MINES-07 PSL week

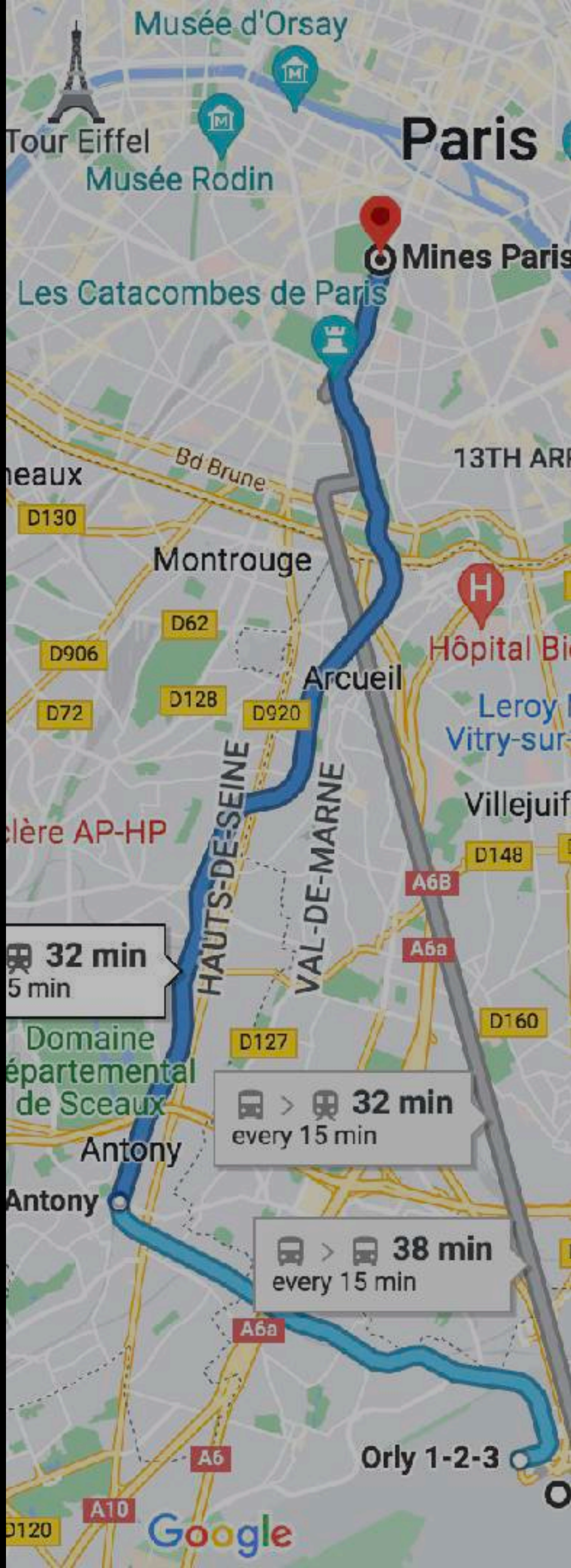
combinatorial & stochastic optimization

sophie.demassey@minesparis.psl.eu

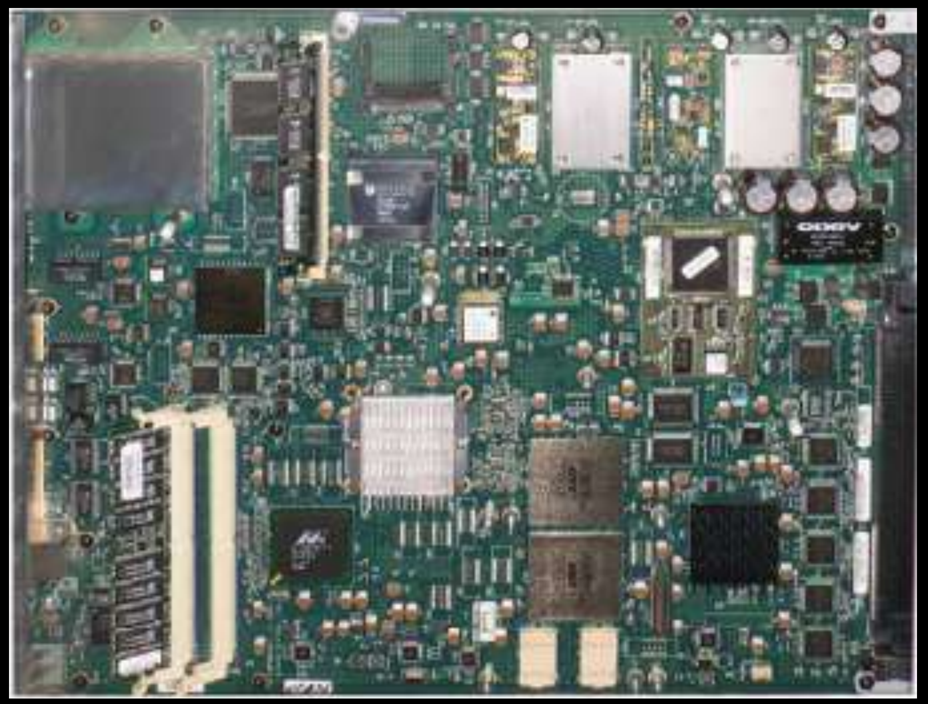
Centre de Mathématiques Appliquées

decision is optimization

select the best/optimum
of all possible alternatives/solutions
regarding a quantitative criterion/objective






select the best solution regarding the objective

decision: operation/strategy, static/dynamic, short/long-term
solution: plan/schedule, path/flow/routing, assignment/layout/design
objective: duration, distance/space, cost/profit/preference, amount/level





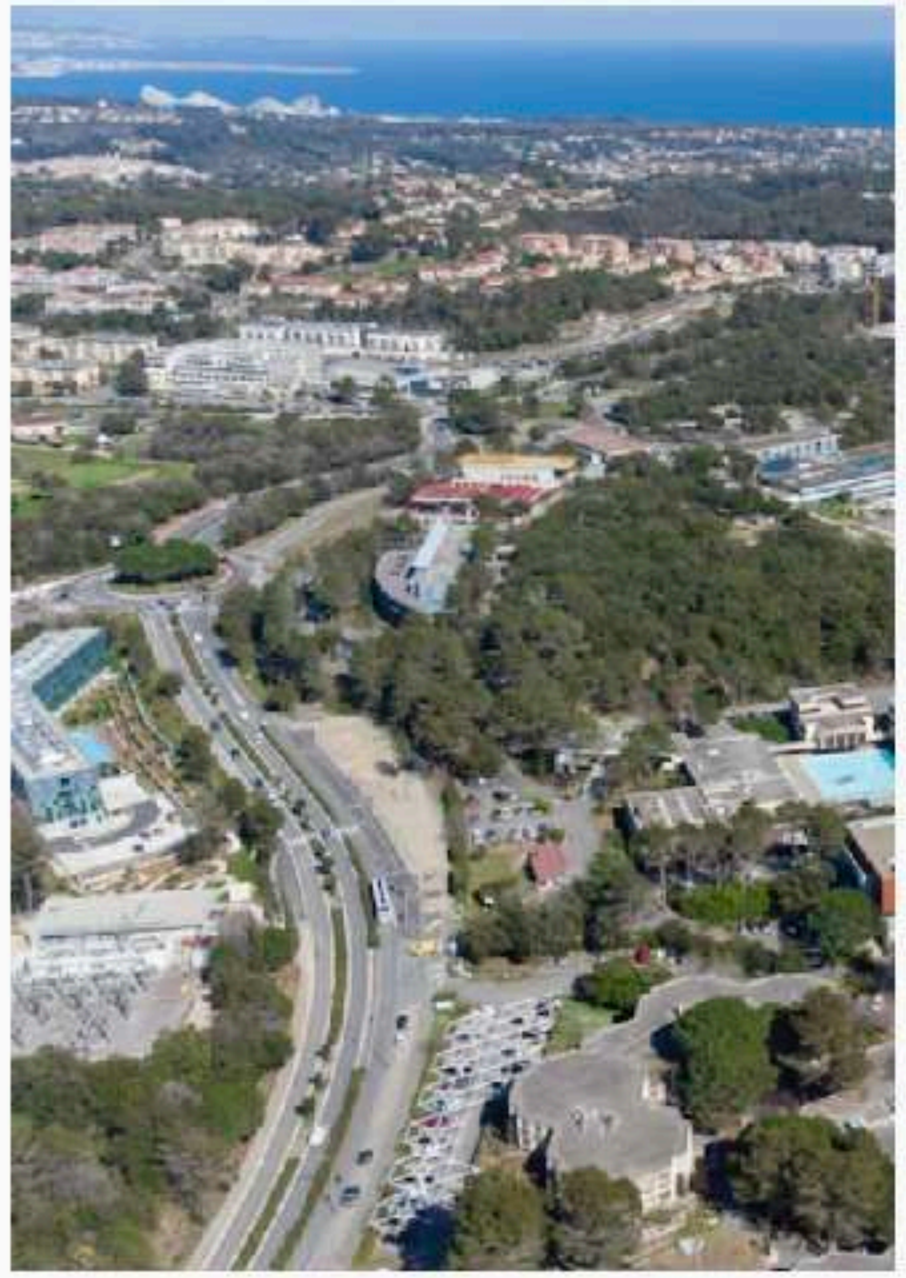
a tool for decision support:
mathematical optimization aka operational research

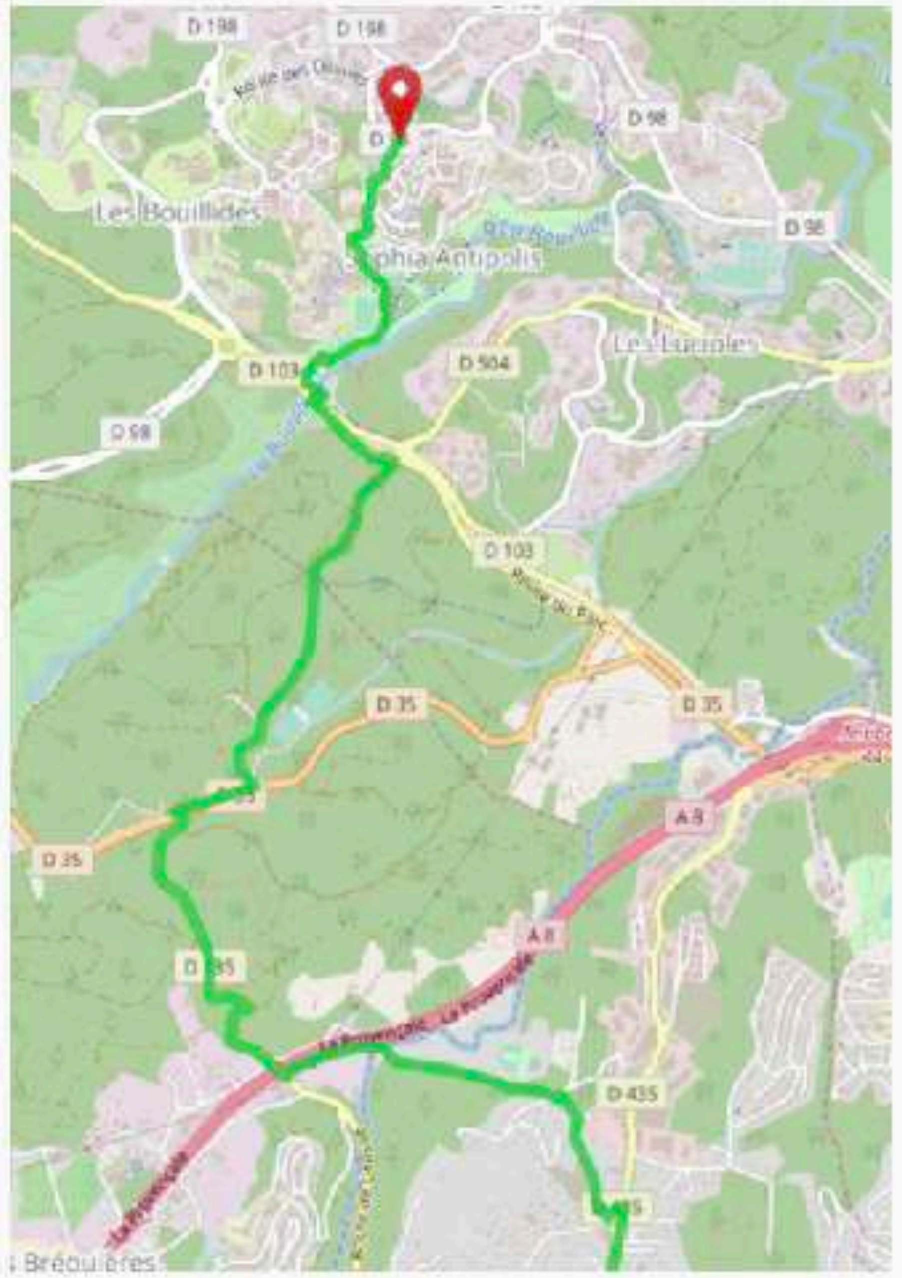
some historical FR players:

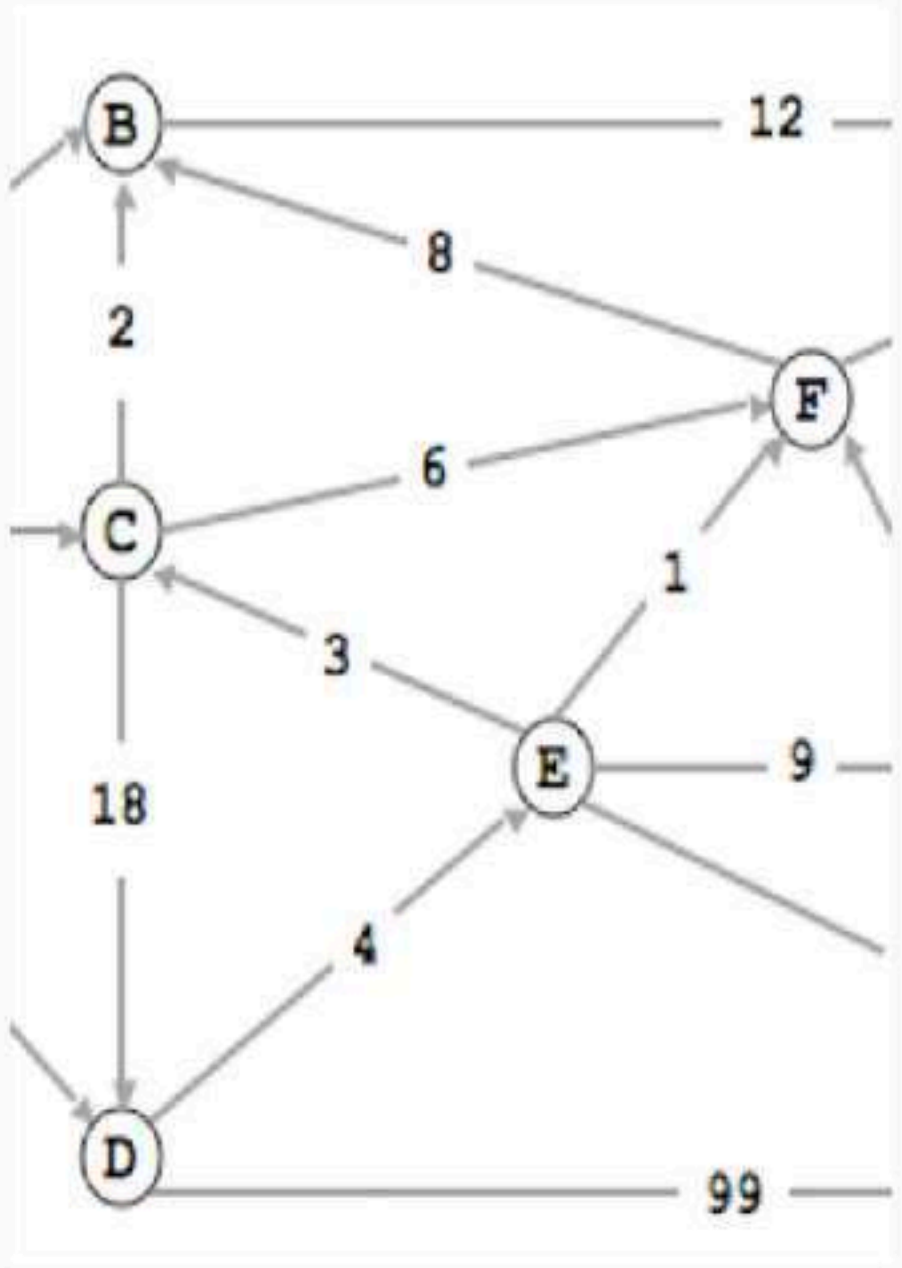
scientist in optimization:
understand the business, do maths/cs, solve problems

mathematical optimization for decision

1. build an abstract model of a concrete system
2. derive a mathematical formulation: relationships/unknowns
3. apply an algorithm to solve the model
4. derive practical solutions







$$min\ z = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}}^{n} d_{ij} x_{ij}$$

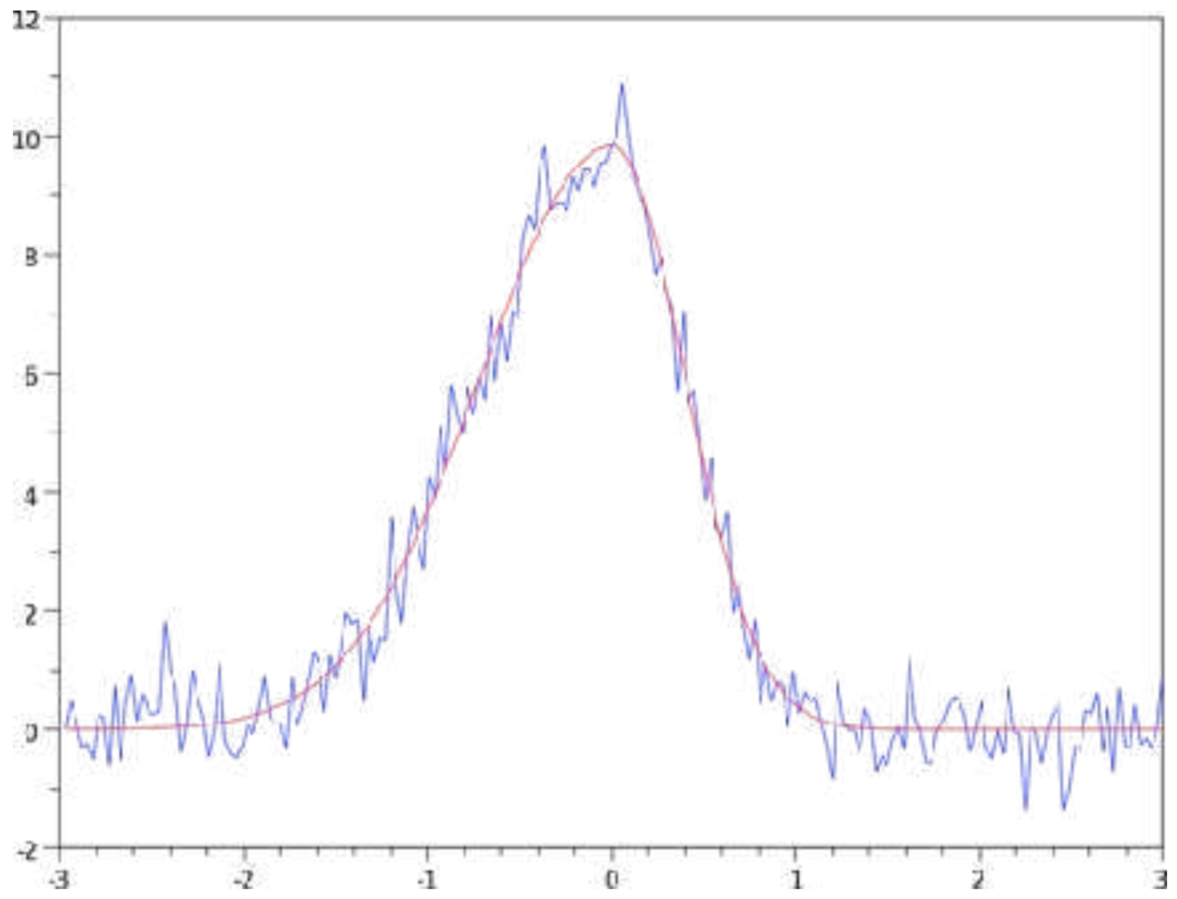
$$\sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij} = 1, \quad \forall i \in N$$

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} x_{ij} = 1, \quad \forall j \in N$$

solve ? theory/practice

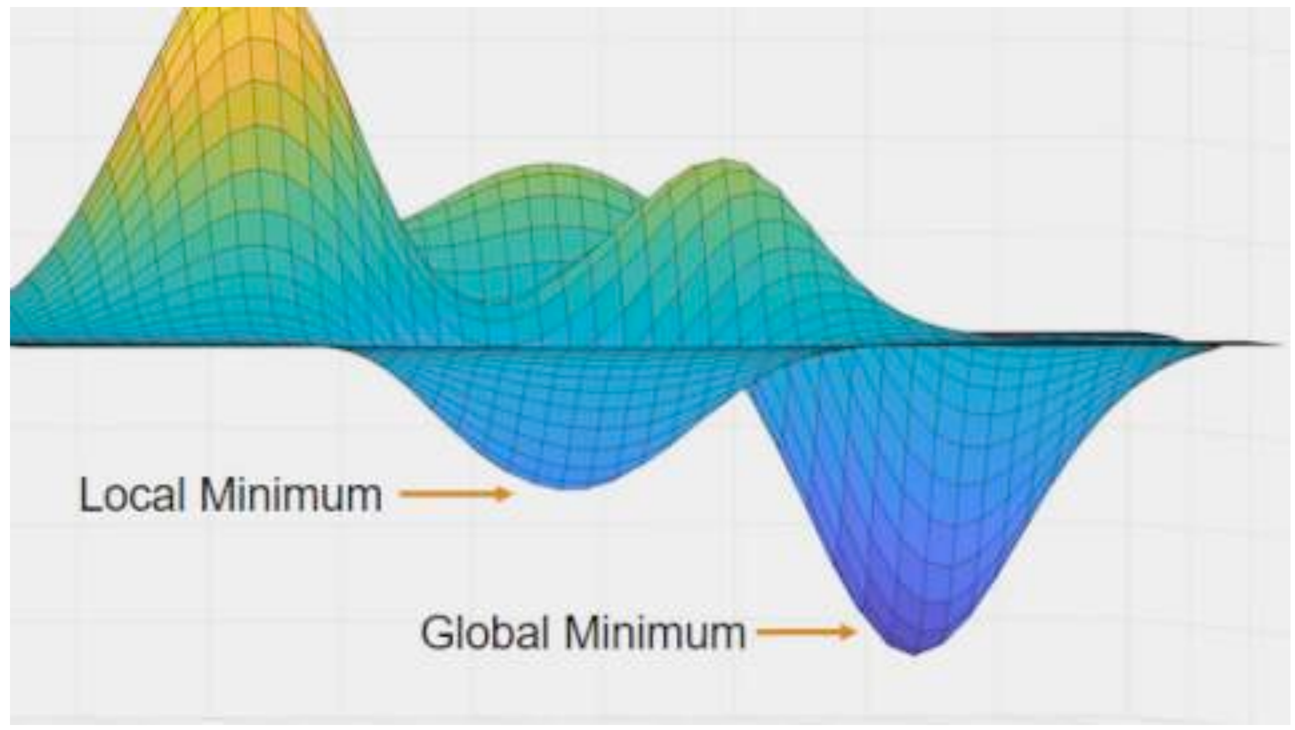
feasibility ?

