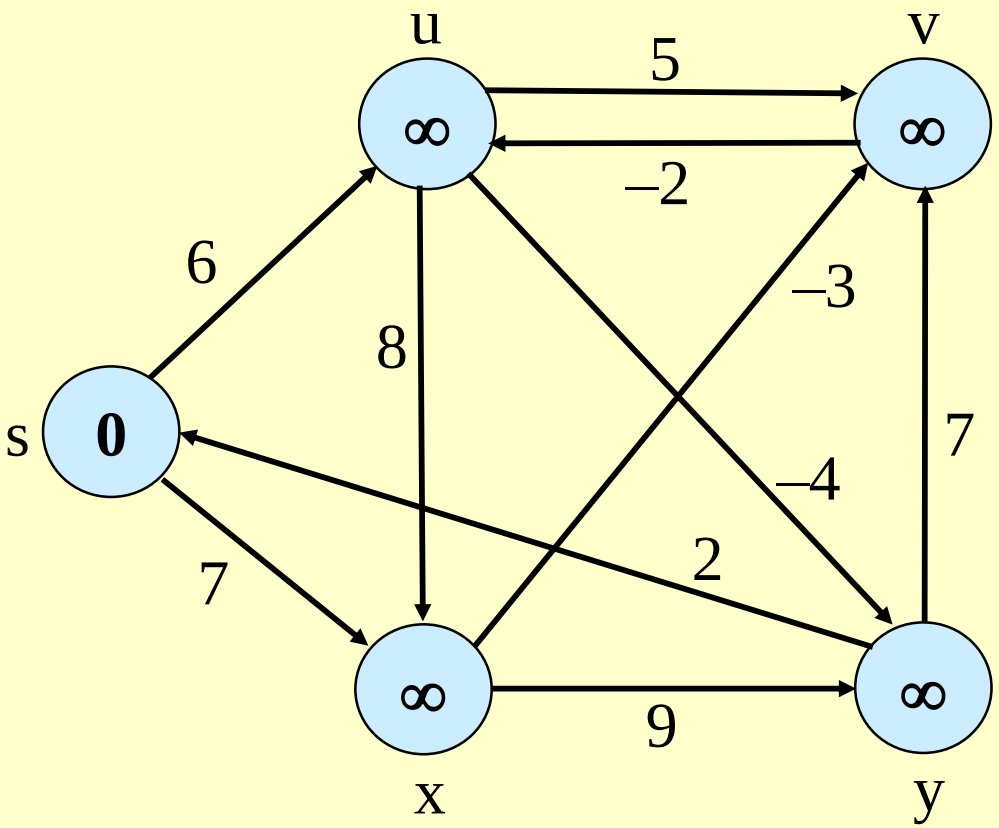
Single-Source Shortest Paths

- **Given:** A single source vertex in a weighted, directed graph.
- Want to compute a shortest path to each possible destination from the source.
- We assume either
  - no negative-weight edges, or
  - no reachable negative-weight cycles.
- Algorithm will compute a shortest-path tree.

Example



Can the shortest paths to the different nodes together have a cycle ?

How could you find the shortest path when edge weight is one for all the edges?

Can BFS be used to find the shortest path when edge weight is the same for all the edges?

If I use negative edge weights can we use BFS?

If I introduce different edge weights for different edges will BFS work?

Simpler Problem: Consider a DAG. How would you find the shortest path to the nodes in the DAG?

Relaxation

Algorithms keep track of d[v], π[v]. **Initialized** as follows:

```
Initialize(G, s)
for each v ∈ V[G] do
      d[v] := ∞;
      π[v] := NIL
od;
d[s] := 0
```

These values are changed when an edge (u, v) is **relaxed**:

