15.082J and 6.855J and ESD.78J

The Successive Shortest Path Algorithm

and the Capacity Scaling Algorithm

for the Minimum Cost Flow Problem

Overview of lecture

- **Quick review of min cost flow and optimality conditions**
- **Pseudoflows**
- **The successive shortest path algorithm**
- **A polynomial time scaling algorithm**

The Minimum Cost Flow Problem

u_{ij} = capacity of arc (i,j).

c_{ij} = unit cost of shipping flow from node i to node j on (i,j).

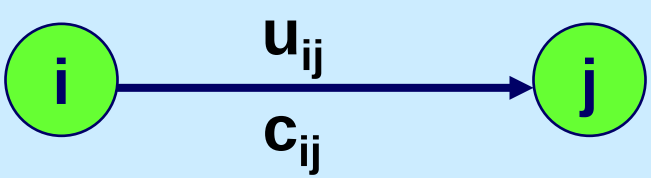
x_{ij} = amount shipped on arc (i,j)

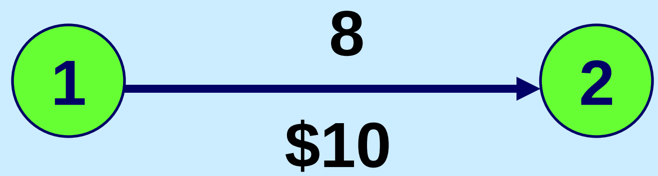
Minimize $\sum_{(i,j)\in A} c_{ij}x_{ij}$

$\sum_j x_{ij} - \sum_j x_{ji} = b_i$ for all $i \in N$.

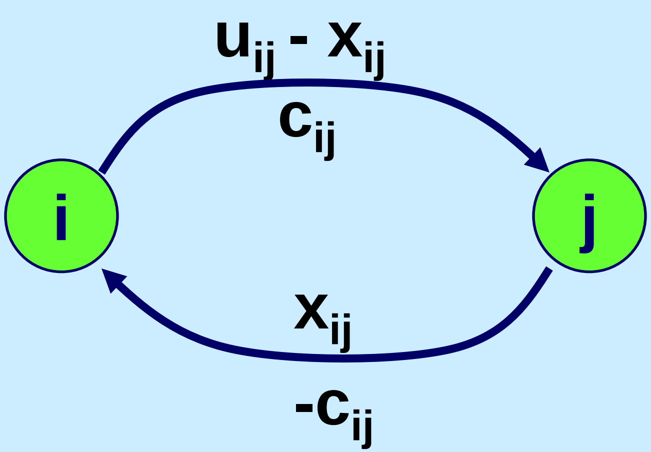
and $0 \leq x_{ij} \leq u_{ij}$ for all $(i,j) \in A$.

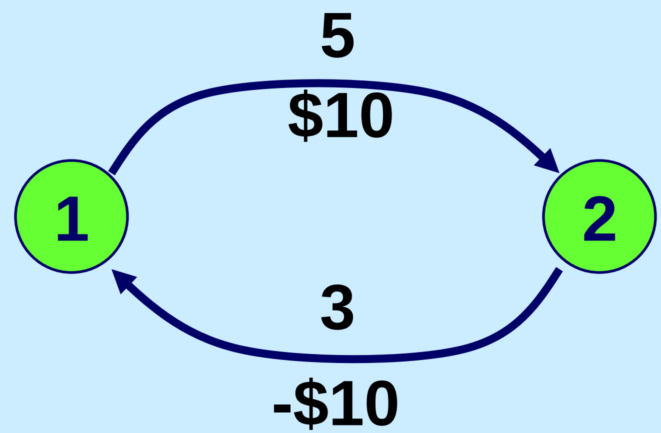
The Residual Network G(x)





Suppose $x_{12} = 3$



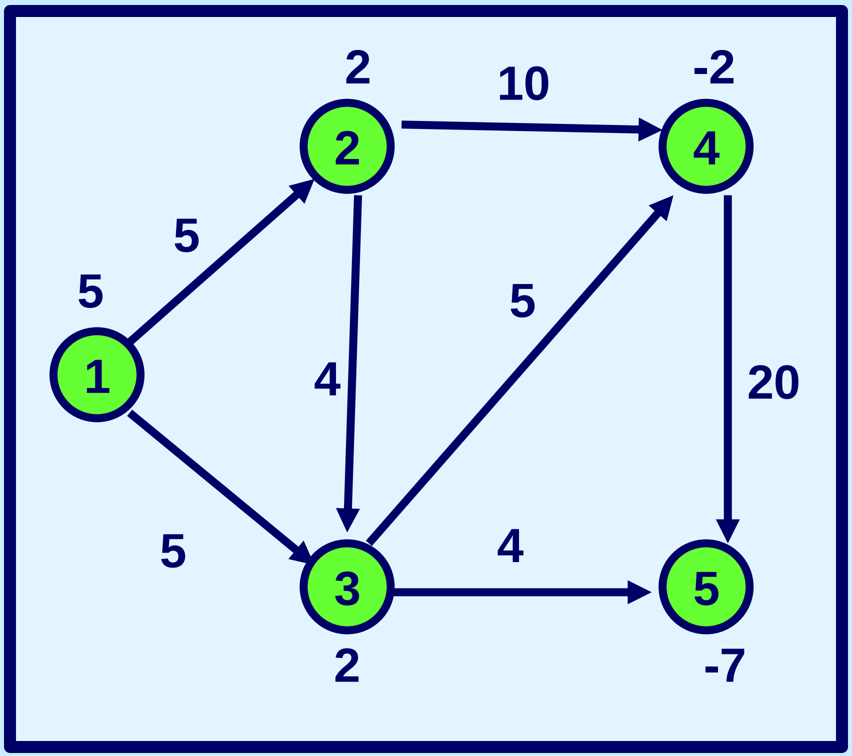


Reducing the flow in (i, j) by 1 is equivalent to sending one unit of flow from j to i. It reduces the cost by c_{ij}.

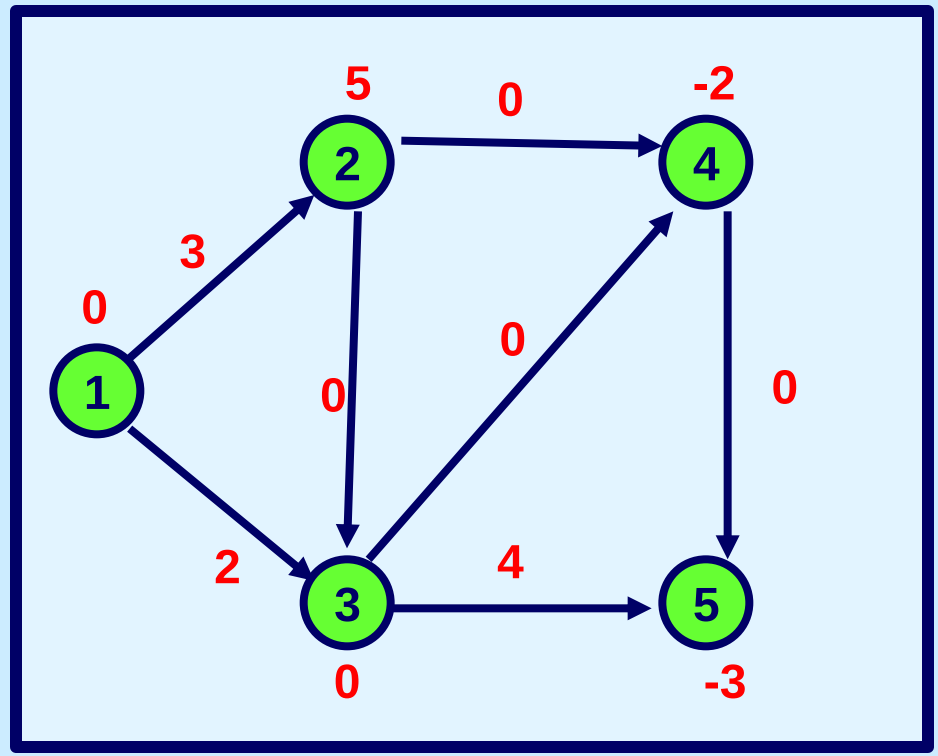
Pseudo-flows

A *pseudo-flow* is a "flow" vector x such that 0 δ x δ u.