- models are approximate
- data are uncertain
- calculations are truncated



optimality ?

- finite time $\neq$ reasonable time
- provable with a gap tolerance
- provable locally vs. globally



mathematical optimization $\neq$ decision support

math optimization also works for:

machine learning find a best model/data match: min empirical risk (supervised), maxreward (reinforcement), min distance (clustering), max homogeneity (decision tree), max margin (svm), max likelihood (markov process).

control find a command $u(t)$ to optimize trajectory $x(t)$ s.t. $x'(t) = g(x(t), u(t))$

game theory, economics, calculus,...

other advanced options for decision:

simulation given a reliable model but no good math formulation

machine learning given historical data but no good reliable model

hybridations evaluate computed solutions by simulation (e.g. black-box optimization), learn mathematical models

math opt for decision (specs 1)

- reliable models: how accurate ? close to reality ?
- optimality certificates: how good is the solution ?
- versatile algorithms: if the problem changes ?
- efficient algorithms: solution times for complex/large problems ?

math opt for decision (specs 2)

- discrete decisions and logical relationships (switch on or off ? if off then no process)

  combinatorial optimization

- uncertain data (approximations and forecasts)

  stochastic optimization

this PSL week:
a quick overview of

monday-wednesday morning
combinatorial optimization

Welington de Oliveira (Mines/CMA)
https://www.oliveira.mat.br

Sophie Demassey (Mines/CMA)
https://sofdem.github.io

stochastic optimization
wednesday afternoon-friday

mixed integer linear programming (MILP)

~~combinatorial optimization~~

combinatorial optimization: beyond MILP

NOT in this course:

- graph theory and combinatorial structures
- metaheuristics and approximation algorithms
- Logic or Constraint Programming
- Linear or Nonlinear Programming (just a glimpse)
- advanced theoretic topics in MI(N)LP

focus on practical MILP

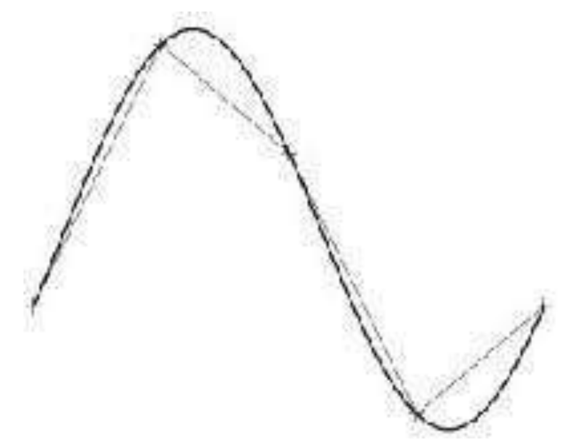
IN this course: a practical approach how to model and solve

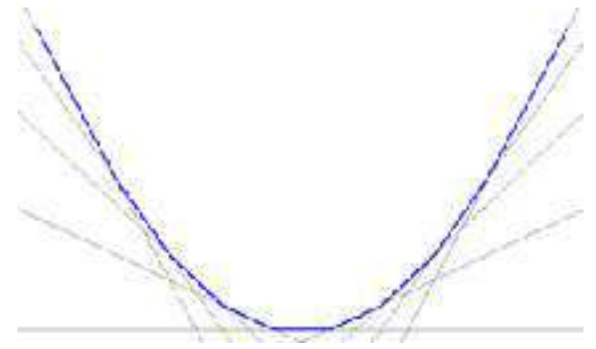
- MILP modeling techniques
- some applications
- notions of complexity
- main techniques to solve MILPs: bounding, branching, cutting
- modern MILP solvers (aka algorithms) and their usage
- steps towards reformulation, duality-based decomposition, and convex MINLP
- technical results without theoretical proofs (see the bibliography to learn more)

why the MILP lense ?

broad applicability:

- logical conditions as binary variables and linear inequalities
- nonlinear relations (physic/economic) as piecewise-linear fits
- convex NonLP ≈ LP $\Longrightarrow$ convex MINLP ≈ MILP (theoretically)



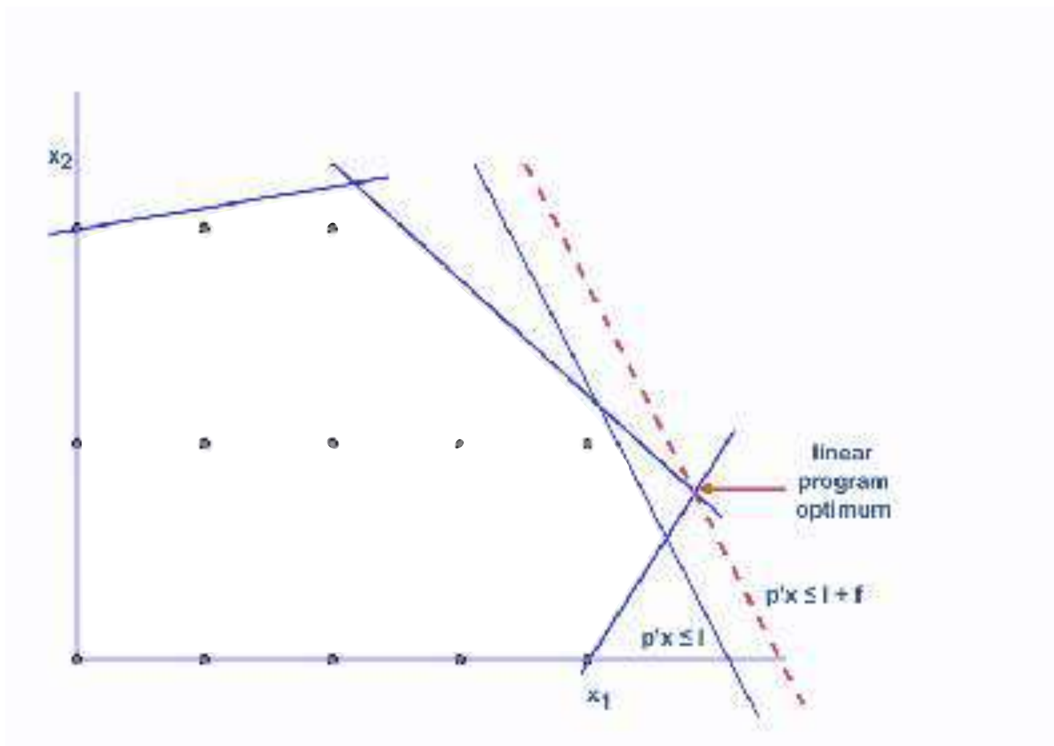


versatility:

- generic form = generic solvers fruit of many research works
- specific problem = specific model + generic solver + specific options

efficiency:

- easy LP + partial enumeration
- sophisticated strategies and algorithmic components



learning goals

after this course, you should be able to:

- identify if an optimization problem is eligible to MILP
- formulate it as a MILP, identify its complexities, and implement the model
- run an off-the-shelf MILP solver, understand the solution process and ways to improve it
- describe main applications of combinatorial optimization: domains and problems
- describe the principle of advanced solution methods and their usage

evaluation & practice

validation be there and **participate**

retake the code of the mini-project (to send by email before march 15)

project:
power generation

- deterministic & stochastic variants

- proposed dev environment: Jupyter Notebook, Google Colab, Gurobi solver, python API: code + report directly through your browser

- goals: model as a MILP, implement and call a solver

- correctness >> completeness

course schedule

	course	project
Mon AM	modeling	model (1)
Mon PM	complexity	model (2)
Tue AM	algorithms	code (1)
Tue PM	modern solvers	code (2)
Wed AM	decomposition	code (3)

ASK for explanations and breaks

the MILP way

a practical view

(1)

Sophie Demassey 2023

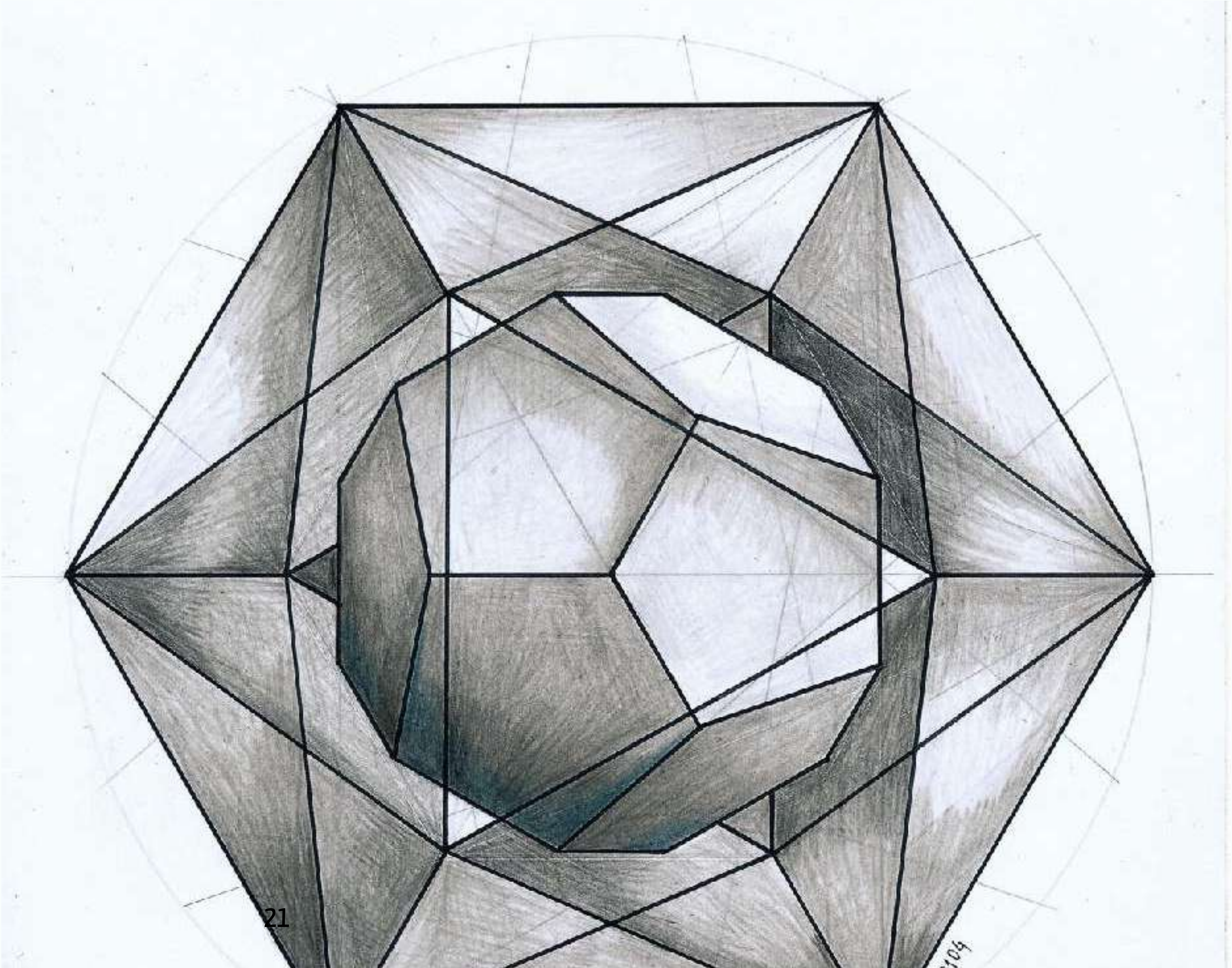
Centre de Mathématiques Appliquées

1 how to model ?

2 how difficult ?

3 how to solve ?

1 how to model ?



Mathematical Program

program = plan (e.g. military)

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to:} & g(x) \leq 0 \\ & x \in X \subseteq \mathbb{R}^n \end{array}$$

Definition

$x \in \mathbb{R}^n$	variables
$f : \mathbb{R}^n \to \mathbb{R}$	objective
$g : \mathbb{R}^n \to \mathbb{R}^m$	constraints

Remark

- $\max f(x) = -\min(-f)(x)$
- $g(x) \geq b \equiv -g(x) + b \leq 0$
- sign $<$ or $\neq$ not allowed in MP (this and beyond: see CLP)

Mixed Integer Linear Program

$$\min\{f(x) \mid g(x) \leq 0,\ x \in \boxed{\mathbb{Z}^p} \times \mathbb{R}^{n-p}\}$$

with **linear** functions f and g:

$$\min c^\top x$$
$$\text{s.t.:}\quad Ax \geq b$$
$$x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}$$

$c \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$

$$\min \sum_{j=1}^{n} c_j x_j \qquad \text{s.t.:}$$
$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i \ \forall i = 1,\ldots,m$$
$$x_j \in \mathbb{Z} \qquad \forall j = 1,\ldots,p$$
$$x_j \in \mathbb{R} \qquad \forall j = p+1,\ldots,n$$

Mixed Integer Linear Program

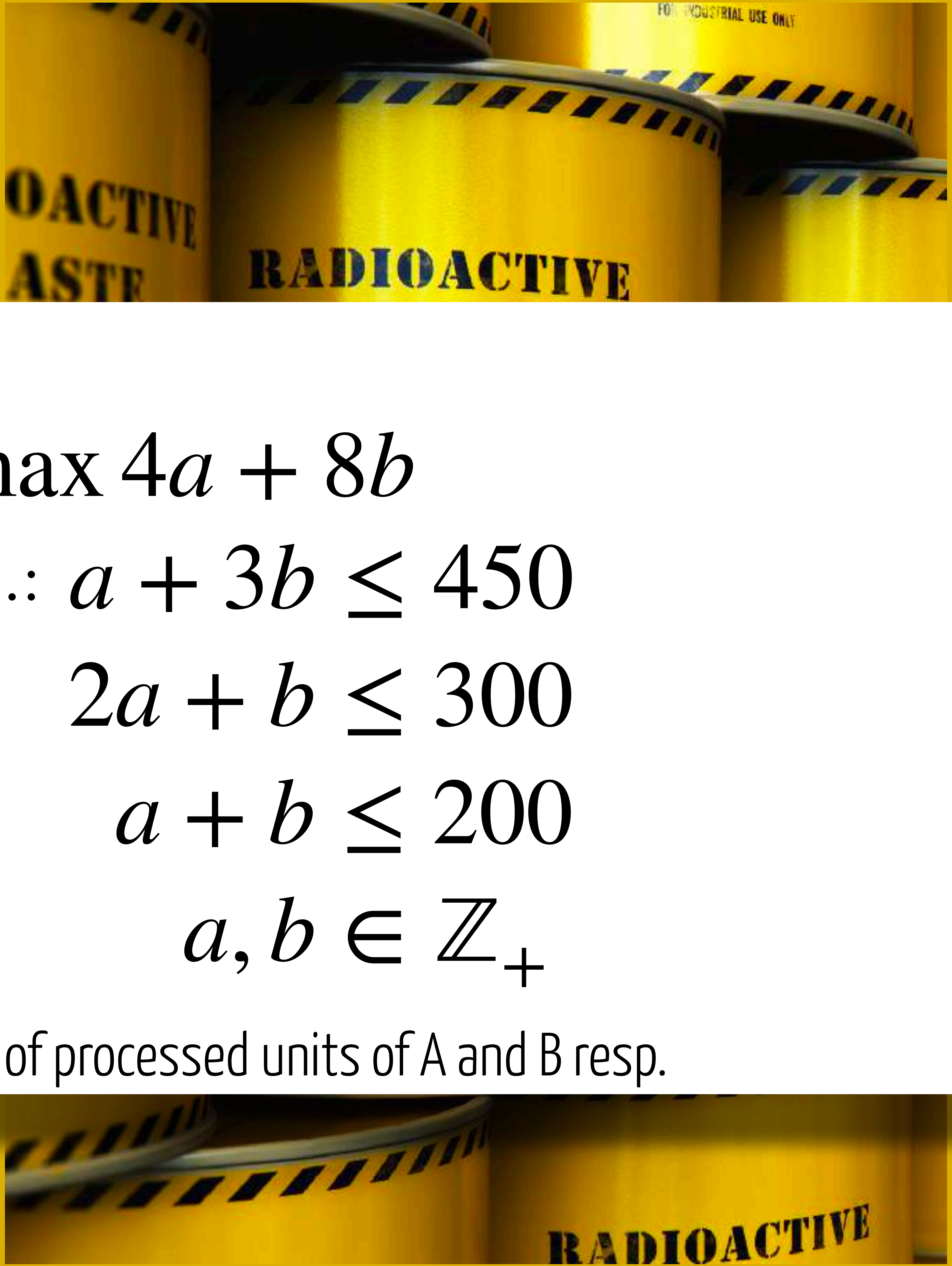
$$\min cx$$
$$\text{s.t.:}\quad Ax \geq b$$
$$x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}$$

terminology

objective	cx
linear constraints	$Ax \geq b$
integrity constraints	$x_1, \ldots, x_p \in \mathbb{Z}$
right hand side (rhs)	$b \in \mathbb{R}^m$
cost vector	$c^\top \in \mathbb{R}^n$
coefficient matrix	$A \in \mathbb{R}^{m \times n}$
solution space	$\{x \in \mathbb{R}^n\}$
feasible set	$\{x \in \mathbb{Z}^p \times \mathbb{R}^{n-p} \mid Ax \geq b\}$

waste management

2 types of nuclear waste A, B with different unit profit/processing time going through 3 processes I, II and III with limited availability

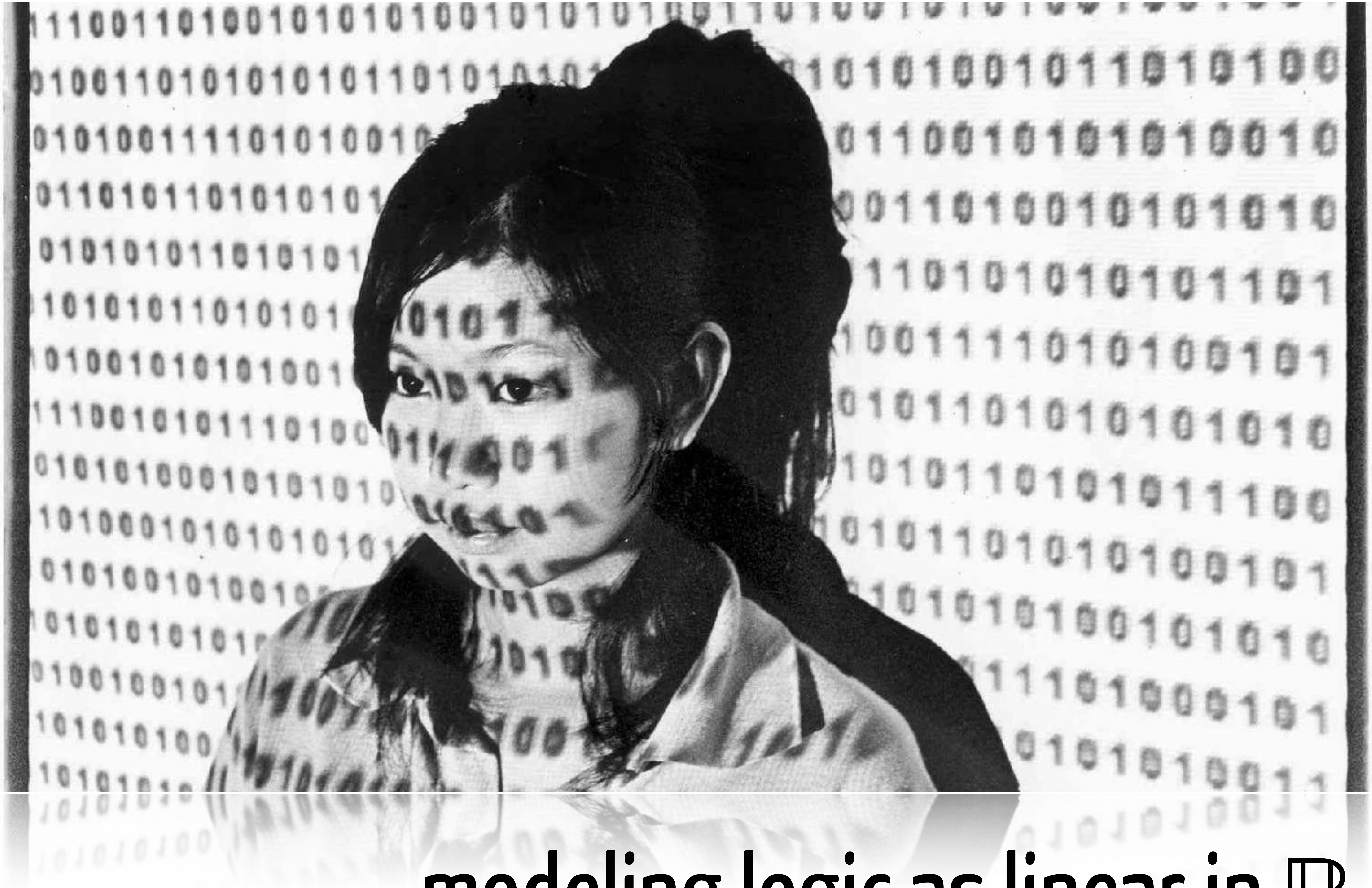


$$
\begin{aligned}
\max\ & 4a + 8b \\
\text{s.t.:}\ & a + 3b \leq 450 \\
& 2a + b \leq 300 \\
& a + b \leq 200 \\
& a, b \in \mathbb{Z}_+
\end{aligned}
$$

a, b number of processed units of A and B resp.

	I	II	III	unit profit
A	1h	2h	1h	4k€
B	3h	1h	1h	8k€
available	450h	300h	200h	

objective: maximize the profit.



modeling logic as linear in 𝔹

~~true~~ 1 or ~~false~~ 0

- is item j selected ? $x_j \in \{0,1\}$
- is item j assigned to item i ? $x_{ij} \in \{0,1\}$
- at most n available items $x_1, \ldots, x_n \in \{0,1\}$
- $z \in \mathbb{R}_+$ is it greater than a ? $x \in \{0,1\}, z \in \mathbb{R}, z \geq ax$

binary variables to model true/false conditions on objects

Integer Knapsack Problem

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\text{s.t. } \sum_{j=1}^{n} w_j x_j \le K$$

$$x_j \in \{0,1\} \qquad j = 1..n$$

Input n items, value c_j and weight w_j for each item j, capacity K.

