
15.082 and 6.855J

**The Successive Shortest Path Algorithm
and the Capacity Scaling Algorithm
for the Minimum Cost Flow Problem**

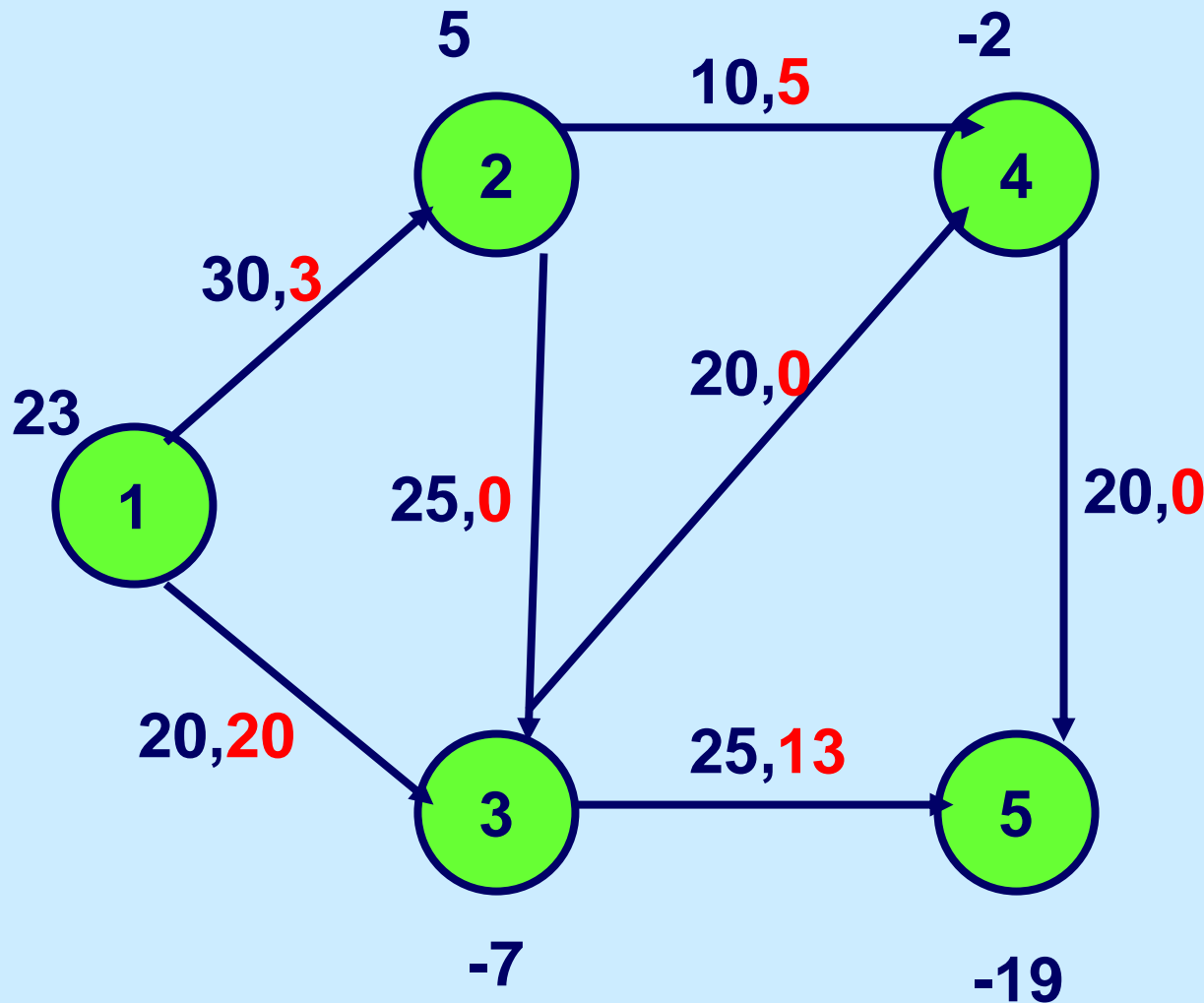
Pseudo-Flows

A pseudo-flow is a "flow" vector x such that
 $0 \leq x \leq u$.