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

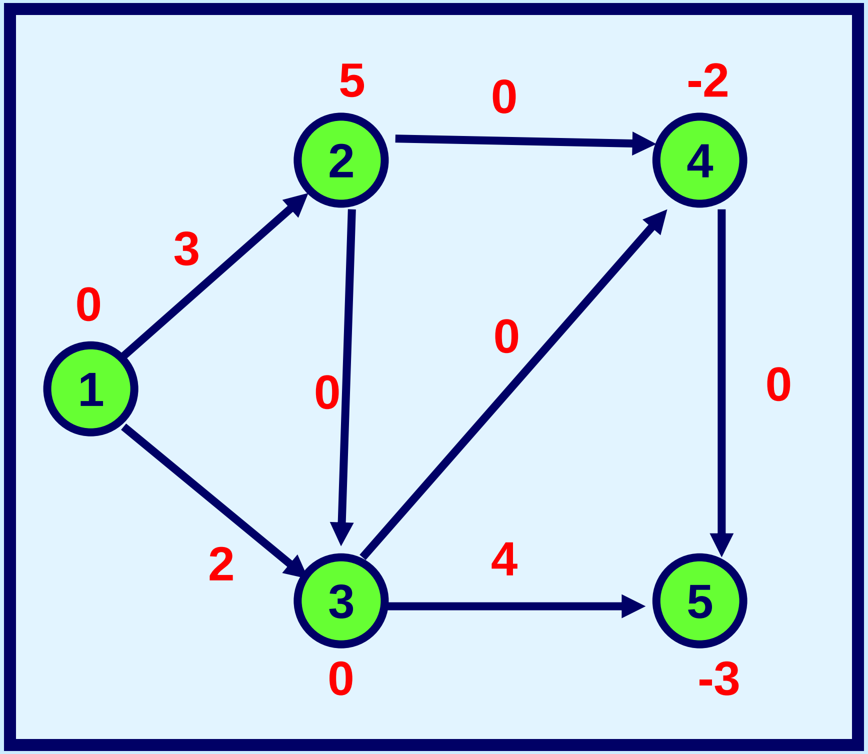


Supplies/demands and capacities

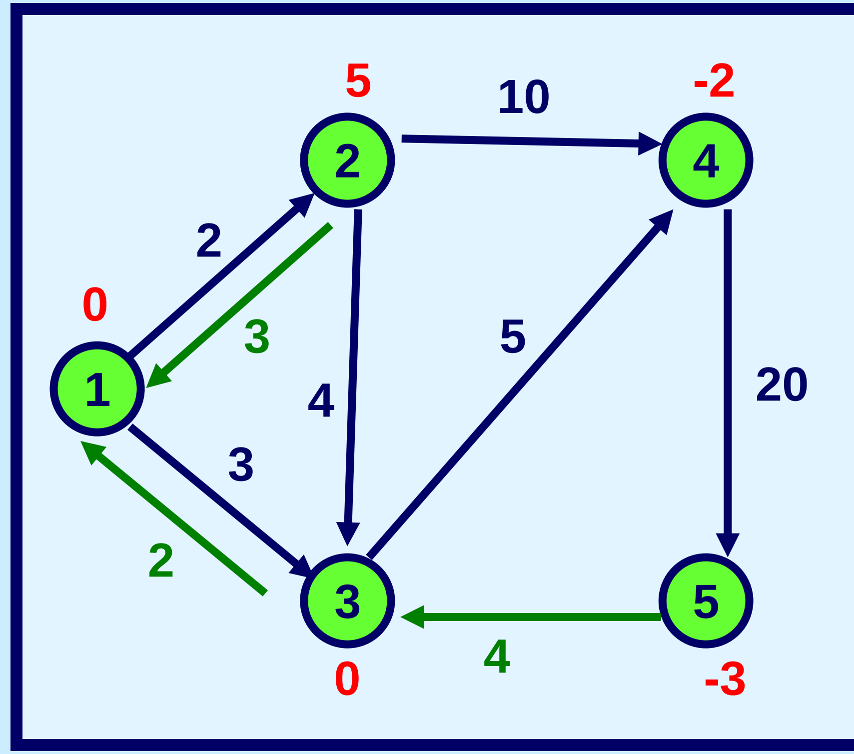


A pseudo-flow and excesses

Pseudo-flows and the residual network



A pseudo-flow and excesses



The residual network

The ***infeasiblity*** ***of the pseudo-flow*** **is** $\sum_{e(i)>0}$ **e(i).** **e.g., 5.**

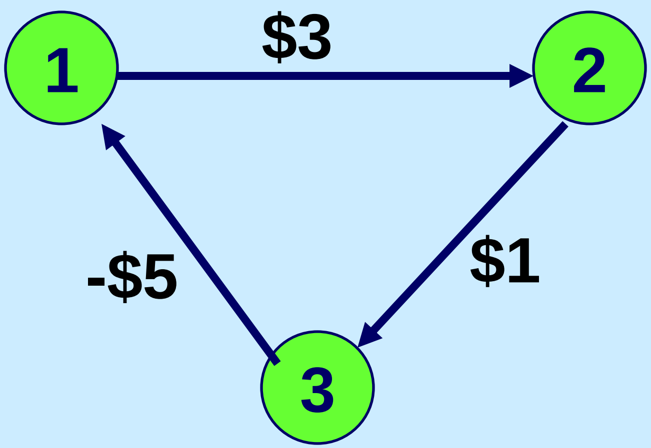
Reduced Costs in G(x)

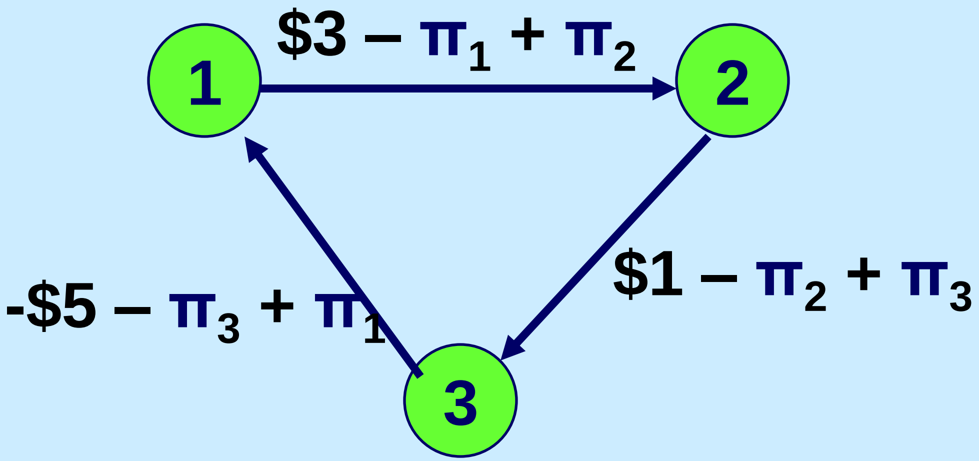
Let π_i denote the **node potential** for node i.

