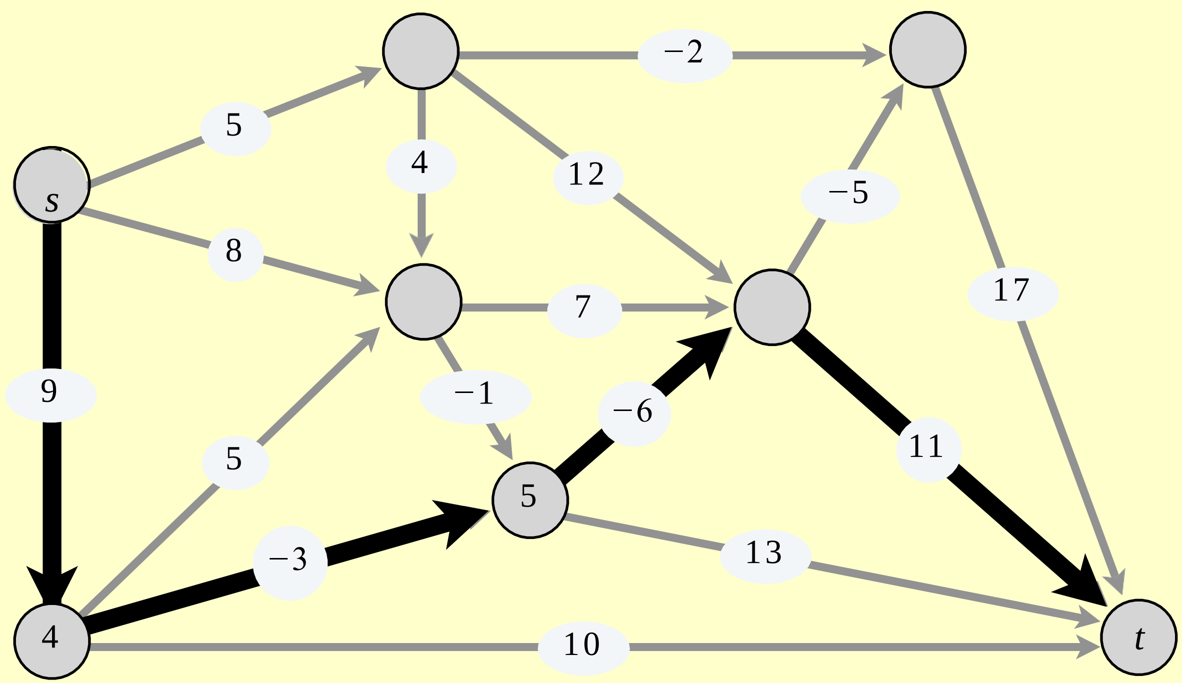
Shortest Paths in a Graph

Shortest paths with negative weights

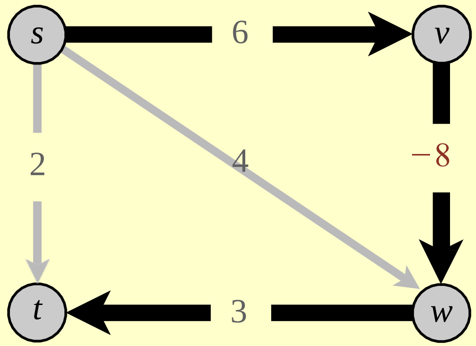
Shortest-path problem. Given a digraph $G = (V, E)$, with arbitrary edge lengths $c_{i,j}$, find shortest path from source node s to destination node t. (assume there exists a path from every node to t)



length of shortest $s \rightsquigarrow t$ path = 9 − 3 − 6 + 11 = 11

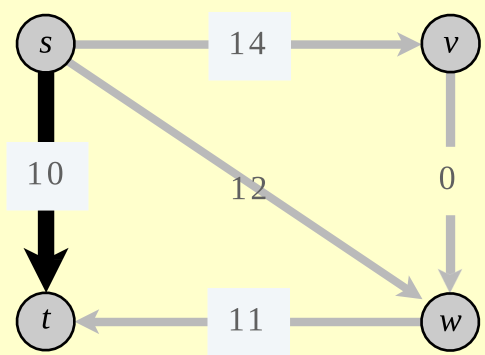
Shortest paths with negative weights: failed attempts

Dijkstra. May not produce shortest paths when edge lengths are negative.



Dijkstra selects the vertices in the order s, t, w, v
But shortest path from s to t is $s \to v \to w \to t$.

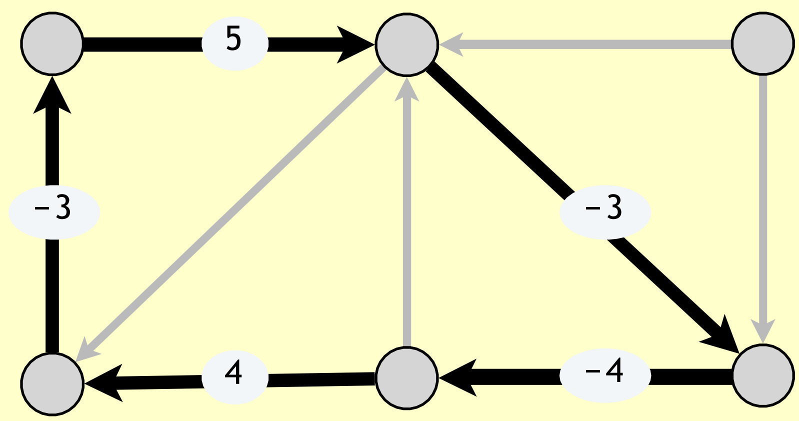
Reweighting. Adding a constant to every edge length does not necessarily make Dijkstra's algorithm produce shortest paths.



