

The Min Cost Flow Problem

– the network simplex algorithm

Jesper Larsen, David Pisinger
jesla, pisinger@man.dtu.dk

DTU Management
Technical University of Denmark

The Min Cost Flow Problem – p.1/77

Min Cost Flow – Definition

The Min Cost Flow problem consists in supplying the sinks from the sources by a flow in the cheapest possible way:

$$\begin{aligned} \min \quad & \sum_{e \in E} w_e x_e \\ & f_x(i) = b_i \quad i \in V \\ & 0 \leq x_e \leq u_e \quad e \in E \end{aligned}$$

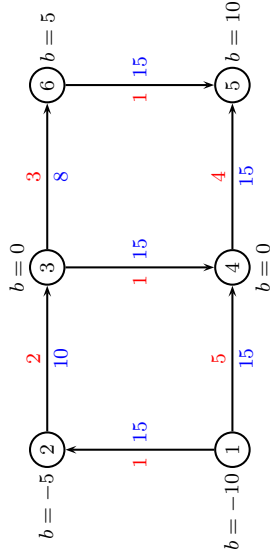
$$\text{where } f_x(i) = \sum_{(j,i) \in E} x_{ji} - \sum_{(i,j) \in E} x_{ij}.$$

Assume graph G is connected
Otherwise solve separate problems

The Min Cost Flow Problem – p.3/77

Min Cost Flow - Terminology

- We consider a directed graph $G = (V, E)$ in which each edge e has a capacity $u_e \in \mathbb{R}_+$ and a unit transportation cost $w_e \in \mathbb{R}$.
- Each vertex i furthermore has a demand $b_i \in \mathbb{R}$. If $b_i \geq 0$ then i is called a **sink**, and if $b_i < 0$ then i is called a **source**.
- We assume that $b_V = \sum_{i \in V} b_i = 0$.



The Min Cost Flow Problem – p.2/77

Min Cost Flow – Network Simplex

Interpretation of Simplex to min cost flow

- Transshipment problem (no capacities)
- Show that for every tree T exactly one tree solution
- Show that any solution can be transformed to equivalent tree solution
- Optimality criteria
- Network simplex for transshipment problem
- Generalize to min cost flow
- Conclusion

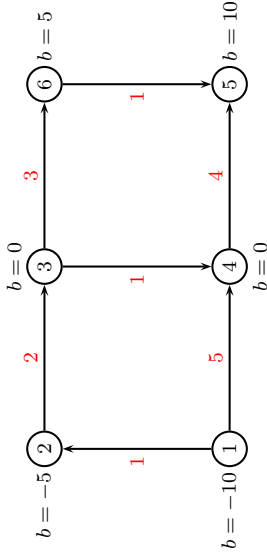
The Min Cost Flow Problem – p.4/77

Min Cost Flow – Characteristics

- If we look at the rows they are linearly dependent (sum them all together).
- But $n - 1$ of the rows are linearly independent, that is, there are $n - 1$ variables in a basic feasible solution.
- Moreover these $n - 1$ variables will correspond to a spanning tree! (in the minimum cost flow problem we just call it a tree).

Transshipment problem

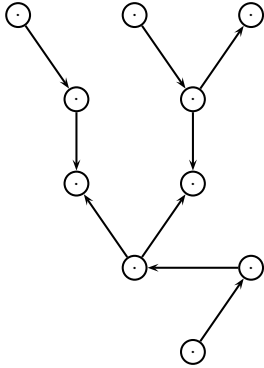
Min Cost Flow problem without capacities ($u_{ij} = \infty$)



Min Cost Flow – Model

	x_{e_1}	x_{e_2}	...	$x_{i,j}$	...	x_{e_m}
	w_{e_1}	w_{e_2}	...	$w_{i,j}$	...	w_{e_m}
1	-1	.	...	...	...	b_1
2	.	.	...	.	...	b_2
...	.	.	...	.	...	.
i	1	.	...	-1	...	b_i
...	.	.	...	.	...	.
j	.	.	...	1	...	b_j
...	.	.	...	.	...	.
n	.	.	...	.	...	b_n
e_1	-1					$-u_1$
e_2		-1				$-u_2$
...	.	.	.	.	.	.
(i,j)				-1		$-u_{i,j}$
...	.	.	.	.	.	.
e_m					-1	$-u_m$

A tree solution

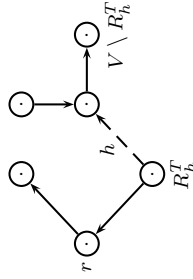


- A **tree solution** of the transshipment problem is a vector $x \in \mathbb{R}^E$ satisfying for some (unoriented) tree T :
 - $f_x(i) = b_i$, for all $i \in V$
 - $x_e = 0$ for all $e \notin T$

A tree T uniquely identifies a tree solution x

Constructing a tree solution I

- Select a vertex r as root of T . Consider an edge h . This edge will partition V into two parts:
 - a part containing r denoted R_h^T , and
 - the remaining nodes $P_h^T = V \setminus R_h^T$.