Let $e(i)$ denote the excess (deficit) at node i .

The infeasibility of the pseudo-flow is $\sum_{e(i)>0} e(i)$.

Supplies/Demands, Capacities, and Flows



What is $e(i)$ for each node i ?

What is the infeasibility of the flow?

Optimality Conditions

Let π be a vector of node potentials.

Let x be a pseudo-flow.

For each arc $(i,j) \in G(x)$, the reduced cost of (i,j) is

$$c^{\pi}_{ij} = c_{ij} - \pi_i + \pi_j.$$

We say that a pseudo-flow x satisfies the **optimality conditions** (is **dual feasible**) if $c^{\pi}_{ij} \geq 0$ for each $(i,j) \in G(x)$.

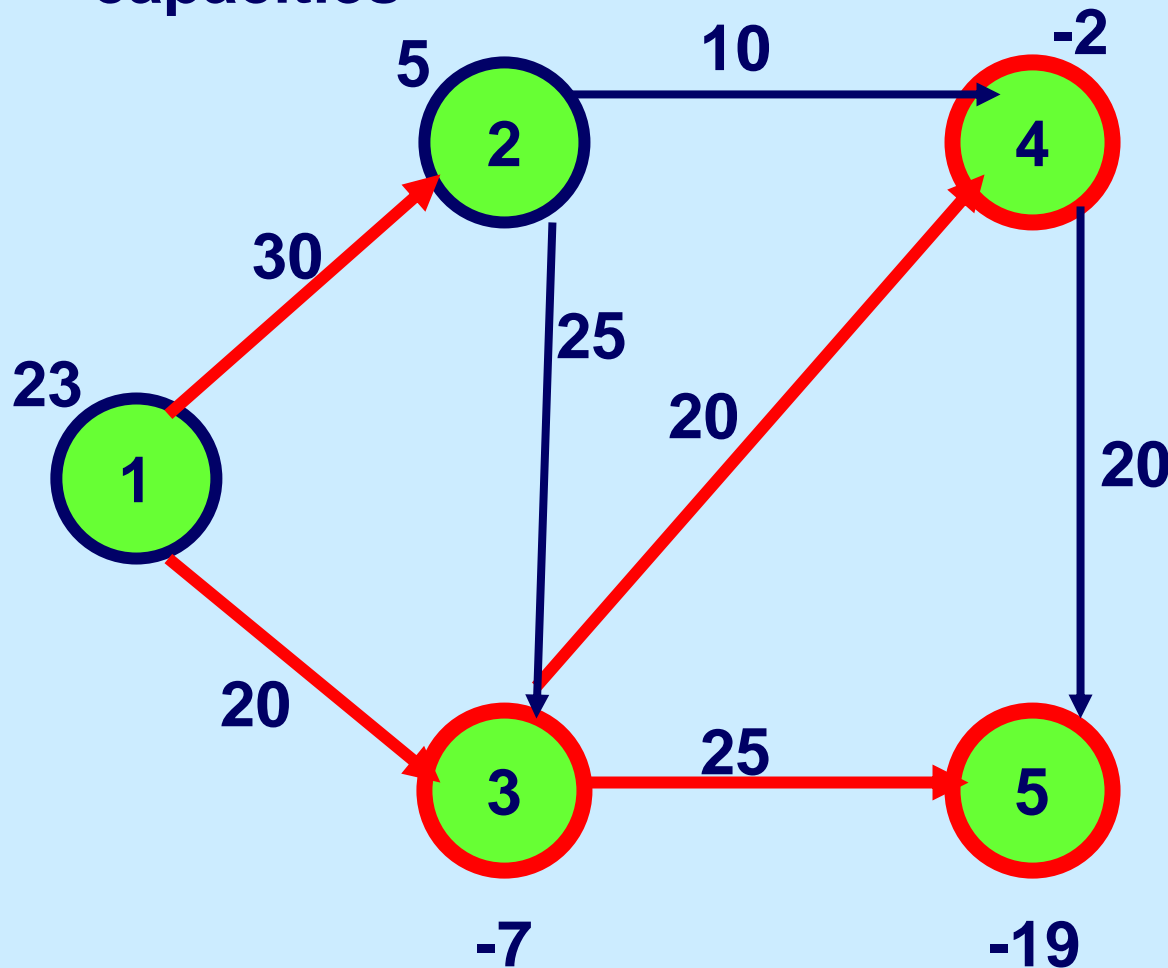
Overview of the algorithm

Start with a pseudo-flow x that satisfies the optimality conditions. :