Adding 8 to each edge weight changes the shortest path from $s \to v \to w \to t$ to $s \to t$ which is not the shortest path in the actual graph.

Negative cycles

Def. A negative cycle is a directed cycle for which the sum of its edge lengths is negative.

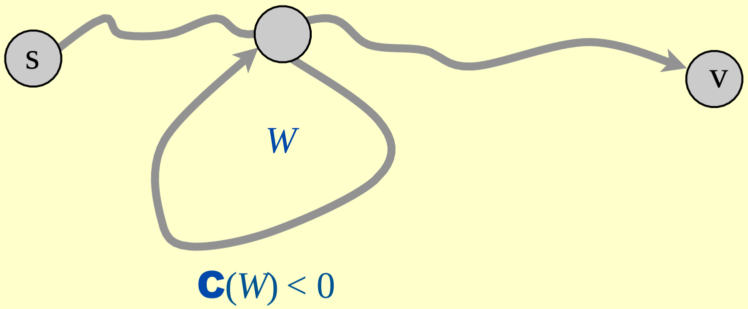


In the given graph, the cycle marked in bold is a negative cycle and the sum of its edge lengths is -1.

Shortest paths and negative cycles

Lemma 1. If some path $s \rightsquigarrow v$ contains a negative cycle, then there does not exist a shortest path $s \rightsquigarrow v$.

Pf. If there exists such a cycle W, then can build a path $s \rightsquigarrow v$ of arbitrarily negative length by detouring around W as many times as desired. ▪

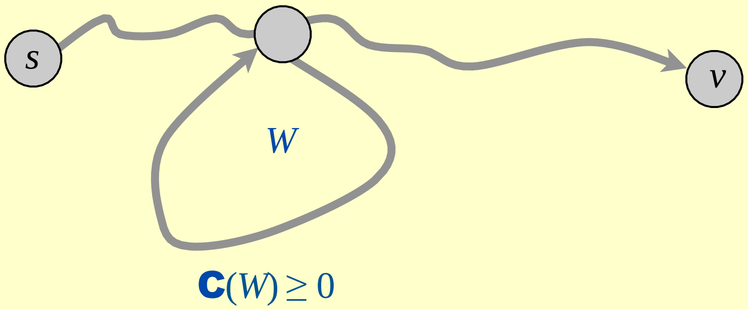


Shortest paths and negative cycles

Lemma 2. If G has no negative cycles, then there exists a shortest path $s \leadsto v$ that is simple (no repetition of nodes) and has $\leq n - 1$ edges.

Pf.

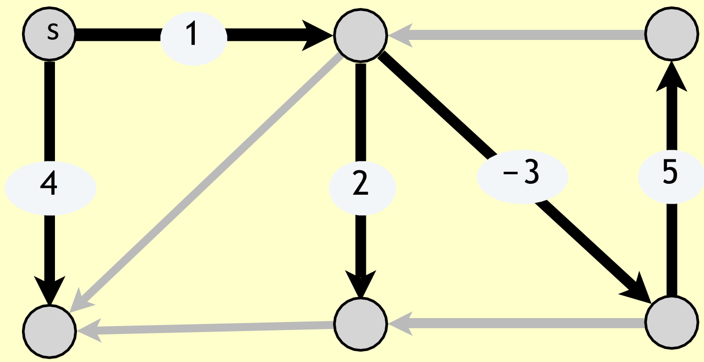
- Among all shortest paths $s \leadsto v$, consider one that uses the fewest edges.
- If that path P contains a directed cycle W, can remove the portion of P corresponding to W without increasing its length. ▪



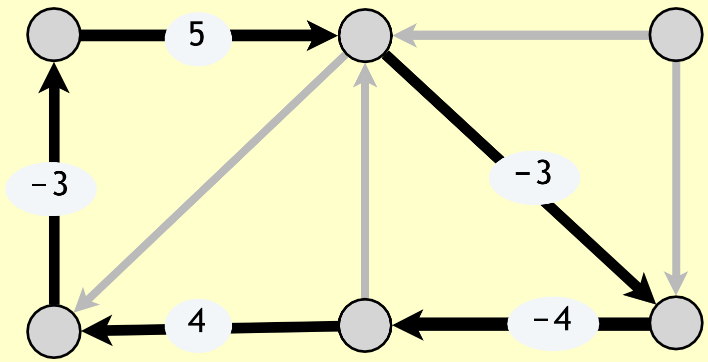
Shortest-paths and negative-cycle problems

Single-source shortest-paths problem. Given a weighed digraph $G = (V, E)$ with edge lengths $C_{i,j}$(but no negative cycles) and a source node s, find a shortest path $s \rightsquigarrow v$ for every node v.

