Bellman-Ford Algorithm

Mashiwat Tabassum Waishy

Lecturer

Department of CSE

STAMFORD UNIVERSITY BANGLADESH

History

- The algorithm was first proposed by Alfonso Shimbel in 1955
- Named after Richard Bellman and Lester Ford Jr who published it in 1956 and 1958
- Edward F. Moore also published the same algorithm in 1957, and for this reason it is also sometimes called the Bellman–Ford–Moore algorithm

Bellman-Ford

- The Bellman-Ford algorithm operates on an input graph, G, with | V | vertices and | E | edges
- Bellman-Ford algorithm is a single-source shortest path algorithm
- Bellman-Ford can also detect negative edge cycles which is a useful feature.
- Returns a Boolean value indicating whether or not there is a negative-weight cycle that is reachable from the source.
- If there is such cycle the algorithm indicates nor solution exists, If there is no such cycle, the algorithm produces the shortest paths and their weights.
- The main contribution of this algorithm is in its ordering of relaxations.

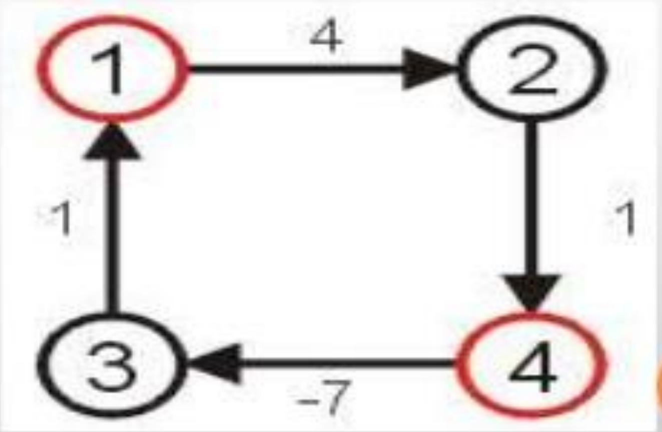
Bellman-Ford

Input: Graph and a source vertex src

Output: Shortest distance to all vertices from src. If there is a negative weight cycle, then shortest distances are not calculated, negative weight cycle is reported.

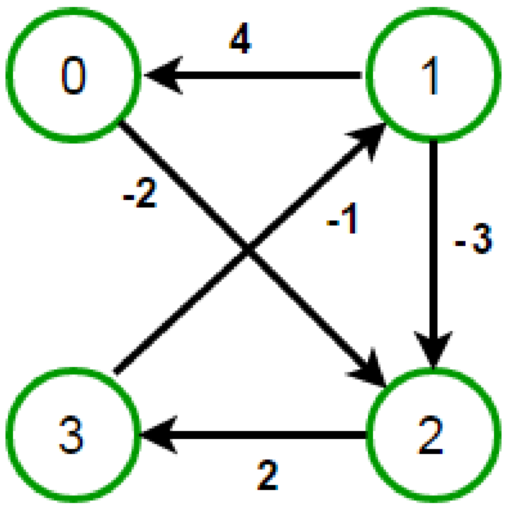
Negative Cycles:

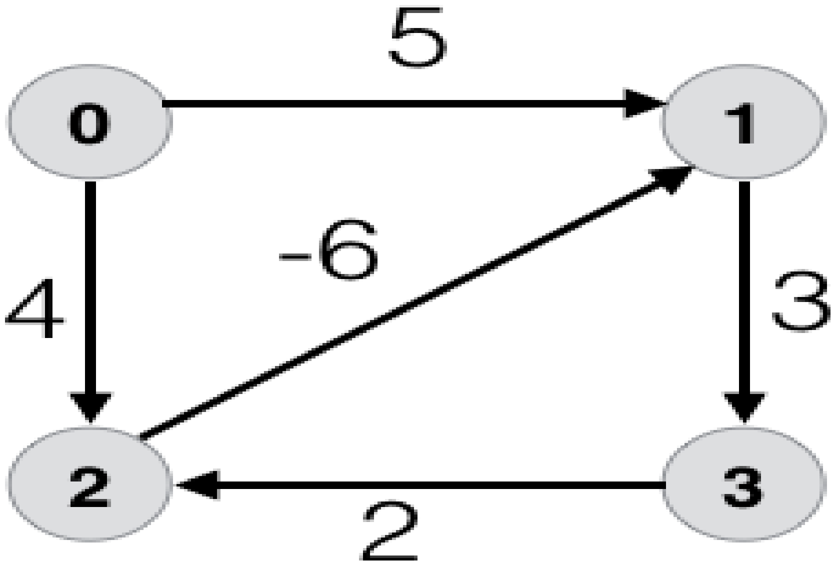
- Clearly, if we have negative vertices, it may be possible to end up in a cycle whereby each pass through the cycle decreases the total *length*
- Thus, a shortest length would be undefined for such a graph
- Consider the shortest path from vertex 1 to 4...
- We will only consider non-negative weights.



Negative Weight Cycle Example

Negative weight cycle: A **negative-weight cycle** is a **cycle** in a graph whose edges sum to a **negative** value.
For example, consider the following graph – It has one **negative-weight cycle**, 1—2—3—1 with sum -2 .



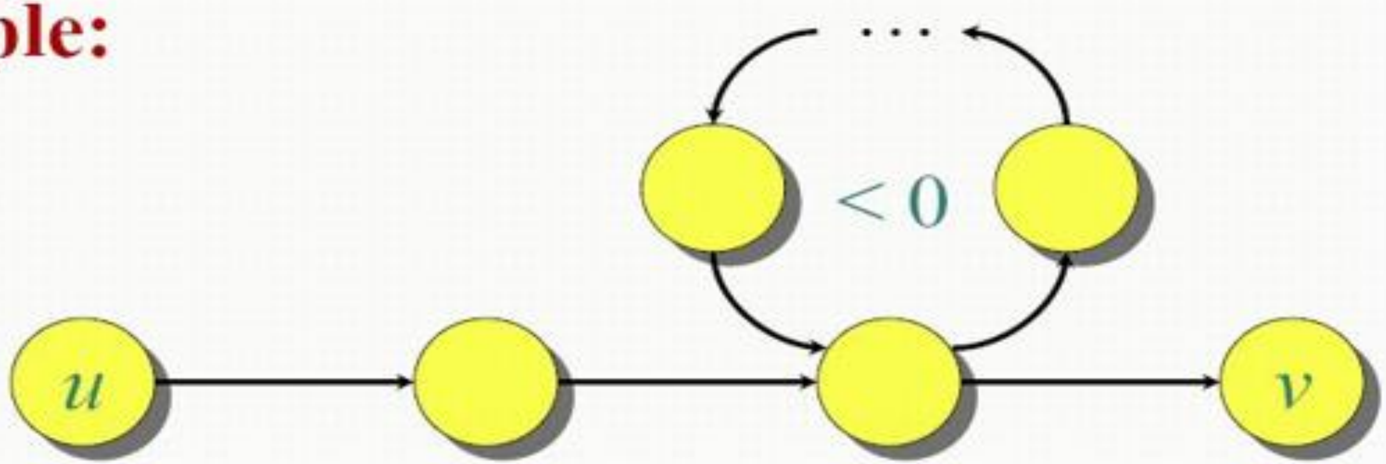


Soution= -∞

Negative-weight cycles

Recall: If a graph $G = (V, E)$ contains a negative-weight cycle, then some shortest paths may not exist.

Example:



Bellman-Ford algorithm: Finds all shortest-path lengths from a ***source*** $s \in V$ to all $v \in V$ or determines that a negative-weight cycle exists.

Bellman-Ford Steps

1) This step initializes distances from the source to all vertices as infinite and distance to the source itself as 0. Create an array dist[] of size |V| with all values as infinite except dist[src] where src is source vertex.

2) This step calculates shortest distances. Do following |V|-1 times where |V| is the number of vertices in given graph.
…..**a)** Do following for each edge u-v
………………If dist[v] > dist[u] + weight of edge uv, then update dist[v]
………………….dist[v] = dist[u] + weight of edge uv

3) This step reports if there is a negative weight cycle in graph. Do following for each edge u-v
……If dist[v] > dist[u] + weight of edge uv, then “Graph contains negative weight cycle”
The idea of step 3 is, step 2 guarantees the shortest distances if the graph doesn’t contain a negative weight cycle. If we iterate through all edges one more time and get a shorter path for any vertex, then there is a negative weight cycle

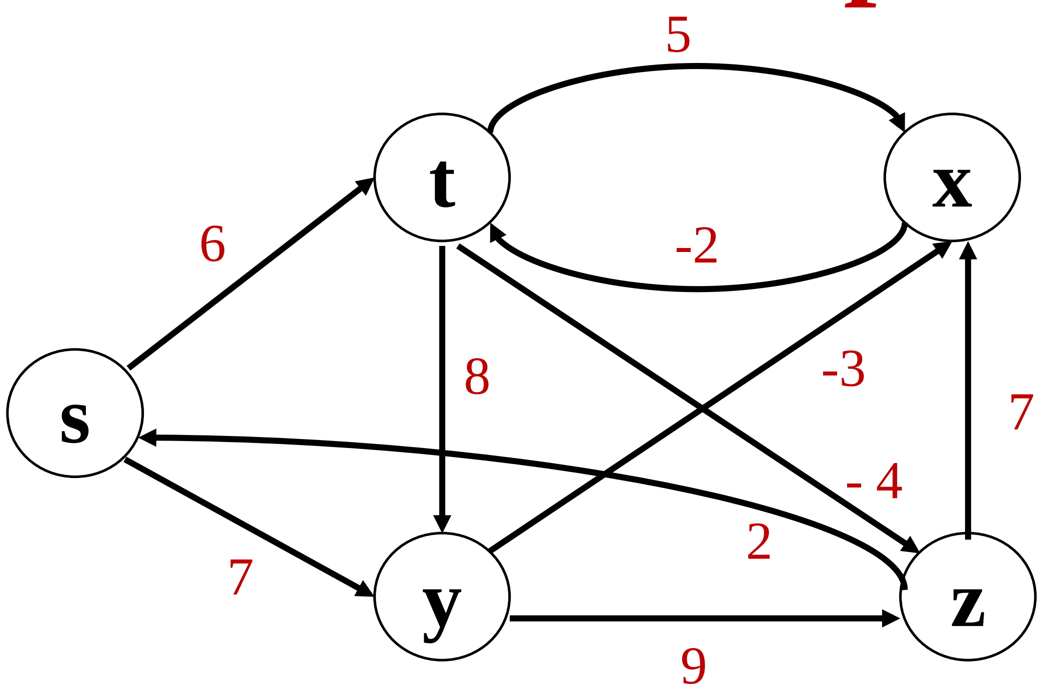
Bellman-Ford Algorithm - Pseudocode

```
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
```

```
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex v ∈ G.V
    v.d = ∞
    v.π = NIL
s.d = 0
```

```
RELAX(u, v, w)
if v.d > u.d + w(u, v)
    v.d = u.d + w(u, v)
    v.π = u
```