At each iteration maintain the optimality conditions, but continually reduce the infeasibility.

Maintaining optimality conditions

A dual feasible network with excesses/deficits and capacities



Suppose that red arcs have a reduced cost of 0

Optimality conditions are maintained if flow is sent on a red path.

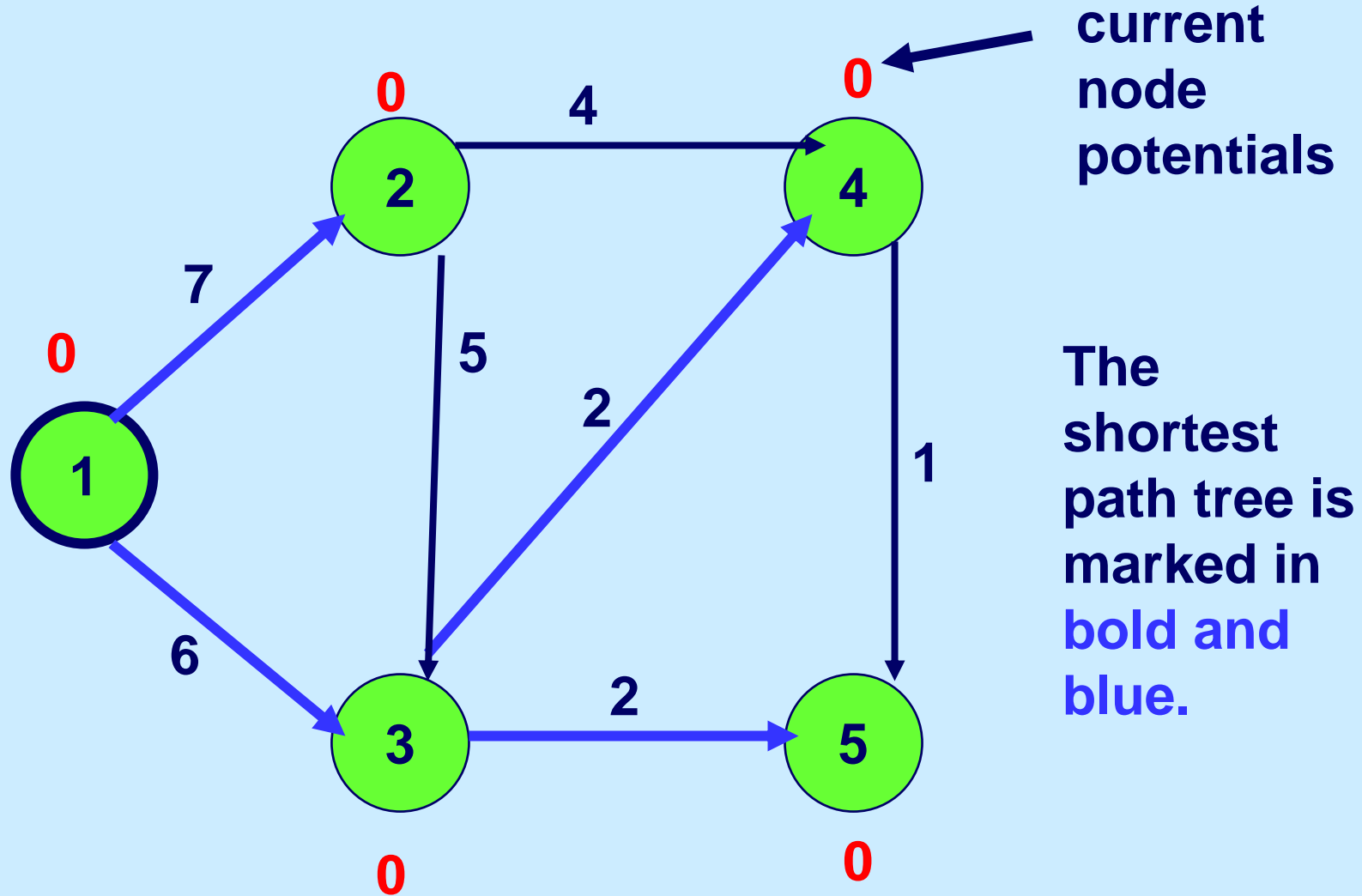
Sending Flow Along Paths with Reduced Cost of 0

