Mincost Flow

Problem Definition:

$$\min \quad \sum_e c(e)f(e)$$
$$\text{s.t.} \quad \forall e \in E: \quad 0 \le f(e) \le u(e)$$
$$\forall v \in V: \quad f(v) = b(v)$$

- $G = (V, E)$ is a directed graph.
- $u : E \to \mathbb{R}_0^+ \cup \{\infty\}$ is the capacity function.
- $c : E \to \mathbb{R}$ is the cost function (note that $c(e)$ may be negative).
- $b : V \to \mathbb{R}$, $\sum_{v \in V} b(v) = 0$ is a demand function.

Solve Maxflow Using Mincost Flow
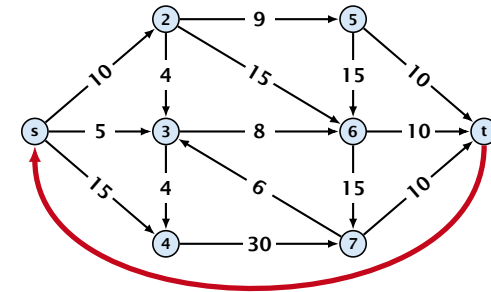
- Given a flow network for a standard maxflow problem.
- Set $b(v) = 0$ for every node. Keep the capacity function u for all edges. Set the cost $c(e)$ for every edge to 0.
- Add an edge from t to s with infinite capacity and cost -1.
- Then, $\mathrm{val}(f*) = -\mathrm{cost}(f_{\min})$, where $f*$ is a maxflow, and $f_{\min}$ is a mincost-flow.

Solve Maxflow Using Mincost Flow

Solve decision version of maxflow:

- Given a flow network for a standard maxflow problem, and a value k.
- Set $b(v) = 0$ for every node apart from s or t. Set $b(s) = -k$ and $b(t) = k$.
- Set edge-costs to zero, and keep the capacities.
- There exists a maxflow of value at least k if and only if the mincost-flow problem is feasible.

Generalization

Our model:

$$\min \quad \sum_e c(e)f(e)$$
$$\text{s.t.} \quad \forall e \in E: \quad 0 \le f(e) \le u(e)$$
$$\forall v \in V: \quad f(v) = b(v)$$

where $b : V \to \mathbb{R}$, $\sum_v b(v) = 0$; $u : E \to \mathbb{R}_0^+ \cup \{\infty\}$; $c : E \to \mathbb{R}$;

A more general model?

$$\min \quad \sum_e c(e)f(e)$$
$$\text{s.t.} \quad \forall e \in E: \quad \ell(e) \le f(e) \le u(e)$$
$$\forall v \in V: \quad a(v) \le f(v) \le b(v)$$

where $a : V \to \mathbb{R}$, $b : V \to \mathbb{R}$; $\ell : E \to \mathbb{R} \cup \{-\infty\}$, $u : E \to \mathbb{R} \cup \{\infty\}$; $c : E \to \mathbb{R}$;

Generalization

Differences

▶ Flow along an edge e may have non-zero lower bound $\ell(e)$.

▶ Flow along e may have negative upper bound $u(e)$.

▶ The demand at a node v may have lower bound $a(v)$ and upper bound $b(v)$ instead of just lower bound = upper bound = $b(v)$.

Reduction I

$$
\begin{aligned}
\min \quad & \textstyle\sum_e c(e)f(e) \\
\text{s.t.} \quad & \forall e \in E: \ \ell(e) \le f(e) \le u(e) \\
& \forall v \in V: \ a(v) \le f(v) \le b(v)
\end{aligned}
$$

We can assume that $a(v) = b(v)$:

Add new node r.

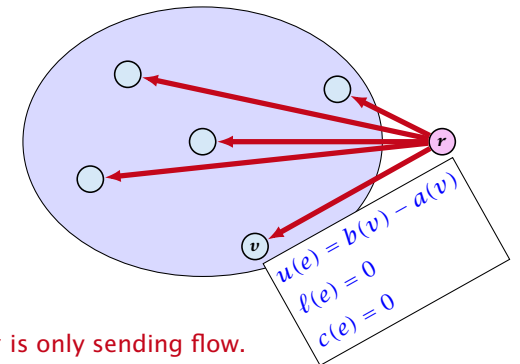Add edge (r, v) for all $v \in V$.

