The Bellman-Ford Shortest Path Algorithm

Class Overview

- The shortest path problem
- Differences
- The Bellman-Ford algorithm
- Time complexity

Shortest Path Problem

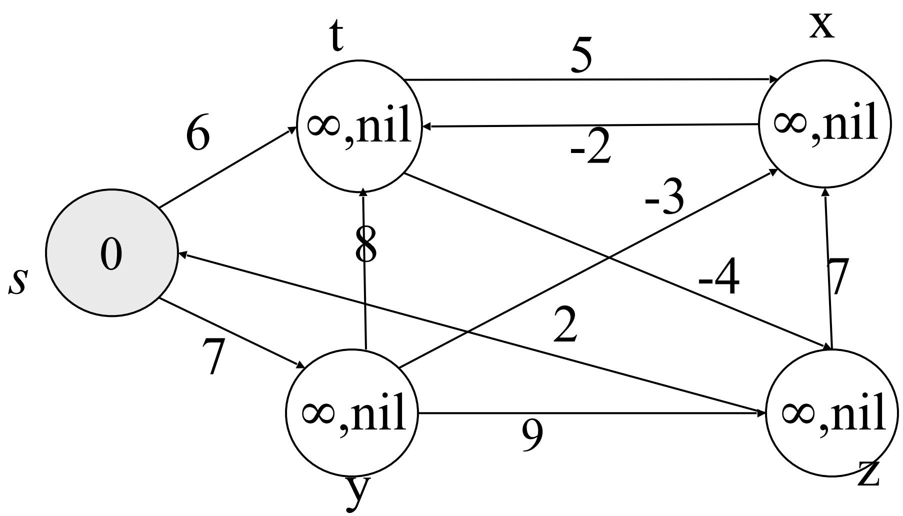
- Weighted path length (cost): The sum of the weights of all links on the path.

- The single-source shortest path problem: Given a weighted graph G and a source vertex s, find the shortest (minimum cost) path from s to every other vertex in G.