Output a maximum value subset of items whose total weight does not exceed K.

x_j is item j packed ?

logic with binaries

x, y binary variables; z continuous variable; a, k, n constants

- either x or y $\quad x + y = 1$
- if x then y $\quad y \geq x$
- if x then z ≤ a $\quad z \leq a + (M - a)(1 - x)$ "big M constraint"

big enough but keep it tight !

linear constraints on binary variables to model logical relations between objects

logic with binaries

x, y binary variables; z continuous variable; a, k, n constants

- either x or y $\quad x + y = 1$
- if x then y $\quad y \geq x$
- if x then z ≤ a $\quad z \leq a + (M - a)(1 - x)$
- if not x then z ≥ a $\quad z \geq a - (M + a)x$
- at most 1 out of n $\quad x_1 + \cdots + x_n \leq 1$
- at least k out of n $\quad x_1 + \cdots + x_n \geq k$

linear constraints on binary variables to model logical relations between objects

Uncapacitated Facility Location Problem

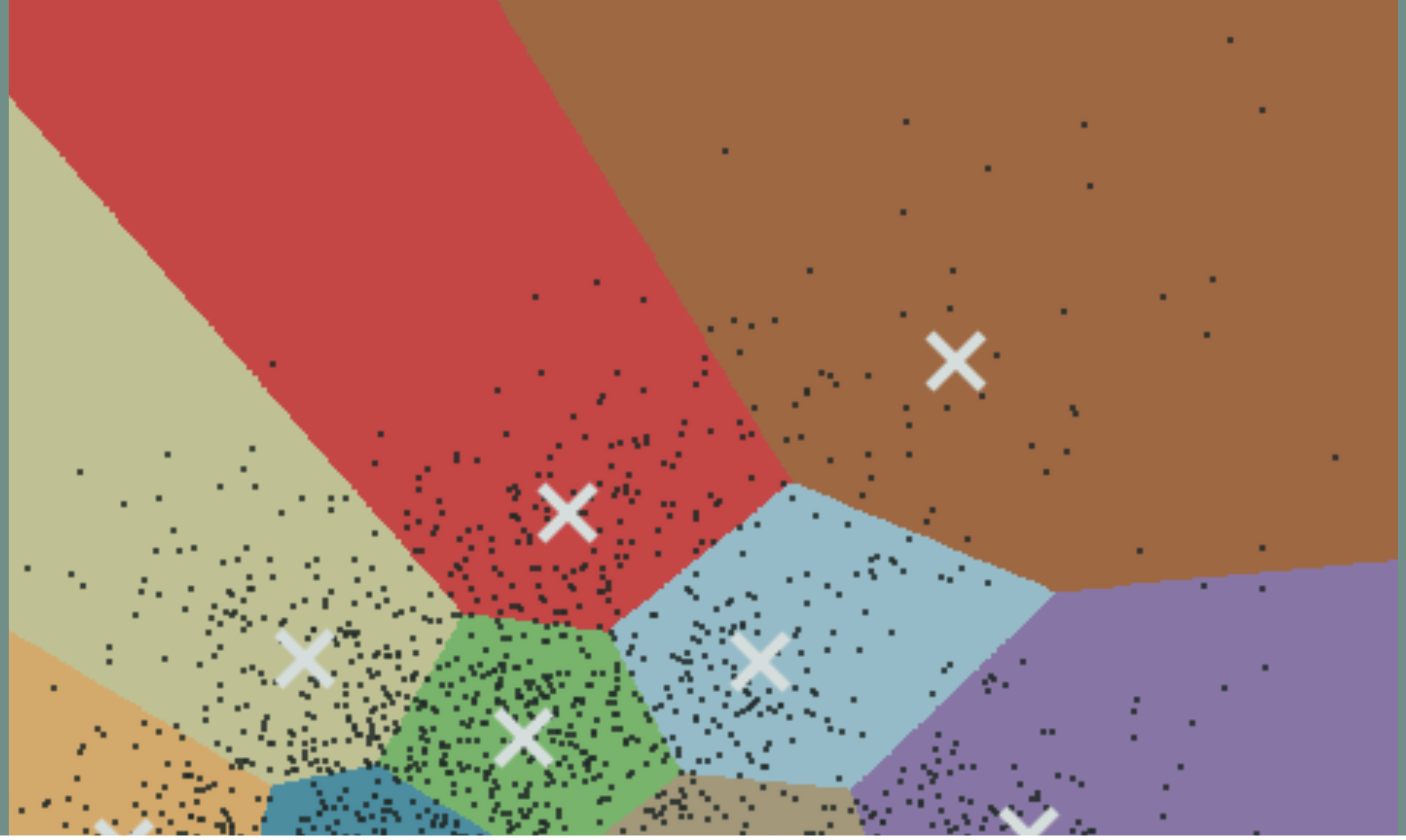
$$\min \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n}\sum_{i=1}^{m} d_{ij} y_{ij}$$

$$\begin{array}{lll}
\text{s.t.} & \displaystyle\sum_{j=1}^{n} y_{ij} = 1 \quad \text{or } \sum_{j=1}^{n} y_{ij} \geq 1 \text{ (if d positive)} & i = 1..m \\
& y_{ij} \leq x_j & j = 1..n,\ i = 1..m \\
& x_j \in \{0,1\} & j = 1..n \\
& y_{ij} \in \{0,1\} & j = 1..n,\ i = 1..m
\end{array}$$

Input n facility locations, m customers, cost c_j to open facility j, cost d_{ij} to serve customer i from facility j

Output a mimimum (opening and service) cost assignment of customers to facilities.

x_j is location j open ? y_{ij} is customer i served from j ?



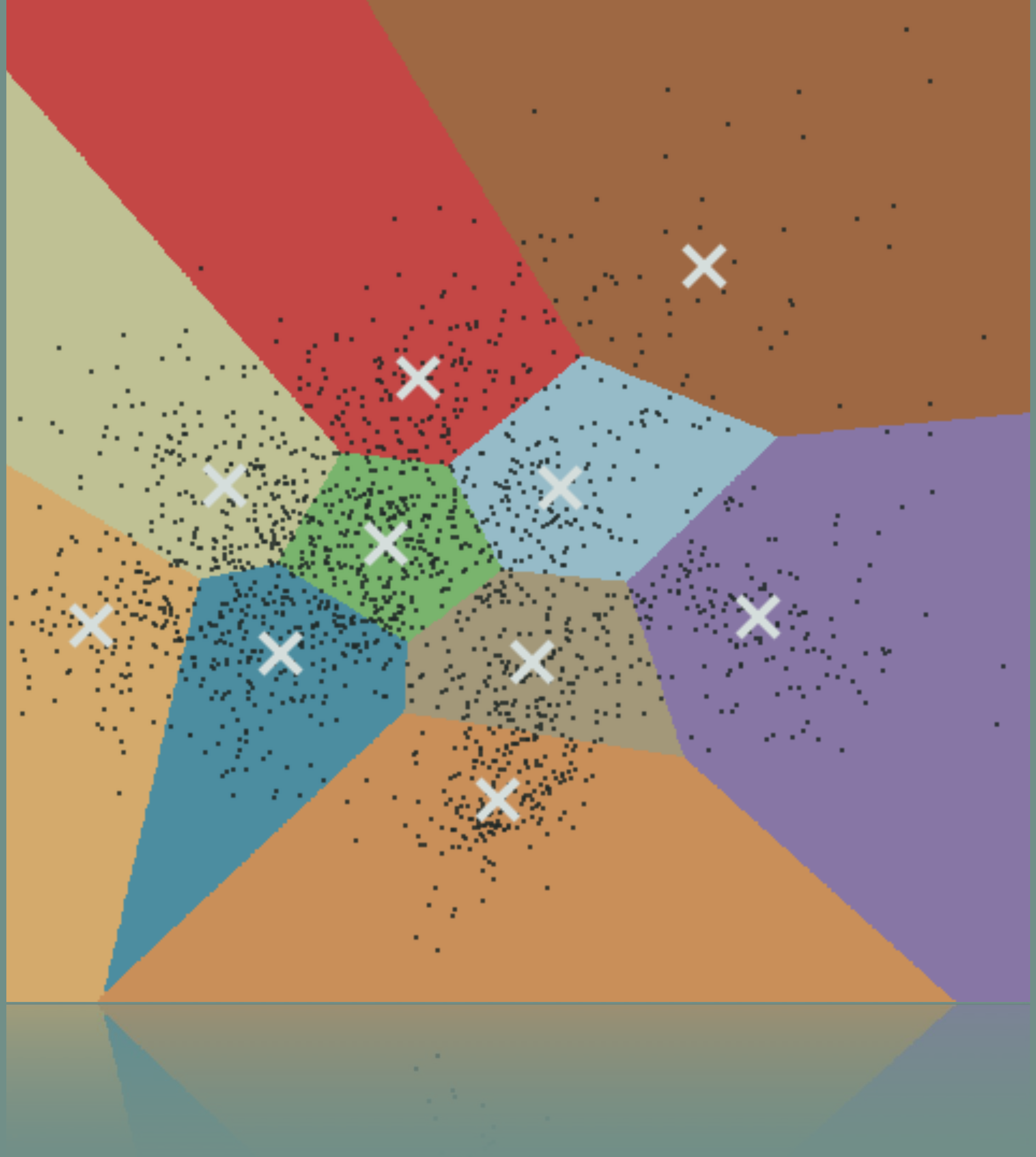
K-median clustering

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} y_{ij}$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{j=1}^{n} y_{ij} = 1 && i = 1..n \\
& y_{ij} \le x_j && i, j = 1..n \\
& \sum_{j=1}^{n} x_j = k && \\
& y_{ij} \in \{0,1\}, x_j \in \{0,1\} && i, j = 1..n
\end{aligned}$$

Input n data points, distance d_{ij} between each two points i,j, number k of clusters. Output k centers minimizing the sum of distances between each point and its nearest center.

x_j is j a center ? y_{ij} is j the nearest center of i ?



Input n data points $m_j \in \mathbb{R}^p$, a number K of clusters. Euclidean distance.

K-median clustering

Output define K points as centers so as to minimize the sum of the distances between each point and its nearest center.

K-mean clustering

Output partition the points into K sets so as to minimize the sum of the distances between each point and the mean of points in its cluster.

K-mean clustering

distances cannot be precomputed:
decision variables and nonlinear constraints

x_{jk} is j assigned to cluster k ?

y_k coordinates of the center of k ?

d_{jk} distance from j to the center of k ?

$$\min \sum_{k=1}^{K} \sum_{j=1}^{n} \boxed{x_{jk} d_{jk}}$$

nonlinear

$$s.t. \quad \boxed{d_{jk} = \sum_{i=1}^{p} (m_j^i - y_k^i)^2 \quad \forall j, k}$$

nonconvex !

$$\sum_{k=1}^{K} x_{jk} = 1 \qquad \forall j$$

$$x_{jk} \in \{0,1\}, y_k^i \in \mathbb{R}, d_{jk} \geq 0$$

K-mean clustering

d_{jk} distance from j to the center of **its cluster** k ?

"convexify"
without the integrity constraints the feasible set it convex

$$\min \sum_{k=1}^{K} \sum_{j=1}^{n} d_{jk}$$

$$s.t. \quad \boxed{d_{jk} \geq \sum_{i=1}^{p} (m_j^i - y_k^i)^2 - \bar{d}_{jk}(1 - x_{jk}) \quad \forall j, k}$$

$$\sum_{k=1}^{K} x_{jk} = 1 \quad \forall j$$

symmetry breaking
(fix an arbitrary order)

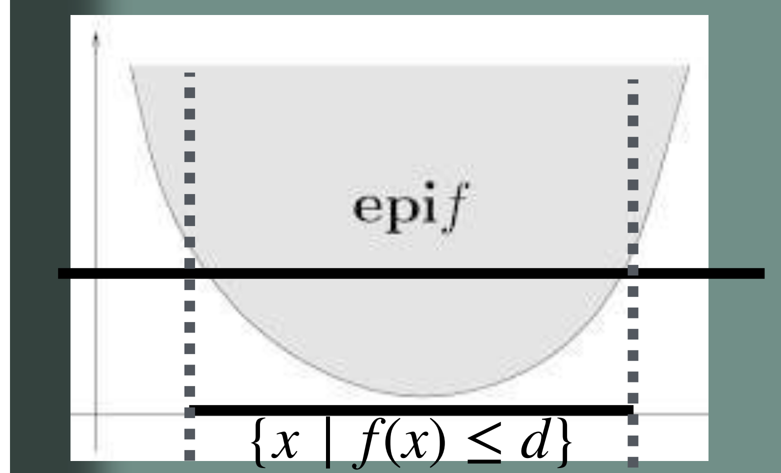
$$\boxed{y_k^1 \leq y_{k+1}^1 \quad \forall k}$$

bounding

$$\boxed{\min_j m_j^i \leq y_k^i \leq \max_j m_j^i \quad \forall i, k}$$

$$x_{jk} \in \{0,1\}, y_k^i \in \mathbb{R}, d_{jk} \geq 0$$

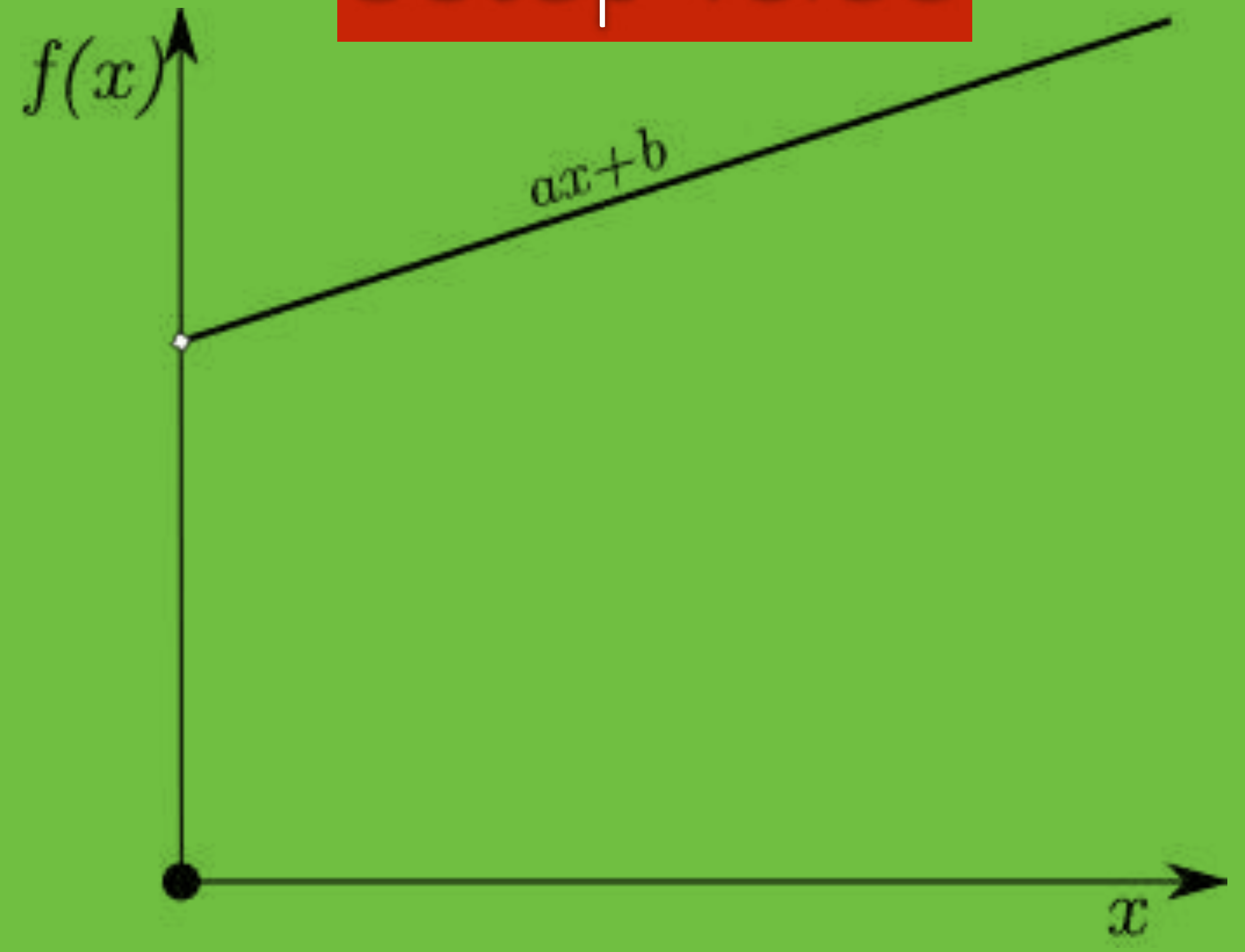
improve the model by reducing the search space



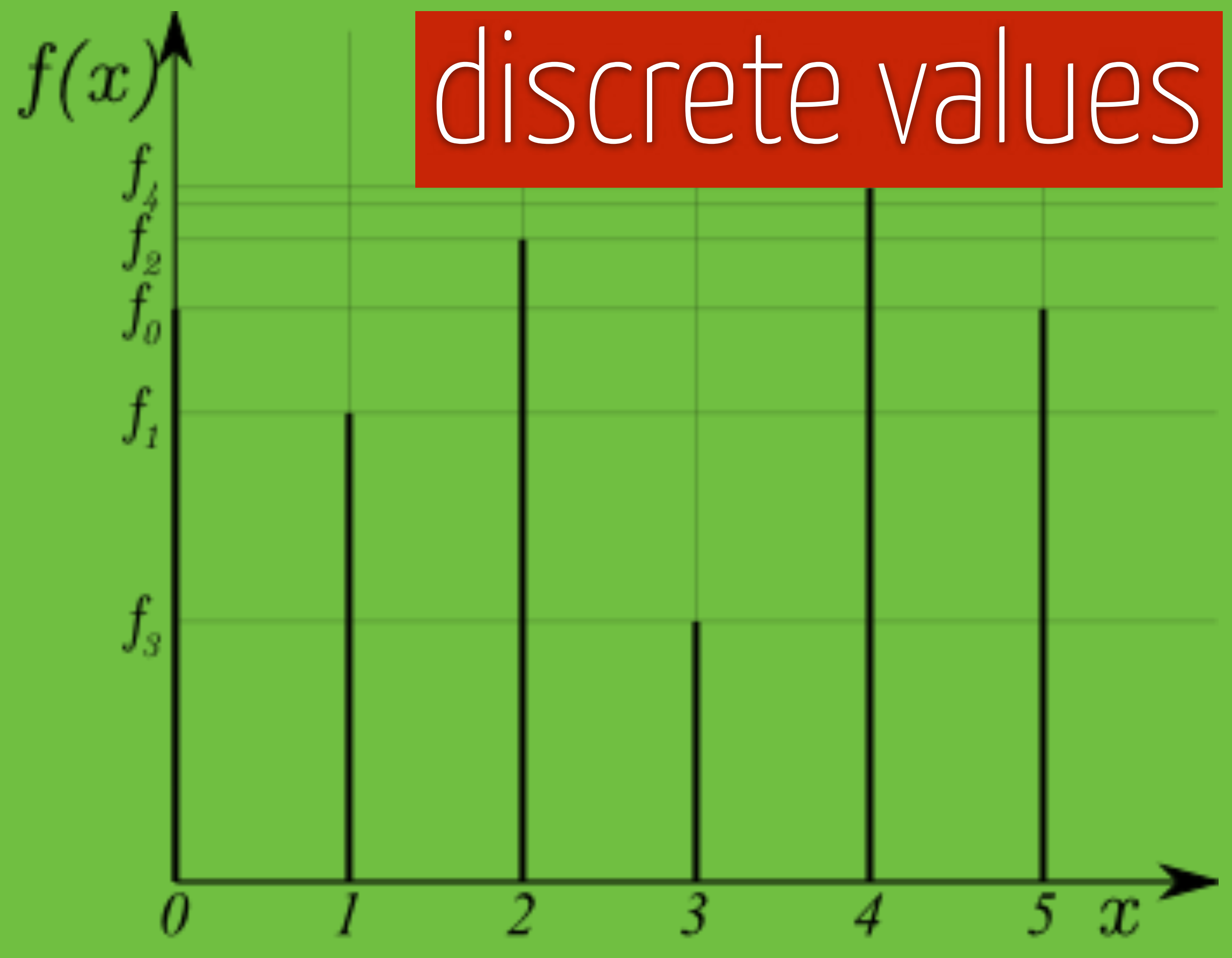
exact reformulation as a convex MINLP... still slower than specialized heuristics