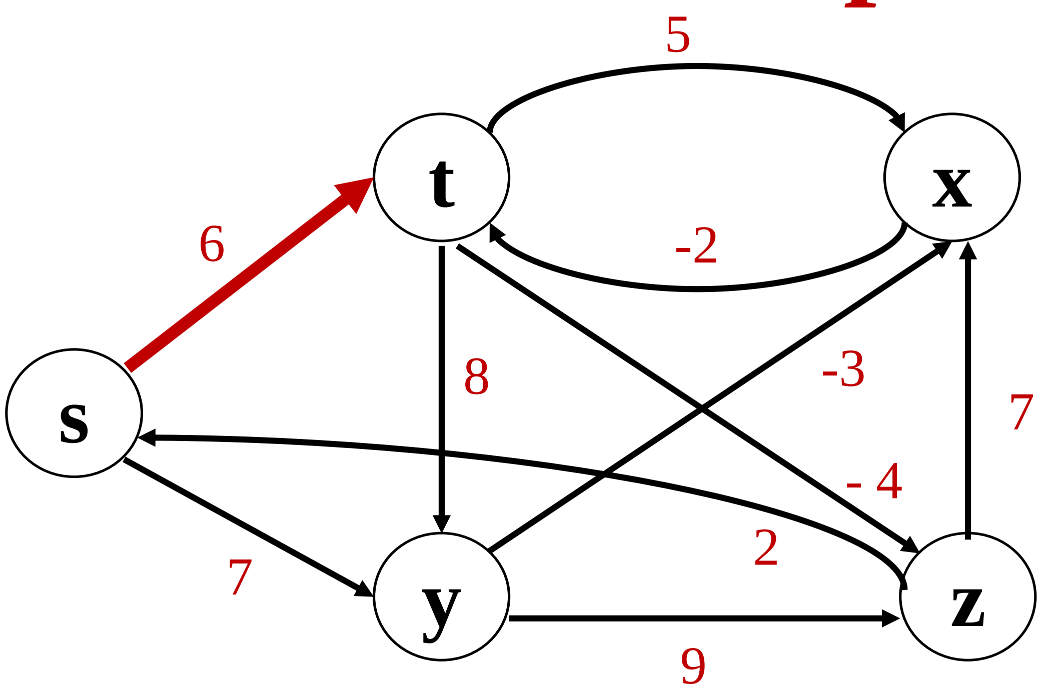
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

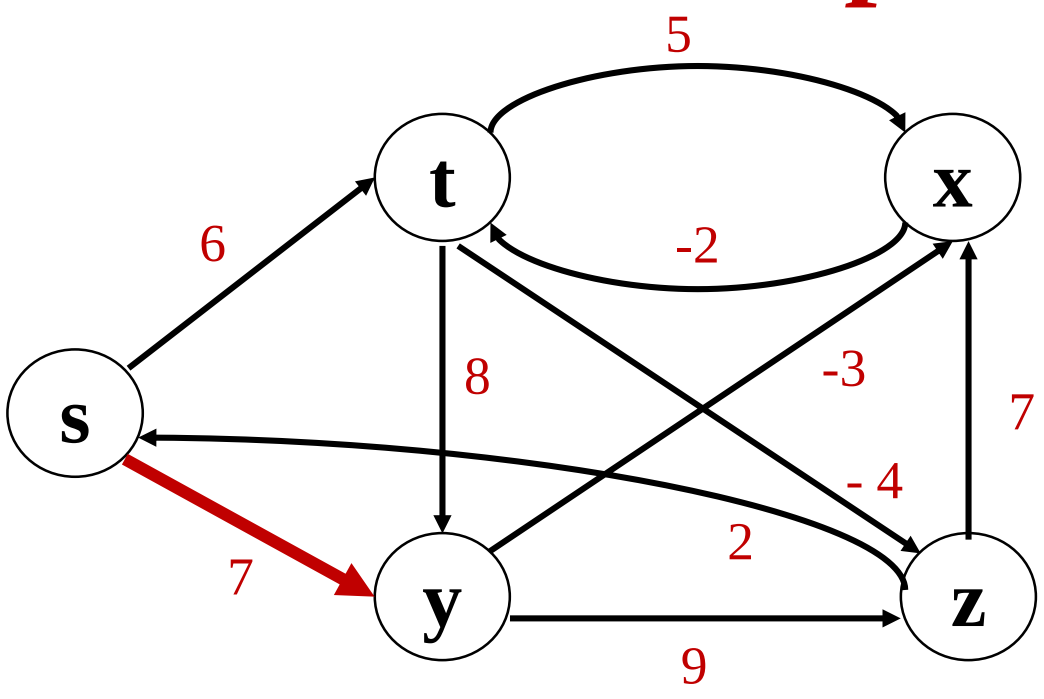
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	∞/NIL	∞/NIL	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

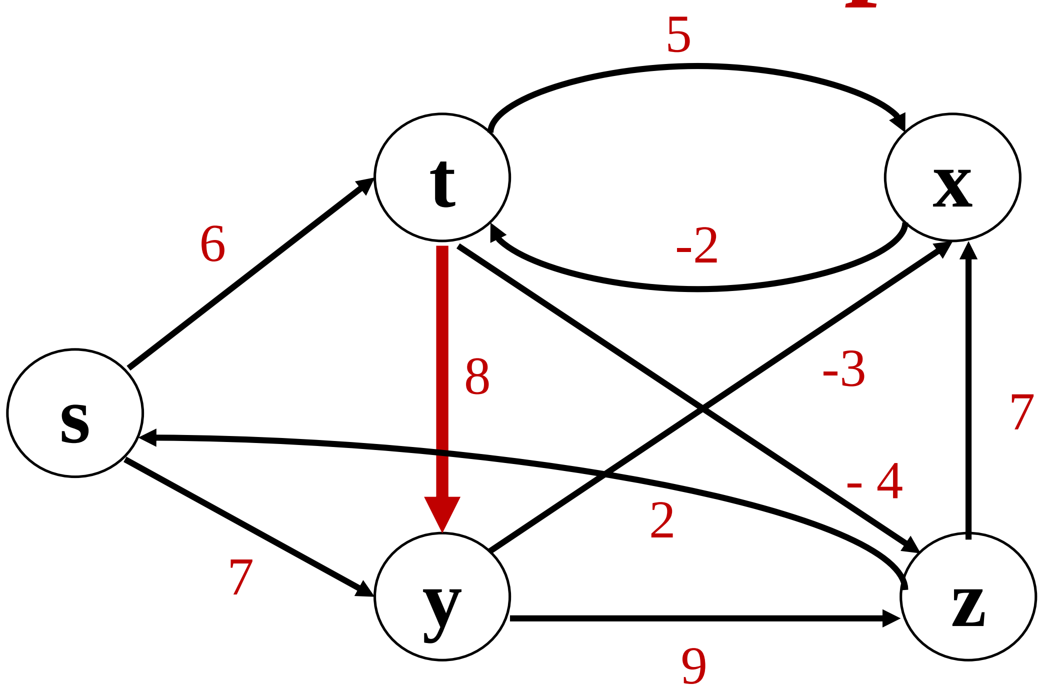
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	∞/NIL	7/s	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

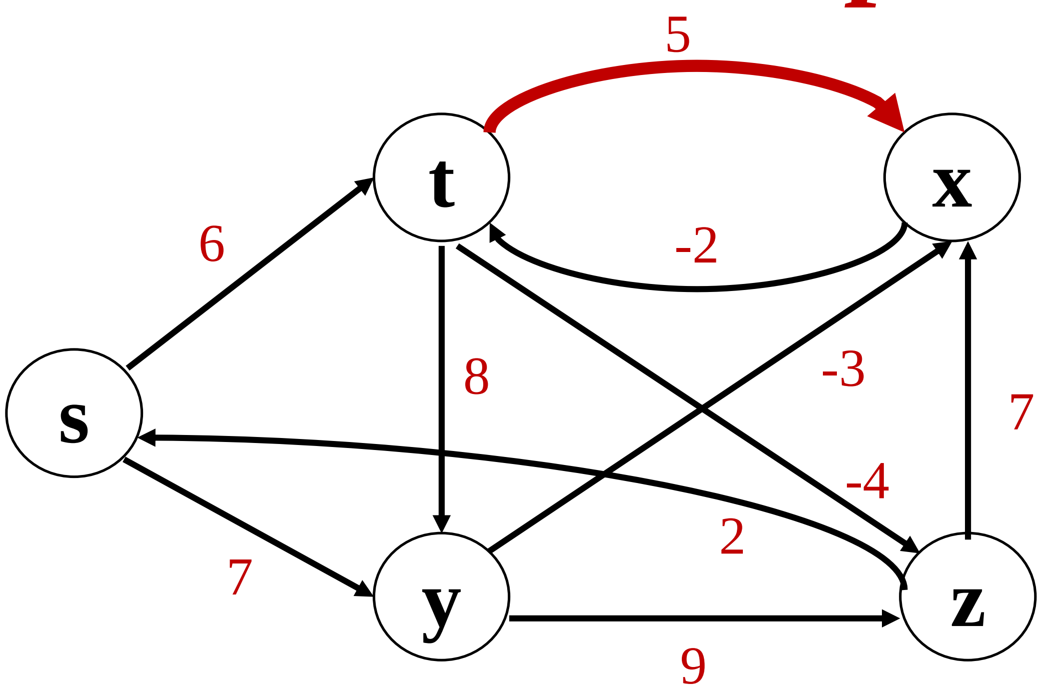
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	∞/NIL	7/s	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