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

For unit of flow out of node i, subtract π_i from the cost.

For unit of flow into node j, add π_j to the cost.





Optimality Conditions

Theorem. Suppose that x^* is flow and π^* is a vector of node potential. Then x^* and π^* are optimal if

1. the flow x^* is feasible and
2. $c^{\pi^*}_{ij} \geq 0$ for all $(i, j) \in G(x^*)$.

   The second property is often called **dual feasibility**.

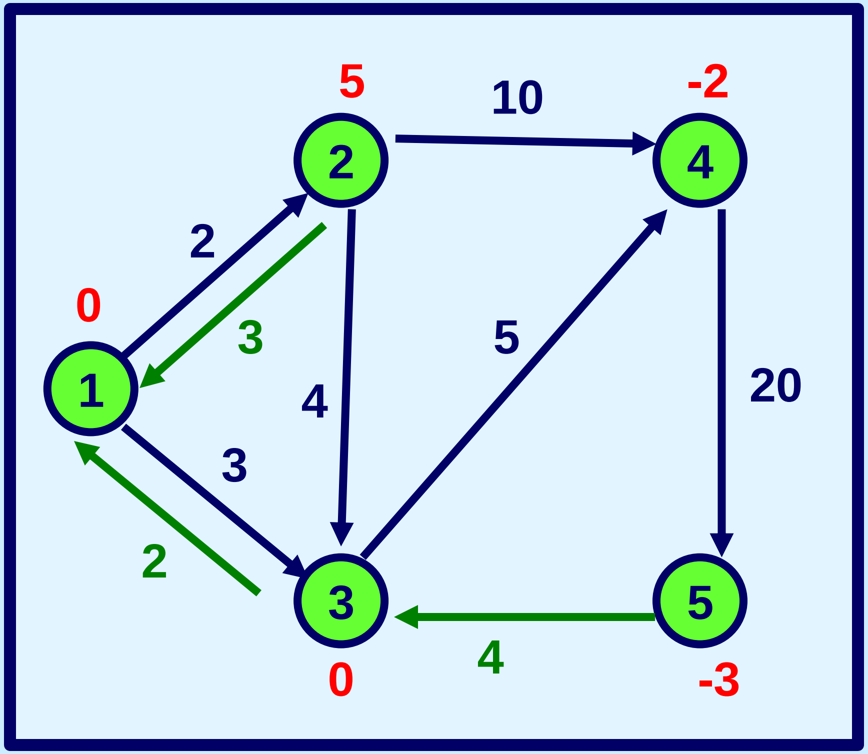
Optimality Conditions

The cycle canceling algorithm starts with a feasible flow and maintains a feasible flow at each iteration. It terminates with dual feasibility.

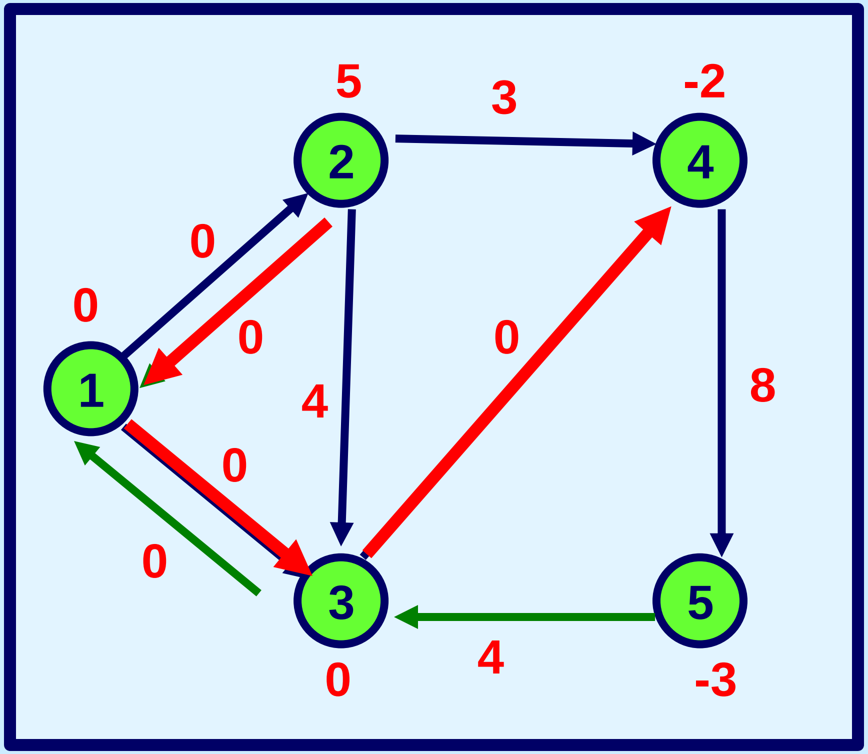
The successive shortest path algorithm starts with (and maintains) an infeasible flow x and node potentials π that are dual feasible. It stops when it obtains a flow that is feasible.

It's main subroutine involves sending flow along a path from a node with excess to a node with deficit.

Augmenting paths



Capacities and excesses in G(x)

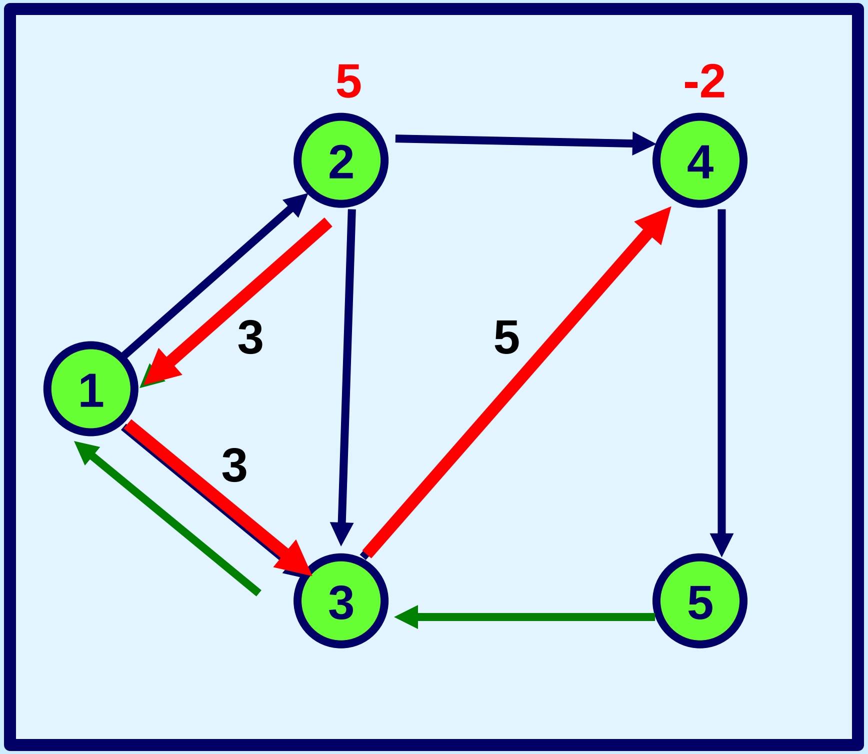


Excesses and reduced costs in G(x)

A path P from i to j in G(x) is augmenting if

$e(i) > 0$, $e(j) < 0$, and $\forall$ $(v, w) \in P$, $c^{\pi}_{vw} = 0$.

