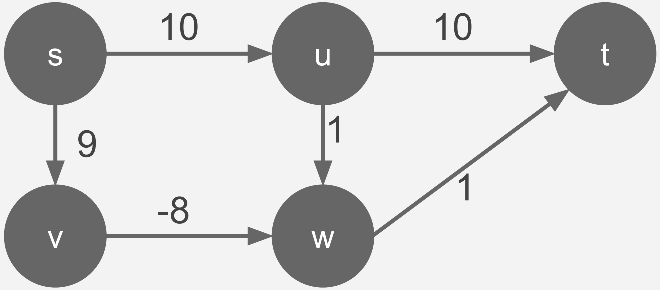
```
dist[s] > dist[v] + c_sv
  20    >   -7    + 9      = TRUE
```
dist[s] = 2, edge[s] = v

```
dist[u] > dist[w] + c_uw
  2     >    1    + 1      = FALSE
```

```
dist[v] > dist[w] + c_su
  -7    >    1    + -8     = FALSE
```

```
dist[u] > dist[t] + c_ut
  2     >    0    + 10     = FALSE
```

```
dist[w] > dist[t] + c_wt
  1     >    0    + 1      = FALSE
```



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist, edge	**s**	**u**	**v**	**w**	**t**
i=0	∞,-	∞,-	∞,-	∞,-	0,t
i=1	∞,-	10,t	∞,-	1,t	0,t
i=2	20,u	2,w	-7,w	1,t	0,t
i=3	2,v	2,w	-7,w	1,t	0,t
i=4					

Shortest paths: dynamic programming

```
dist[s] > dist[u] + c_su
   2    >    10   + 10     = FALSE
```

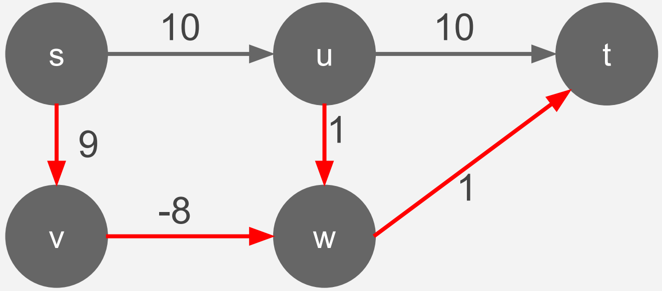
```
dist[s] > dist[v] + c_sv
   2    >   -7    + 9      = FALSE
```

```
dist[u] > dist[w] + c_uw
   2    >    1    + 1      = FALSE
```

```
dist[v] > dist[w] + c_su
  -7    >    1    + -8     = FALSE
```

```
dist[u] > dist[t] + c_ut
   2    >    0    + 10     = FALSE
```

```
dist[w] > dist[t] + c_wt
   1    >    0    + 1      = FALSE
```



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist, edge	**s**	**u**	**v**	**w**	**t**
i=0	∞,-	∞,-	∞,-	∞,-	0,t
i=1	∞,-	10,t	∞,-	1,t	0,t
i=2	20,u	2,w	-7,w	1,t	0,t
i=3	2,v	2,w	-7,w	1,t	0,t
i=4	2,v	2,w	-7,w	1,t	0,t

Shortest paths: Bellman Ford, O(mn) time and O(n) space

Direction: shortest paths from source s

```
For v ∈ V
  dist[v] = ∞
  edge[v] = null
dist[s] = 0

For i = 1 : |V|-1
  For (v, w) ∈ E
    If dist[w] > dist[v] + c_vw
      dist[w] = dist[v] + c_vw
      edge[w] = v
```

Initial distances

dist[t] = 0: SP's to t
dist[s] = 0: SP's from s

The longest simple path would include only |V|-1 edges

Shortest paths: dynamic programming

```
dist[u] > dist[s] + c_su
   ∞    >    0    + 10    = TRUE
dist[u] = 10, edge[u] = s
```

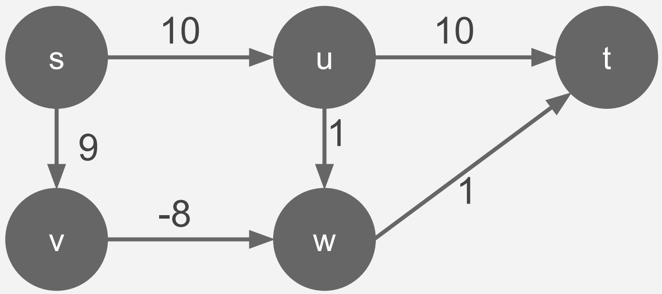
```
dist[v] > dist[s] + c_sv
   ∞    >    0    + 9     = TRUE
dist[v] = 9, edge[v] = s
```

```
dist[w] > dist[u] + c_uw
   ∞    >    10   + 1     = TRUE
dist[w] = 11, edge[w] = u
```

```
dist[w] > dist[v] + c_vw
   ∞    >    9    + -8    = TRUE
dist[w] = 1, edge[w] = v
```

```
dist[t] > dist[u] + c_ut
   ∞    >    10   + 10    = TRUE
dist[t] = 20, edge[t] = u
```

```
dist[t] > dist[w] + c_wt
   ∞    >    1    + 1     = TRUE
dist[t] = 2, edge[t] = w
```



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist, edge	s	u	v	w	t
i=0	0,s	∞,-	∞,-	∞,-	∞,-
i=1	0,s	10,s	9,s	1,v	2,w
i=2					
i=3					
i=4					

Shortest paths: dynamic programming

```
dist[u] > dist[s] + c_su
  10    >    0    + 10     = FALSE
```

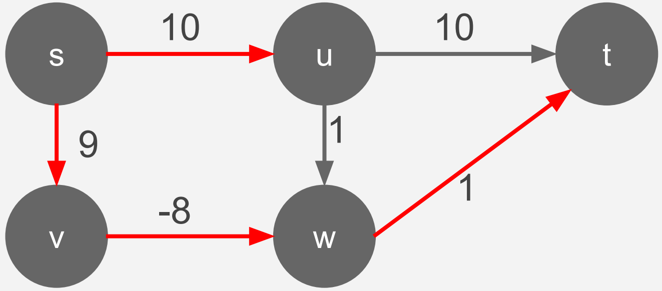
```
dist[v] > dist[s] + c_sv
  9     >    0    + 9      = FALSE
```

```
dist[w] > dist[u] + c_uw
  1     >   10    + 1      = FALSE
```

```
dist[w] > dist[v] + c_vw
  1     >    9    + -8     = FALSE
```

```
dist[t] > dist[u] + c_ut
  2     >   10    + 10     = FALSE
```

```
dist[t] > dist[w] + c_wt
  2     >    1    + 1      = FALSE
```



edges:
s->u=10
s->v=9
u->w=1
v->w=-8
u->t=10
w->t=1

dist, edge	s	u	v	w	t
i=0	0,s	∞,-	∞,-	∞,-	∞,-
i=1	0,s	10,s	9,s	1,v	2,w
i=2	0,s	10,s	9,s	1,v	2,w
i=3					
i=4					