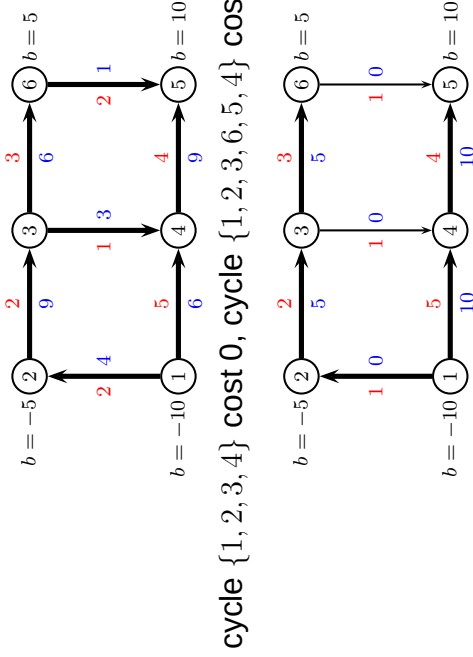


- If h starts in R_h^T and ends in P_h^T then set $x_h = -\sum_{i \in R_h^T} b_i$
- If h starts in P_h^T and ends in R_h^T then set $x_h = \sum_{i \in R_h^T} b_i$

The Min Cost Flow Problem – p.9/77

Solution vs. tree solution

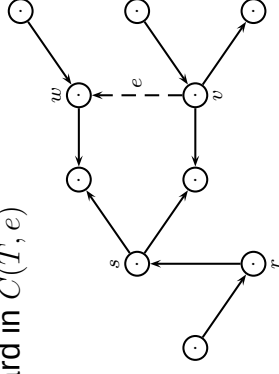
- If $G = (V, E, w, b)$ has an optimal solution, it also has an optimal tree solution.



The Min Cost Flow Problem – p.10/77

Negative cost circuit

- The Network Simplex Method moves from tree solution to tree solution using negative cost circuits C_e^T consisting of tree edges *and exactly one non-tree edge* e
- We have the following properties:
 - $C(T, e) \subseteq T \cup \{e\}$
 - e is forward in $C(T, e)$

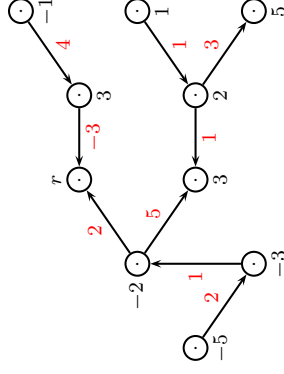


The Min Cost Flow Problem – p.11/77

Set the potentials

- Set the vertex potential $y_i, i \in V$ equal to the cost of the path in T from r to i (counted with sign: plus for a forward edge, minus for a backward edge). It now holds:
 - For all $i, j \in V$, the cost of the path from i to j in T equals $y_j - y_i$ (why?). But then the reduced costs $\bar{w}_{ij}$ (defined by $w_{ij} + y_i - y_j$) satisfy:

$$\forall e \in T : \bar{w}_e = 0 \quad \forall e \notin T : \bar{w}_e = c(C_e^T)$$

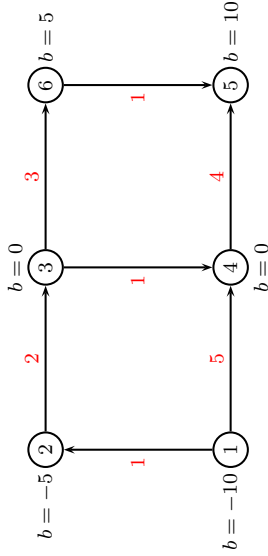


The Min Cost Flow Problem – p.12/77

Algorithm

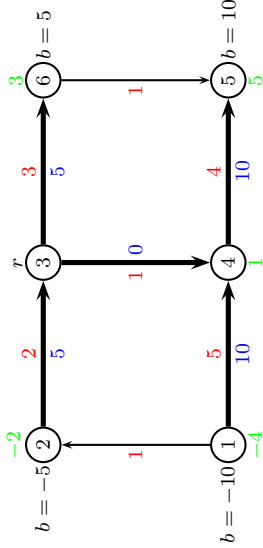
find a tree T whose associated flow x is feasible
 compute y
 while there exists an arc $e = (u, v)$ such that $\bar{w}_e = w_e + y_u - y_v < 0$
 find such an arc e
 if $C(T, e)$ has no reverse arc, **stop**
 compute $\theta = \min\{x_j : j \text{ reverse arc of } C(T, e)\}$
 find a reverse arc h of $C(T, e)$ with $x_h = \theta$
 augment x by θ on $C(T, e)$
 set $T := T \cup \{e\} \setminus \{h\}$
 update y