Augmenting along a path



The augmenting path with residual capacities.

The capacity of an augmenting path P from node i to node j is

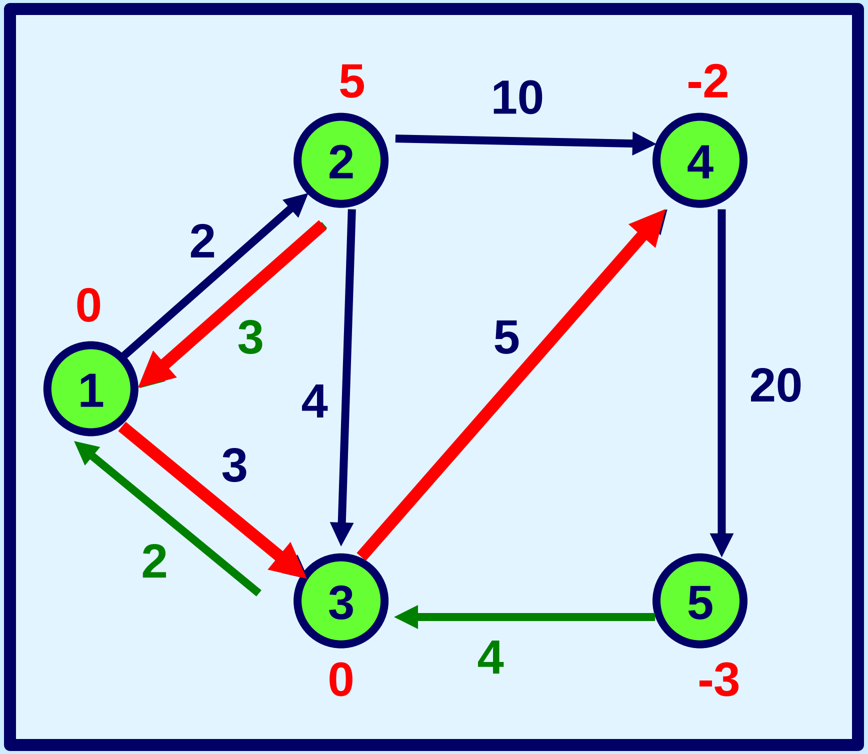
$\min \{ e(i), -e(j), \min_{(v,w)\in P} r_{vw}\}$.

In this case, the capacity is

min {5, 2, 3} = 2

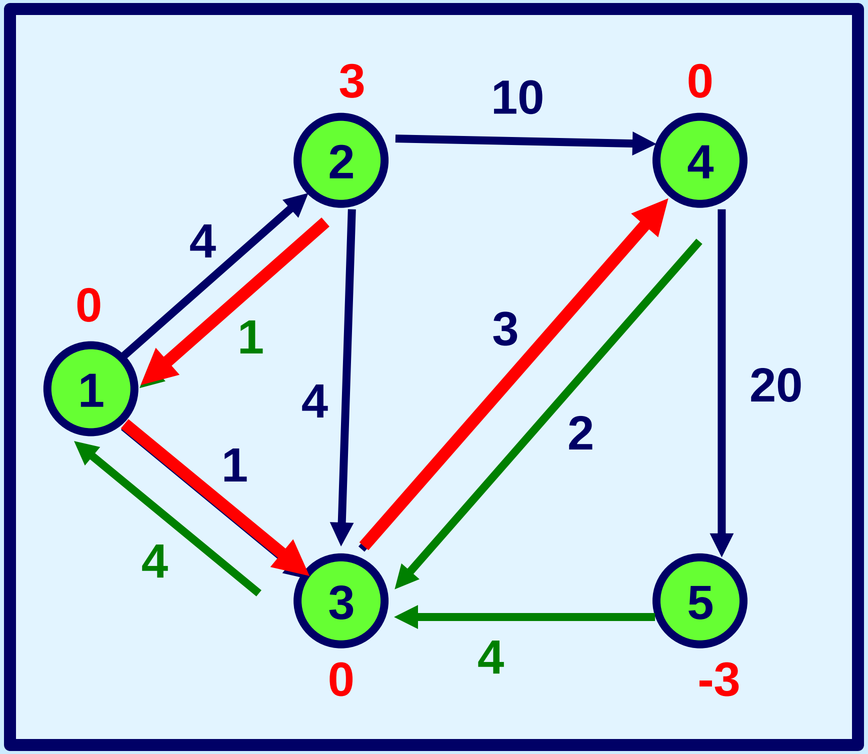
To augment along a path P is to send δ units of flow from i to j in P, where δ = capacity(P).

Before and after the augmentation



Capacities and excesses in G(x)