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

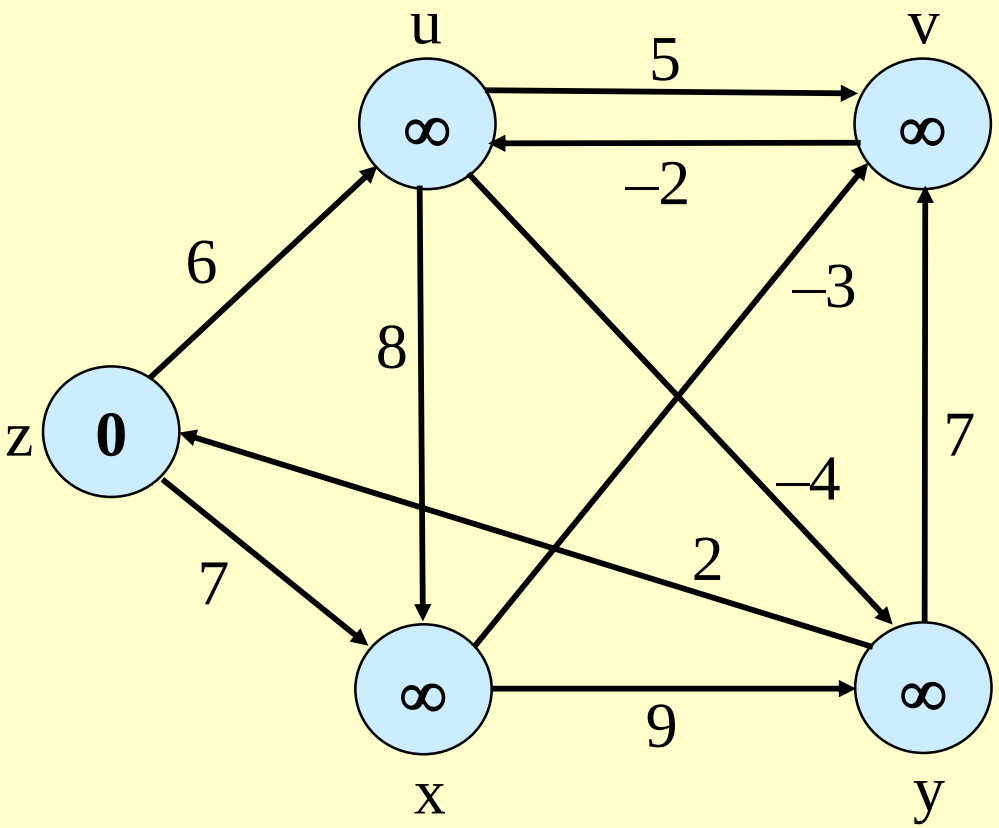
Bellman-Ford Algorithm

Can have negative-weight edges. Will "detect" reachable negative-weight cycles.

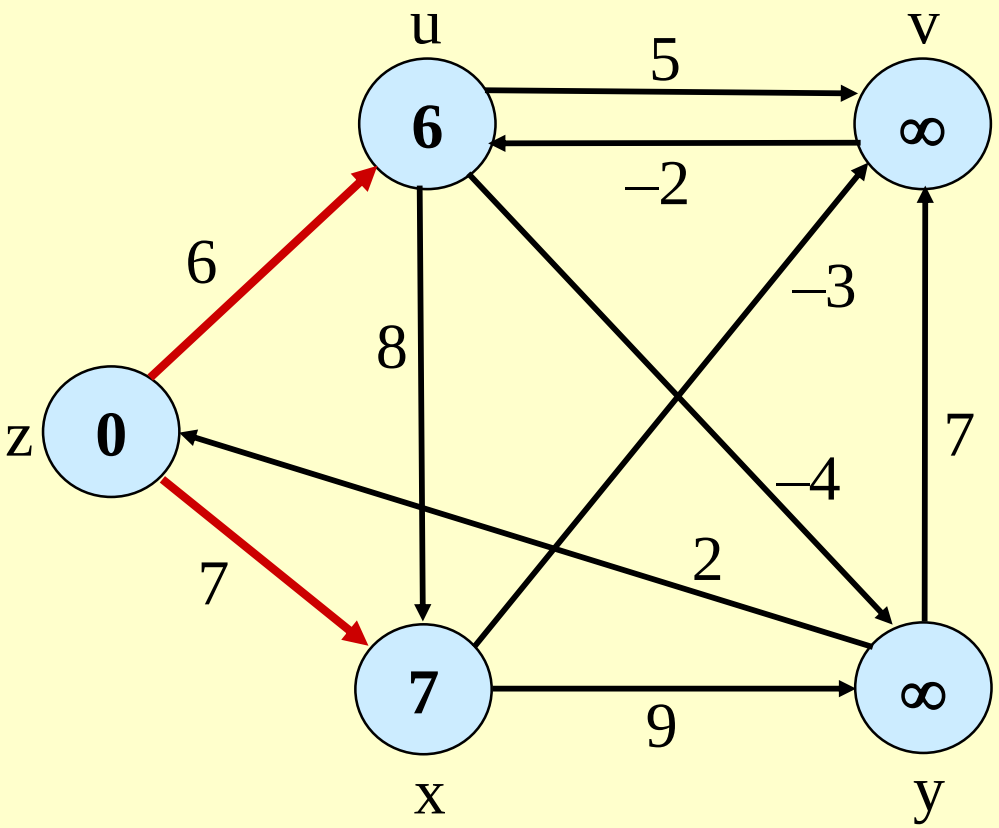
```
Initialize(G, s);
for i := 1 to |V[G]| –1 do
for each (u, v) in E[G] do
      Relax(u, v, w)
od
od;
for each (u, v) in E[G] do
if d[v] > d[u] + w(u, v) then
      return false
fi
od;
return true
```