modelling nonlinear functions

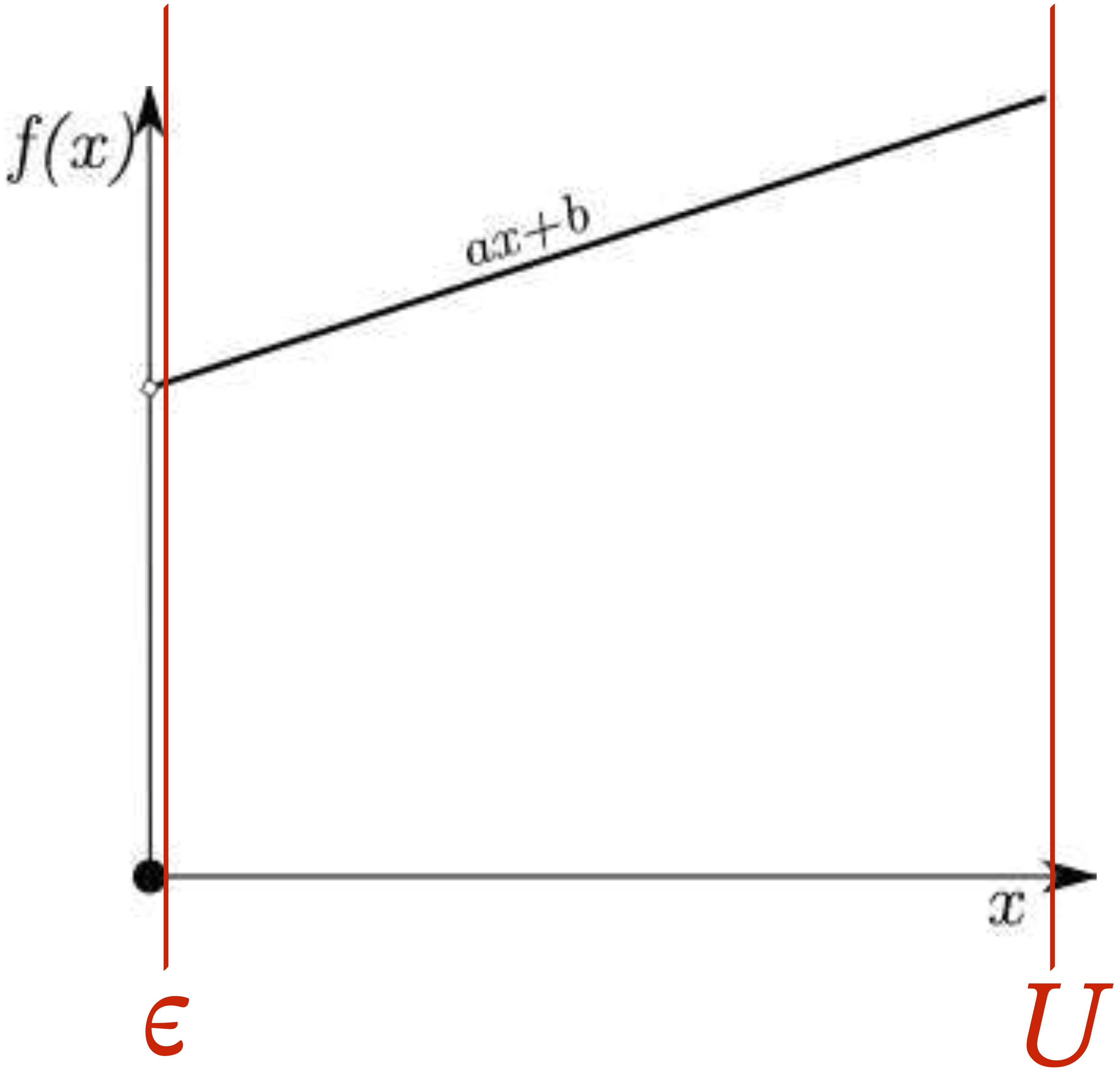
setup value



discrete values



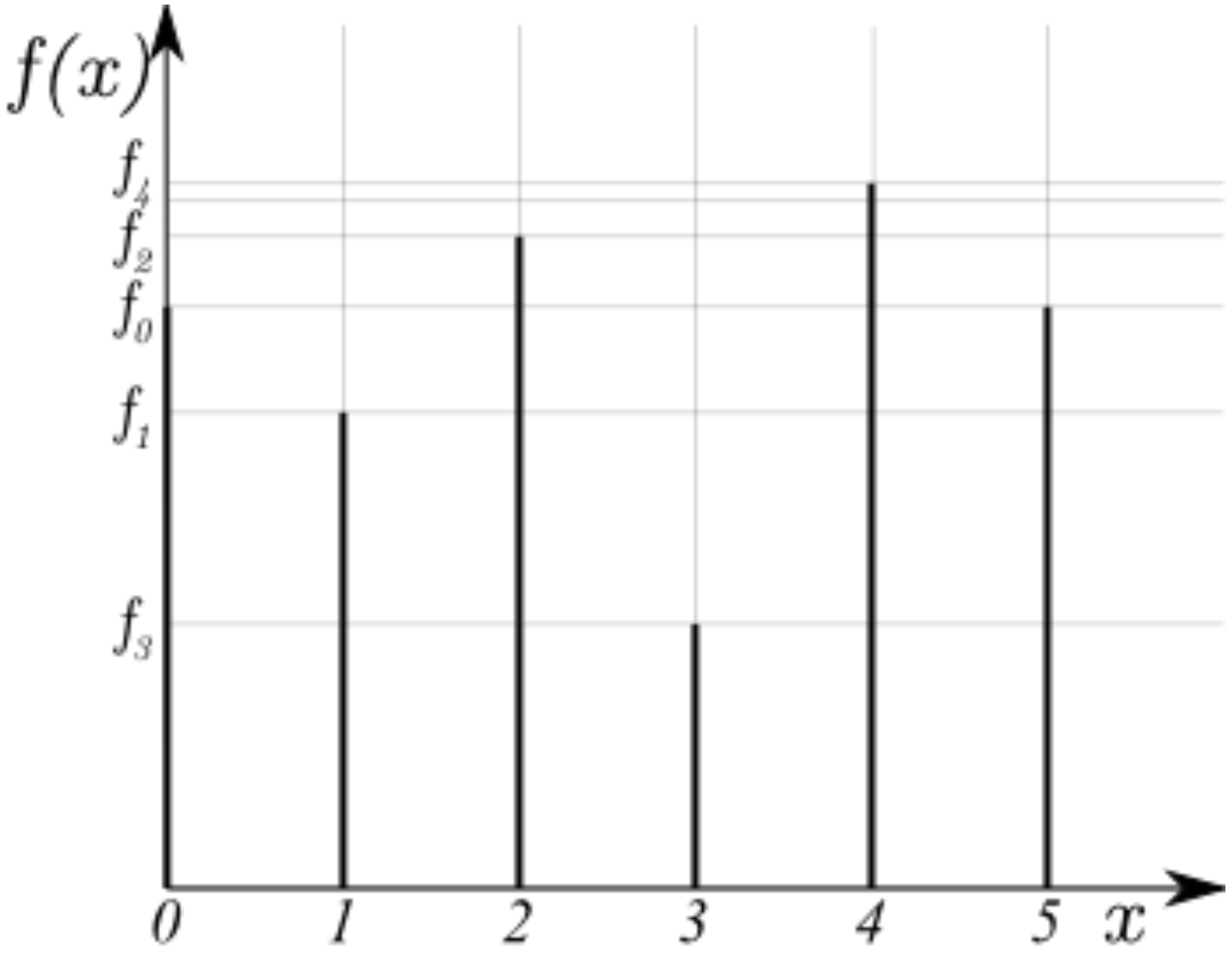
piecewise linear



setup value

$$f(x) = ax + b\delta$$
$$\epsilon\delta \leq x \leq U\delta$$
$$\delta \in \{0, 1\}$$

δ is x positive ?



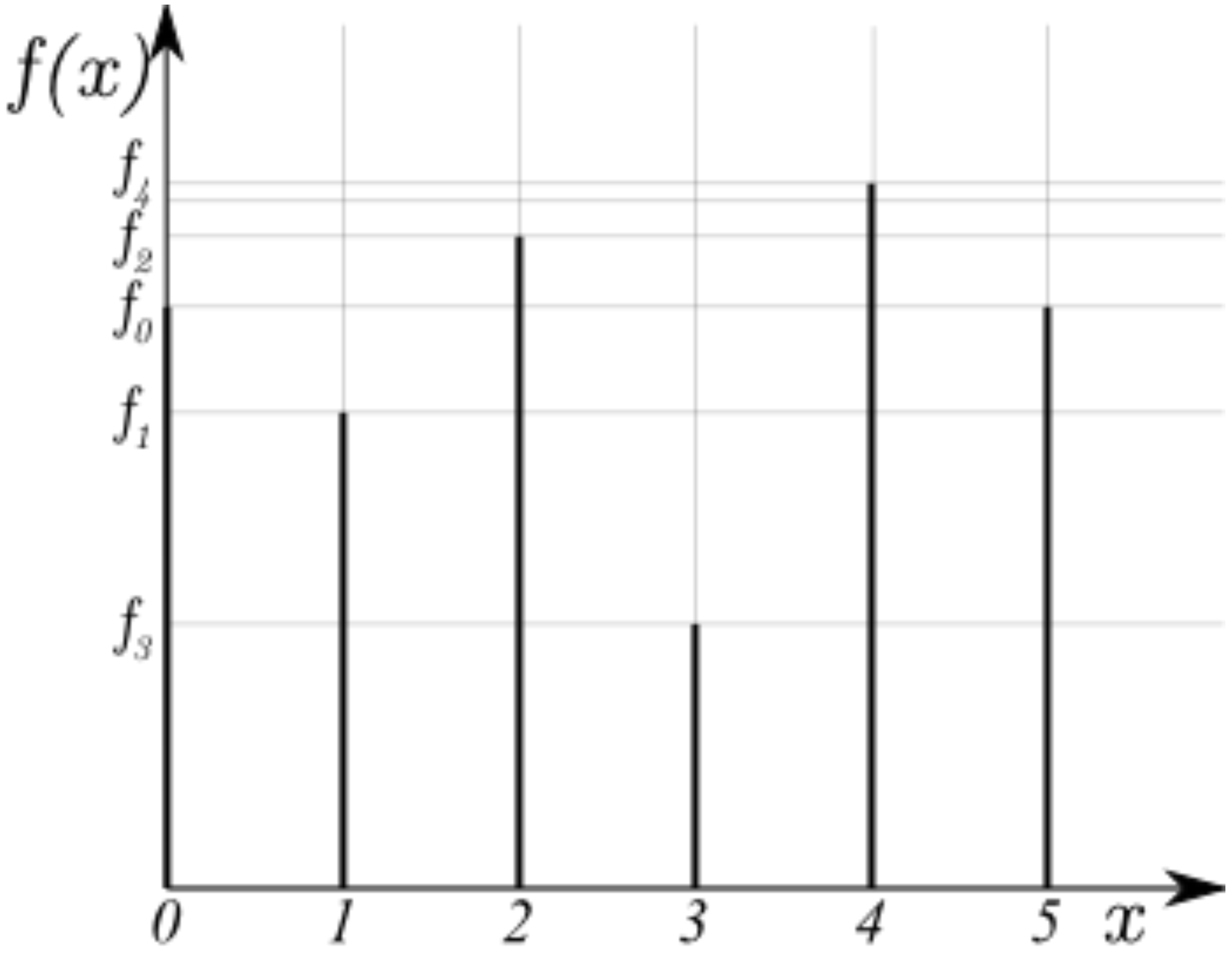
discrete values

$$f(x) = \sum_i \delta_i f_i$$
$$\sum_i i\delta_i = x$$
$$\sum_i \delta_i = 1$$
$$\delta_i \in \{0,1\} \; i = 0..n$$

δ_i is $x=i$ (and $f(x)=f_i$) ?

Special Ordered Set of type 1:

<u>ordered</u> set of variables, all zero except at most one



discrete values

$f(x) = \sum_i \delta_i f_i$

$\sum_i i\delta_i = x$

$\sum_i \delta_i \geq 1$

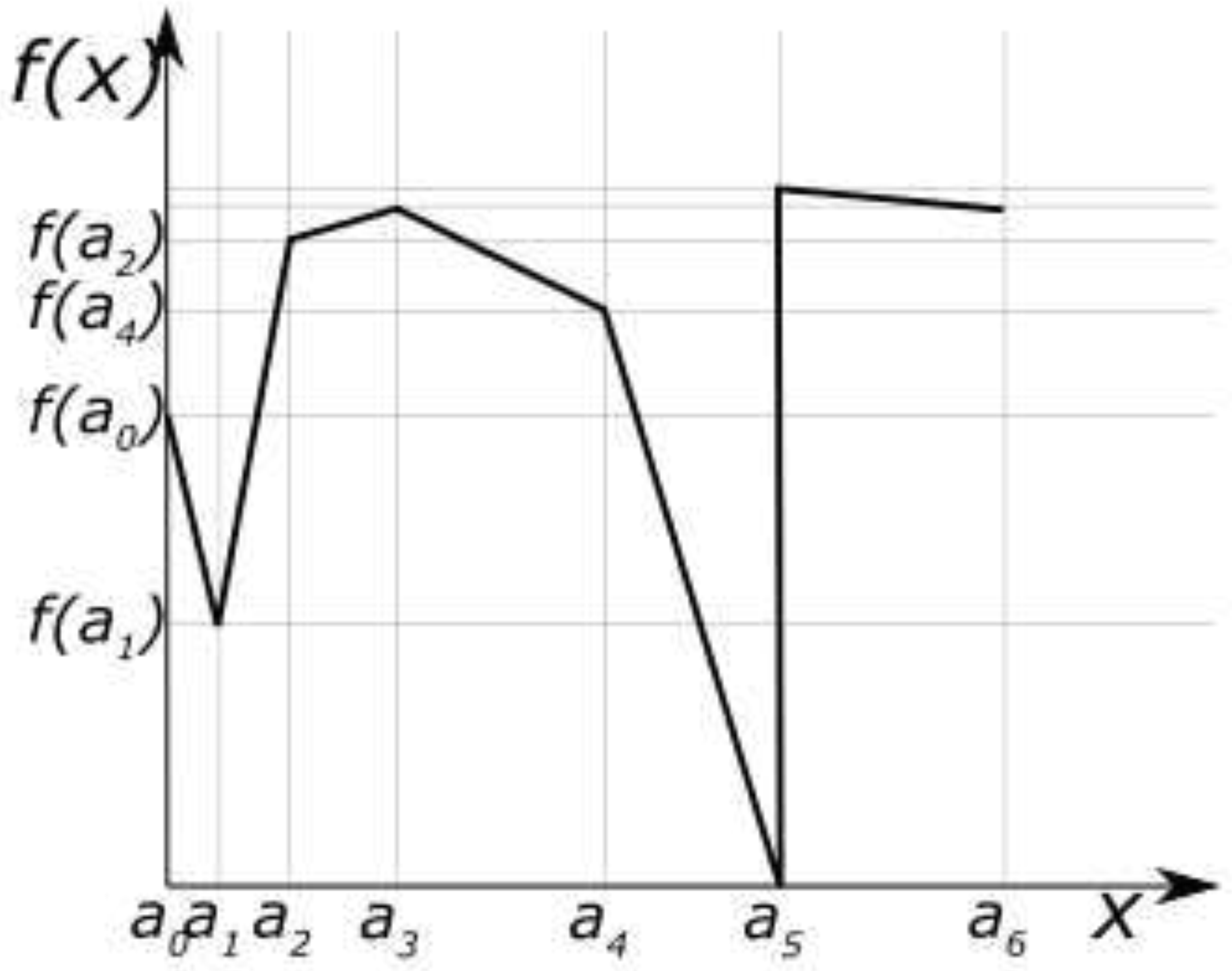
$\delta_i \in \{0, 1\}\ i = 0..n$

SOS1(δ)

δ_i is $x=i$ (and $f(x)=f_i$) ?

Special Ordered Set of type 2:

ordered set of variables, all zero except at most two consecutive



piecewise linear

$$f(x) = \sum_i \lambda_i f(a_i)$$
$$\sum_i a_i \lambda_i = x$$
$$\sum_i \lambda_i = 1$$
$$\lambda_i \in [0,1] \quad i = 0..n$$

SOS2(λ)

λ_i is $x=a_i$? (then $\lambda_i a_i + \lambda_{i+1} a_{i+1}$ in $[a_i, a_{i+1}]$ if $\lambda_i + \lambda_{i+1} = 1$)

modeling with $\mathbb{Z}$

$$x_i = 5$$

to order i is the 5th item

to count 5 items are selected

to measure time task i starts at time 5

to measure space item i is located on floor 5

$$\simeq \delta_{i5} = 1$$

Binary Integer Linear Program (BIP) $\{0,1\}^n$

Integer Linear Program (IP) $\mathbb{Z}^n$

Mixed Integer Linear Program (MIP) $\mathbb{Z}^n \cup \mathbb{Q}^n$


project:
power generation

Input

demand D_p (MW) for each period $p \in \{0, \ldots, P-1\}$ of length Δ_p (h),

N_t units of each type $t \in T$ with power output range $[\underline{L}_t, \overline{L}_t]$ (MW).

Base cost C_t^b (€/h) to operate a unit at its min level

+ cost C_t^r (€/MWh) per each extra MWh.

1/ basic power generation problem

Output

a number of units to commit and their production level to meet the demand on each period and minimize the operation costs.

- no need to know the activity of each individual unit
- be careful with equations in power (MW) or in energy (MWh)
- keep the same order of magnitude for data

Input

demand D_p (MW) for each period $p \in \{0, \ldots, P-1\}$ of length Δ_p (h),

N_t units of each type $t \in T$ with power output range $[\underline{L}_t, \overline{L}_t]$ (MW).

Base cost C_t^b (€/h) to operate a unit at its min level

+ cost C_t^r (€/MWh) per each extra MWh.

$\mathtt{x}_{tp}$ number of units of type t to commit on period p

$\mathtt{l}_{tp}$ extra output (MW) of the units of type t on period p

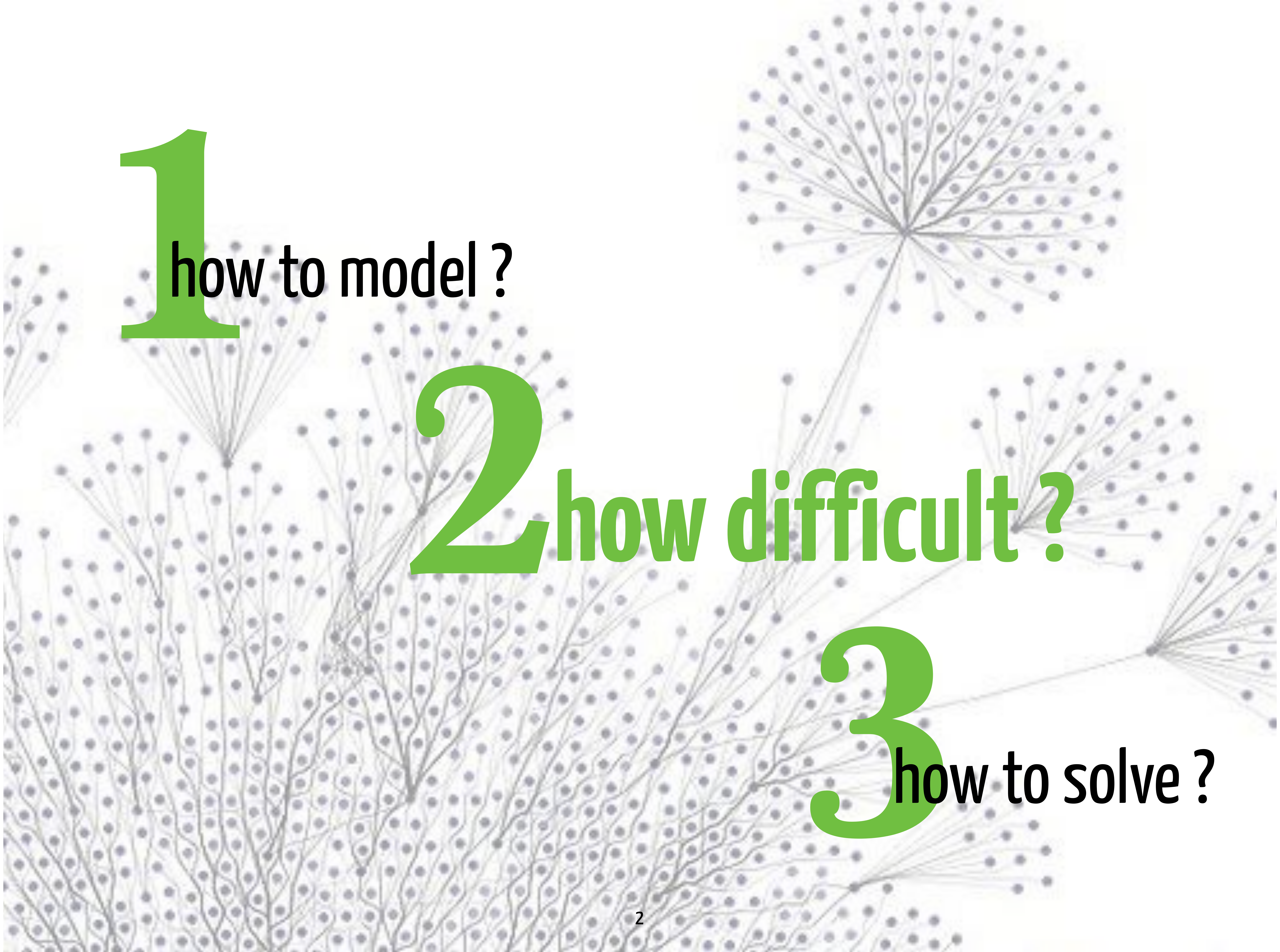
the MILP way

a practical view

(2)

Sophie Demassey 2023

CMA
Centre de Mathématiques Appliquées



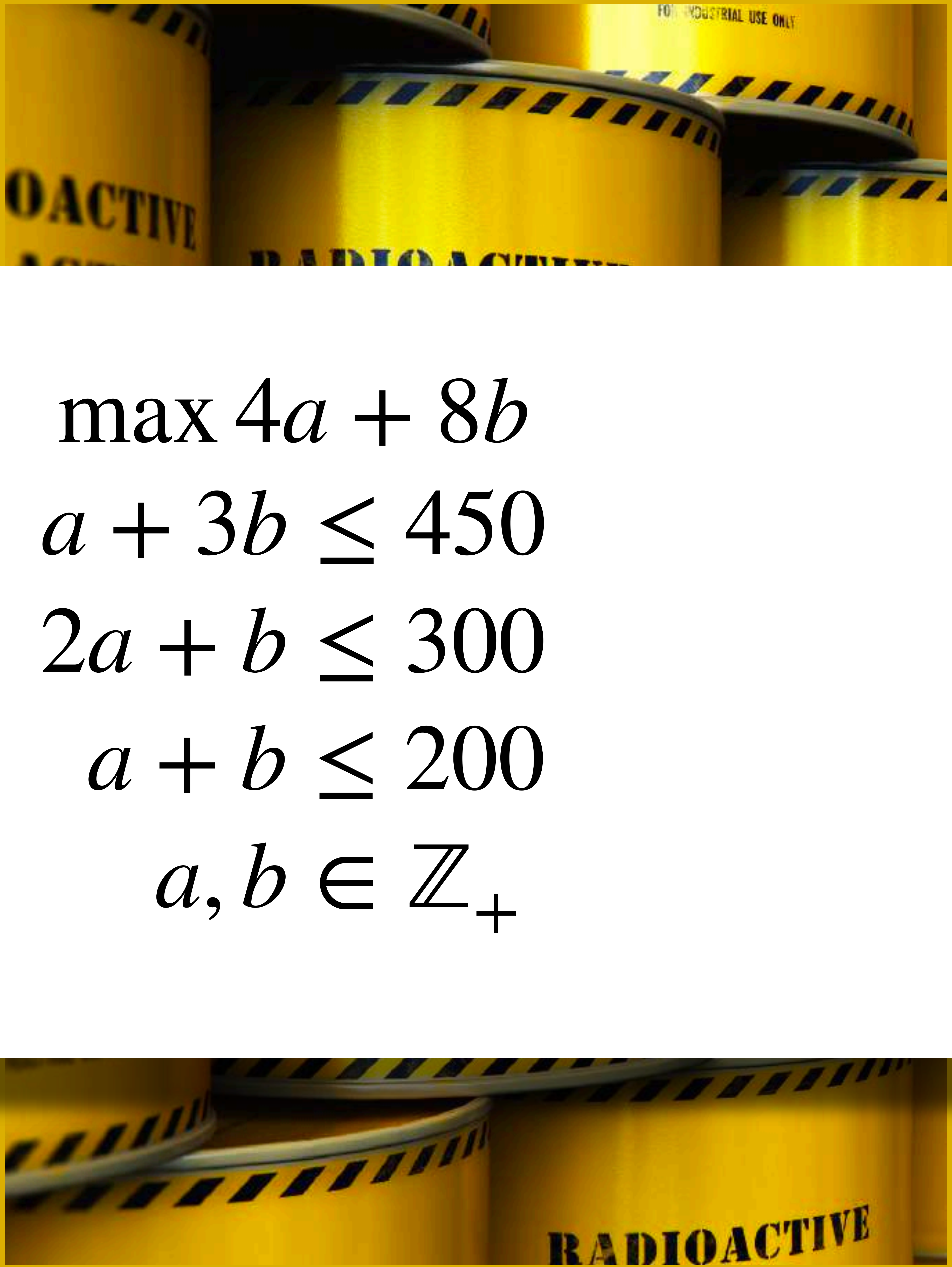
1 how to model ?

2 how difficult ?