Set $\ell(e) = c(e) = 0$ for these edges.

Set $u(e) = b(v) - a(v)$ for edge (r, v).

Set $a(v) = b(v)$ for all $v \in V$.
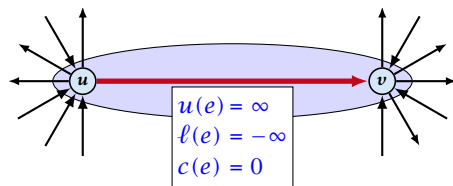
Set $b(r) = -\sum_{v \in V} b(v)$.

$-\sum_v b(v)$ is negative; hence r is only sending flow.

$$
\begin{aligned}
u(e) &= b(v) - a(v) \\
\ell(e) &= 0 \\
c(e) &= 0
\end{aligned}
$$

Reduction II

$$
\begin{aligned}
\min \quad & \textstyle\sum_e c(e)f(e) \\
\text{s.t.} \quad & \forall e \in E: \ \ell(e) \le f(e) \le u(e) \\
& \forall v \in V: \ f(v) = b(v)
\end{aligned}
$$

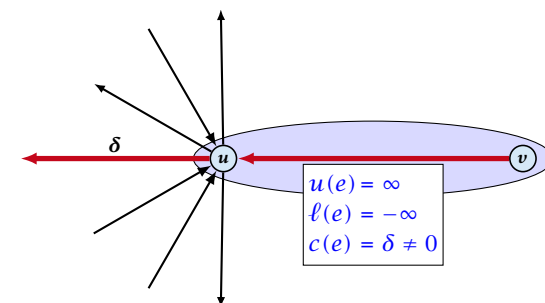**We can assume that either $\ell(e) \ne -\infty$ or $u(e) \ne \infty$:**

$$
\begin{aligned}
u(e) &= \infty \\
\ell(e) &= -\infty \\
c(e) &= 0
\end{aligned}
$$

If $c(e) = 0$ we can contract the edge/identify nodes u and v.

If $c(e) \ne 0$ we can transform the graph so that $c(e) = 0$.

Reduction II

We can transform any network so that a particular edge has cost $c(e) = 0$:

δ

$$
\begin{aligned}
u(e) &= \infty \\
\ell(e) &= -\infty \\
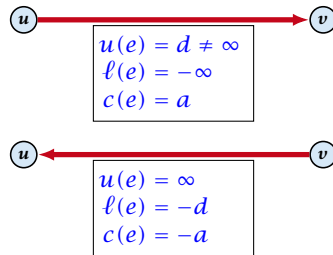c(e) &= \delta \ne 0
\end{aligned}
$$

Additionally we set $b(u) = 0$.

Reduction III

$$\begin{aligned}\min \quad & \sum_e c(e)f(e) \\ \text{s.t.} \quad & \forall e \in E: \ \ell(e) \leq f(e) \leq u(e) \\ & \forall v \in V: \ f(v) = b(v)\end{aligned}$$

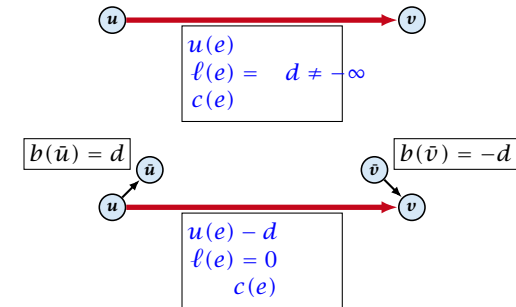**We can assume that $\ell(e) \neq -\infty$:**

Replace the edge by an edge in opposite direction.