If each arc of $P \in G(x)$ has a reduced cost of 0, then sending flow along P maintains the optimality conditions.

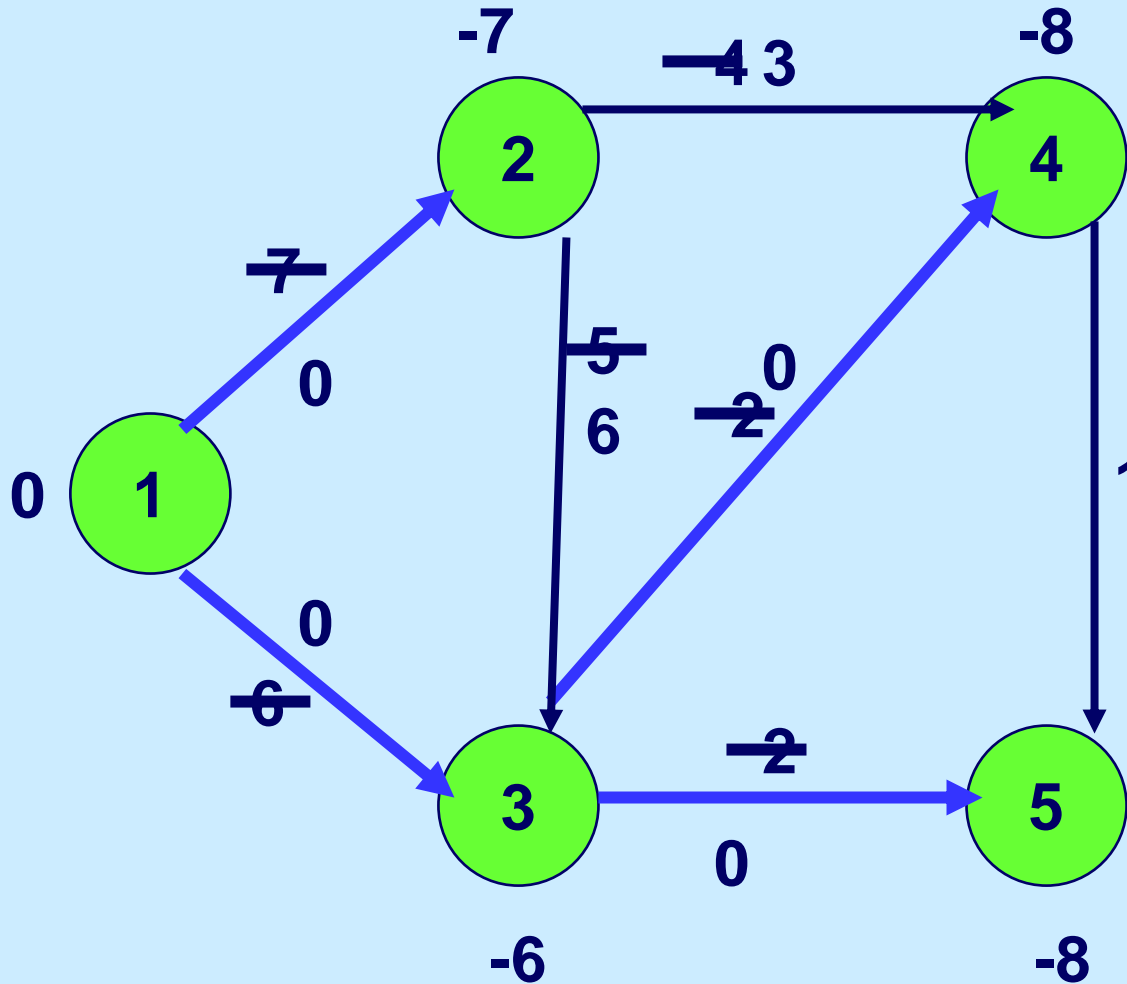
To reduce infeasibility, send flow from an excess node to a deficit node.