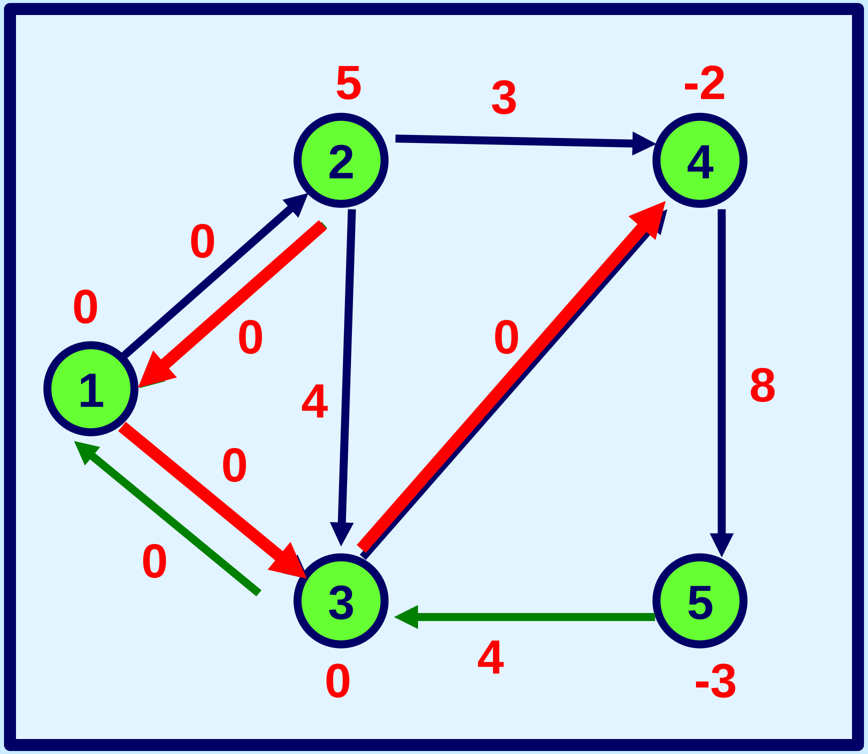
before augmenting on P



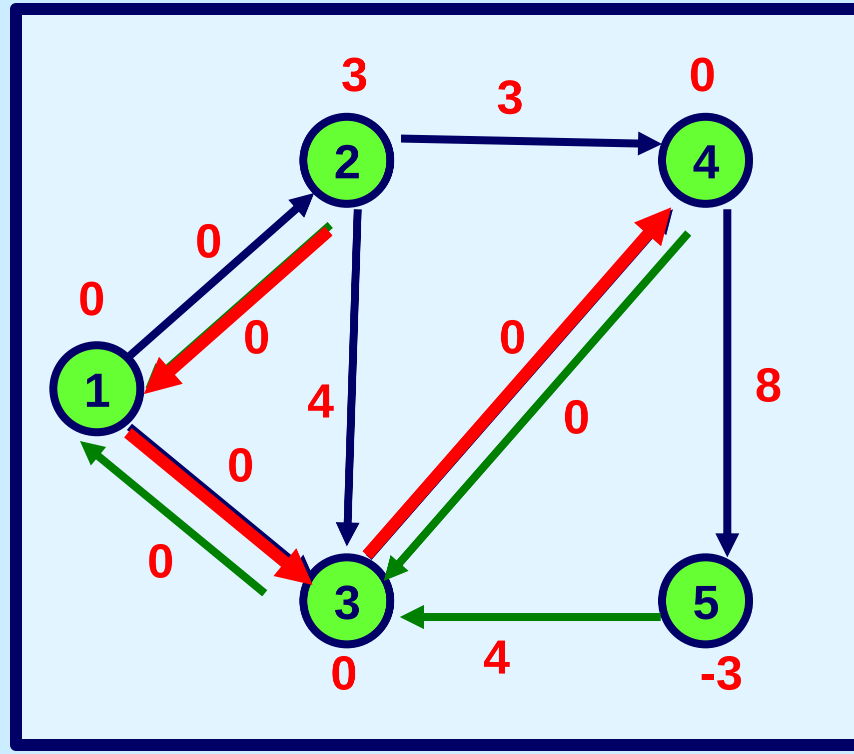
Capacities and excesses in G(x)

after augmenting on P

The effect of augmentations



Excesses and reduced costs before the augmentation



Excesses and reduced costs after the augmentation

Augmentations reduce the infeasiblity by at least 1.

Augmentations maintain dual feasibility. Any arc added to G(x) is the reversal of an arc with a reduced cost of 0.

The Successive Shortest Path Algorithm

Initialize with a dual feasible pseudo-flow (x, π)

while x is infeasible do

select a node i with e(i) > 0

adjust the node potentials so that it remains dual feasible and there is an augmenting path P from node i to a node j with e(j) < 0;

(if the adjustment is not possible, then there is no feasible flow for the problem)

augment along path P

update the residual network and excesses;

The SSP Algorithm

The next steps

- **Show how to find an initial dual feasible pseudoflow.**
- **Show how to adjust the node potentials**
- **Show that the problem is infeasible if there is no possible adjustment of the node potentials**
- **Proof of finiteness**
- **Proof of correctness of the algorithm**

Finding an initial dual feasible pseudoflow

Let $A' = \{(i, j) : u_{ij} = \infty\}$.

We first choose π so that $c^{\pi}_{ij} \geq 0$ for $(i, j) \in A'$.

Step 1. Let $C = 1 + \max \{ | c_{ij} | : (i, j) \in A'\}$.

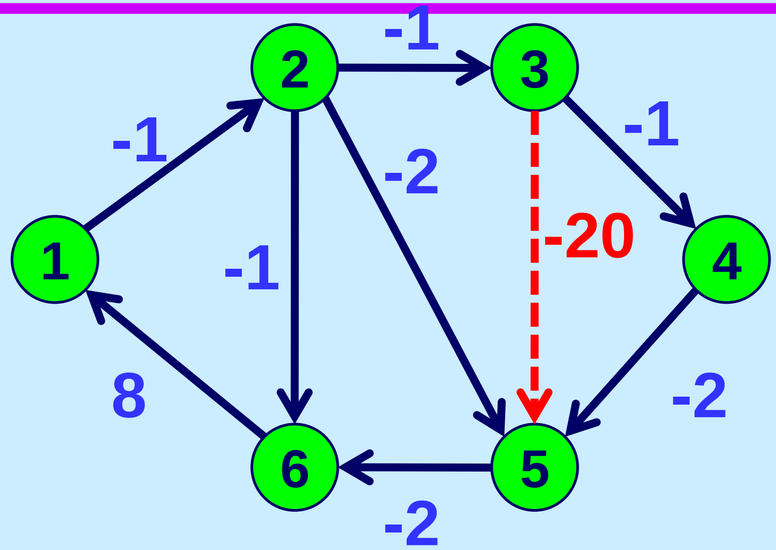
If there is no directed path from 1 to j in A', then add arc (1, j) to A' with cost nC.

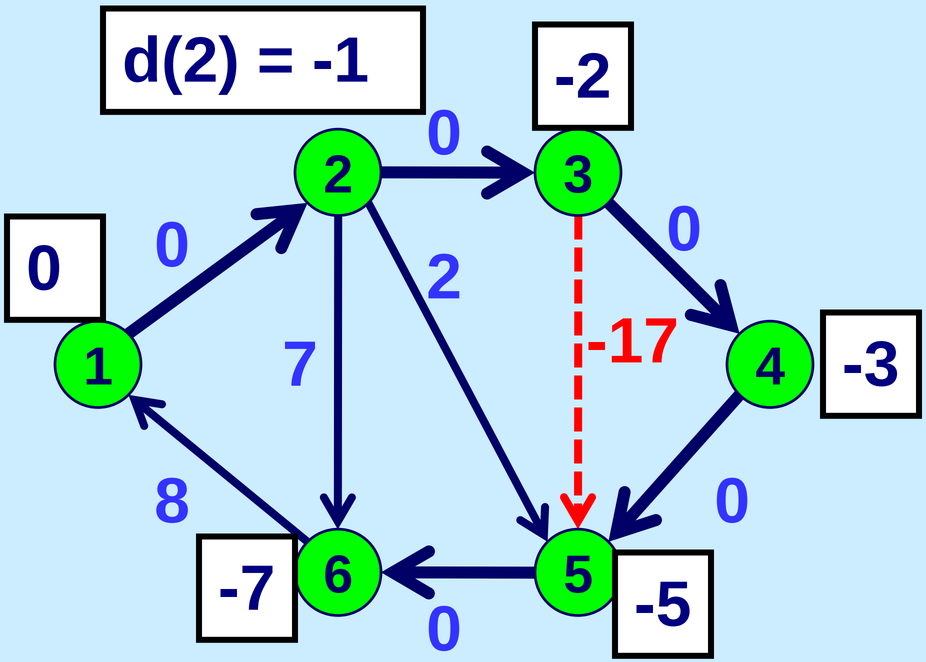
Step 2. Use a shortest path algorithm to find the shortest path length from 1 to node j in A'. If there is a negative cost cycle, quit. One can send an infinite amount of flow around a negative cost cycle.

Step 3. Let $\pi(j) = -d(j)$ for all j.

Step 4. For each arc $(i, j) \in A \backslash A'$ with $c^{\pi}_{ij} < 0$, let $x_{ij} = u_{ij}$.

Finding an initial dual feasible pseudoflow





A' = solid lines. They have infinite capacity. The costs are on the arcs.

The dotted red line has a capacity of 10.

The numbers next to nodes are the shortest path distances from node 1.