3 how to solve ?

waste management

2 types of nuclear waste A, B with different unit profit/processing time going through 3 processes I, II and III with limited availability



$$
\begin{aligned}
\max 4a + 8b& \\
a + 3b &\leq 450 \\
2a + b &\leq 300 \\
a + b &\leq 200 \\
a, b &\in \mathbb{Z}_+
\end{aligned}
$$

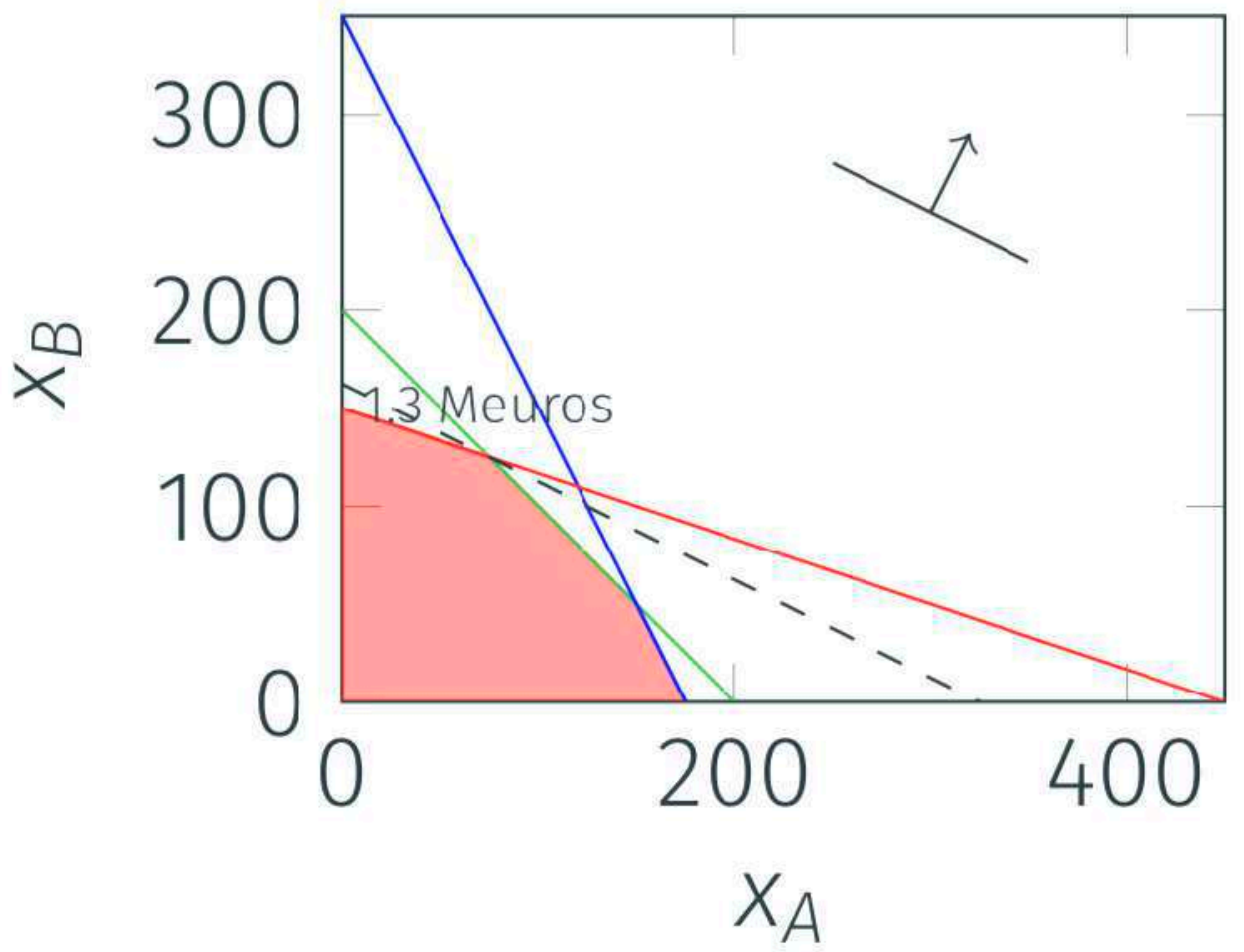
	I	II	III	profit
A	1h	2h	1h	4k€
B	3h	1h	1h	8k€
available	450h	300h	200h	

objective: maximize the profit.

waste management

$$\max 4a + 8b$$
$$a + 3b \leq 450 \quad \text{active}$$
$$2a + b \leq 300$$
$$a + b \leq 200 \quad \text{active}$$
$$a, b \geq 0$$
$$\cancel{a, b \in \mathbb{Z}}$$

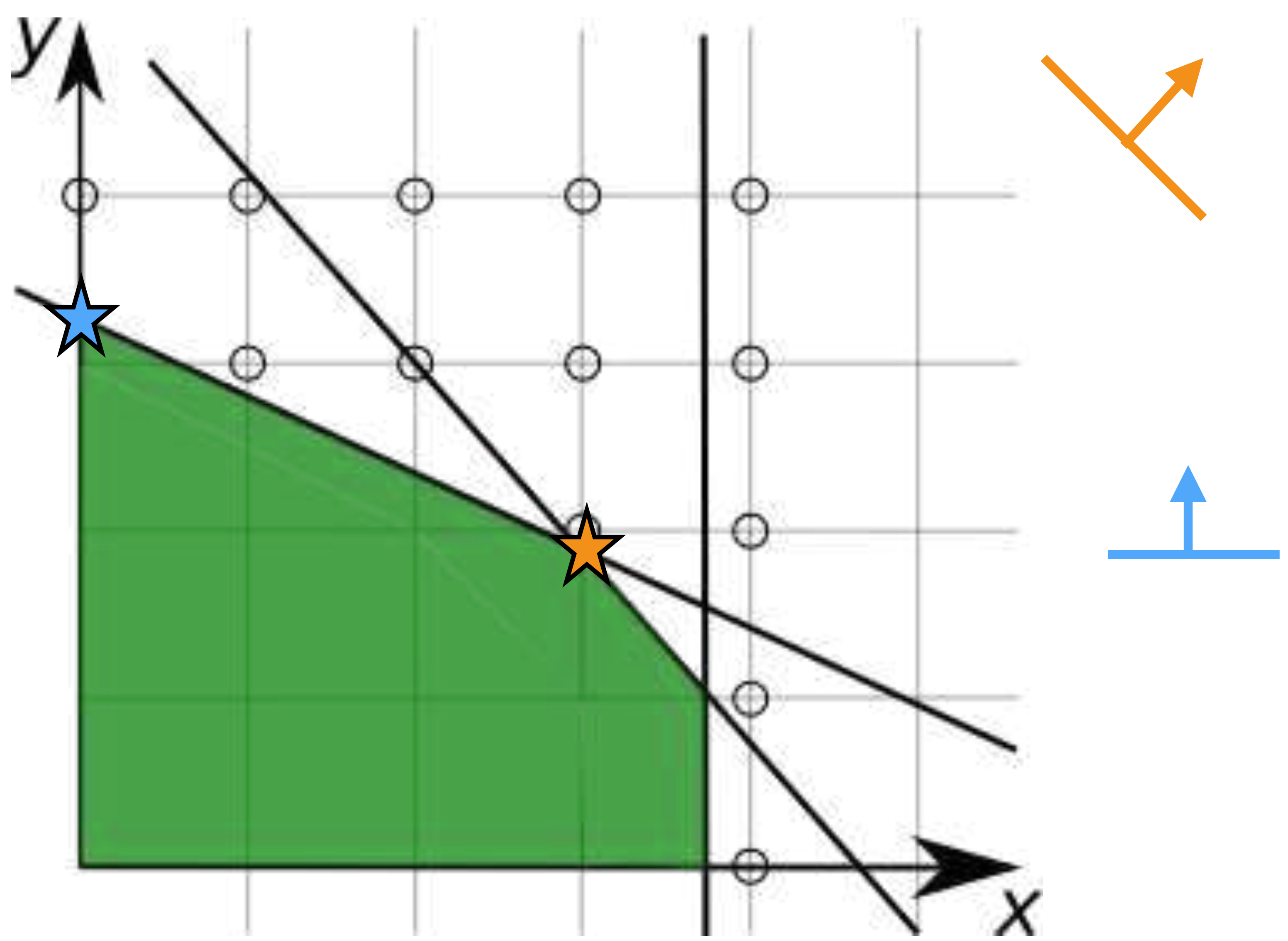
LP relaxation



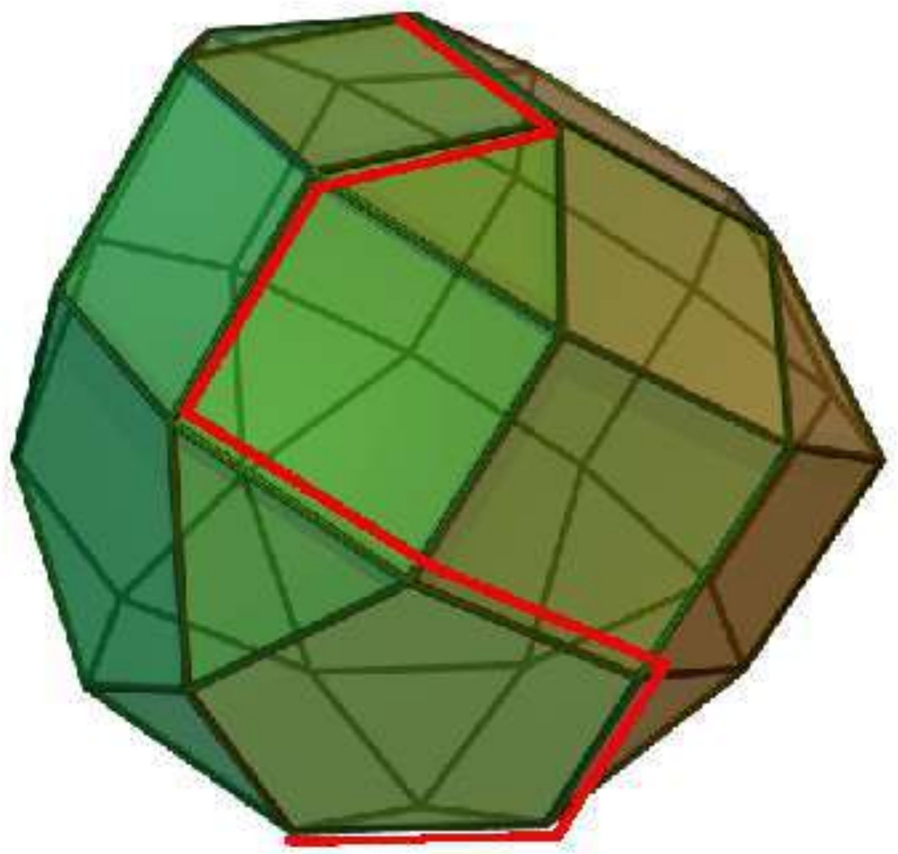
LP solution: $a^* + 3b^* = 450,\ a^* + b^* = 200 \Rightarrow (a^*, b^*) = (\frac{150}{2}, \frac{250}{2})$

Linear Programming cheat sheet

LP is easy



- MILP without integrality = LP-relaxation
- linear inequality = halfspace
- LP feasible set = polyhedron
- convex optimization
- if LP is feasible and bounded, at least one vertex is optimal
- primal simplex algorithm: visit adjacent vertices as cost decreases
- interior point method runs in polynomial time (but simplex often faster)
- strong duality: $\min_x \{cx \mid Ax \geq b, x \geq 0\} = \max_u \{ub \mid uA \leq c, u \geq 0\}$



Market Split Problem

$$
\begin{aligned}
\min \ & \sum_{j=1}^{m} s_j^+ + s_j^- \\
\text{s.t.} \ & \sum_{i=1}^{n} a_{ij} x_i + s_j^+ - s_j^- = \frac{d_j}{2} && j = 1..m \\
& x_i \in \{0, 1\} && i = 1..n \\
& s_j^+ \geq 0, s_j^- \geq 0 && j = 1..m
\end{aligned}
$$

x_i is retailer i assigned to division 1 ?

s_j gap to the 50% split goal for product j

Input 1 company, 2 divisions, m products with availabilities d_j, n retailers with demands a_{ij} in each product j.

Output an assignment of the retailers to the divisions approaching a 50/50 production split.

MIPLIB markshare_5_0

```
[sofdem:~/Documents/Code/gurobi]$ gurobi.sh mymip.py markshare_5_0.mps.gz
Changed value of parameter Presolve to 0
   Prev: -1   Min: -1   Max: 2   Default: -1
Optimize a model with 5 rows, 45 columns and 203 nonzeros
Found heuristic solution: objective 5335
Variable types: 5 continuous, 40 integer (40 binary)

Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0    5 5335.00000    0.00000   100%     -    0s

*62706364 28044              38      1.0000000    0.00000   100%   2.1 1241s

Explored 233848403 nodes (460515864 simplex iterations) in 3883.56 seconds
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 1.000000000000e+00, best bound 1.000000000000e+00, gap 0.0%
Optimal objective: 1
```

Input 5 products, 40 retailers

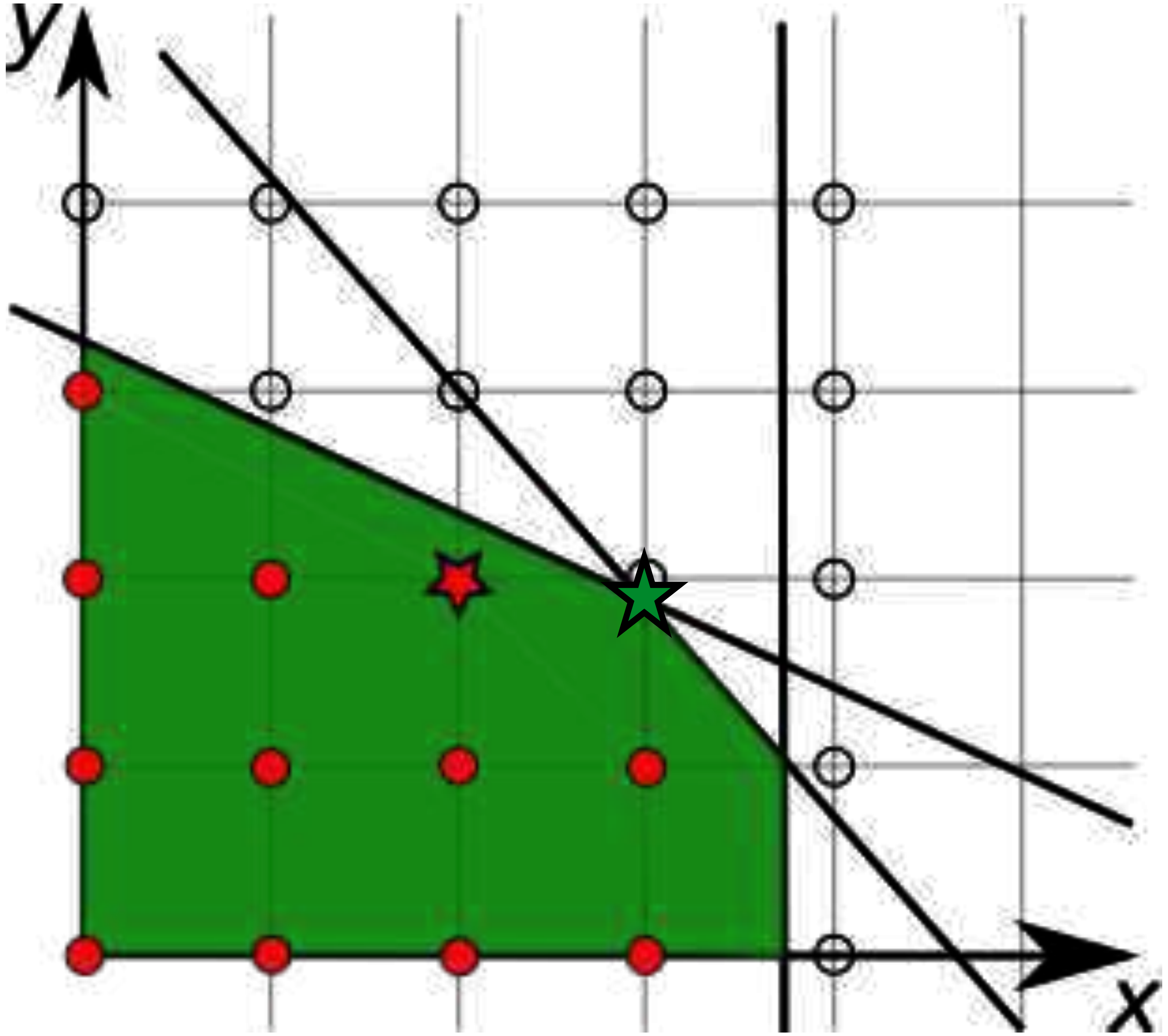
Output . . . . (hold the line please) . . .

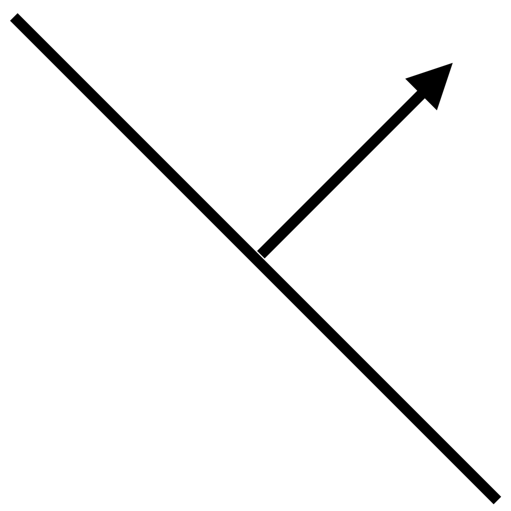
Int Opt = 1

Time to the solution = 20 minutes

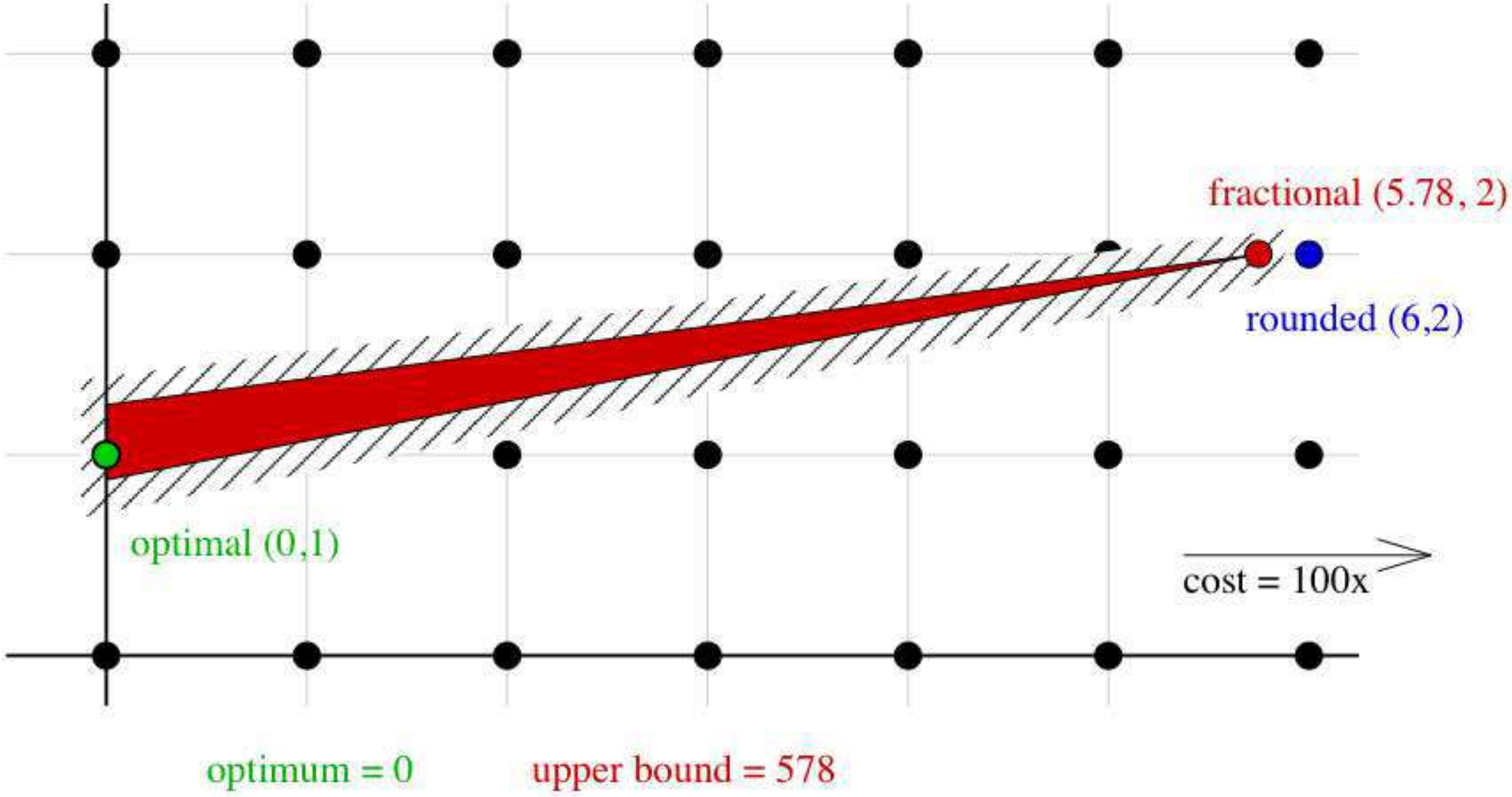
Time of optimality proof > 1 hour

MILP ≠ LP-relaxation





MILP ≠ round LP-relaxation



general MILP is NP-hard

- small problems are easy
- some specific problems are easy

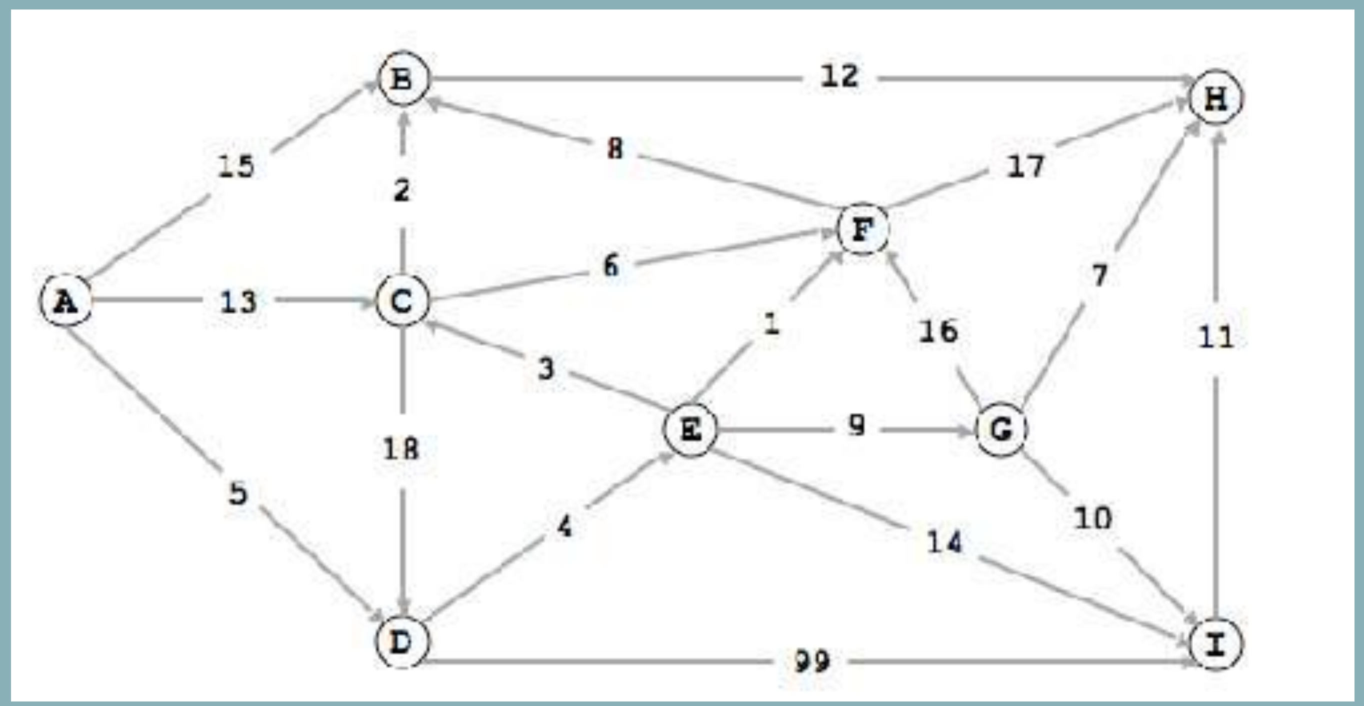
1||Cmax Scheduling Problem

$$
\begin{aligned}
&\min\ s_{n+1} \quad = p_1 + \ldots + p_n && \\
&\text{s.t. } s_{n+1} \geq s_j + p_j && j = 1..n \\
&\quad s_j - s_i \geq M x_{ij} + (p_i - M) && i,j = 1..n \\
&\quad x_{ij} + x_{ji} = 1 && i,j = 1..n; i < j \\
&\quad s_j \in \cancel{\mathbb{Z}_+} \ \geq 0 && j = 1..n+1 \\
&\quad x_{ij} \in \{0,1\} && i,j = 1..n
\end{aligned}
$$