Next: how to ensure that there is a path whose reduced cost is 0.

Find a shortest path node from node i, and update potentials.



Update the Potentials



To obtain new node potentials, subtract the shortest path distances from the old potentials.

The shortest path tree now has 0 reduced cost arcs.

The Successive Shortest Path Algorithm

begin

find an initial dual feasible pseudo-flow (x, π)
{More on this later}

while there is a node with positive excess **do**

begin

$S = \{i : e(i) > 0\}; T = \{j : e(j) < 0\};$

select a node $s \in S$ and a node $t \in T$ such that
there is a path from s to t ;

determine shortest path distances $d(j)$ from node
 s to all other nodes in $G(x)$ with respect c^π ;

update $\pi = \pi - d$;

send flow along a 0-reduced cost path from s to t ;

update the residual network and S and T ;

end

end

The SSP Algorithm

Finding an Initial Dual Feasible Solution

Case 1. All original costs are non-negative.

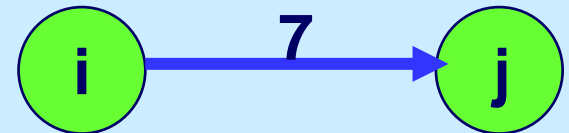
Then take $x = 0$ and $\pi = 0$.

In this case, the reduced costs are the original costs.

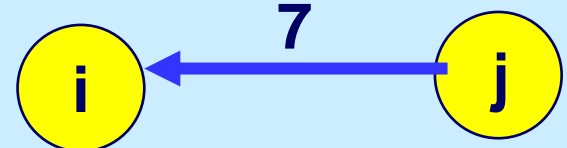
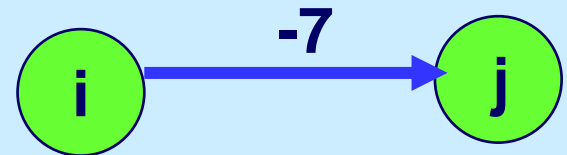
Case 2. All arcs have a finite capacity.

Set $\pi = 0$ and choose x as follows:

Case 2a. Arc (i,j) has a non-negative cost. Then let $x_{ij} = 0$



Case 2b. Arc (i,j) has a negative cost. Then let $x_{ij} = u_{ij}$.



Arc in $G(x)$ after saturating (i,j) ₁₀

Analysis of the Successive Shortest Path Algorithm

Each augmentation reduces the infeasibility by at least 1.

What is a bound on the initial infeasibility?

The time per augmentation is the time to run Dijkstra's algorithm (or any shortest path algorithm with non-negative distances.)

The Capacity Scaling Algorithm

- ◆ It's like the shortest augmenting path algorithm, except that we try to send flow along paths with large capacity from nodes with large excess to nodes with large deficit.
- ◆ In a Δ -scaling phase, each augmentation will send at least Δ units of flow. Initially Δ is very large;
when no augmentations of size Δ exist, then $\Delta := \Delta/2$.

The Capacity Scaling Algorithm

- ◆ Initially, choose Δ to be the largest power of 2 so that there is a path of capacity Δ from a node with excess $\geq \Delta$ to a node with excess $\leq -\Delta$.
- ◆ Initialization of the Δ -scaling phase: saturate each arc of $G(x)$ with capacity $\geq \Delta$ with negative reduced cost.
- ◆ Optimality conditions are maintained for arcs with capacity $\geq \Delta$.