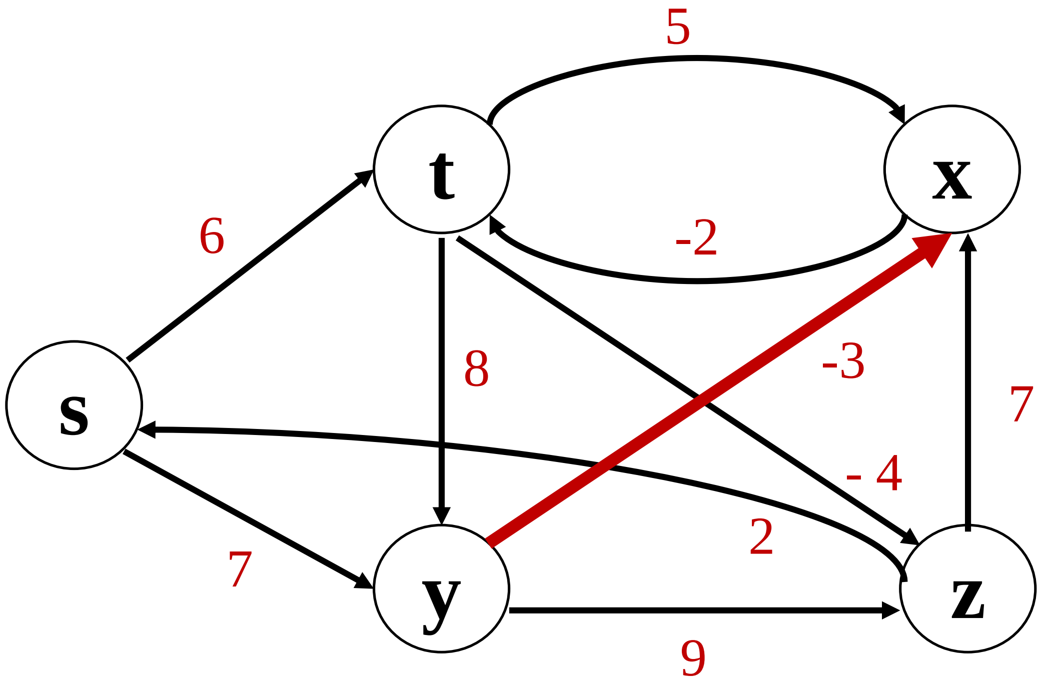
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	11/t	7/s	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

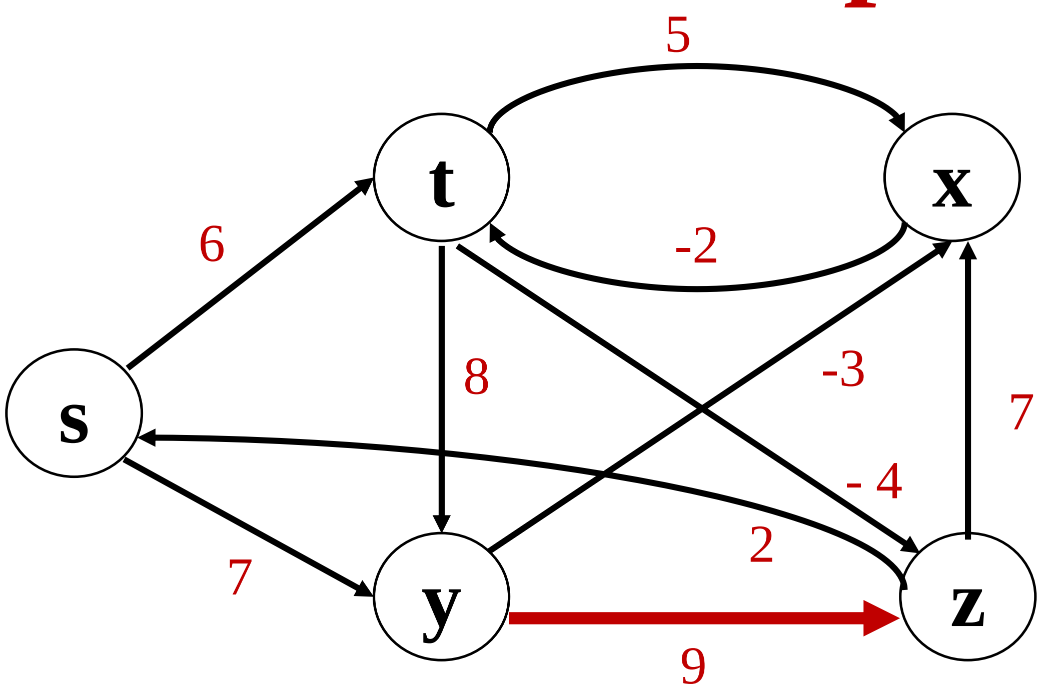
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	4/y	7/s	∞/NIL

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

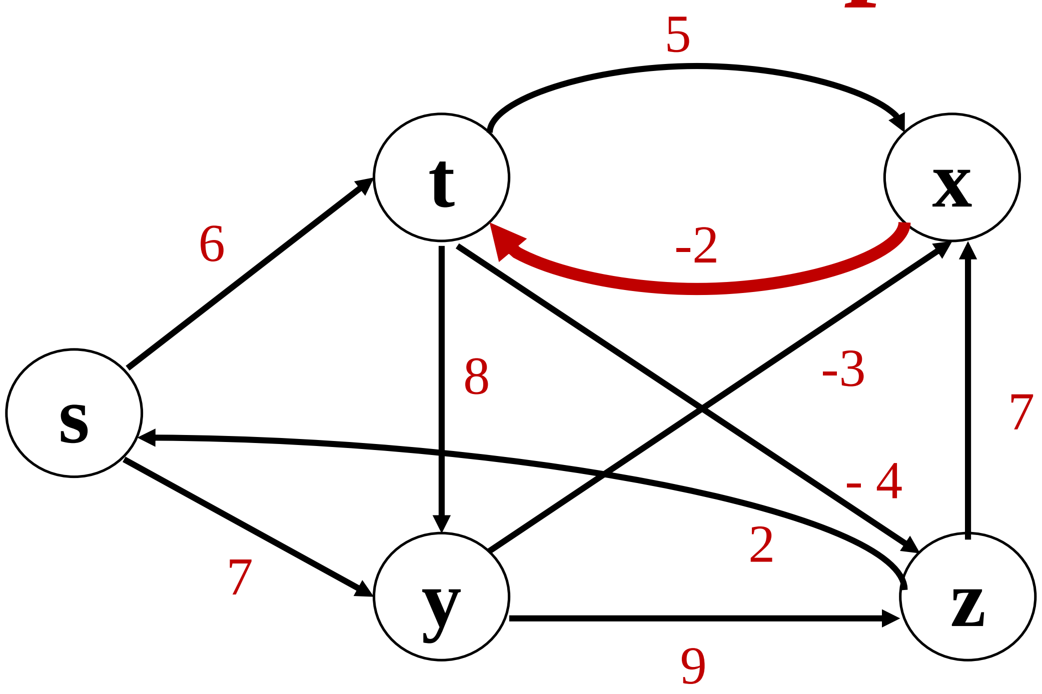
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	6/s	4/y	7/s	16/y

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

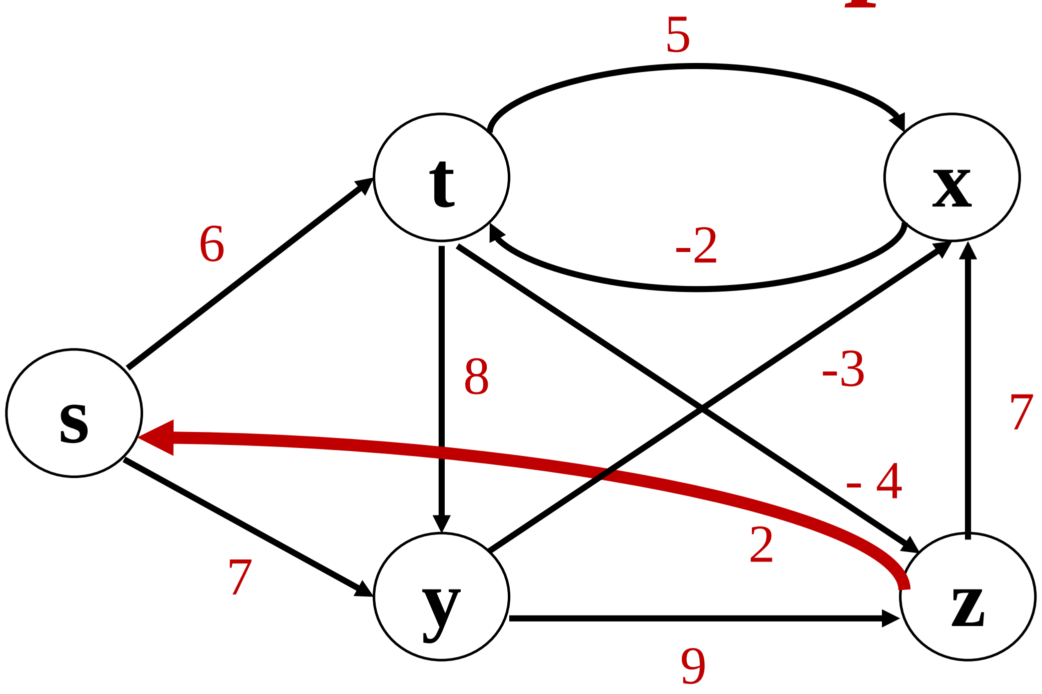
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	16/y