Time Complexity is O(VE).

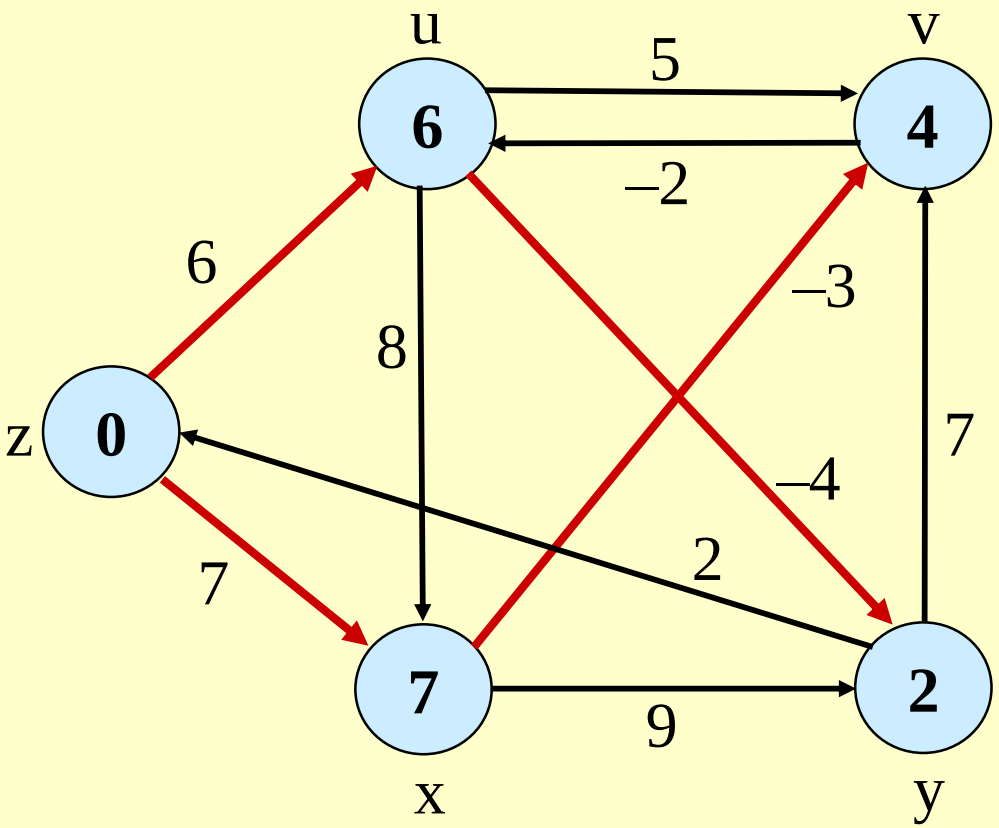
Example



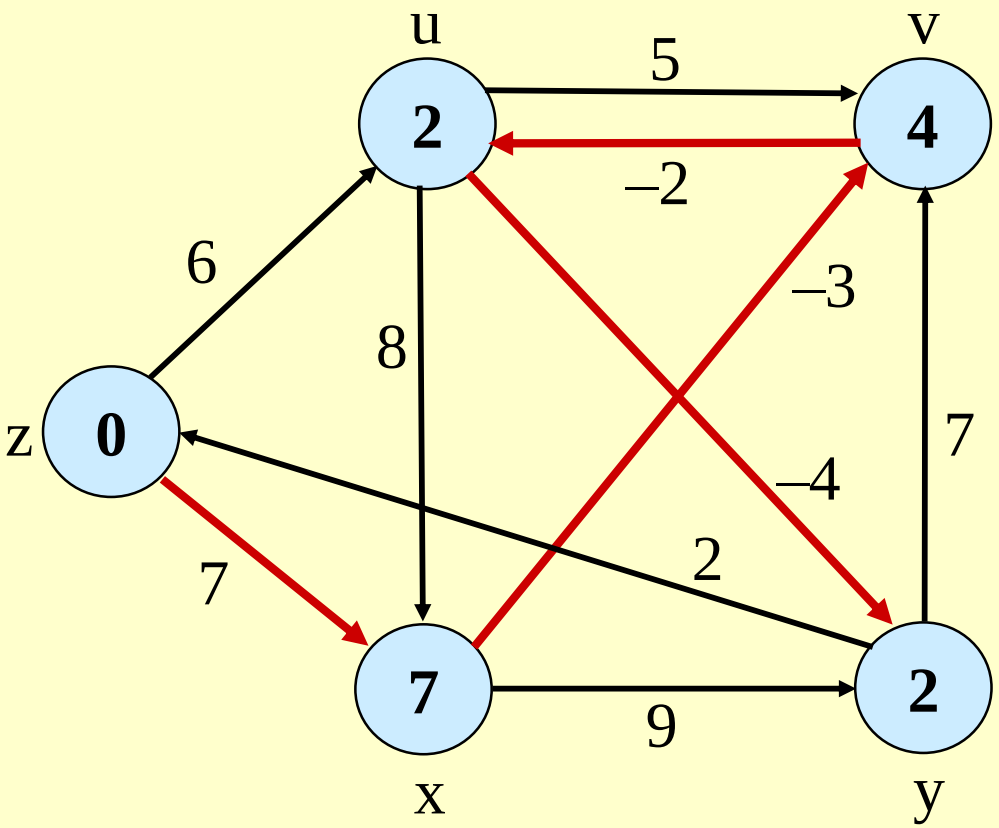
Example



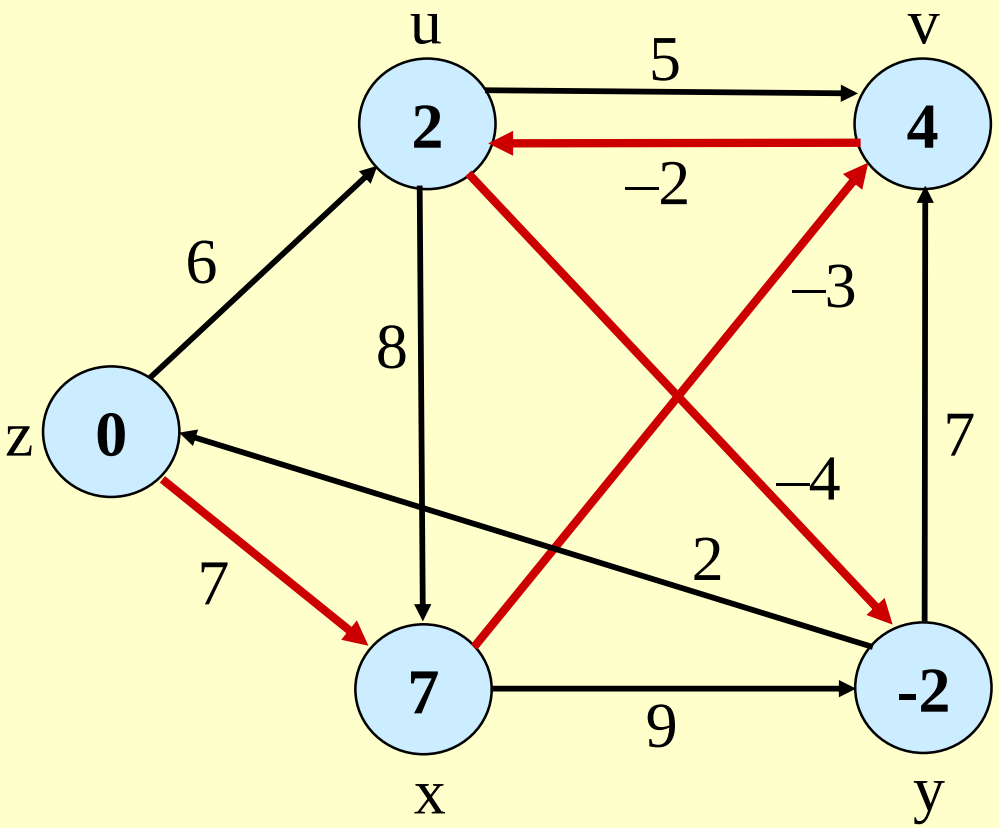
Example



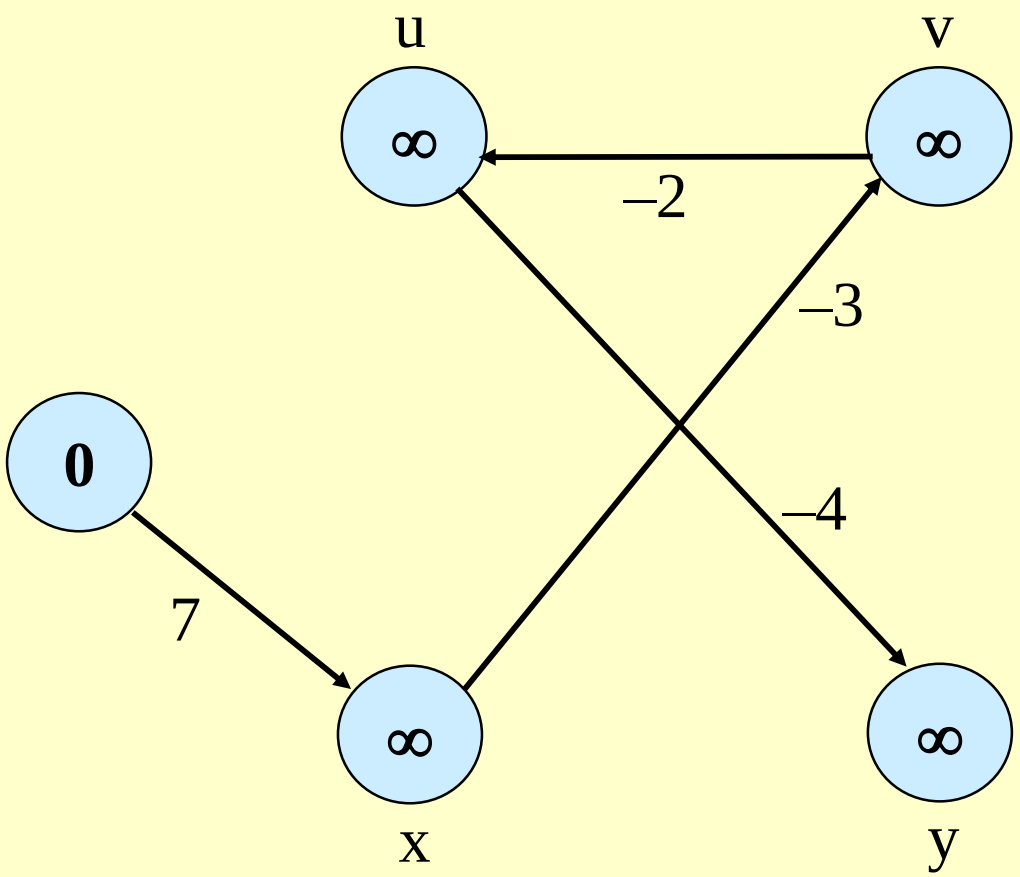
Example



Example (2nd iteration)



Shortest Path Tree



Questions

- If d[v] != ∞ , then there exists *some* path from s to v ?
- d[v] can increase with time ?
- Therefore, if d[v] = δ(s, v) at any time, then d[v] can decrease further?
- After i iterations of relaxing on all edges, if the shortest path to v has i edges, then d[v] = δ(s, v)?
- How many edges can any shortest path contain?