Major subroutine

$G(x, \Delta) = G(x)$ restricted to arcs (i,j) with $r_{ij} \geq \Delta$

$S(\Delta) := \{i \in N : e(i) \geq \Delta\}; \quad T(\Delta) := \{i \in N : e(i) \leq -\Delta\};$

Subroutine Augmentation(s,t, Δ).

conditions: there is a node $s \in S(\Delta)$, a node $t \in T(\Delta)$, and a path in $G(x, \Delta)$ from s to t

begin

determine shortest path distances $d(\cdot)$ from node s to all other nodes in $G(x, \Delta)$ with respect to the reduced costs c^π ;

update $\pi := \pi - d$;

augment Δ units of flow along a shortest path from s to t in $G(x, \Delta)$;

update x , S , T , and $G(x, \Delta)$;

end;

Algorithm *capacity scaling*;

begin

initialize with a dual feasible pseudo-flow (x, π) ;
initialize Δ and $G(x)$;

while $\Delta \geq 1$

begin $\{\Delta$ -scaling phase $\}$

for every arc (i, j) in the residual network $G(x)$

if $r_{ij} \geq \Delta$ and $c_{ij} < 0$ **then** send r_{ij} unit of flow
along (i, j) ;

while there is a node $s \in S(\Delta)$, a node $t \in T(\Delta)$,
and a path in $G(x, \Delta)$ from s to t **do**

Augmentation(s, t, Δ)

$\Delta := \Delta/2$;

end;

end;

Capacity Scaling

Augmentations Are Per Scaling Phase

Assume: there is a large capacity path from node 1 to each node and from each node to node 1.

At the end of the Δ -scaling phase:

a. There is no node i with excess $e(i) \geq \Delta$, or

b. There is no node j with excess $e(j) \leq -\Delta$.

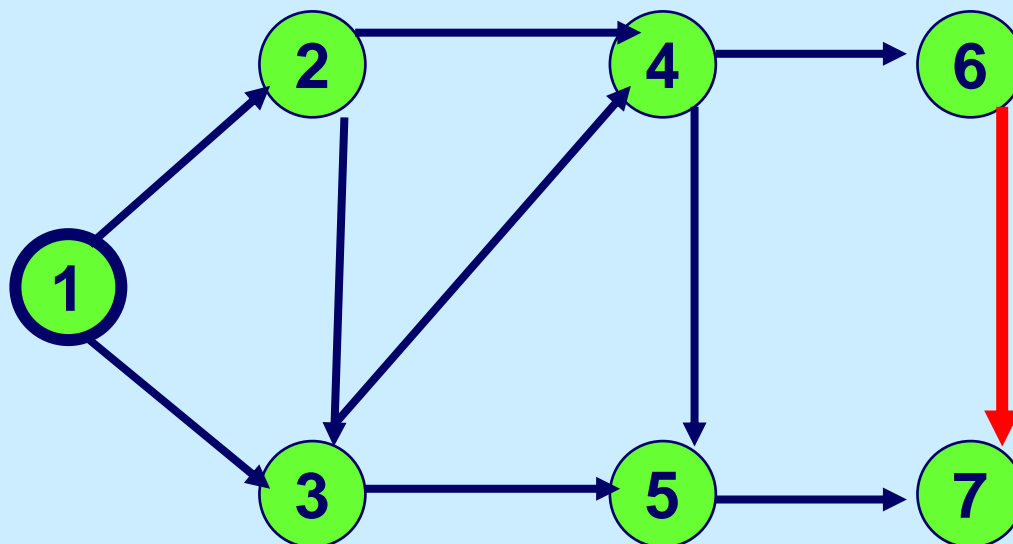
And each arc in $G(x, \Delta)$ has a non-negative reduced cost.

The infeasibility at the end of the Δ -scaling phase is at most $n\Delta$.

The infeasibility at the beginning of the Δ -scaling phase is $O(m\Delta)$.

The number of augmentations per scaling phase is $O(m)$.