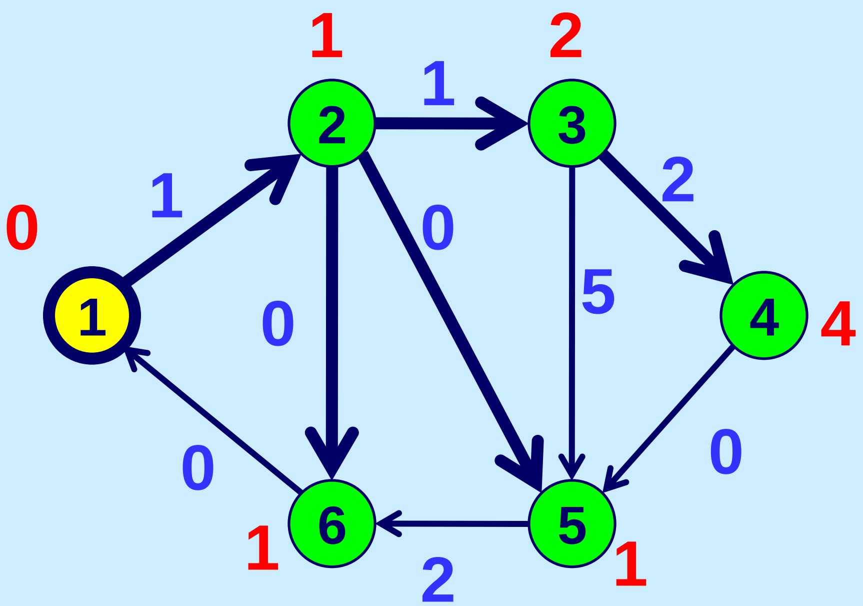
$c^{\pi}_{ij} = c_{ij} + d(i) – d(j)$

Adjusting node potentials

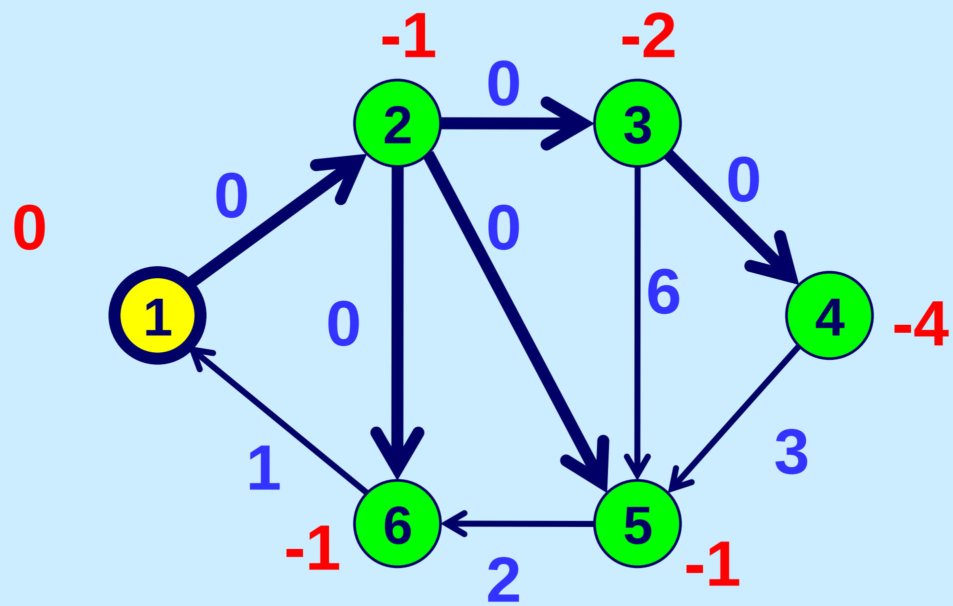
Select node i with e(i) > 0.

Let d(j) = shortest path from node i to node j in G(x).

Replace π_j by π_j - d(j) for all j ∈ N.



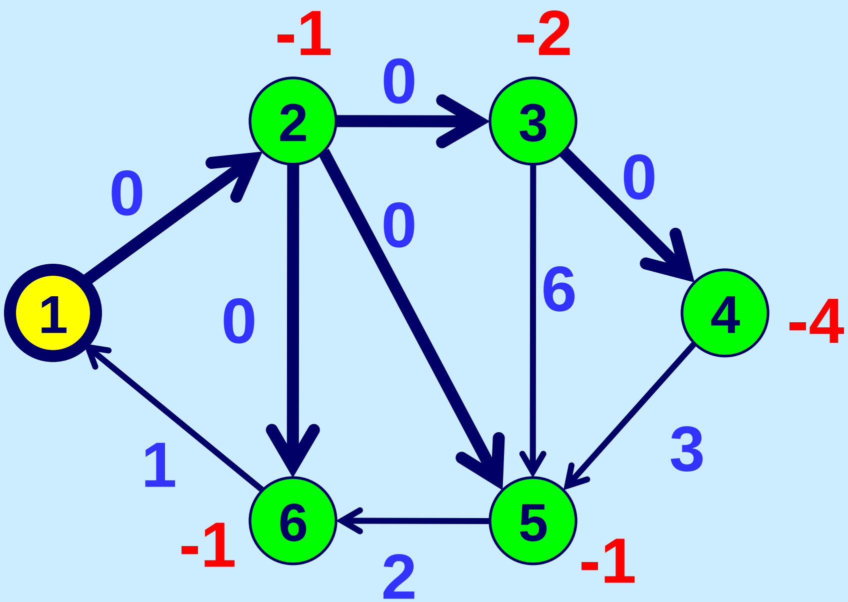
G(x) with reduced costs on arcs and shortest path distances. e(1) > 0.

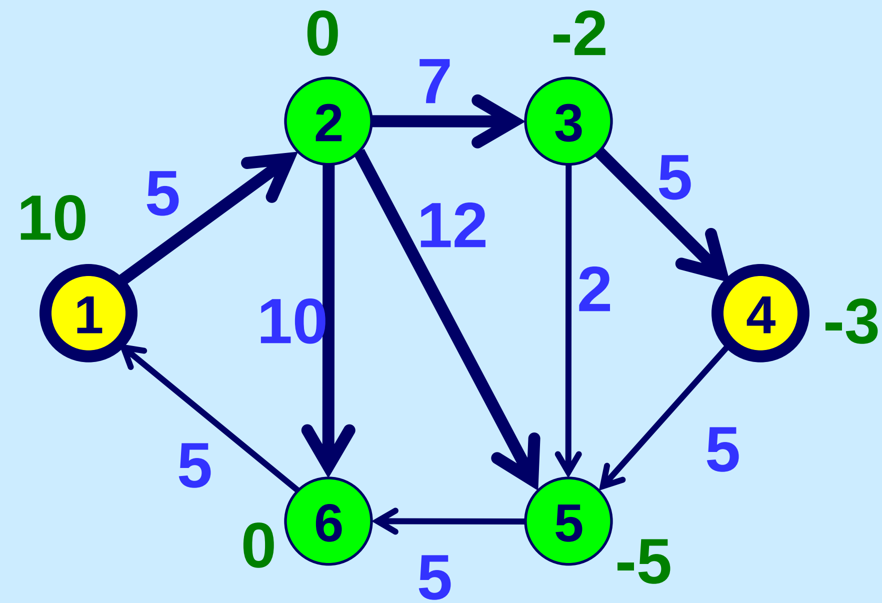


Node potentials π_j - d(j)and reduced costs

On Node Potentials

After adjusting node potentials, there is an path with 0-reduced cost arcs in G(x) from node i to each other node (assuming that there is a path from node 1).