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

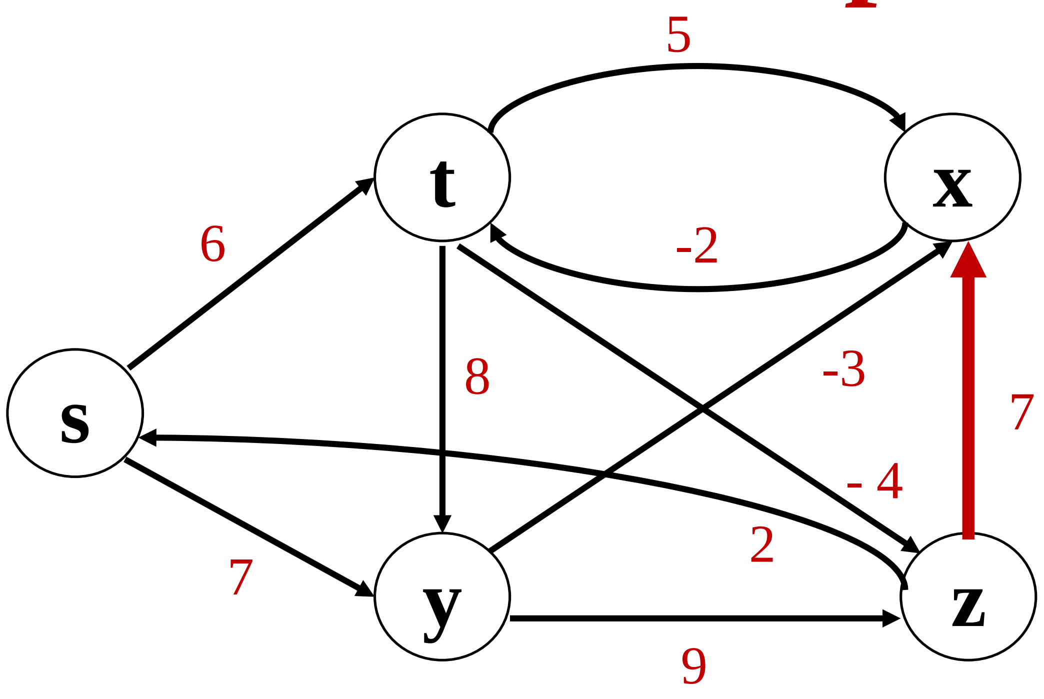
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	16/y

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

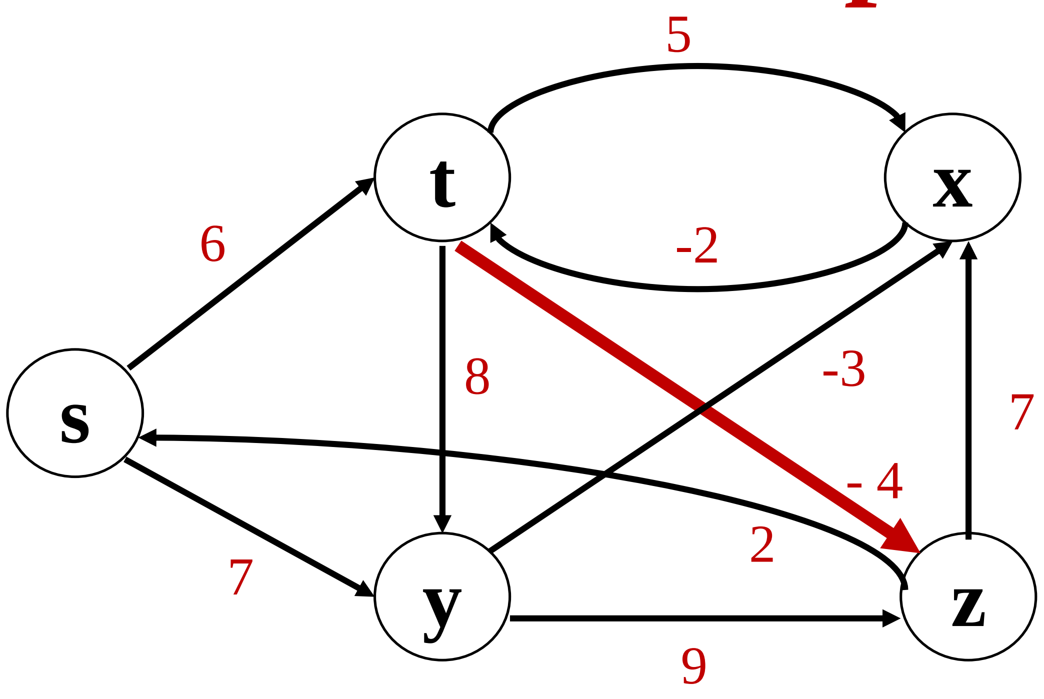
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	16/y

<u>Edge List:</u> (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

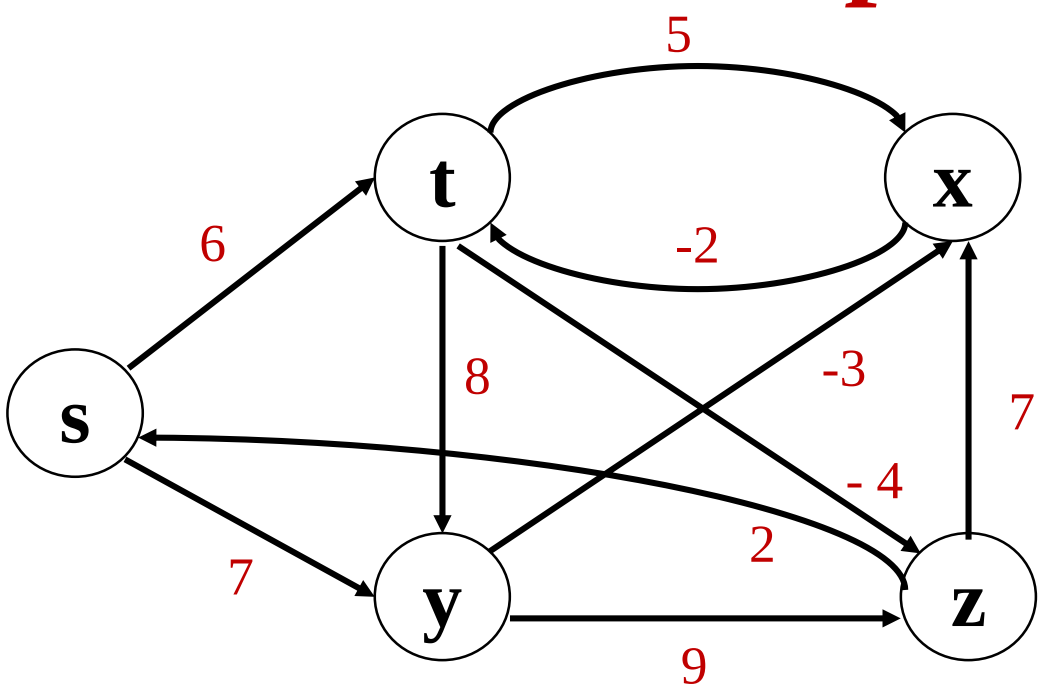
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

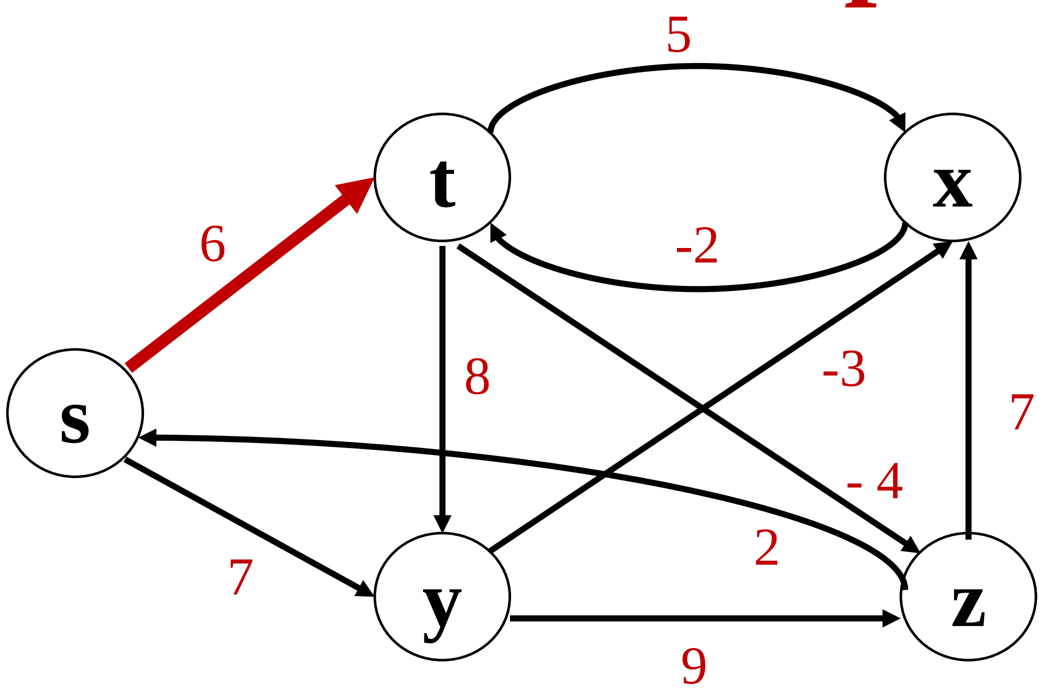
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