Questions

- d[v], if not ∞, is the length of *some* path from s to v.
- d[v] either stays the same or decreases with time
- Therefore, if d[v] = δ(s, v) at any time, this holds thereafter
- Note that d[v] $\geq$ δ(s, v) always
- After i iterations of relaxing on all edges, if the shortest path to v has i edges, then d[v] = δ(s, v).

Properties of Relaxation

Lemma 24.11: ($\forall$ v:: d[v] $\geq \delta(s, v)$) is an invariant.

Implies d[v] doesn't change once d[v] = $\delta(s, v)$.

Corollary 24.12: If there is no path from s to v, then d[v] = $\delta(s, v) = \infty$ is an invariant.

Bellman-Ford Algorithm

Can have negative-weight edges. Will "detect" reachable negative-weight cycles.

```
Initialize(G, s);
for i := 1 to |V[G]| –1 do
for each (u, v) in E[G] do
      Relax(u, v, w)
od
od;
for each (u, v) in E[G] do
if d[v] > d[u] + w(u, v) then
      return false
fi
od;
return true
```

Time Complexity is O(VE).

- So if Bellman-Ford has not converged after V(G) - 1 iterations, then there cannot be a shortest path tree, so there must be a negative weight cycle.

Questions

Is it possible that you get the shortest paths in the first iteration ?

What permutation of the edges could I be lucky to select that could give me the shortest path in the first iteration?

If it is given that no negative cycle exists, then, how many iterations are required?

Suppose it is not known whether a negative cycle exists, then on seeing a reduction in distance to a point, can we conclude that a negative cycle exists?

Would a change in edge permutation per iteration still give a valid shortest path?

Is it possible that you get the shortest paths in the first iteration ?
YES

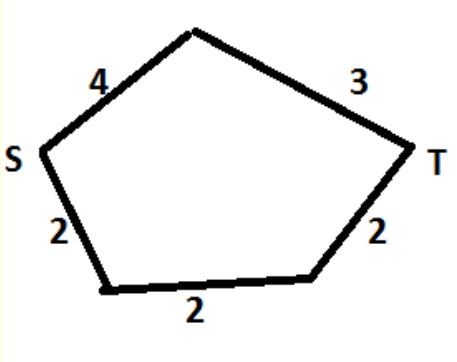
What permutation of the edges could I be lucky to select that could give me the shortest path in the first iteration? Edges in the shortest path tree taken in order of their appearance in the tree.

If it is given that no negative cycle exists, then, how many iterations are required? Whenever two iteration
show no change. Thus <=V.
Suppose it is not known whether a negative cycle exists, then on seeing a reduction in distance to a point, can we conclude that a negative cycle exists? Yes if distance to source becomes negative
Would a change in edge permutation per iteration still give a valid shortest path? Yes

The shortest path between two nodes will be the same if the weight of each edge in the graph is increased by a positive constant K.

The shortest path between two nodes will be the same if the weight of each edge is divided by a positive constant K.

The shortest path between two nodes will be the same if the weight of each edge in the graph is increased by a positive constant K.



False:

Consider figure. From S to T the shortest path is 2+2+2.

Increase all edge weights by 1. The shortest path is now 4+1+3+1 and not 2+1+2+1+2+1

The shortest path between two nodes will be the same if the weight of each edge is divided by a positive constant K.

True: It is like changing the unit of distance.

Suggestion

Instead of relaxing all edges why not relax only edges incident on a node whose current estimate is not infinity ? – true! Will be effecient.
Complexity ? O(E) per iteration ☺ as all nodes
could be neighbors of the source.