Network Simplex

- Algorithm:
  1. compute the dual vector p
  2. compute the reduced costs $\bar{c}$; STOP if $\bar{c} \geq 0$
  3. $B :=$ set of backward edges of the cycle formed by an edge e with $\bar{c}_e < 0$; STOP if $B = \emptyset$
  4. push flow around this cycle and update basis and basic solution

- Degeneracy: if some tree edge has flow 0
  - → θ^* could be 0
  - → tree changes, but not the flow
  - → cycling can happen!
  - → anticycling rules!

- Running time (of one iteration):
  - compute p : $O(n)$
  - compute $\bar{c}$: $O(m)$
  - cycle : $O(n)$
  - updating the flow : $O(n)$

$$\overline{}$$
$$O(m) \qquad \text{(since } m \geq n-1\text{)}$$

Important: data structures that support access to edges & nodes of graph and trees in constant time

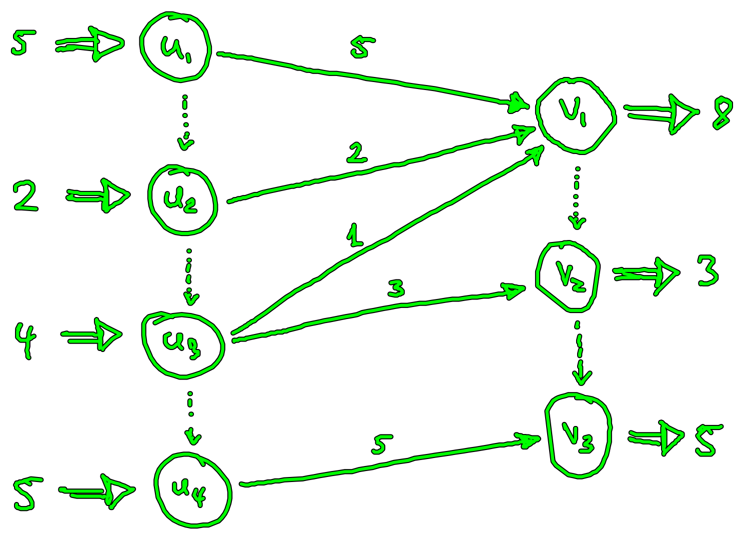
The number of iteration can be $\Omega(2^n)$, but is most often $O(n)$.

- Finding an initial basic feasible solution:
  - Add auxiliary edges from each source to each sink (if not already there) with cost $n \cdot \max\{c_e \mid e \in E\} + 1$

Let $u_1, \dots, u_\ell$ be the source nodes and $v_1, \dots, v_k$ the sink nodes.

Put flow on all auxiliary edges „from top to bottom":

Ex.:



$$\sum_{v \in V} b_v = 0$$

- Possible outcomes of the algorithm
  - directed cycle with reduced cost < 0 (step 3)
    $\longrightarrow$ unbounded problem
  - optimal solution (step 2) <u>with</u> auxiliary edges
    $\longrightarrow$ (original) problem is infeasible
  - optimal solution (step 2) without auxiliary edges
    $\longrightarrow$ problem is solved and output is an optimal primal solution together with an optimal dual solution !

- Capacitated problems:

Suppose we have bounds on the flow of an edge e:

$$0 \le d_e \le f_e \le u_e \qquad \forall e \in E$$

Partition the edges not in the tree into two sets D and U with

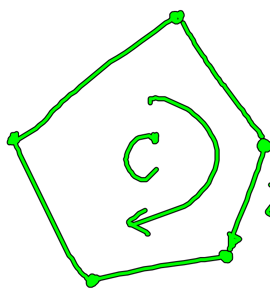
$$f_e = d_e \quad \forall e \in D \quad , \qquad f_e = u_e \quad \forall e \in U$$

Basis change: Compute $\bar{c}_e$ as before for all $e \in D \cup U$. Then

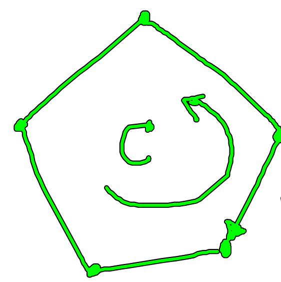
$$\left.\begin{array}{r}\text{Either}\\ \text{Or}\end{array}\right\} \text{find an edge } e \left\{\begin{array}{l}\in D\\ \in U\end{array}\right\} \text{ with } \left\{\begin{array}{l}\bar{c}_e < 0\\ \bar{c}_e > 0\end{array}\right\}.$$

This gives a cycle C.

Orient C such that e is a $\left\{\begin{array}{l}\text{forward}\\ \text{backward}\end{array}\right\}$ arc



$\bar{c}_e < 0$, $f_e = d_e$ $(e \in D)$



$\bar{c}_e > 0$, $f_e = u_e$ $(e \in U)$

$F :=$ set of forward edges of C, $B :=$ set of backward edges of C

$$\Theta^* := \min \left\{ \min_{e' \in B} \{ f_{e'} - d_{e'} \}, \min_{e' \in F} \{ u_{e'} - f_{e'} \} \right\}$$

$\longrightarrow$ At least one edge e' will have it's flow value set either to d_e or u_e.

If $e' \in T$ then e' leaves T and goes to either D or U and e replaces e' in T

If $e' = e$ then e changes from D to U or vice versa.

$\longrightarrow$ degeneracy: if $\Theta^* = 0$ then the basis changes, but not the flow

$\uparrow$

T and either D or U