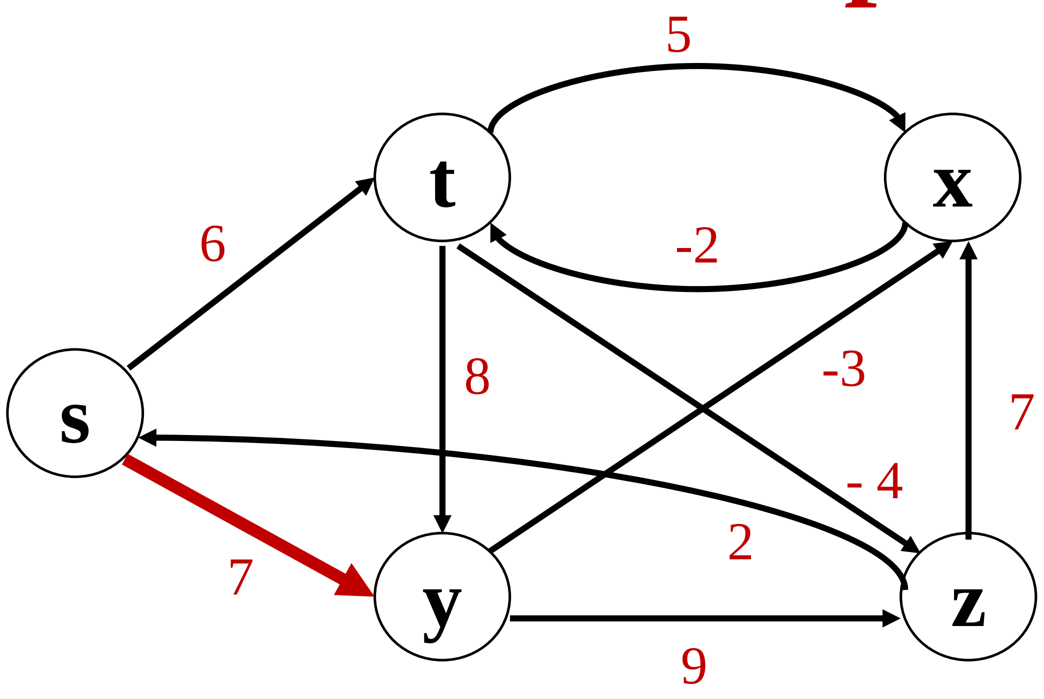
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

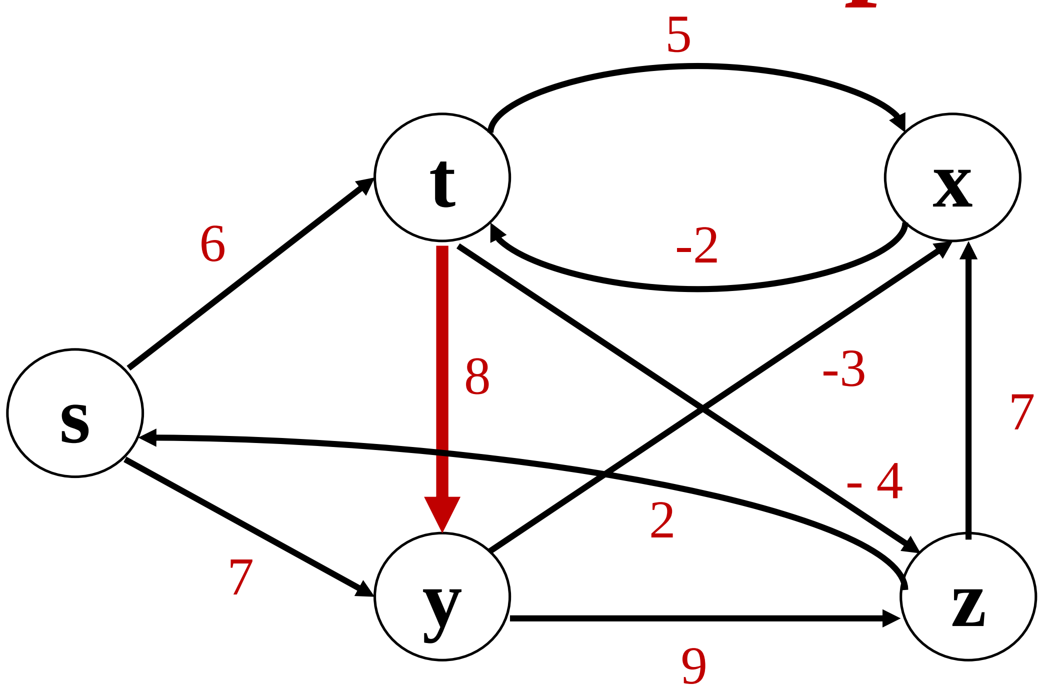
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

<u>Edge List:</u> (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

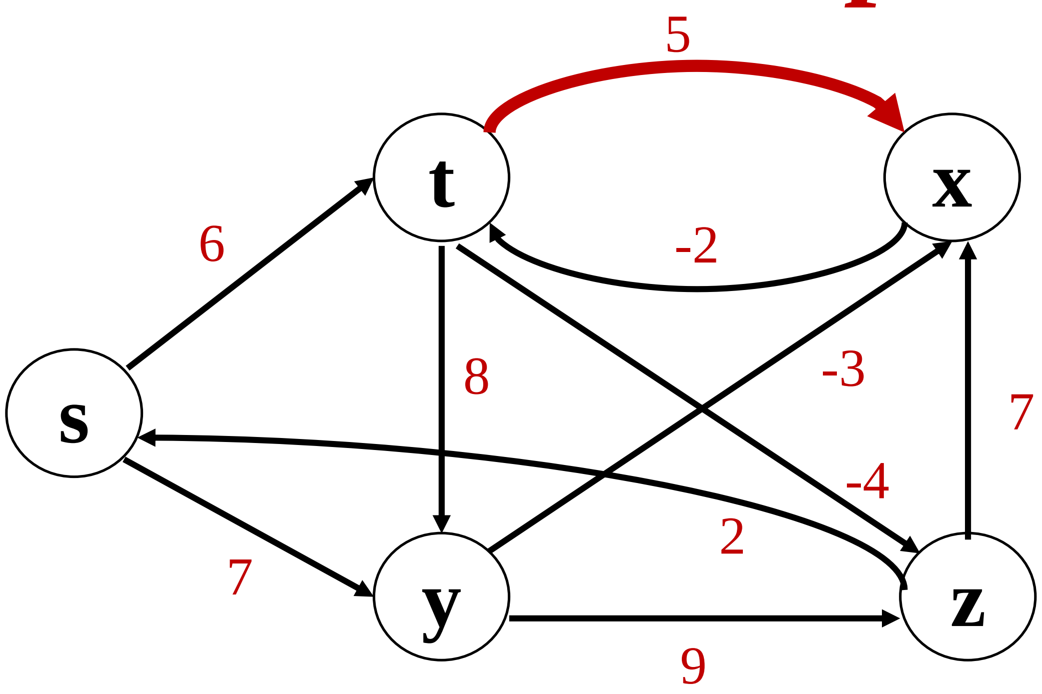
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

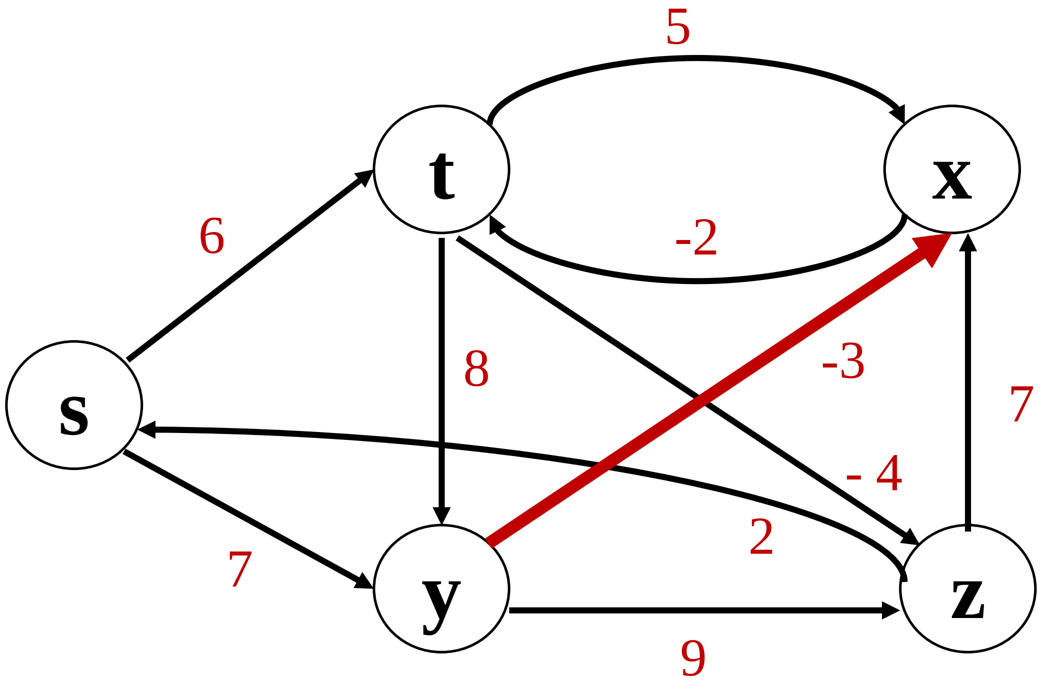
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

<u>Edge List:</u> (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

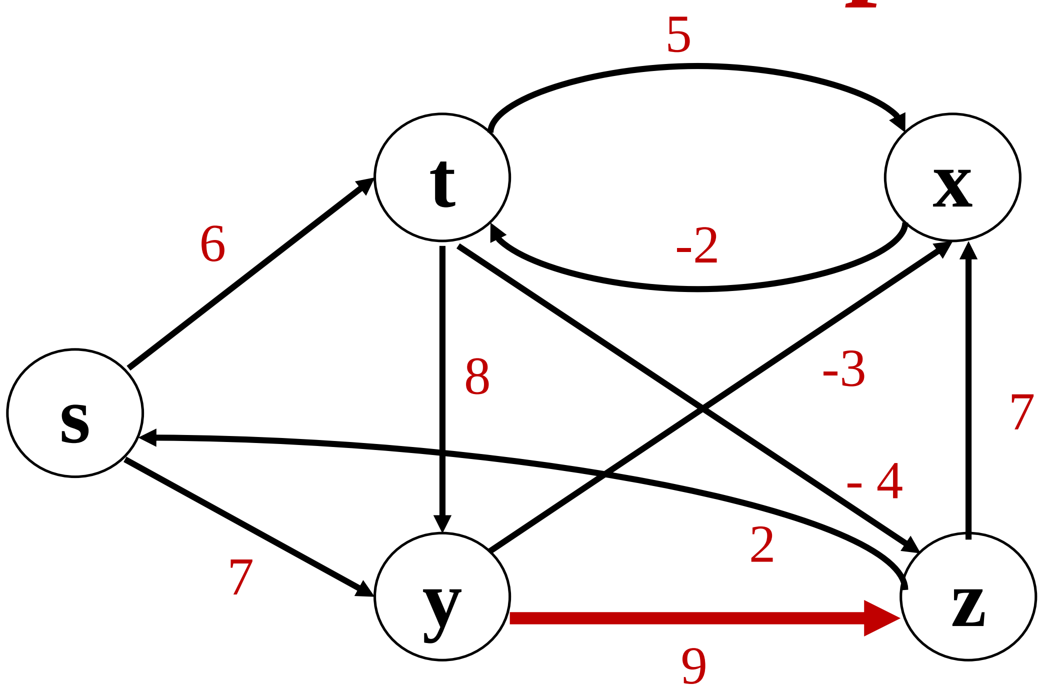
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