Input n tasks, duration pi for each task i, one machine

Output a minimal makespan schedule of the tasks on the machine without overlap

x_{ij} does i precede j ? s_j starting time of j



Capacitated Transhipment Problem

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

$$\text{s.t.} \sum_{j\in\delta^+(i)} x_{ij} - \sum_{j\in\delta^-(i)} x_{ij} = b_i \qquad i \in V$$

$$x_{ij} \le h_{ij} \qquad (i,j) \in A$$

$$x_{ij} \in \cancel{\mathbb{Z}_+} \geq 0 \qquad (i,j) \in A$$

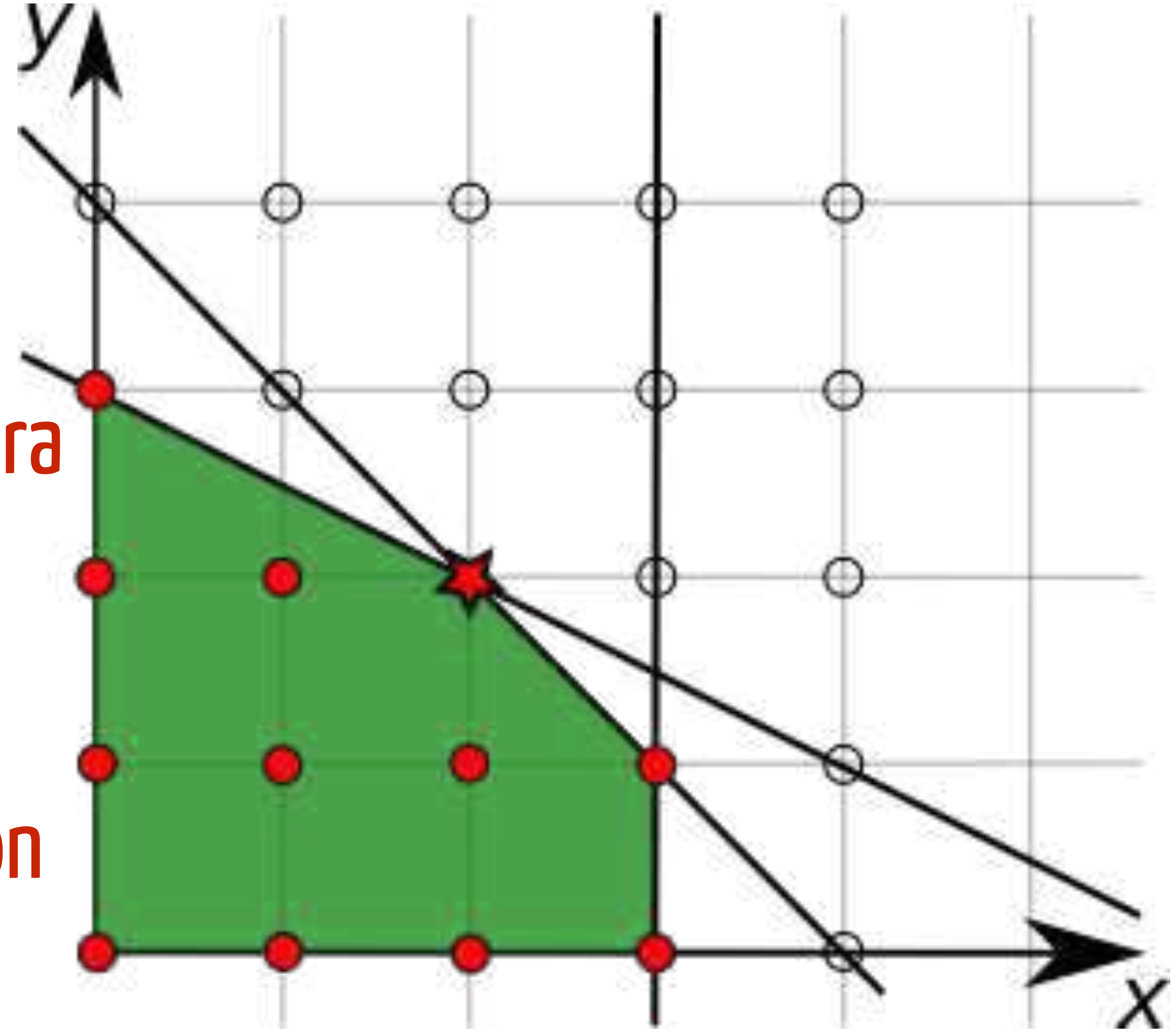
Input digraph (V,A), demand or supply b_i at each node i, capacity h_{ij} and unit flow cost c_{ij} for each arc (i,j)

Output a mimimum cost integer flow to satisfy the demand

x_{ij} flow on arc (i,j)

LP = ILP sometimes

integral polyhedra
=
convex hull
=
ideal formulation



totally unimodular matrix
(theory)

$$(P) = \max\{\ cx \mid Ax \leq b, x \in \mathbb{Z}^n_+\ \}$$

- basic feasible solutions of the LP relaxation $(\bar{P})$ take the form: $\bar{x} = (\bar{x}_B, \bar{x}_N) = (B^{-1}b, 0)$ where B is a square submatrix of (A, I_m)
- Cramer's rule: $B^{-1} = B^*/det(B)$ where B^* is the adjoint matrix (made of products of terms of B)
- Proposition: if (P) has integral data (A, b) and if $det(B) = \pm 1$ then $\bar{x}$ is integral

Definition

A matrix A is totally unimodular (TU) if every square submatrix has determinant $+1$, -1 or 0.

Proposition

If A is TU and b is integral then any optimal solution of $(\bar{P})$ is integral.

totally unimodular matrix
(practice)

How to recognize TU ?

Sufficient condition

A matrix A is TU if

- all the coefficients are $+1$, -1 or 0
- each column contains at most 2 non-zero coefficient
- there exists a partition (M_1, M_2) of the set M of rows such that each column j containing two non zero coefficients satisfies $\sum_{i \in M_1} a_{ij} - \sum_{i \in M_2} a_{ij} = 0$.

Proposition

A is TU $\iff$ A^t is TU $\iff$ (A, I_m) is TU
where A^t is the transpose matrix, I_m the identitiy matrix

Interlude

Show that the Transhipment ILP is ideal
Show that the Scheduling ILP is NOT ideal

project:
power generation

Input electric power demand D_p for each time period $p \in \{0,\ldots,P-1\}$ of Δ_p hours, N_t power generation units of each type $t \in T$ with power output range $[\underline{L}_t, \overline{L}_t]$. A reserve factor F. A base hourly cost C_t^b to operate a unit at its min level + a cost C_t^r per extra MWh.

a basic power generation problem

Output a number of units to commit and their production level to meet both the demand and the reserve on each period so as to minimize the operation costs.

- no need to know the activity of each individual unit
- be careful with equations in power or in energy
- choose units to enforce the homogeneity of the values

Input electric power demand D_p (MW) for each time period $p \in \{0,\dots,P-1\}$ of Δ_p hours, N_t power generation units of each type $t \in T$ with power output range $[\underline{L}_t, \overline{L}_t]$ (MW). A reserve factor $F \in [0,1]$. A base hourly cost C_t^b (eur/h/unit) to operate a unit at its min level + a cost C_t^r (eur/MWh) per extra MWh.

$$\min \sum_{t,p} (\Delta_p C_t^b x_{tp} + \Delta_p C_t^r l_{tp})$$
$$\sum_t (\underline{L}_t x_{tp} + l_{tp}) \geq D_p \ \forall p$$
$$\sum_t \overline{L}_t x_{tp} \geq (1+F) * D_p \ \forall p$$
$$0 \leq l_{tp} \leq (\overline{L}_t - \underline{L}_t) x_{tp} \ \forall t,p$$
$$0 \leq x_{tp} \leq N_t \ \forall t,p$$
$$x_{tp} \in \mathbb{Z} \ \forall t,p$$

x_{tp} number of committed units of type t on period p

l_{tp} extra load (MW) of all units of type t on period p

startup costs

Input the number A_t of active units at time 0, a positive startup cost C^s_t to turn a unit on.

$y_{tp} \geq \max(0, x_{tp} - x_{tp-1})$ in any feasible solution and
$y_{tp} \leq \max(0, x_{tp} - x_{tp-1})$ in any optimal solution (prove it)

$$\min \ldots + \sum_{t,p} C^s_t y_{tp}$$

$$y_{t0} \geq x_{t0} - A_t \ \forall t$$
$$y_{tp} \geq x_{tp} - x_{tp-1} \ \forall t, p \neq 0$$

$$y_{tp} \in \mathbb{Z}_+ \ \forall t, p$$

y_{tp} number of units of type t starting on period p

hydro power generation

Input hydro units $h \in H$ with fixed power output L_h (MW), hourly reservoir depth reduction R_h (m/h) and hourly cost C_h^b (eur/h) when on, and with startup cost C_h^s (eur); A_h the commitment status (true/false) of the unit before time 0. At end, the unique reservoir must be replenished to its initial level; pumping electric consumption E (MWh/m) for 1 meter depth increase.

x_{hp} hydro unit h committed on period p

y_{hp} hydro unit h started on period p

u_p reservoir depth increase (m/h) by pumping on period p

hydro power generation

$$min \ldots + \sum_{h,p} (\Delta_p C^b_h x_{hp} + C^s_h y_{hp})$$

$$\sum_t (\underline{L}_t x_{tp} + l_{tp}) + \sum_h L_h x_{hp} \geq D_p + E u_p \ \forall p$$

$$\sum_t \overline{L}_t x_{tp} + \sum_h L_h \geq (1 + F) * D_p \ \forall p$$

$$\sum_p \sum_h R_h \Delta_p x_{hp} = \sum_p \Delta_p u_p$$

$$y_{hp} \geq x_{hp} - x_{hp-1} \ \forall h, p$$

$$x_{h(-1)} = A_h$$

$$u_p \in \mathbb{R}_+ \ \forall p$$

$$x_{hp}, y_{hp} \in \{0,1\} \ \forall h, p$$

Input (noncyclic)

up/down times: minimum time Δ_t^+, Δ_t^- (h) unit $t \in T$ may remain on or off; time $\Delta_{0it}^+, \Delta_{0it}^-$ (h) the ith unit of type $t \in T$ has been on/off before period 0.

ramp rates: maximum power increase/decrease L_t^+, L_t^- (MW) between two consecutive periods; maximum power L_t^S (MW) when turned on; maximum power L_t^E (MW) before turned off; load L_{0it} (MW) for ith unit of type $t \in T$ before period 0.

Input (cyclic)

the status before period 0 $(\Delta_{0it}^+, \Delta_{0it}^-, L_{0it})$ are duplicated from period P-1.

Physical limits of the units: minimum up/down times and maximum ramp up/down rates

commitment must be monitored for units individually

minimum uptime

Let $P_t^+(p) = \{0 \leq p' \leq p \mid \sum_{k=p'}^{p-1} \Delta_k < \Delta_t^+\}.$

Show that an unit of type t cannot been turned on more than once during $P_t^+(p)$.

Show that if an unit of type t is off at time p then it has not been turned on at any time $p' \in P_t^+(p)$.

Reformulate these assertions as a linear relation between the binary variables modelling the unit status and status change at appropriate periods.

minimum uptime (noncyclic case)

If unit i of type t has been on for exactly $\Delta^+_{0it} > 0$ hours before time 0, what can you say about its status and status change at any period in

$$P^+_{it} = \{p \geq 0 \mid \sum_{k=0}^{p-1} \Delta_k < \Delta^+_t - \Delta^+_{0it}\}?$$

Fix binary variables modelling the status and status change of a unit at given periods according to this assertion.

the MILP way

a practical view

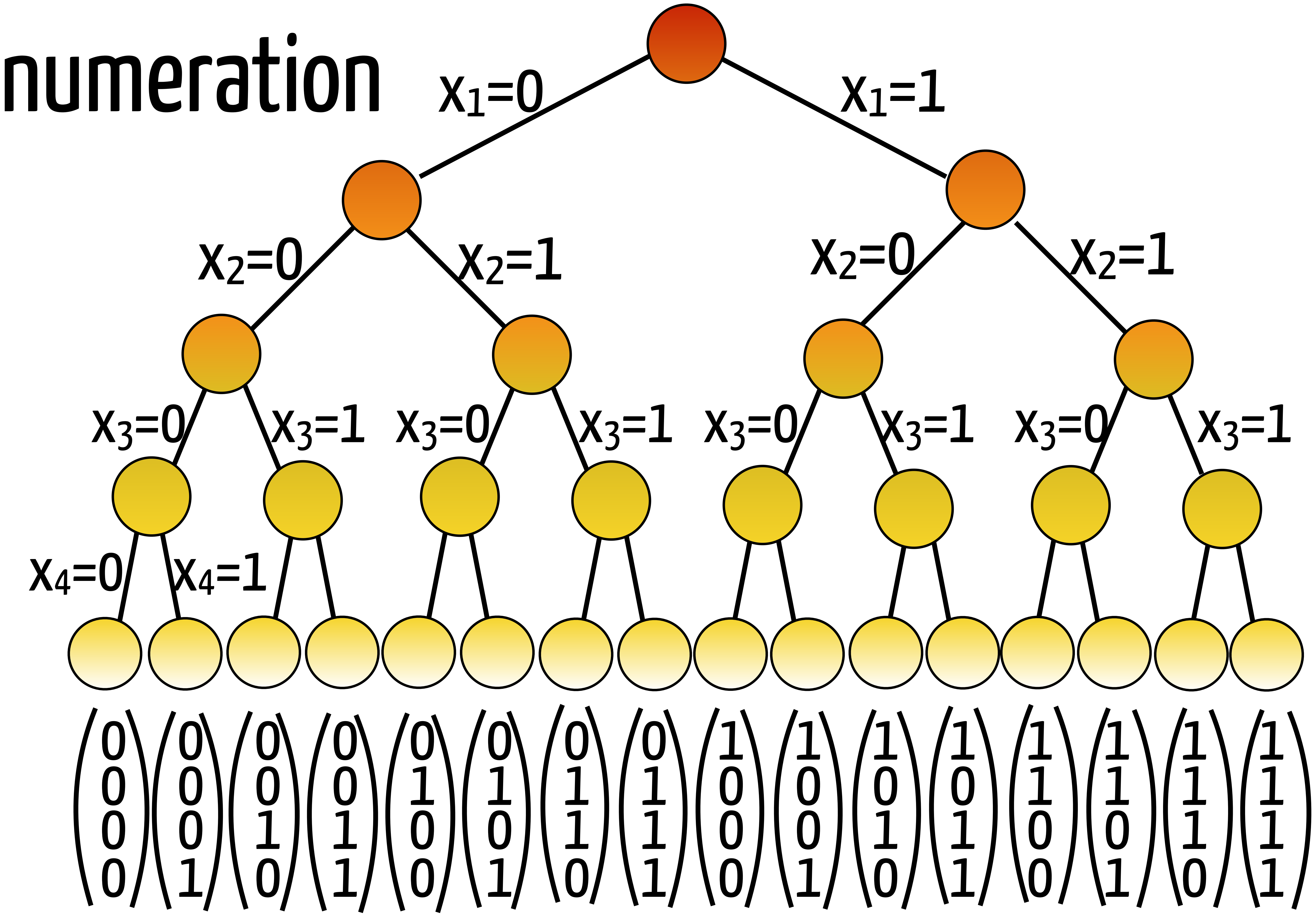
(3)

Sophie Demassey 2023

Centre de Mathématiques Appliquées


1 how to model ?
2 how difficult ?
3 how to solve ?

Complete enumeration

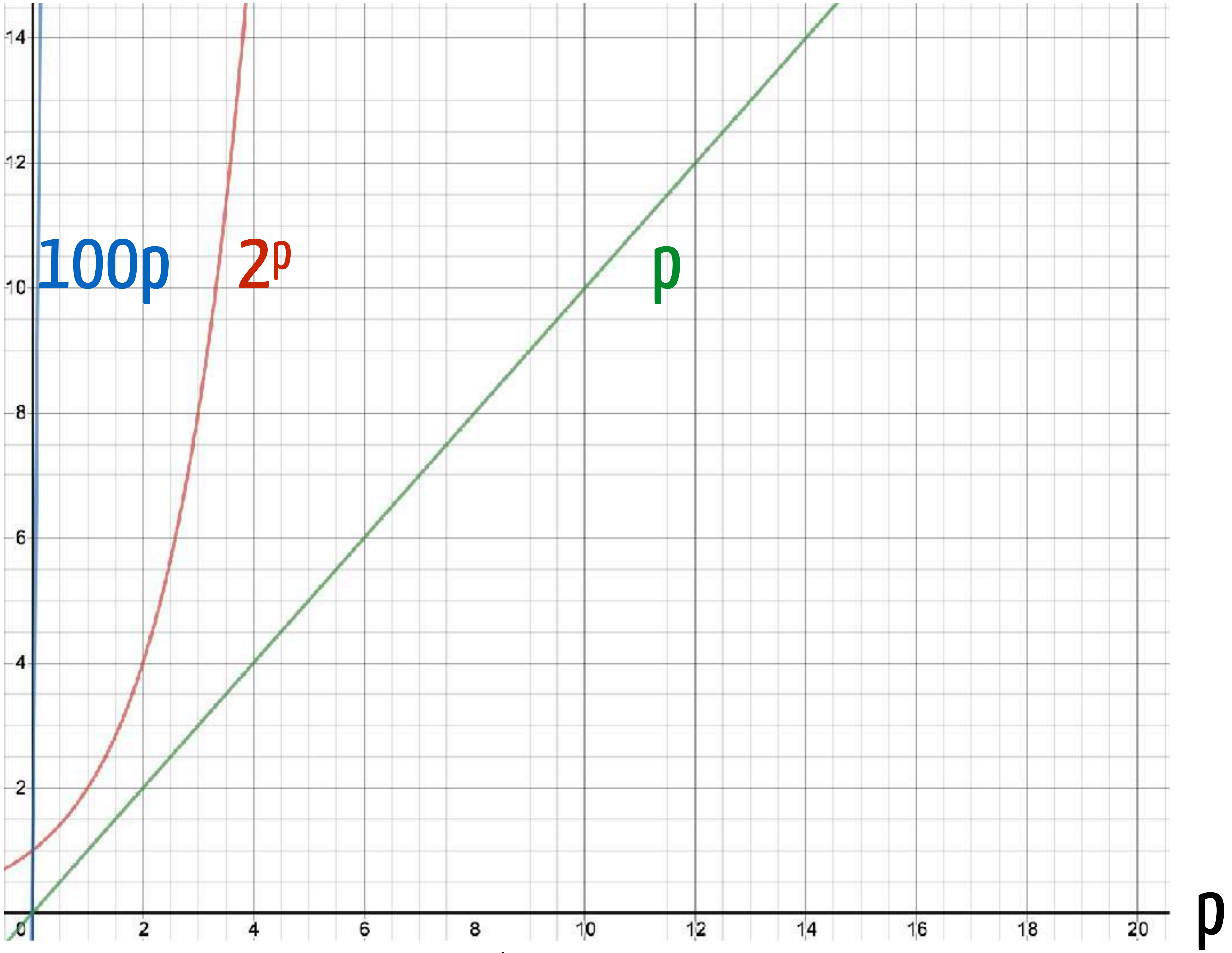


MILP with p binaries

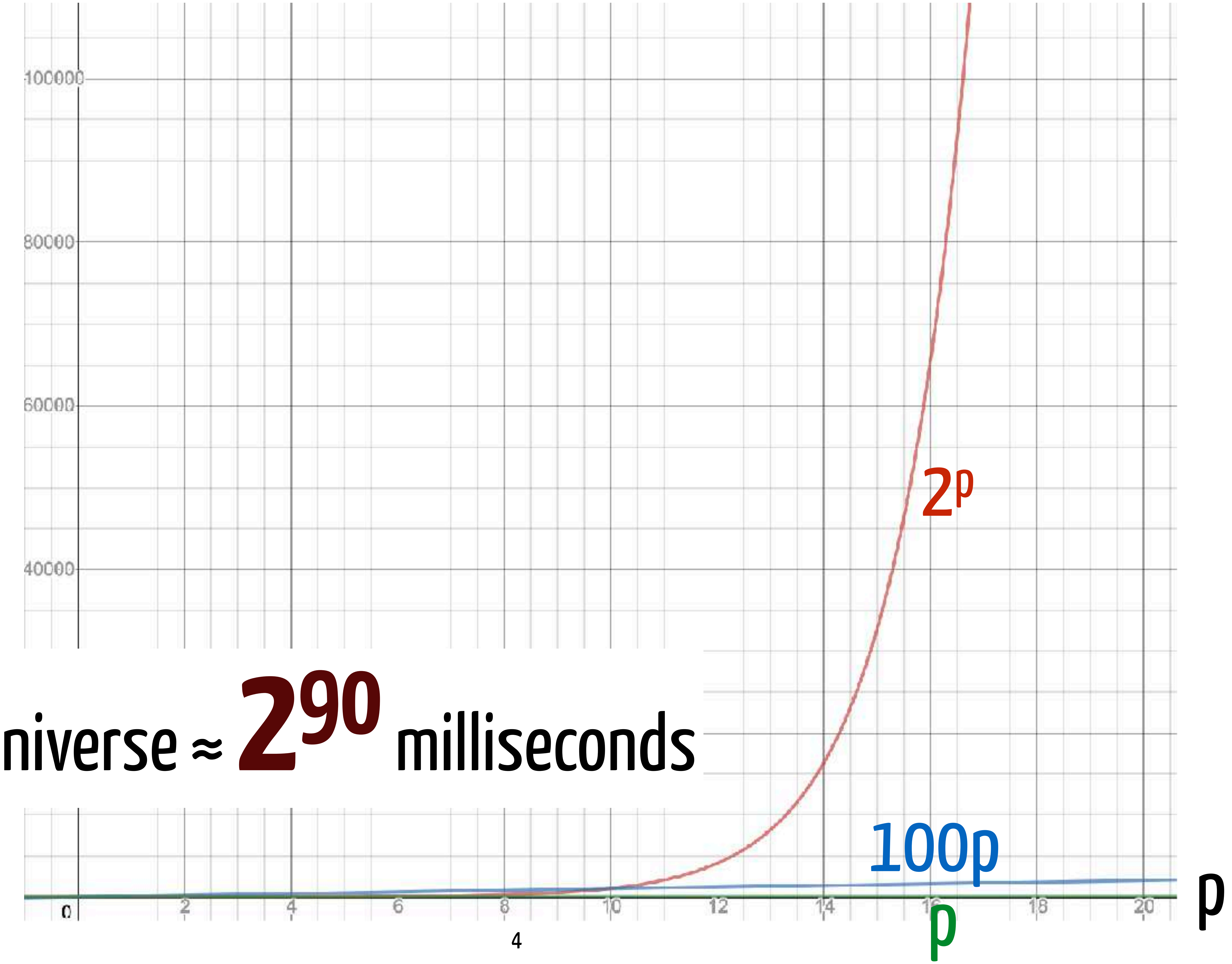
$$\min\{cx \mid Ax \geq b, x \in \{0,1\}^p \times \mathbb{R}^{n-p}\}$$

