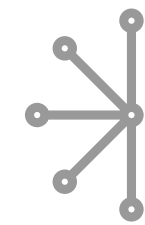




Technische  
Universität  
Braunschweig



Institut für Betriebssysteme  
und Rechnerverbund  
Algorithmik

# Mathematical Methods of Algorithmics

## Tutorial 0 — Examples, Modeling & Solving in Practice

# Board: Example Simplex with Dictionaries

$$\max 6x_1 + 8x_2 + 5x_3 + 9x_4$$

$$\text{s.t. } 2x_1 + x_2 + x_3 + 3x_4 \leq 5$$

$$x_1 + 3x_2 + x_3 + 2x_4 \leq 3$$

$$x_1, x_2, x_3, x_4 \geq 0$$

## Idea of Modeling

- We can solve LPs, so if we can model problem  $A$  as LP, we can solve  $A$
- Model: Efficiently turn concrete problem instance into concrete LP
- The solution of that LP should give us the solution to  $A$
- Similar to reductions from complexity theory

## Solving LPs

- How can we solve LPs in practice?

# Maximum Network Flow

**Given:** directed graph  $G = (V, E)$  with edges with capacities  $c(e) \in \mathbb{R}_{\geq 0}$

- See board for an example
- For example, think about pipelines, or cars on roads or ships on rivers

**Desired:**

- Flow  $f(e)$  from source  $s \in V$  to sink  $t \in V$ ,  $0 \leq f(e) \leq c(e)$  for all edges
- Except for source & sink: Flow conservation — what comes in must go out
- Maximize the flow value, i.e., the value flowing into the sink

**Model (encode this problem) as LP!**

## Variables:

- $x_{vw}$ , flow value on edge  $vw \in E$

**Objective:**  $\max \sum_{vt \in E} x_{vt} - \sum_{tv \in E} x_{tv}$

## Constraints:

Flow value on each edge:  $\forall vw \in E : 0 \leq x_{vw} \leq c(vw)$

Flow conservation:  $\forall v \in V \setminus \{s, t\} : \sum_{vw \in E} x_{vw} = \sum_{wv \in E} x_{wv}$

## Vertex Capacities:

- At most  $c(v)$  incoming flow into  $v \in V \setminus \{s, t\}$

## Minimum Flow:

- Some edges with 'minimal' capacity

## Minimum Cost Maximum Flow:

- It costs  $w(e)f(e)$  units to ship  $f(e)$  units along edge  $e$
- Two-staged process: Find maximum flow first, minimize costs later!

**Enables us to adapt our algorithm to new constraints quickly!**

# LP Solvers in Practice

Different solvers exist; differences in quality (performance) are quite drastic.

## Commercial solvers:

- CPLEX
- Gurobi

Both are good, Gurobi tends to be a bit faster and is more actively developed.

Both have free academic licenses for students/researchers.

## Open source toolkits:

- SCIP (tends to be the fastest open-source toolkit)
- COIN-OR (CLP, CBC)
- GLPK (GNU Linear Programming Kit)

All these toolkits can also handle Mixed Integer Linear Programs (Integrality Constraints).

**Note:** Those restrictions make the problem NP-hard and can make solving much slower.