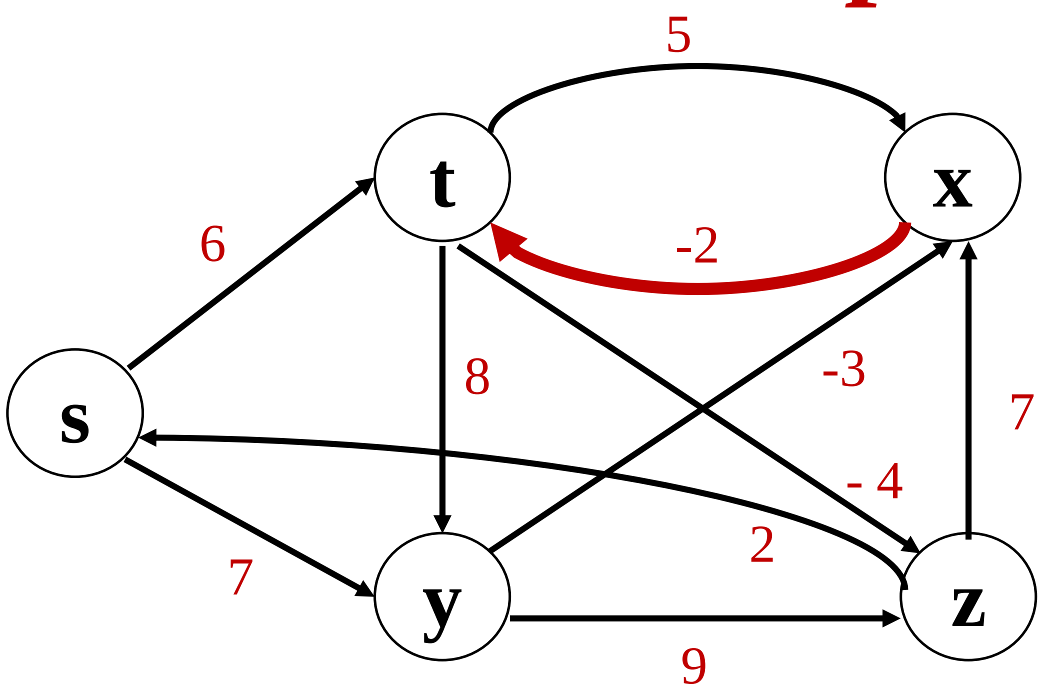
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

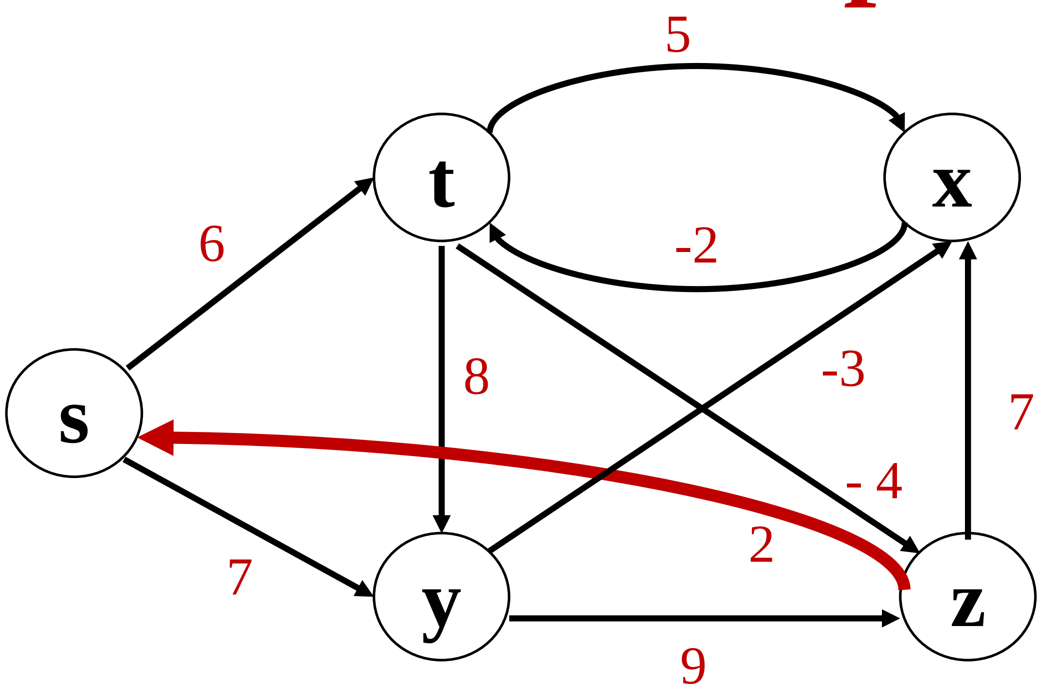
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

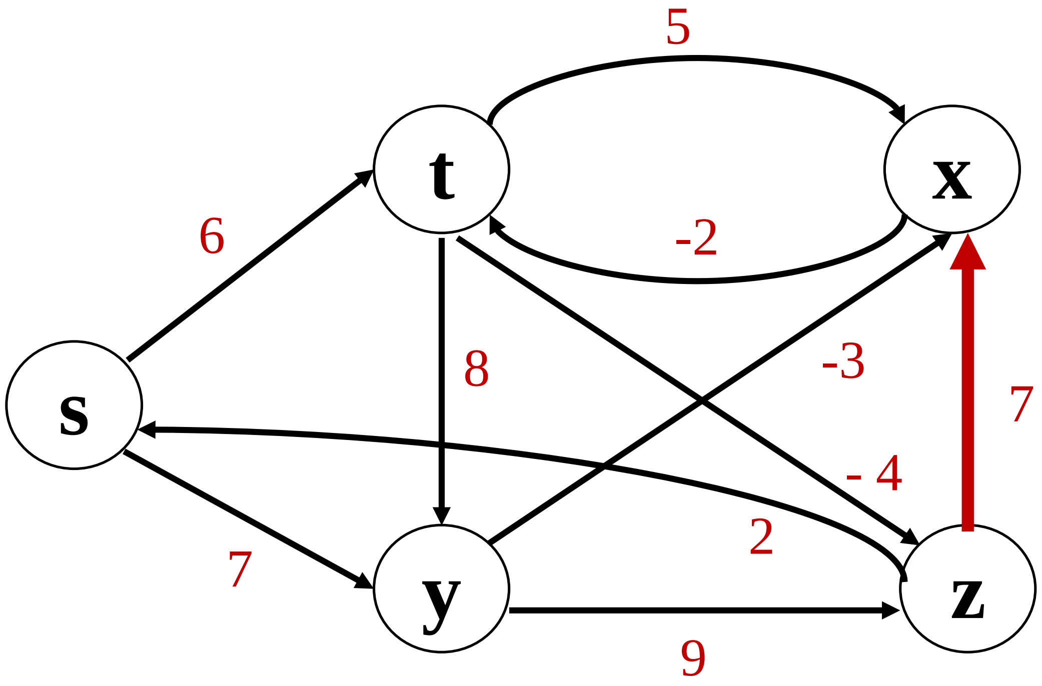
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

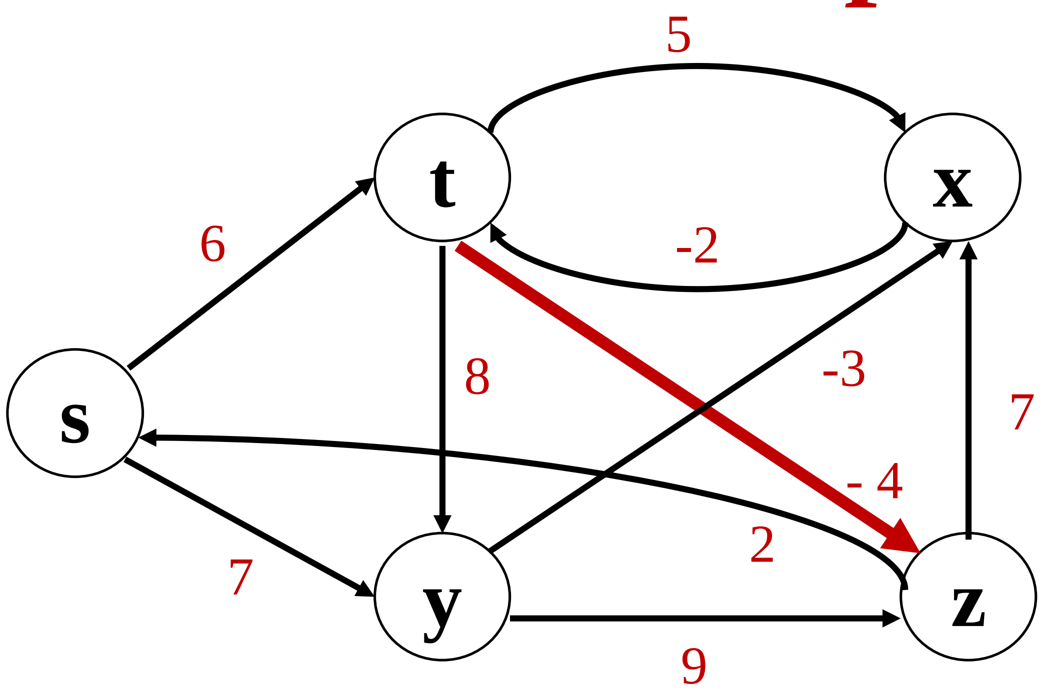
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