Reduction IV

$$\begin{aligned}\min \quad & \sum_e c(e)f(e) \\ \text{s.t.} \quad & \forall e \in E: \ \ell(e) \leq f(e) \leq u(e) \\ & \forall v \in V: \ f(v) = b(v)\end{aligned}$$

We can assume that $\ell(e) = 0$:

The added edges have infinite capacity and cost $c(e)/2$.

Applications

Caterer Problem

▶ She needs to supply r_i napkins on N successive days.

▶ She can buy new napkins at p cents each.

▶ She can launder them at a fast laundry that takes m days and cost f cents a napkin.

▶ She can use a slow laundry that takes $k > m$ days and costs s cents each.

▶ At the end of each day she should determine how many to send to each laundry and how many to buy in order to fulfill demand.
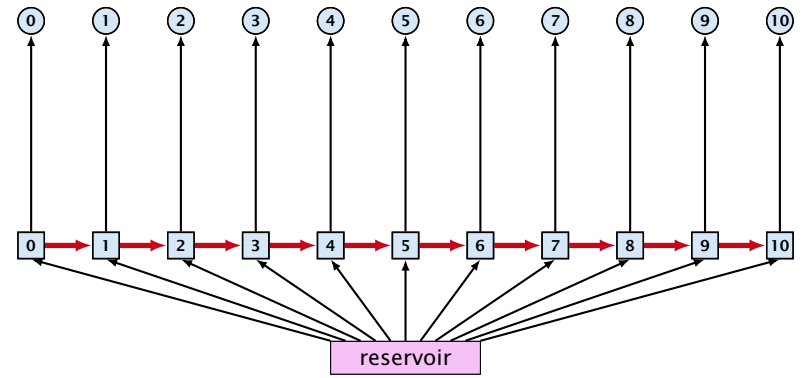
▶ Minimize cost.
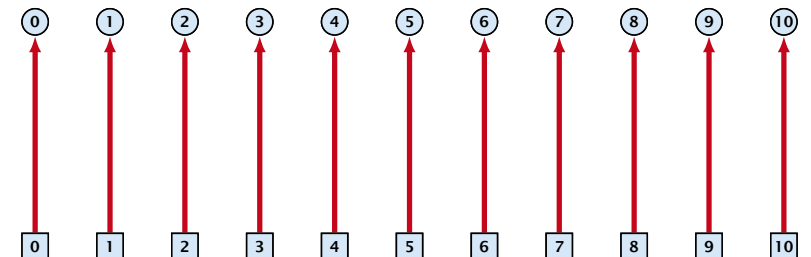
day edges:

upper bound: $u(e_i) = \infty$;
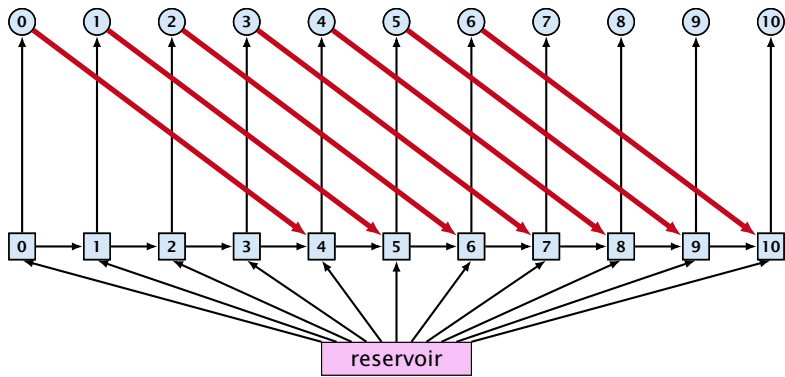lower bound: $\ell(e_i) = r_i$;
cost: $c(e) = 0$
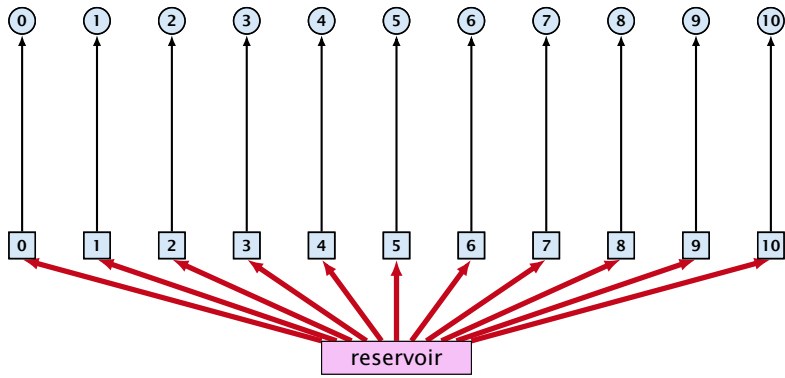
buy edges:

upper bound: $u(e_i) = \infty$;
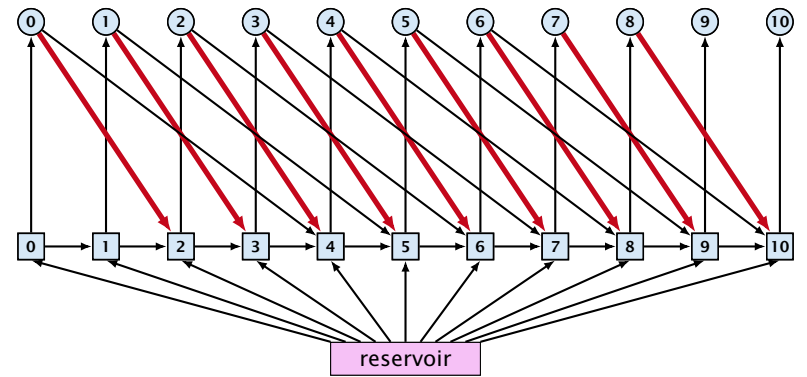lower bound: $\ell(e_i) = 0$;
cost: $c(e) = p$

forward edges:

upper bound: $u(e_i) = \infty$;
lower bound: $\ell(e_i) = 0$;
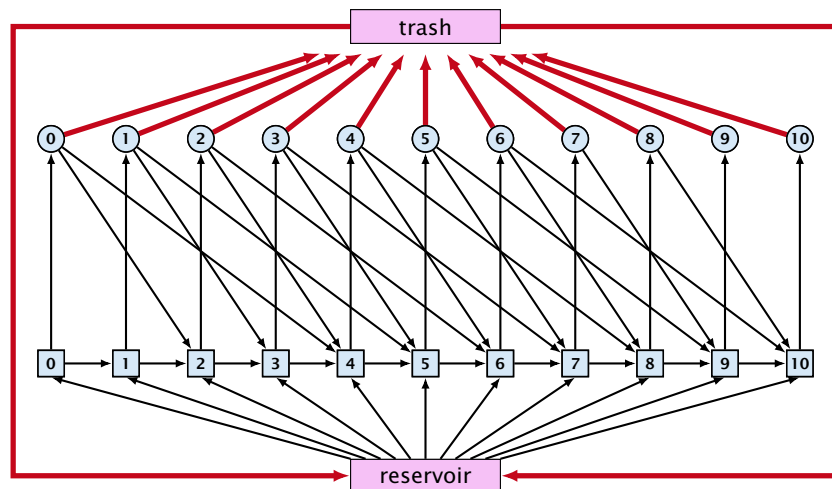cost: $c(e) = 0$

slow edges:

upper bound: $u(e_i) = \infty$;
lower bound: $\ell(e_i) = 0$;
cost: $c(e) = s$

fast edges:

upper bound: $u(e_i) = \infty$;
lower bound: $\ell(e_i) = 0$;
cost: $c(e) = f$

trash

0 1 2 3 4 5 6 7 8 9 10

0 → 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10

reservoir

trash edges:
upper bound: $u(e_i) = \infty$;
lower bound: $\ell(e_i) = 0$;
cost: $c(e) = 0$

Residual Graph

Version A:
The residual graph G' for a mincost flow is just a copy of the graph G.

If we send $f(e)$ along an edge, the corresponding edge e' in the residual graph has its lower and upper bound changed to $\ell(e') = \ell(e) - f(e)$ and $u(e') = u(e) - f(e)$.