$= \mathbf{2^p}$ LPs to solve

Combinatorial explosion



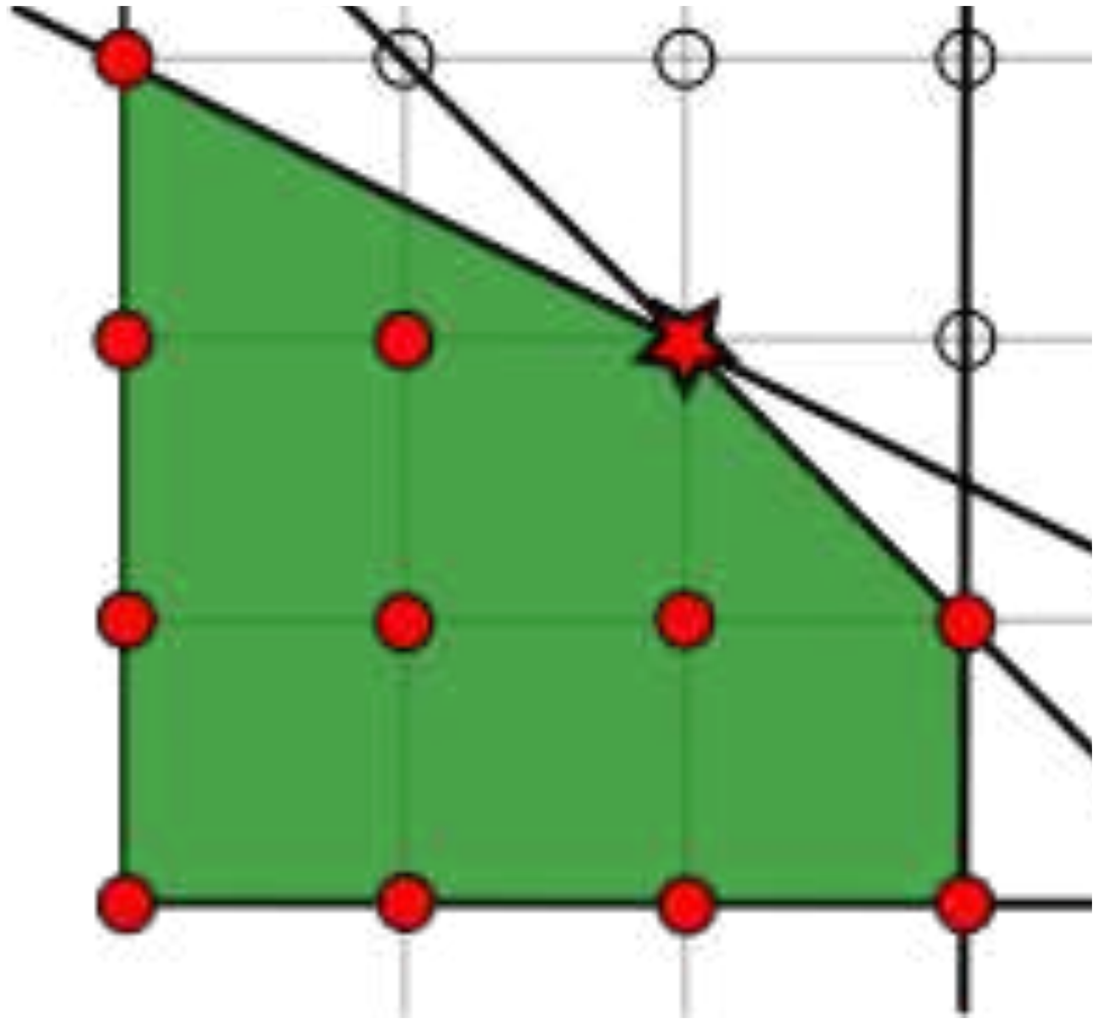
Combinatorial explosion



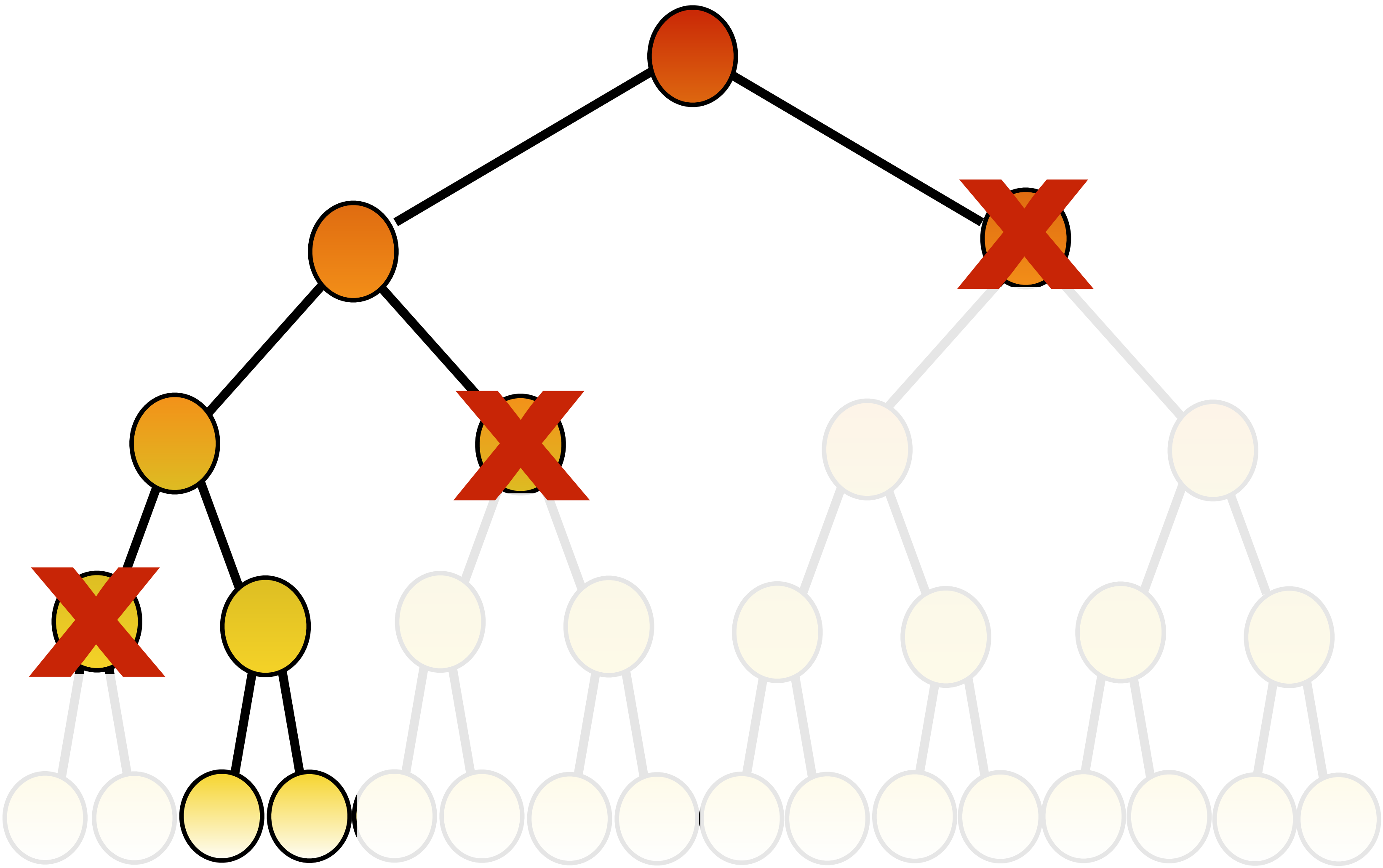
age of the universe ≈ 2^{90} milliseconds

Two options

1 compute
an ideal formulation



2 evaluate partial solutions
progressively

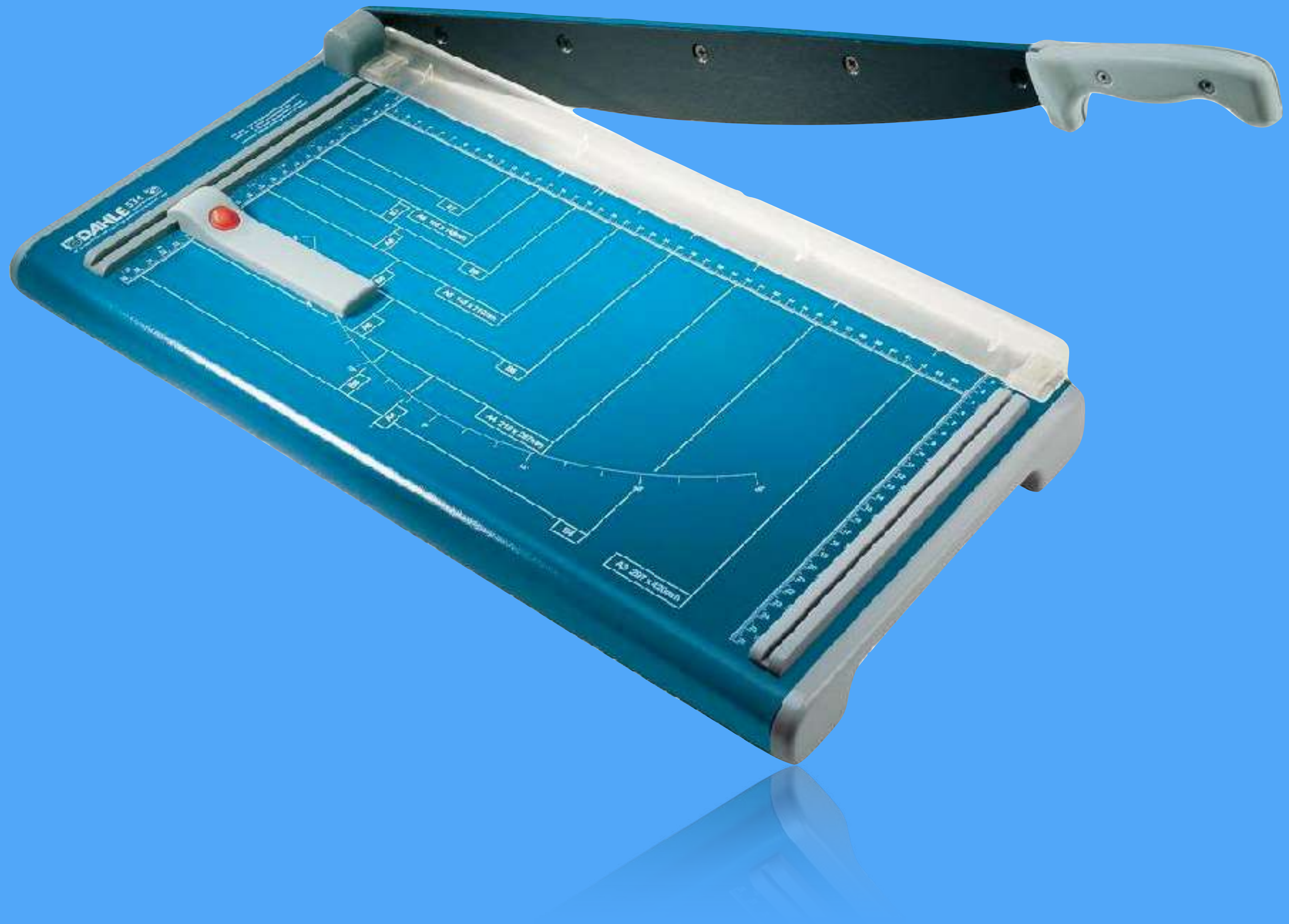


1 **Cut Generation** compute an ideal formulation

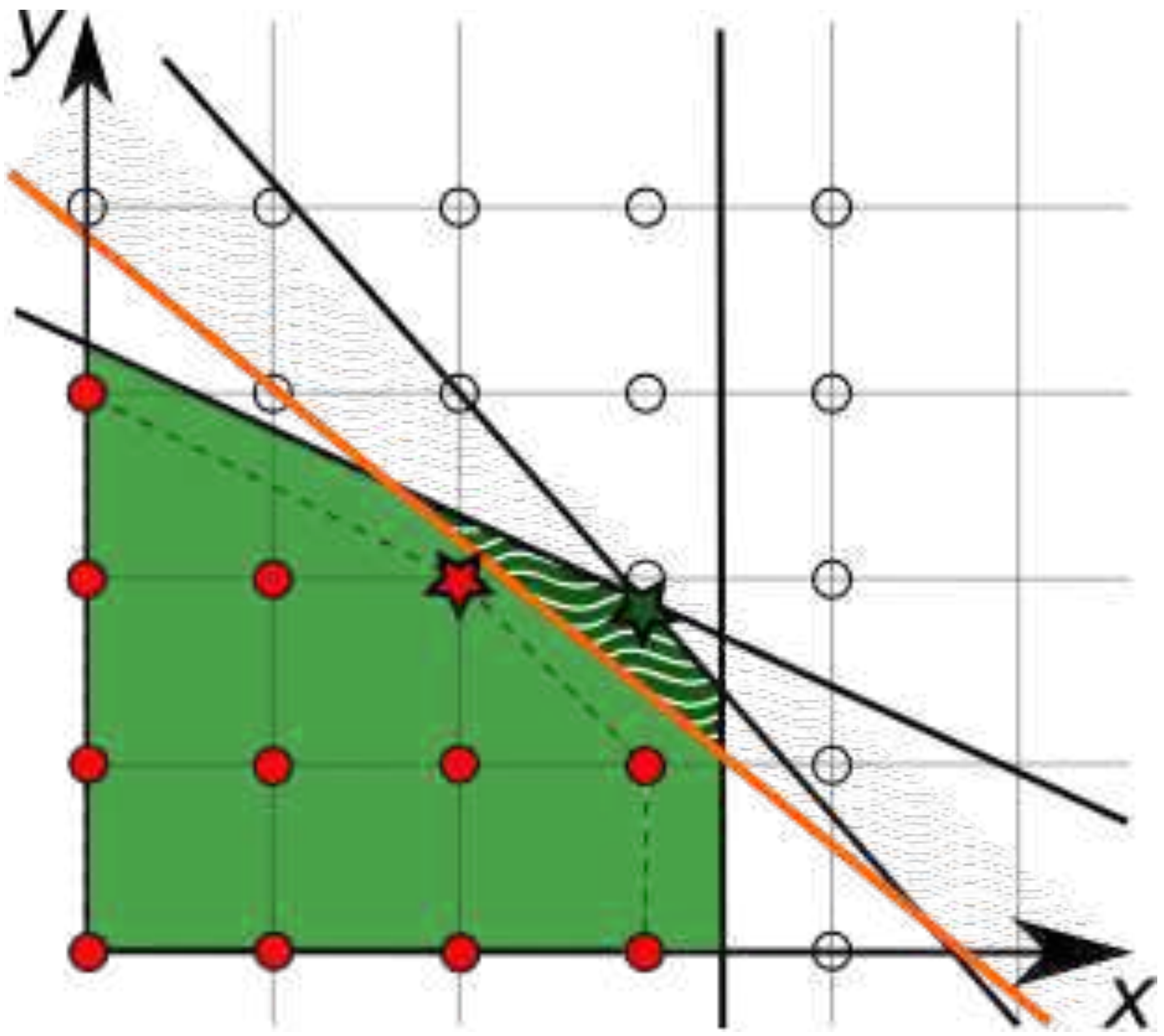
2 **Branch&Bound** evaluate partial solutions progressively

3 **modern Branch&Cut** mix up+presolve+heuristics

4 **decomposition methods** (Branch&Price, Lagrangian relaxation, Benders)



Cutting Plane Algorithm



Cut valid inequality that separates a relaxed LP solution

Farkas Lemma cuts are linear combinations of constraints

cutting plane algorithm

1. solve the LP relaxation of (P), get $\bar{\mathbf{x}}$
2. if $\bar{\mathbf{x}}$ is integral STOP: feasible then optimal for (P)
3. find cuts C for (P,$\bar{\mathbf{x}}$) from template T
4. add constraints C to (P) then 1.

separation subproblem

templates

general-purpose

mixed integer rounding, split, Chvátal-Gomory

structure-based

clique, cover, flow cover, zero half

problem-specific

subtour elimination (TSP), odd-set (matching)

ex 1 Chvátal-Gomory cuts

$(P) : \max\{cx \mid Ax \leq b, x \in \mathbb{Z}_+\}$

For any $u \in \mathbb{R}^m_+$ the following inequalities are valid:

1. surrogate: $\sum_j \sum_i u_i a_{ij} x_j \leq \sum_i u_i b_i$ $\quad (u \geq 0)$

2. round off: $\sum_i \lfloor \sum_j u_i a_{ij} \rfloor x_j \leq \sum_i u_i b_i$ $\quad (x \geq 0)$

3. Chvátal-Gomory: $\sum_i \lfloor \sum_j u_i a_{ij} \rfloor x_j \leq \lfloor \sum_i u_i b_i \rfloor$ $\quad (\lfloor uA \rfloor x \in \mathbb{Z})$

variants in the choice of u, ex: Gomory or MIR cuts

to go further: Kaparis & Letchford 2010
Separation algorithms for 0-1 knapsack polytopes

ex 2 Cover cuts

$$S = \{y \in \{0,1\}^7 | 11y_1 + 6y_2 + 6y_3 + 5y_4 + 5y_5 + 4y_6 + y_7 \leq 19\}$$

- (y_3, y_4, y_5, y_6) is a minimal cover for $11y_1 + 6y_2 + 6y_3 + 5y_4 + 5y_5 + 4y_6 + y_7 \leq 19$ as $6 + 5 + 5 + 4 > 19$ then $y_3 + y_4 + y_5 + y_6 \leq 3$ is a cover inequality
- we can derive a stronger valid inequality $y_1 + y_2 + y_3 + y_4 + y_5 + y_6 \leq 3$ by noting that y_1, y_2 has greater coefficients than any variable in the cover
- note furthermore that (y_1, y_i, y_j) is a cover $\forall i \neq j \in \{2, 3, 4, 5, 6\}$ then $2y_1 + y_2 + y_3 + y_4 + y_5 + y_6 \leq 3$ is also valid

lifting

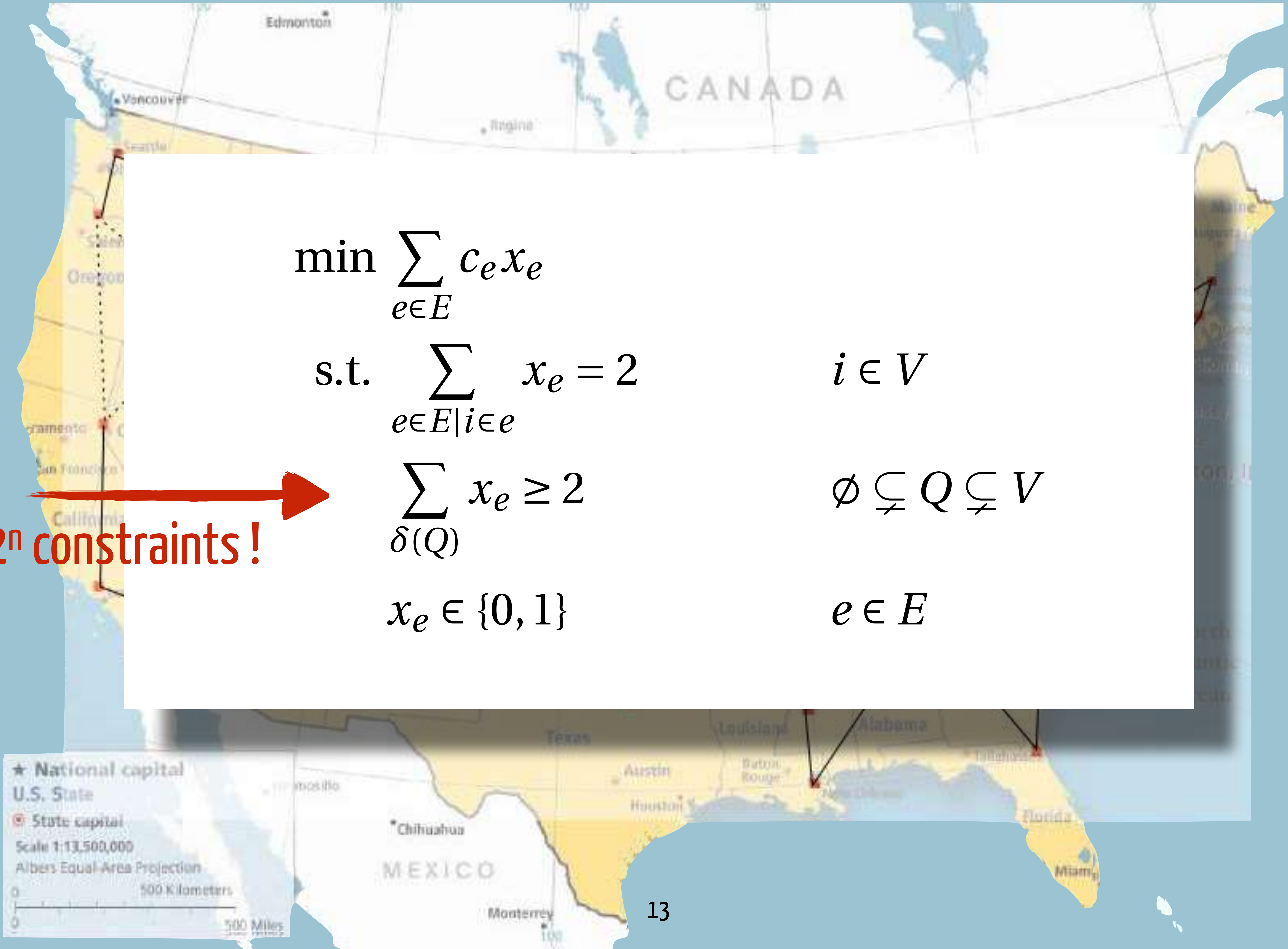
separation: solve knapsack $\min\{\sum_j (1-\bar{y}_j)x_j \mid \sum a_j x_j \geq b + \epsilon, x \in \{0,1\}^n\}$

get coefficients x^* of the cover inequality $\sum x_j^* y_j \leq \sum x_j^* - 1$

if $\sum (1-\bar{y}_j)x_j^* < 1$ then it is a cut (not satisfied by current LP solution $\bar{y}$)

ex3 Subtour for TSP

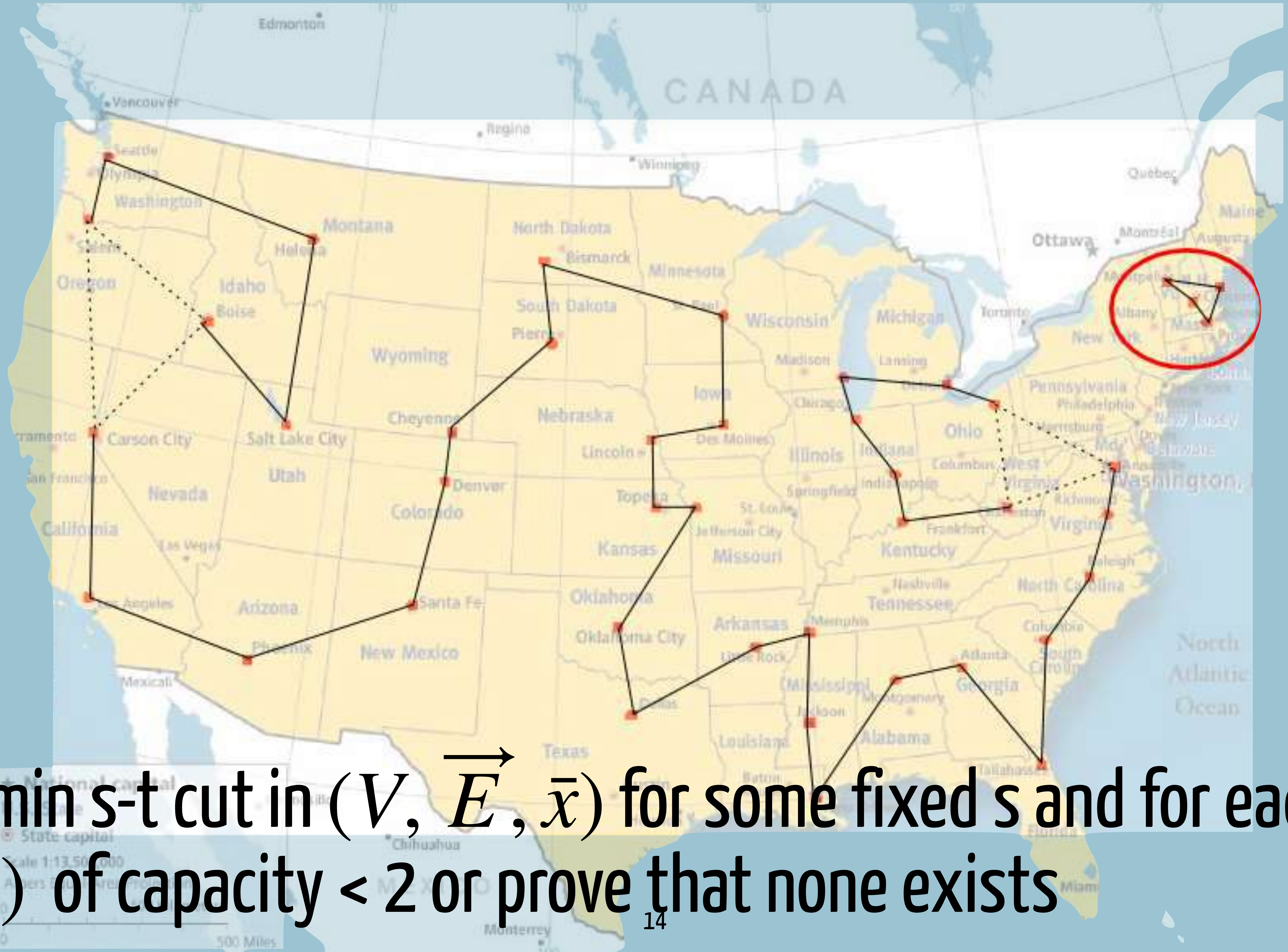
ex3 Subtour for TSP



$$
\begin{aligned}
\min \; & \sum_{e \in E} c_e x_e & \\
\text{s.t.} \; & \sum_{e \in E \mid i \in e} x_e = 2 & i \in V \\
& \sum_{\delta(Q)} x_e \geq 2 & \emptyset \subsetneq Q \subsetneq V \\
& x_e \in \{0, 1\} & e \in E
\end{aligned}
$$