Node potentials π_j - d(j)and reduced costs

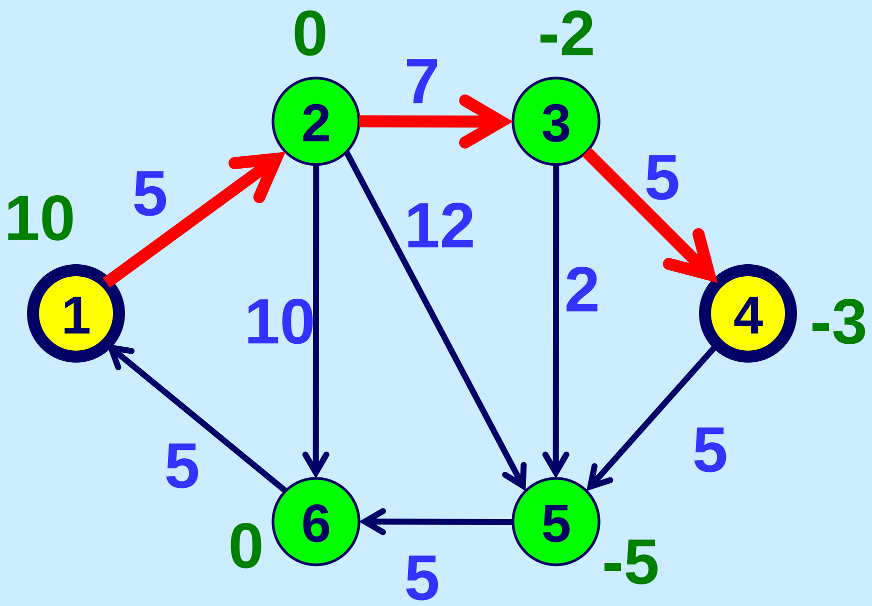
Excesses and capacities.

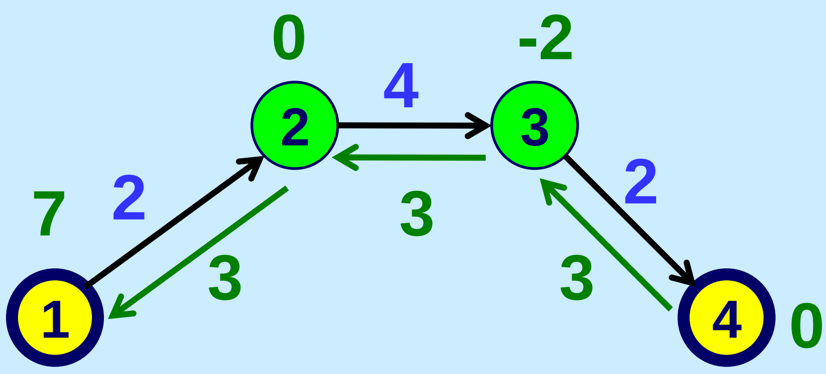
Augment along a path from node i

Select a node j with e(j) < 0 that is reachable from node i.

Let P be the path from node i to node j.

Send min {e(i), -e(j), min {r_{vw} : (v, w) ∈ P} } units in P.





Excesses and capacities.

Send min {10, 3, 5} units of flow from 1 to 4. Update residual capacities and excesses.

Proof of finiteness and correctness

Finiteness comes from the fact that the infeasibility is always integer and nonnegative, and it decreases by at least one at each iteration.

The algorithm usually terminates a feasible flow (and also with dual feasibility) and is thus optimal.

It can also terminate if there is no path from a node i with e(i) > 0 to any node with negative excess. In the latter case, the problem is infeasible.

Analysis of the Successive Shortest Path Algorithm

Each augmentation reduces the infeasibility by at least 1.

Initial infeasibility is at most $mU_{max} + nb_{max}$

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

Running time: $O((mU_{max} + nb_{max})\ m \log nC)$

Mental Break

Thomas Edison invented the light bulb. The invention helped him deal with a personal (non-professional) issue. What was it?

He was afraid of the dark

Apollo 11 did not have much fuel when it landed back on earth. How much fuel remained?

About 20 seconds worth.

How long does it take all of the stars of the Milky Way galaxy to revolve around the center of the galaxy?

Around 200 million years.

Mental Break

A neutron star is so dense that it can quickly spin on its axis an entire revolution without tearing itself apart. How fast?

It can spin in 1/30th of a second.

A full moon is brighter than a half moon. How much brighter?

9 times as bright.

In 1961-2, MIT students developed the first interactive computer game. What was it called?

Spacewar.

The Capacity Scaling Algorithm

- A polynomial time variant of the successive shortest path algorithm.
- Relies on the generic scaling approach.
- Send flow from a node i with $e(i) \geq \Delta/2$.
- Send flow along paths with capacity $\geq \Delta/2$.

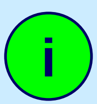
Δ-optimality

A pseudoflow x and node potentials π are called **Δ-optimal** if

1. $e(j) < \Delta$ for all j
2. $\forall (i, j)$, if $c^{\pi}_{ij} < 0$, then $r_{ij} < \Delta$

Suppose **Δ = 64**. Which of the following are permitted if the pseudoflow is Δ-optimal?

$e(i) = 64$



not permitted

$e(j) = -128$

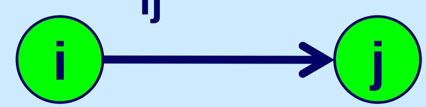
permitted

$r_{ij} = 64$

$c^{\pi}_{ij} = -1$

not permitted

$r_{ij} = 63$



$c^{\pi}_{ij} = -6300$

permitted

The Capacity Scaling Algorithm

Capacity Scaling Algorithm

let x, π be an initial dual feasible pseudo-flow and node potentials

let $e^* = 1 + \max \{e(i) : i \in N \}$;

$\Delta := 2^K$, where $K = \lceil \log e^* \rceil$

while $\Delta > 1$ **do**

y := ImproveApprox(x, Δ)