Negative-cycle problem. Given a digraph $G = (V, E)$ with edge lengths $C_{i,j}$, find a negative cycle (if one exists).



shortest-paths tree



negative cycle

Shortest paths with negative weights: dynamic programming

Def. $OPT(i, v)$ = Length of shortest path $s \rightsquigarrow v$ (for any $v \in V$) that uses $\leq i$ edges.

Goal. $OPT(n-1, v)$ for each v.

by Lemma 2, if no negative cycles, there exists a shortest $s \rightsquigarrow v$ path that is simple

Case 1. Shortest path $s \rightsquigarrow v$ uses $\leq i-1$ edges.

- $OPT(i, v) = OPT(i-1, v)$.

optimal substructure property

Case 2. Shortest path $s \rightsquigarrow v$ uses exactly i edges.

- if (w, v) is the last edge in such shortest path $s \rightsquigarrow v$, incur a cost of C_{wv}.
- Then, select the best path $s \rightsquigarrow w$ using $\leq i-1$ edges.

Bellman equation.

$$OPT(i, v) = \min\left\{OPT(i-1, v), \min_{(w,v)\in E}\{OPT(i-1, w) + C_{wv}\}\right\} \qquad if\ i > 0$$

Shortest paths with negative weights: Bellman-Ford Algorithm

SHORTEST-PATHS(V, E, C, s)

$M[0, s] \leftarrow 0.$

FOREACH node $v \in V$:

$M[0, v] \leftarrow \infty.$

FOR i = 1 TO $n - 1$

FOREACH node $v \in V$:

$M[i, v] \leftarrow M[i-1, v].$

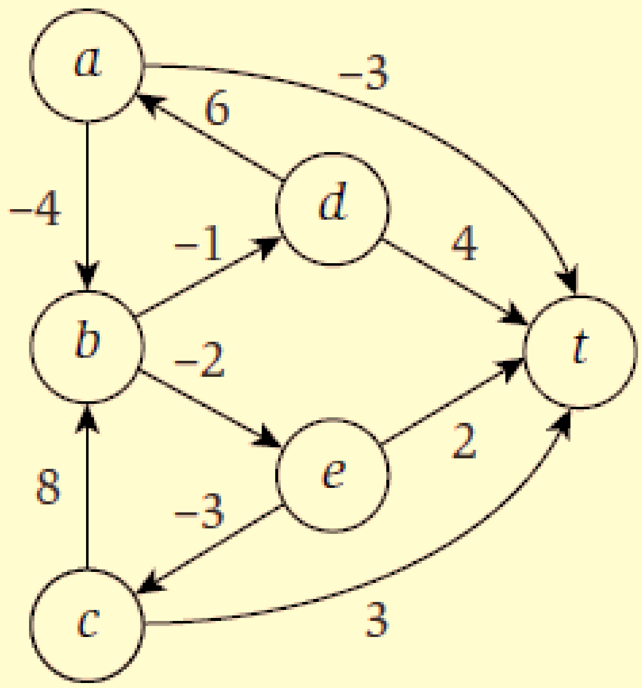
FOREACH edge $(w, v) \in$ E:

$M[i, v] \leftarrow \min \{ M[i, v], M[i-1, w] + C_{wv} \}.$

$O(|V|^3)$

$$OPT(i, v) = \begin{cases} 0 & i = 0, v = s \\ \infty & i = 0, v \neq s \\ \min\left\{OPT(i-1, v), \min_{(w,v)\in E}\{OPT(i-1, w) + C_{wv}\}\right\} & if\ i > 0 \end{cases}$$

Example



	a	*b*	*c*	*d*	*e*	*t*
0	0	∞	∞	∞	∞	∞
1	0	-4	∞	∞	∞	-3
2	0	-4	∞	-5	-6	-3
3	0	-4	-9	-5	-6	-4
4	0	-4	-9	-5	-6	-6
5	0	-4	-9	-5	-6	-6

$$OPT(i,v) = \begin{cases} 0 & i = 0, v = s \\ \infty & i = 0, v \neq s \\ \min\left\{OPT(i-1,v), \min\limits_{(w,v)\in E}\{OPT(i-1,w) + C_{wv}\}\right\} & if\ i > 0 \end{cases}$$

Bellman-Ford Algorithm using Relax( ) operation

```
BELLMAN-FORD(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
for i = 1 to |G.V| − 1
    for each edge (u, v) ∈ G.E
        RELAX(u, v, w)
for each edge (u, v) ∈ G.E
    if v.d > u.d + w(u, v)
        return FALSE
return TRUE
```

```
Alg.: INITIALIZE(G, s)
for each v ∈ V
    do d[v] := ∞
       π[v] := NIL
d[s] := 0
```

```
Relax(u, v, w)
    if d[v] > d[u] + w(u, v) then
        d[v] := d[u] + w(u, v);
        π[v] := u
    fi
```

Example