Transshipment - Example



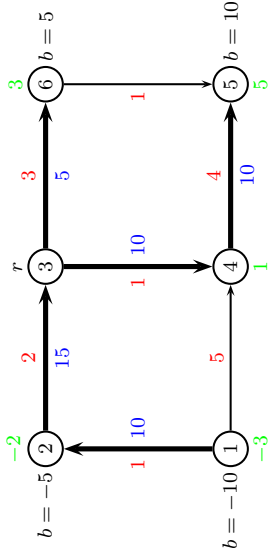
Example

Initialize by solving maximum flow, finding tree



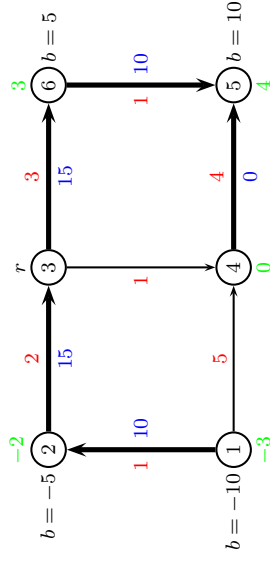
reduced cost $\bar{w}_{12} = 1 + (-4) - (-2) = -1$
 reduced cost $\bar{w}_{65} = 1 + 3 - 5 = -1$
 find $\theta = \min\{x_{14}\} = \min\{10\} = 10$
 find edge $(1, 4)$ where $x_{14} = \theta$
 augment cycle $C = \{1, 2, 3, 4\}$ by $\theta = 10$

Example



reduced cost $\bar{w}_{14} = 5 + (-3) - 1 = 1$
 reduced cost $\bar{w}_{65} = 1 + 3 - 5 = -1$
 find $\theta = \min\{x_{34}, x_{45}\} = \min\{10, 10\} = 10$
 find edge $(3, 4)$ where $x_{34} = \theta$
 augment cycle $C = \{3, 6, 5, 4\}$ by $\theta = 10$

Example



reduced cost $\bar{w}_{14} = 5 + (-3) - 0 = 2$

reduced cost $\bar{w}_{34} = 1 + 0 - 0 = 1$

stop

Tree Solution Extended

- The concept of a tree solution is extended to capacity constrained problems – a non-tree edge e may have flow either 0 or u_e .
- $E \setminus T$ is hence split into two edge sets, L and U . Edges in L must have flow 0 - edges in U must be filled to their capacity.
- The tree solution x must satisfy:

$$\forall i \in V : f_x(i) = b_i$$

$$\forall e \in L : x_e = 0$$

$$\forall e \in U : x_e = u_e$$

Min Cost Flow Problem

What if edge capacities are present?

- We consider the Min Cost flow - problem given by the network $G = (V, E)$ with demands b_i , $i \in V$, capacities u_e , $e \in E$ and costs w_e , $e \in E$.
- Remember the optimality conditions for a feasible flow $x \in \mathbb{R}^E$:

$$\bar{w}_e < 0 \Rightarrow x_e = u_e \quad (\neq \infty)$$

$$\bar{w}_e > 0 \Rightarrow x_e = 0$$

A unique tree solution

- As before, the tree solution for T is unique:
- Select a vertex r called the *root* of T . Consider an edge h in T . h partitions V into two:
 - a part containing r denoted R_h^T , and
 - a remainder $P_h^T = V \setminus R_h^T$.
- Consider now the *modified demand* $B(P_h^T)$ in P_h^T given that a tree solution is sought:

$$B(P_h^T) = \sum_{i \in P_h^T} b_i - \sum_{\{(i,j) \in U : i \in R_h^T, j \in P_h^T\}} u_{ij} + \sum_{\{(i,j) \in U : i \in P_h^T, j \in R_h^T\}} u_{ij}$$

- if h starts in R_h^T and ends in P_h^T then set $x_h = B(P_h^T)$
- if h starts in P_h^T and ends in R_h^T then set $x_h = -B(P_h^T)$

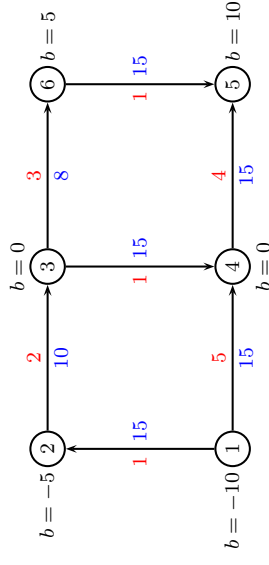
Network Simplex for Min Cost Flow

- Idea: Start with a feasible tree solution - find it e.g using a Max Flow algorithm.
- We now search for *either* a non-tree edge e with $x_e = 0$ and negative reduced cost or a non-tree edges e with $x_e = u_e$ and positive reduced cost. Given e and (T, L, U) , the corresponding circuit is denoted $C_e^{T,L,U}$.
- In this, the flow must be *increased* in forward edges respecting capacity constraints and *decreased* in backward edges respect the non-negativity constraints.