Version B:
The residual graph for a mincost flow is exactly defined as the residual graph for standard flows, with the only exception that one needs to define a cost for the residual edge.

For a flow of z from u to v the residual edge (v, u) has capacity z and a cost of $-c((u, v))$.

---

10 Mincost Flow

A circulation in a graph $G = (V, E)$ is a function $f : E \to \mathbb{R}^+$ that has an excess flow $f(v) = 0$ for every node $v \in V$.

A circulation is feasible if it fulfills capacity constraints, i.e., $f(e) \le u(e)$ for every edge of G.

Lemma 73
A given flow is a mincost-flow if and only if the corresponding residual graph G_f does not have a feasible circulation of negative cost.

⇒ Suppose that g is a feasible circulation of negative cost in the residual graph.

Then $f + g$ is a feasible flow with cost $\mathrm{cost}(f) + \mathrm{cost}(g) < \mathrm{cost}(f)$. Hence, f is not minimum cost.

⇐ Let f be a non-mincost flow, and let f^* be a min-cost flow. We need to show that the residual graph has a feasible circulation with negative cost.

Clearly $f^* - f$ is a circulation of negative cost. One can also easily see that it is feasible for the residual graph. (after sending $-f$ in the residual graph (pushing all flow back) we arrive at the original graph; for this f^* is clearly feasible)

For previous slide:
$g = f^* - f$ is obtained by computing $\Delta(e) = f^*(e) - f(e)$ for every edge $e = (u,v)$. If the result is positive set $g((u,v)) = \Delta(e)$ and $g((v,u)) = 0$. Otherwise set $g((u,v)) = 0$ and $g((v,u)) = -\Delta(e)$.

10 Mincost Flow

Lemma 74
A graph (without zero-capacity edges) has a feasible circulation of negative cost if and only if it has a negative cycle w.r.t. edge-weights $c : E \to \mathbb{R}$.

Proof.
▶ Suppose that we have a negative cost circulation.
▶ Find directed cycle only using edges that have non-zero flow.
▶ If this cycle has negative cost you are done.
▶ Otherwise send flow in opposite direction along the cycle until the bottleneck edge(s) does not carry any flow.
▶ You still have a circulation with negative cost.
▶ Repeat.

10 Mincost Flow

Algorithm 45 CycleCanceling($G = (V, E), c, u, b$)
1: establish a feasible flow f in G
2: **while** G_f contains negative cycle **do**
3:     use Bellman-Ford to find a negative circuit Z
4:     $\delta \leftarrow \min\{u_f(e) \mid e \in Z\}$
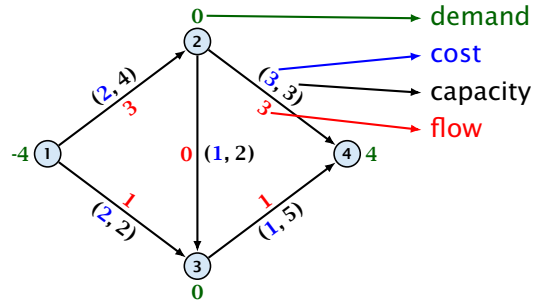5:   augment δ units along Z and update G_f
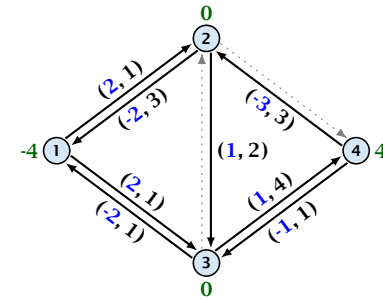
How do we find the initial feasible flow?
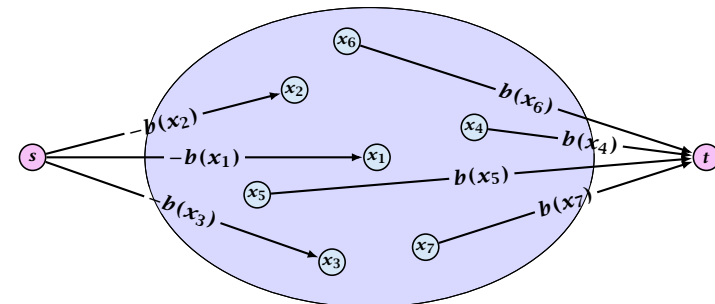
▶ Connect new node s to all nodes with negative $b(v)$-value.
▶ Connect nodes with positive $b(v)$-value to a new node t.
▶ There exist a feasible flow in the original graph iff in the resulting graph there exists an s-t flow of value