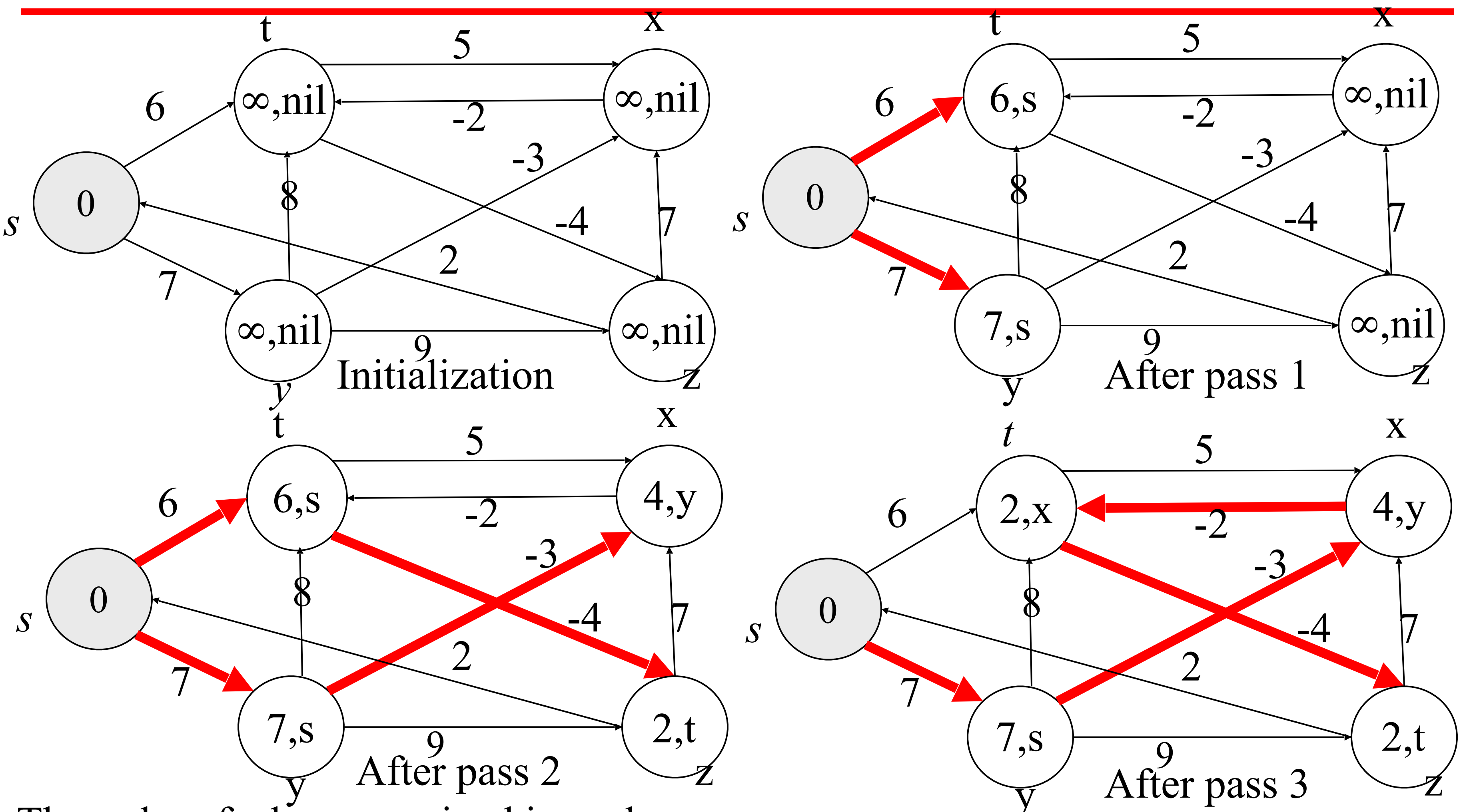
Differences

- Negative link weight: The Bellman-Ford algorithm works; Dijkstra's algorithm doesn't.
- Distributed implementation: The Bellman-Ford algorithm can be easily implemented in a distributed way. Dijkstra's algorithm cannot.
- Time complexity: The Bellman-Ford algorithm is higher than Dijkstra's algorithm.

The Bellman-Ford Algorithm



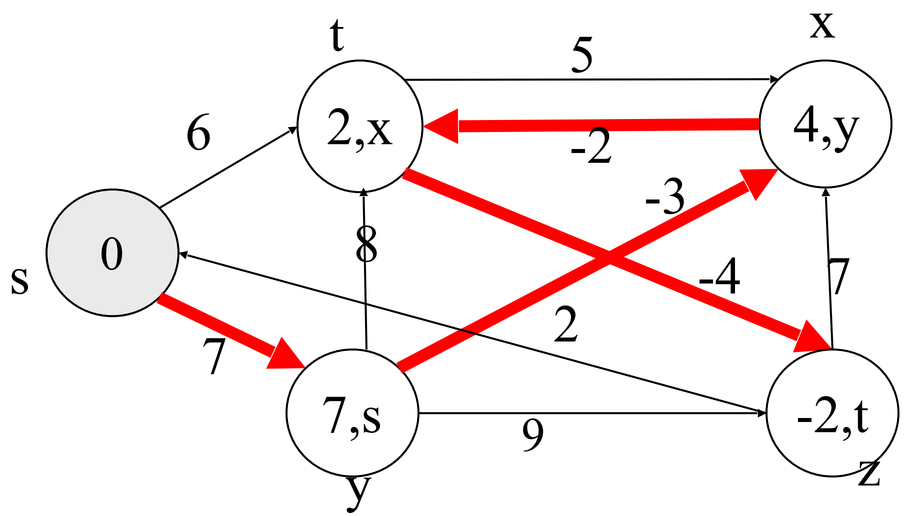
The Bellman-Ford Algorithm



The order of edges examined in each pass:

(t, x), (t, z), (x, t), (y, x), (y, t), (y, z), (z, x), (z, s), (s, t), (s, y)

The Bellman-Ford Algorithm



After pass 4

The order of edges examined in each pass:
(t, x), (t, z), (x, t), (y, x), (y, t), (y, z), (z, x), (z, s), (s, t), (s, y)

The Bellman-Ford Algorithm

Bellman-Ford(G, w, s)

```
Initialize-Single-Source(G, s)
for i := 1 to |V| - 1 do
    for each edge (u, v) ∈ E do
        Relax(u, v, w)
for each vertex v ∈ u.adj do
    if d[v] > d[u] + w(u, v)
        then return False    // there is a negative cycle
return True
```

Relax(u, v, w)

```
if d[v] > d[u] + w(u, v)
    then  d[v] := d[u] + w(u, v)
          parent[v] := u
```

Time Complexity

Bellman-Ford(G, w, s)

```
Initialize-Single-Source(G, s)                  → O(|V|)
for i := 1 to |V| - 1 do
    for each edge (u, v) ∈ E do
                                                → O(|V||E|)
        Relax(u, v, w)
for each vertex v ∈ u.adj do                    → O(|E|)
    if d[v] > d[u] + w(u, v)
        then return False    // there is a negative cycle
return True
```

Time complexity: O(|V||E|)