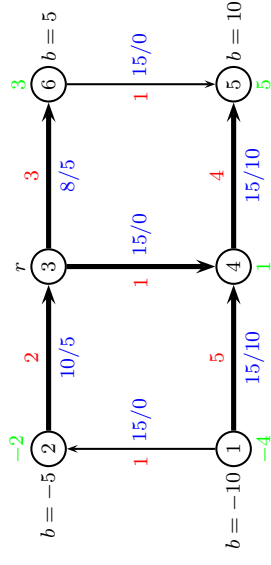
Algorithm

- find a triple (T, L, U) whose associated flow x is feasible
- compute y
- while there exists an arc $e = (u, v) \in L$ such that $\bar{w}_e < 0$
 - ... or $e \in U$ and $\bar{w}_e > 0$
 - find such an arc e
 - if $C(T, L, U, e)$ has no reverse arc
 - ... and no forward arc of finite capacity, **stop**
 - compute $\theta_1 = \min\{x_j : j \text{ reverse arc of } C(T, L, U, e)\}$
 - compute $\theta_2 = \min\{u_j - x_j : j \text{ forward arc of } C(T, L, U, e)\}$
 - let $\theta = \min\{\theta_1, \theta_2\}$
 - find a reverse arc h of $C(T, L, U, e)$ with $x_h = \theta$
 - ... or forward arc with $u_h - x_h = \theta$
 - augment x by θ on $C(T, L, U, e)$
 - set $T := T \cup \{e\} \setminus \{h\}$
 - update L, U by deleting h and adding e as appropriate
 - update y

Example

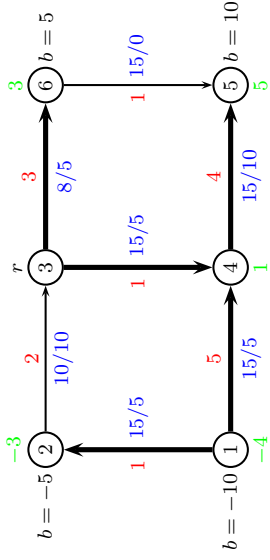


Example



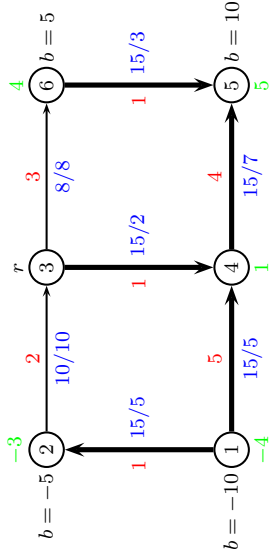
$L = \{(1, 2), (6, 5)\}, U = \{\}$
 reduced cost $\bar{w}_{12} = 1 + (-4) - (-2) = -1$
 reduced cost $\bar{w}_{65} = 1 + 3 - 5 = -1$
 cycle $C = \{1, 2, 3, 4\}$
 find $\theta = 5$
 augment cycle $C = \{1, 2, 3, 4\}$ by $\theta = 5$

Example



$L = \{(6, 5)\}, U = \{(2, 3)\}$
 reduced cost $\bar{w}_{23} = 2 + (-3) - 0 = -1$
 reduced cost $\bar{w}_{65} = 1 + 3 - 5 = -1$
 cycle $C = \{3, 6, 4, 5\}$
 find $\theta = 3$
 augment cycle $C = \{3, 6, 4, 5\}$ by $\theta = 3$

Example



$L = \{\}, U = \{(2, 3), (3, 6)\}$
 reduced cost $\bar{w}_{23} = 2 + (-3) - 0 = -1$
 reduced cost $\bar{w}_{36} = 3 + 0 - 4 = -1$
 stop

Conclusion

Advantages over augmenting circuit

- Potentials y can be calculated in linear time
- Augmenting edge can be found in linear time
- Can be combined with selection criteria to reach polynomial running time

Relation to simplex

- basis of $n - 1$ edges
- most negative reduced cost enters basis
- bounds handled implicitly

CPLEX uses network simplex when appropriate structure