Consider what happens at the end of the $\Delta = 32$ scaling phase



Suppose

$$r_{67} = 28$$

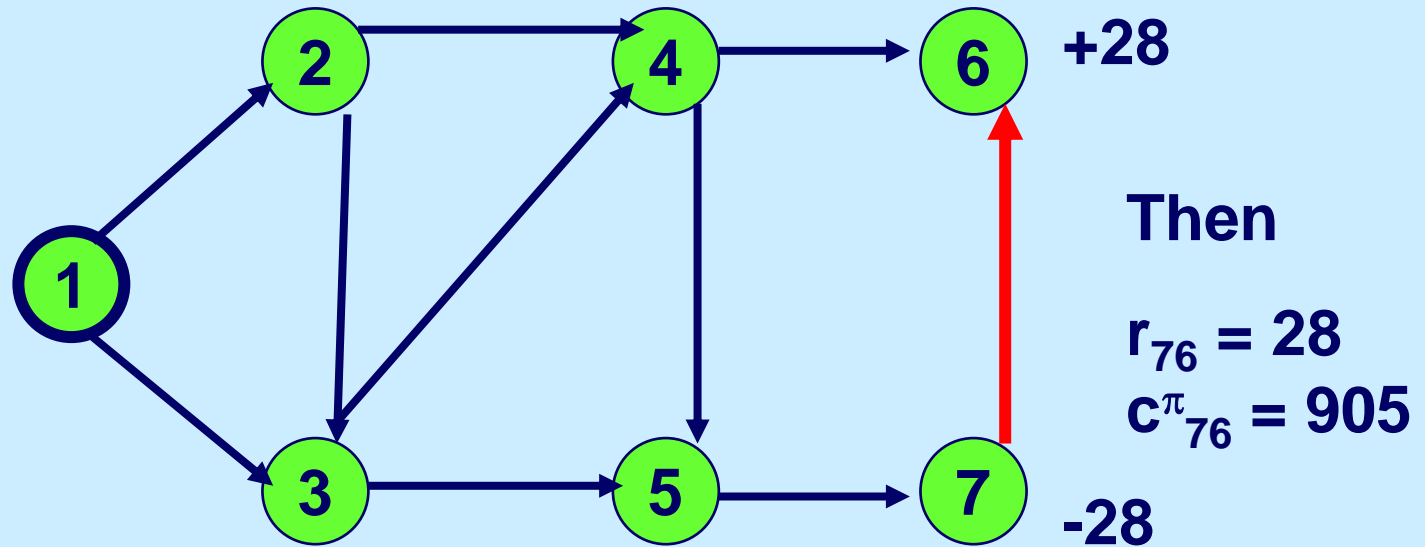
$$c_{67}^\pi = -905$$

Either there is no node with $e(i) \geq 32$, or else there is no node with $e(i) \leq -32$.

The infeasibility is $\leq 32n$ at the end of the phase

All arcs in $G(x, \Delta)$ have non-negative reduced cost.

Saturating arcs at the beginning of the $\Delta = 16$ scaling phase



Saturate arc (6,7) at the beginning of the 16-scaling phase.

$e(6)$ increases by 28; $e(7)$ decreases by 28;

The infeasibility may increase by up to Δ per saturation.

The infeasibility at the beginning of the Δ -scaling phase is at most $n\Delta + 2m\Delta = O(m\Delta)$.

Summary

The successive shortest path algorithm and the capacity scaling algorithm both rely on sending flow in paths that have a reduced cost of 0.

Both algorithms maintain dual feasibility conditions. The first in $G(x)$, the second in $G(x, \Delta)$.

The number of augmentations per scaling phase of the capacity scaling algorithm is $O(m)$.

Let B^* denote the largest excess for the initial dual feasible pseudo-flow.

The number of scaling phases is $O(\log B^*)$.

$$B^* < \max b(j) + n \max (u_{ij}).$$

Some Final Remarks

The capacity scaling algorithm can be made strongly polynomial with some modest changes.

The resulting algorithm has $O(m \log m)$ augmentations, even though the number of scaling phases can be as large as $O(m \log m)$.

The only way to have lots of scaling phases is to have very few augmentations per scaling phase. This comes out of the analysis, which is included in the book.

Next lecture: Network simplex algorithm.