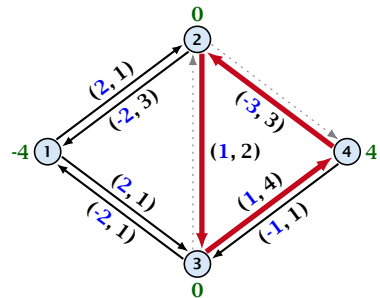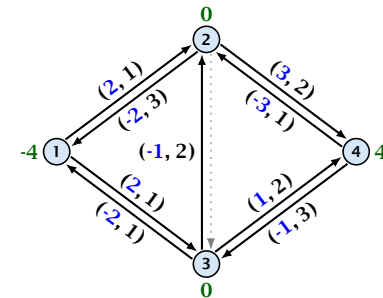$$\sum_{v:b(v)<0} (-b(v)) = \sum_{v:b(v)>0} b(v) \; .$$
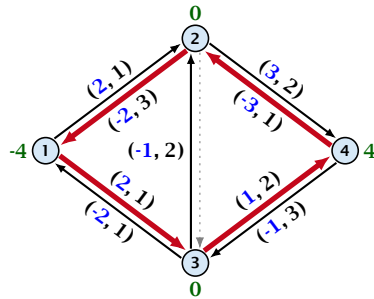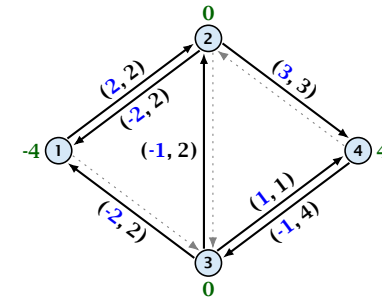
10 Mincost Flow

10 Mincost Flow

10 Mincost Flow

10 Mincost Flow

10 Mincost Flow

(2, 1) (3, 2)
(-2, 3) (-3, 1)
-4 (-1, 2) 4
(2, 1) (1, 2)
(-2, 1) (-1, 3)
0

---

10 Mincost Flow

0

(2, 2) (3, 3)
(-2, 2)
-4 (-1, 2) 4
(-2, 2) (1, 1)
(-1, 4)
0

---

10 Mincost Flow

Lemma 75

The improving cycle algorithm runs in time $\mathcal{O}(nm^2CU)$, for integer capacities and costs, when for all edges e, $|c(e)| \leq C$ and $|u(e)| \leq U$.

- ▶ Running time of Bellman-Ford is $\mathcal{O}(mn)$.
- ▶ Pushing flow along the cycle can be done in time $\mathcal{O}(n)$.
- ▶ Each iteration decreases the total cost by at least 1.
- ▶ The true optimum cost must lie in the interval $[-mCU, \ldots, +mCU]$.

Note that this lemma is weak since it does not allow for edges with infinite capacity.

---

10 Mincost Flow

A general mincost flow problem is of the following form:

$$\begin{array}{ll} \min & \sum_e c(e)f(e) \\ \text{s.t.} & \forall e \in E: \ \ell(e) \leq f(e) \leq u(e) \\ & \forall v \in V: \ a(v) \leq f(v) \leq b(v) \end{array}$$

where $a: V \to \mathbb{R}$, $b: V \to \mathbb{R}$; $\ell: E \to \mathbb{R} \cup \{-\infty\}$, $u: E \to \mathbb{R} \cup \{\infty\}$ $c: E \to \mathbb{R}$;

Lemma 76 (without proof)

A general mincost flow problem can be solved in polynomial time.