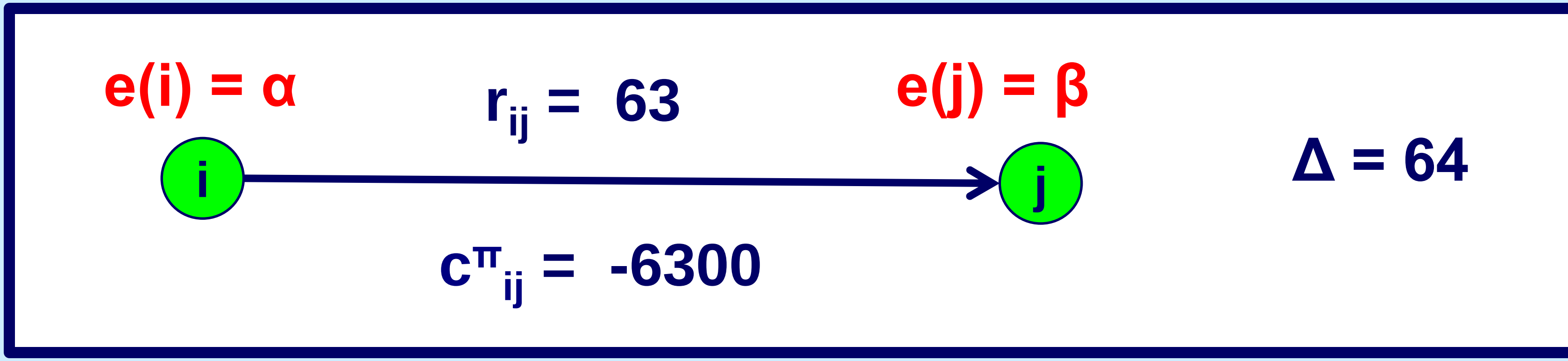
x := y

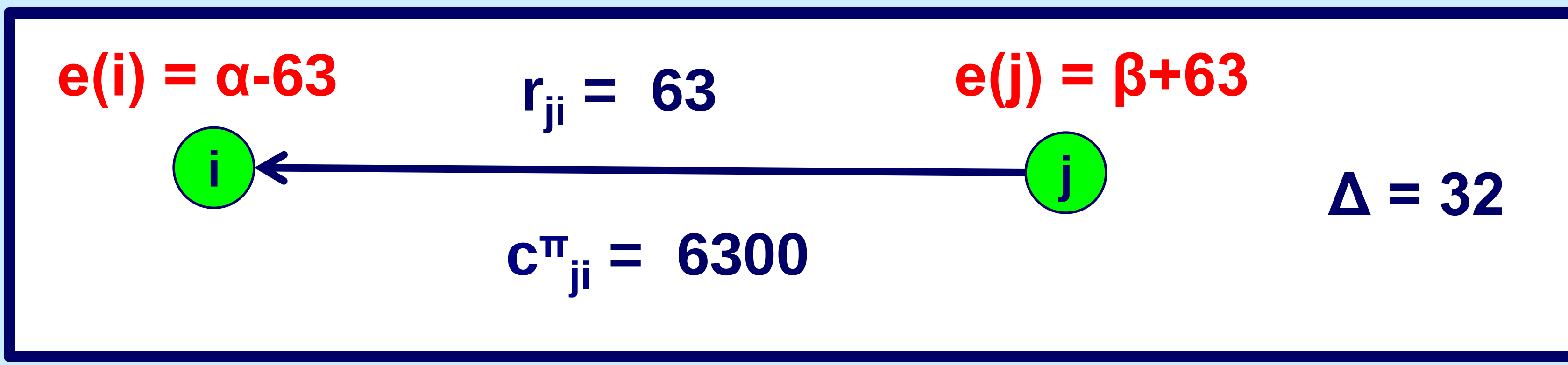
$\Delta := \Delta/2$

ImproveApprox(x, Δ) converts a Δ-optimal pseudo-flow into a Δ/2 optimal pseudo-flow.

Initializing ImproveApprox

For any arc (i, j) with $c^{\pi}_{ij} < 0$ and $r_{ij} \geq \Delta/2$, send r_{ij} units of flow in (i, j).





After this operation, if $c^{\pi}_{vw} < 0$, then $r_{vw} < \Delta/2$ for all $(v, w) \in G(x)$.

ImproveApprox(x, Δ)

for any arc (i, j) with $c^{\pi}_{ij} < 0$ and $r_{ij} \geq \Delta/2$, send r_{ij} units of flow in (i, j);

update r and e( );

while there is a node i with $e(i) \geq \Delta/2$ **do**

let $G(x, \Delta/2) = \{ (i, j) \text{ with } r_{ij} \geq \Delta/2\}$;

select a node i with $e(i) \geq \Delta/2$;

let d(j) = min cost of a path from i to j in $G(x, \Delta/2)$; replace π by π - d( )

augment along an augmenting path from i to some node j with $e(j) < 0$ (assume that the path exists).

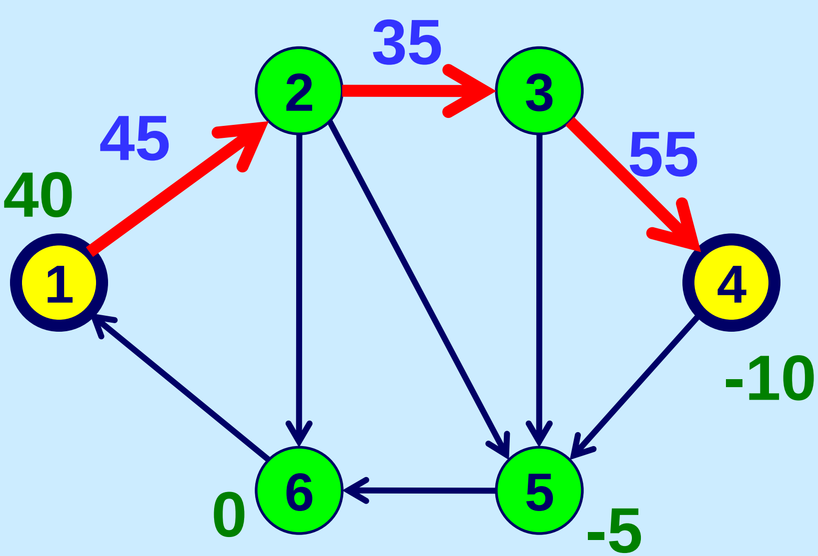
update the residual network and excesses;

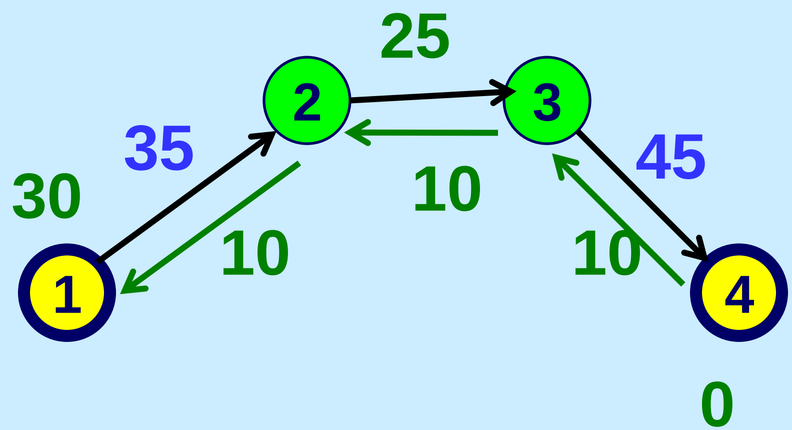
An augmentation when $\Delta = 64$

- $e(1) \geq 32$
- $e(4) < 0$

- $r_{12} \geq 32$
- $r_{23} \geq 32$
- $r_{34} \geq 32$

- $c^{\pi}_{12} = 0$
- $c^{\pi}_{23} = 0$
- $c^{\pi}_{34} = 0$





After an augmentation from i to j, either the infeasibility decreases by at least $\Delta/2$, or $e(j) \geq 0$.

Number of Augmentations

- $\sum e(i) < n\Delta$ at the beginning of the Δ-scaling phase

$e(j) = \beta+63$

$\Delta = 64$

Recall that we send r_{ij} units of flow in arcs with negative reduce costs. Each saturating push increases e(j) by at most $\Delta - 1$.

- $\sum e(i) < n\Delta + m\Delta$ after the saturating pushes at the beginning of the Δ-scaling phase.
- Each augmentation
  - (i) eliminates a node with deficit or
  - (ii) reduces $\sum e(i)$ by at least $\Delta/2$

(i) occurs fewer than n times

(ii) occurs fewer than 2m + 2n times.

Running time analysis

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

The number of scaling phases is O(log M), where

$M < m\ u_{ma}x + n\ b_{max}$.

The running time per augmentation is the time to solve a single shortest path problem using Dijkstra's algorithm.

The total running time is O((m log M)(m + n log nC)).

Summary

The successive shortest path algorithm. Maintains dual feasibility and moves towards a feasible flow.

Capacity scaling algorithm: a scaling variant of the successive shortest path problem.

The capacity scaling algorithm was the first polynomial time algorithm for the min cost flow problem.

Edmonds-Karp 1972.

Some Final Remarks

The number of augmentations can be reduced from $O(m \log M)$ to $O(m \log n)$ Orlin 1988.

The algorithm is slightly different, and the analysis is more complex.

Next lectures: Network flow applications and LP, and Network simplex algorithm.

15.082J / 6.855J / ESD.78J Network Optimization

Fall 2010