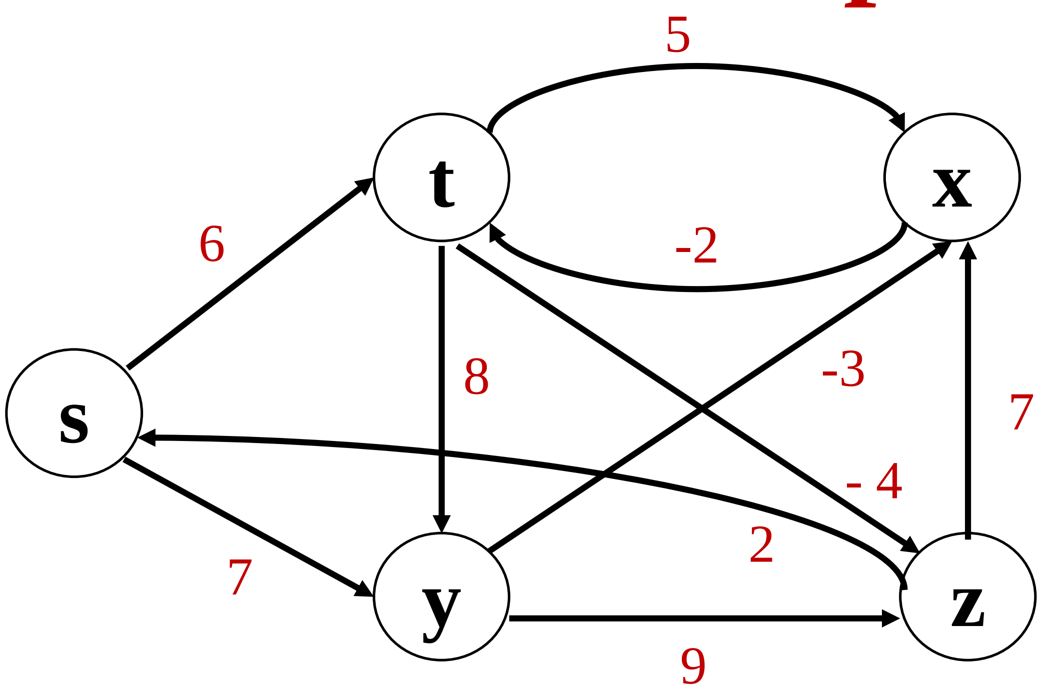
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

<u>Edge List:</u> (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

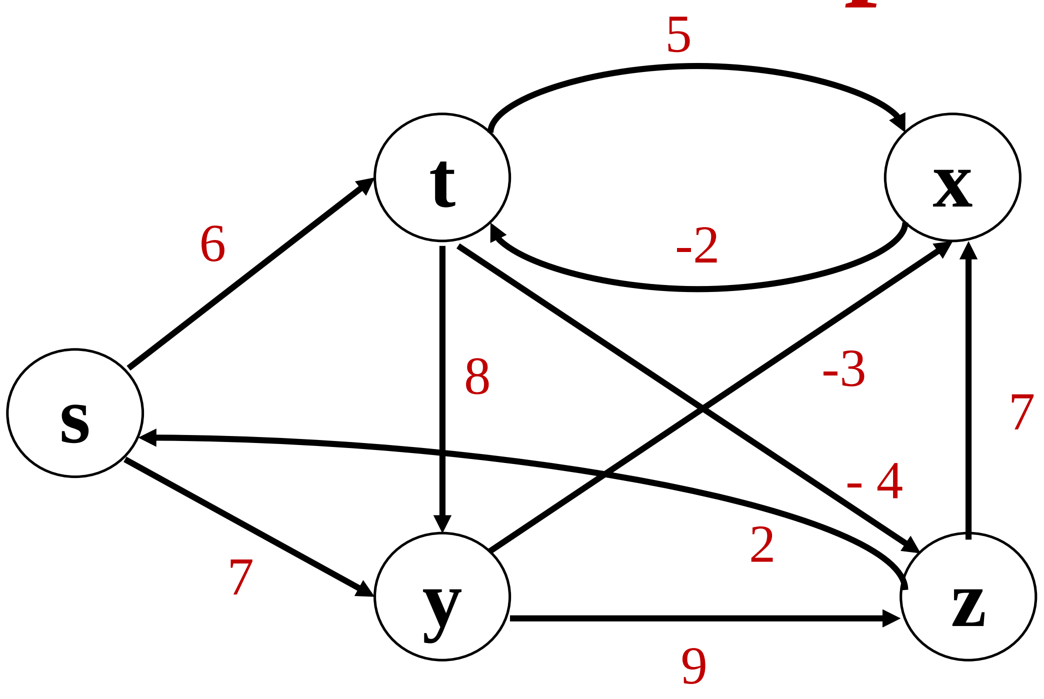
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t

<u>Edge List:</u> (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

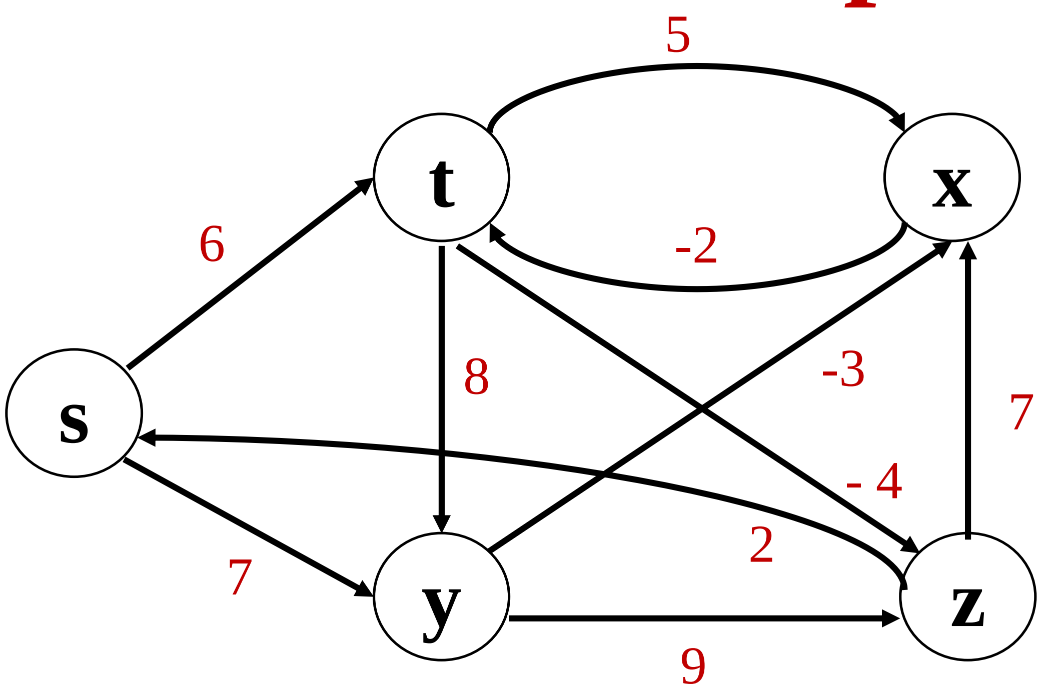
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t
3rd	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

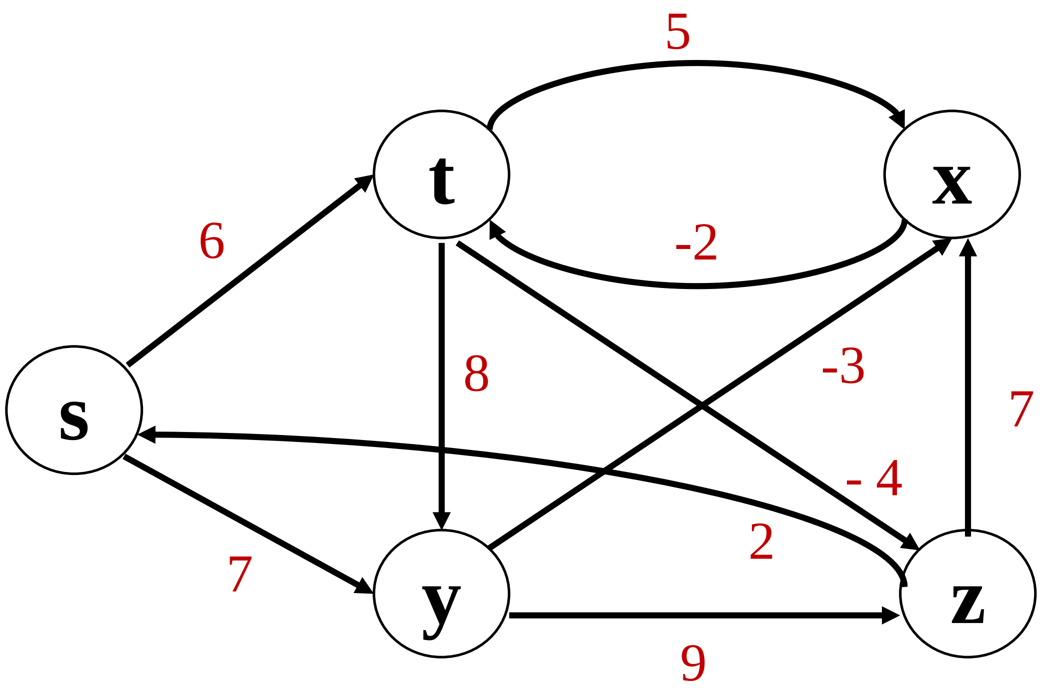
Bellman-Ford Example



Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t
3rd	0	2/x	4/y	7/s	-2/t
4th	0	2/x	4/y	7/s	-2/t

Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

Bellman-Ford Example

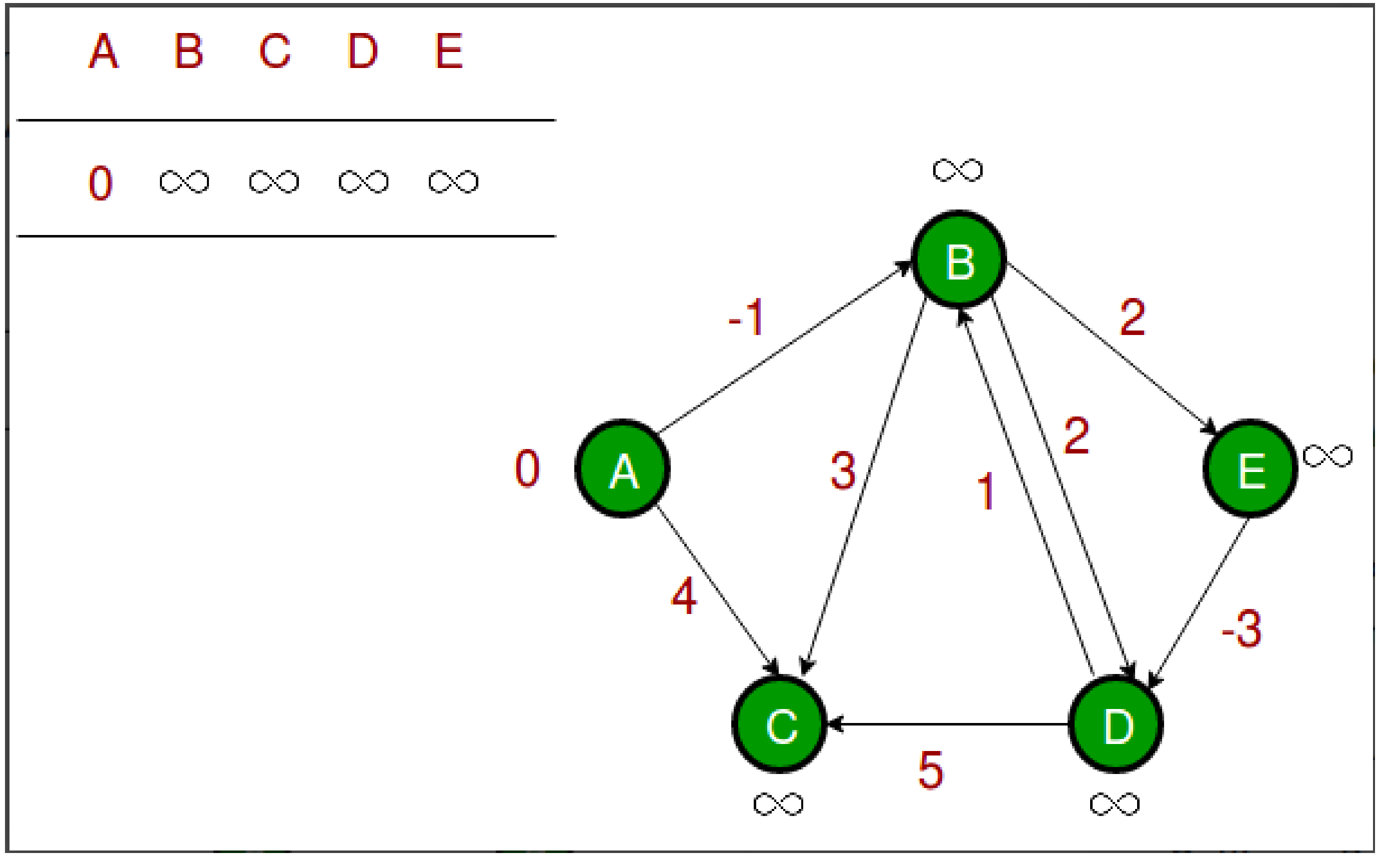


Vertex	s	t	x	y	z
Initial	0/NIL	∞/NIL	∞/NIL	∞/NIL	∞/NIL
1st	0	2/x	4/y	7/s	-2/t
2nd	0	2/x	4/y	7/s	-2/t
3rd	0	2/x	4/y	7/s	-2/t
4th	0	2/x	4/y	7/s	-2/t
5th					

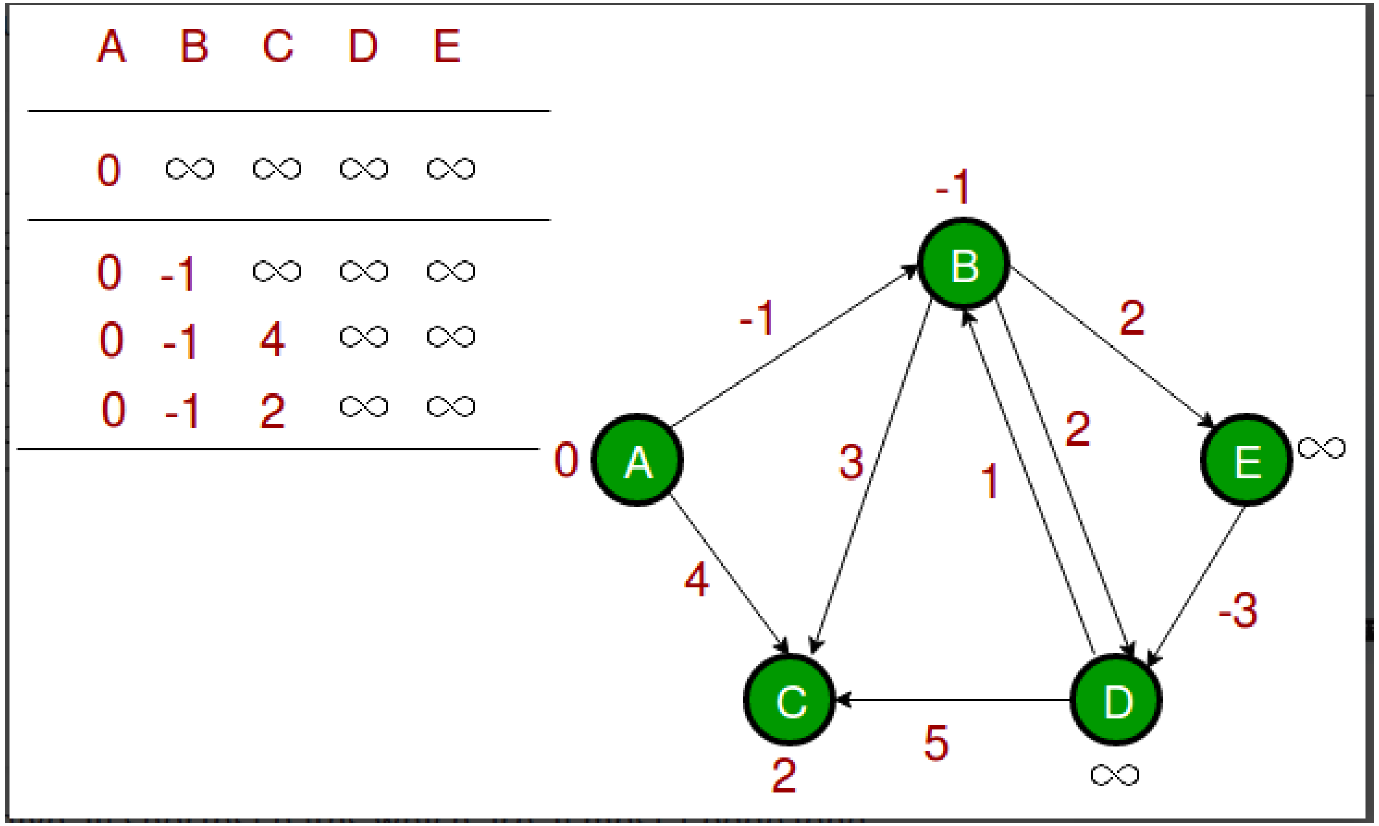
Edge List: (s,t) , (s,y) , (t,y) , (t,x) , (y,x) , (y,z) , (x,t) , (z,s) , (z,x) , (t,z)

If distance is decreased(relaxed) in 5th iteration, it means there exist a negative weight cycle.

Example

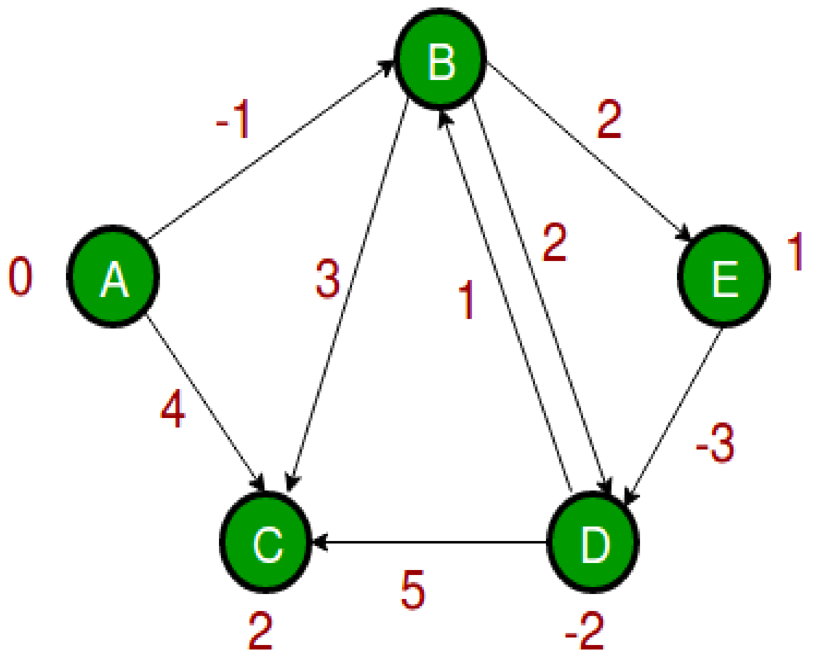


Example



Example

A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1
0	-1	2	1	1
0	-1	2	-2	1



Complexity

The Bellman-Ford algorithm runs in time $O(VE)$, since the initialization in line 1 takes $\Theta(V)$ time, each of the $|V| - 1$ passes over the edges in lines 2–4 takes $\Theta(E)$ time, and the **for** loop of lines 5–7 takes $O(E)$ time.

Applications

- For the Internet specifically, there are many protocols that use Bellman-Ford

  Information protocol.
- for example the Routing Information Protocol(RIP).
- A second example is the interior gateway routing protocol to help machines exchange routing data within a system.

Thank you