2^n constraints !

ex 3 Subtour for TSP

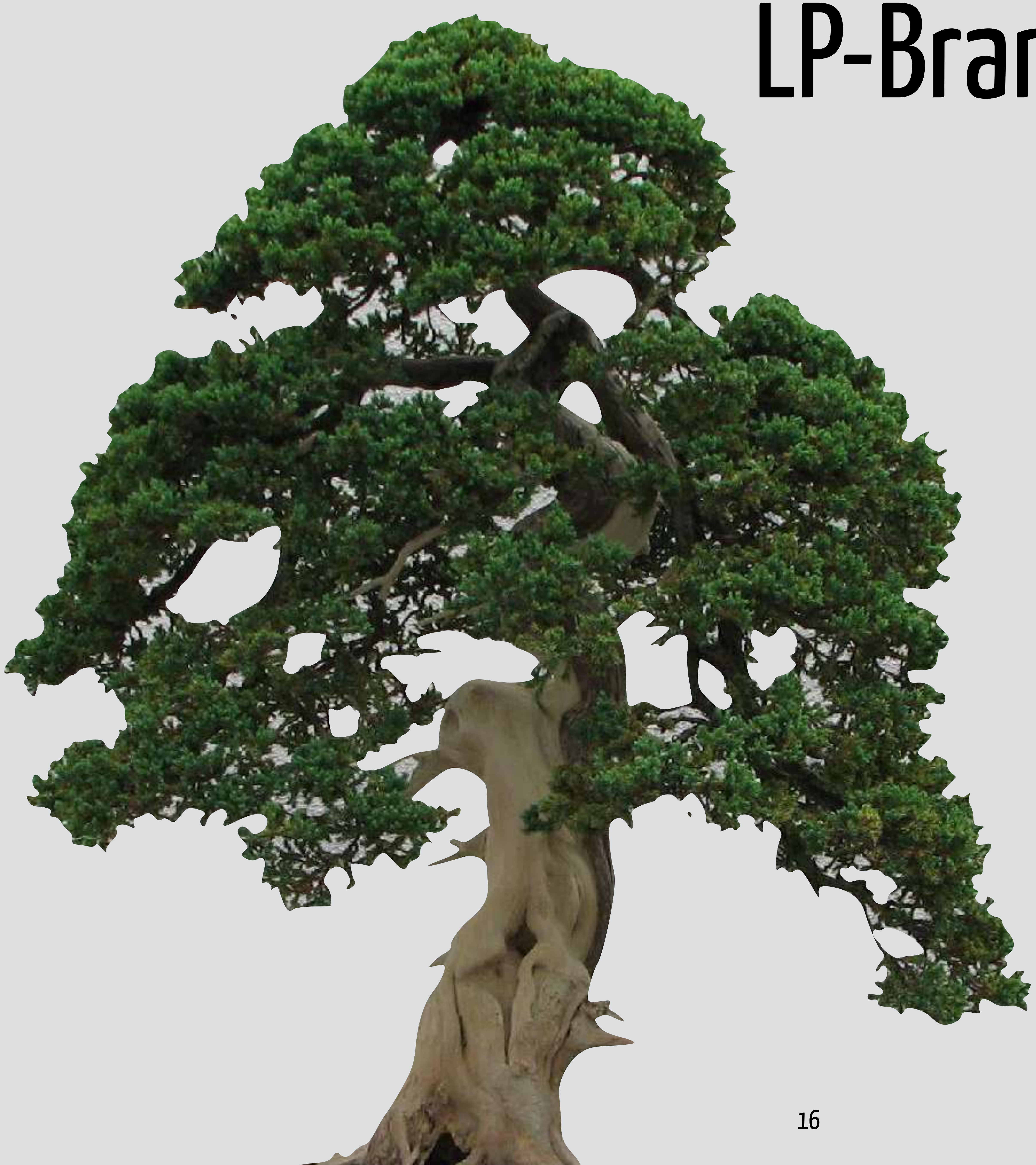


separation: solve min s-t cut in $(V, \overrightarrow{E}, \bar{x})$ for some fixed s and for each $t \in V \setminus \{s\}$ to find a cutset $\delta(Q)$ of capacity < 2 or prove that none exists

limits depending on the templates

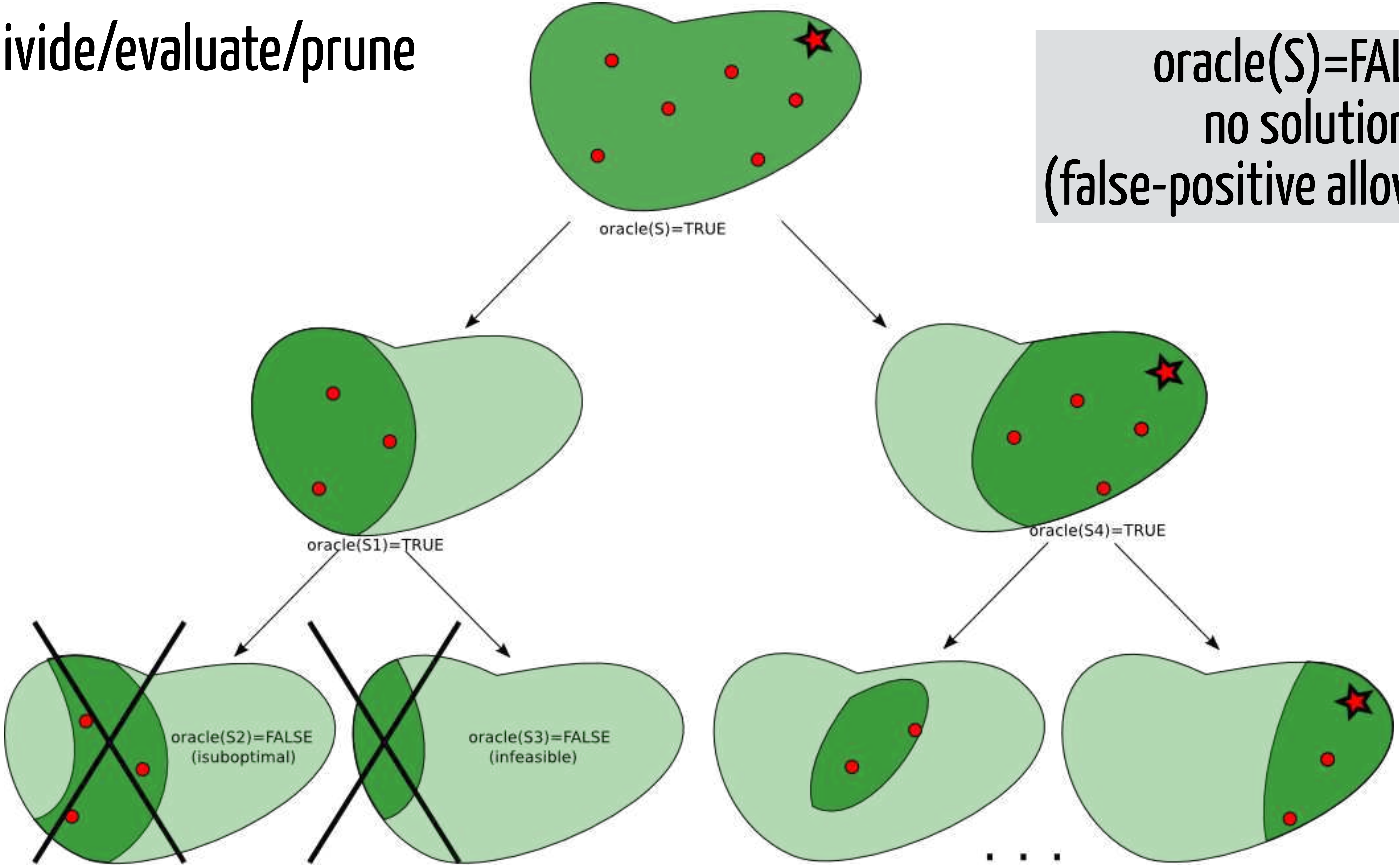
- the algorithm may stop prematurely
- the algorithm may not converge
- the algorithm may converge slowly
- the separation procedure may be NP-hard
- the LP relaxation grows
- the LP relaxation structure changes

LP-Branch and Bound



Search tree

divide/evaluate/prune

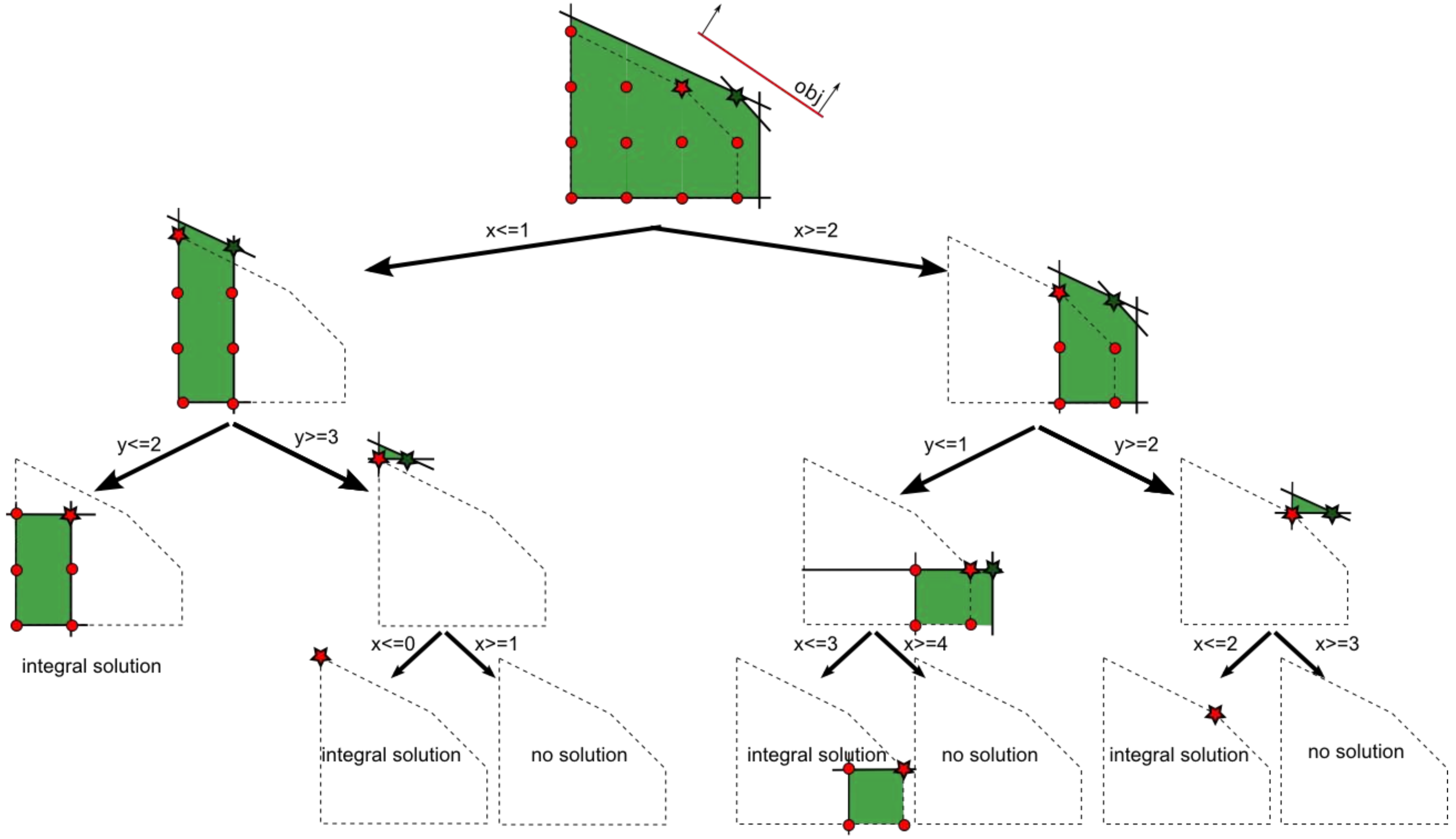


oracle(S)=FALSE if
no solution in S
(false-positive allowed)

LP-based branch and bound

1. evaluate by solving the LP relaxation and compare bounds
2. divide with variable bounding (hyperplanes)

```
oracle(S) = FALSE if either:
 — the LP relaxation is unfeasible on S
 — the relaxed LP solution x̄ is not better than
   the best integer solution found so far x*
 — x̄ is integer (then update x*)
```


obj
x<=1
x>=2
y<=2
y>=3
y<=1
y>=2
integral solution
x<=0
x>=1
x<=3
x>=4
x<=2
x>=3
integral solution
no solution
integral solution
no solution
integral solution
no solution

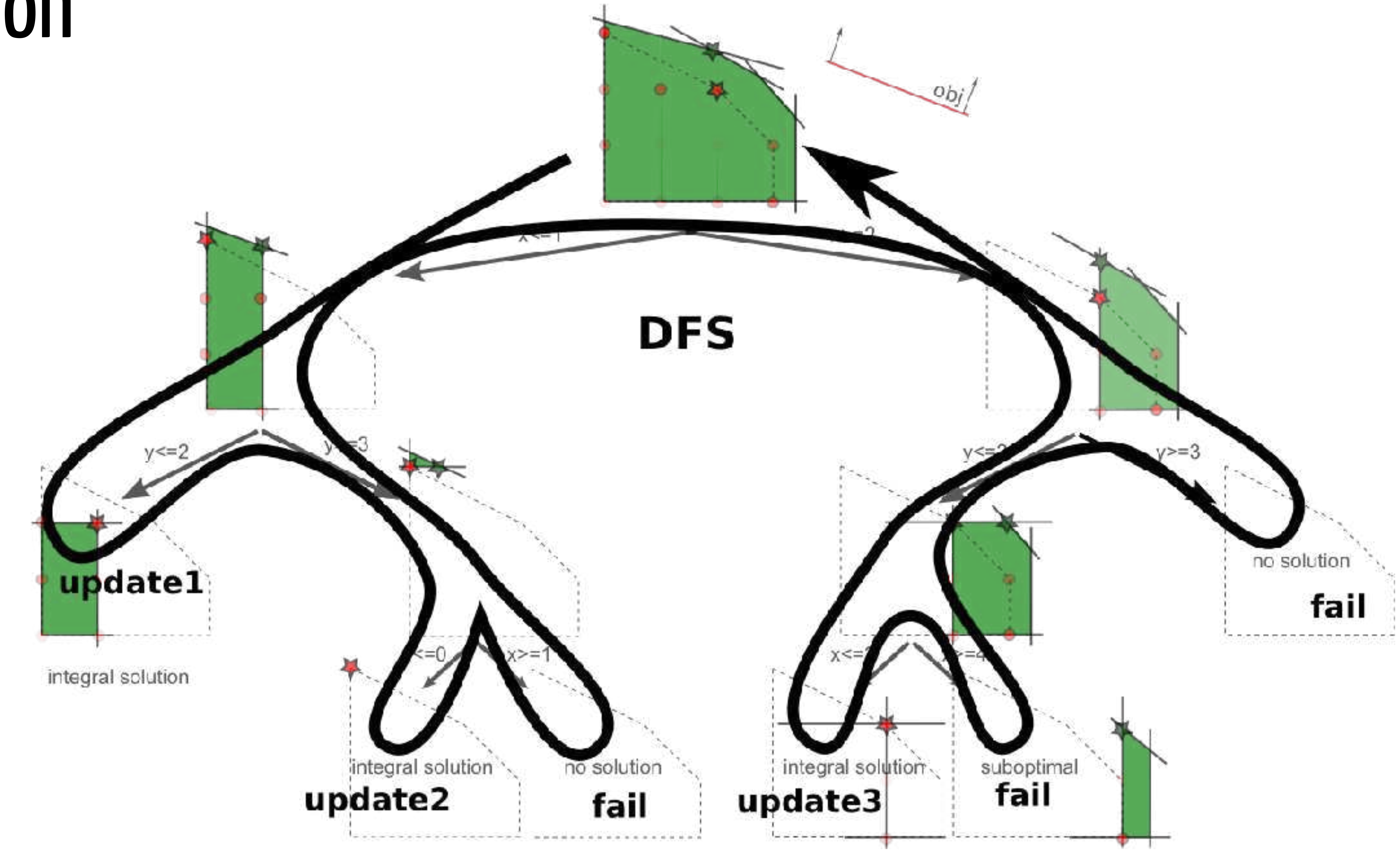
branching

node selection
which order to visit nodes ?

variable selection
how to separate nodes ?

constraint branching
versus variable branching

node selection

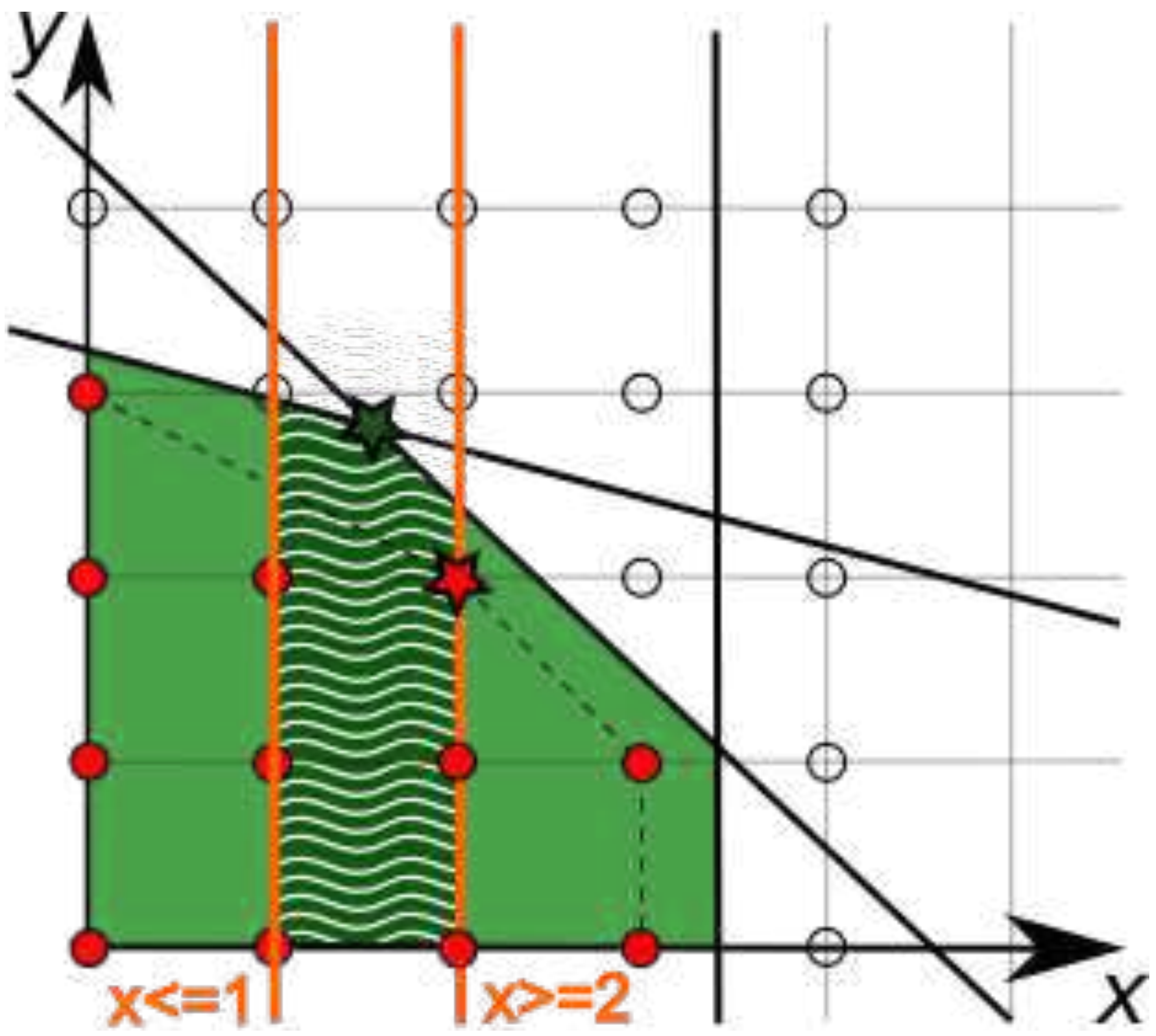


Best Bound First Search explore less nodes, manages larger trees

Depth First Search sensible to bad decisions at or near the root

DFS (up to n solutions) **+ BFS** (to prove optimality)

variable selection



most fractional easy to implement but not better than random

strong branching best improvement among all candidates (impractical)

pseudocost branching record previous branching success for each var (inaccurate at root)

reliability branching pseudocosts initialised with strong branching

constraint branching

example: GUB dichotomy

- if (P) contains a GUB constraint $\sum_C x_i = 1$, $x \in \{0,1\}^n$
- choose $C' \subseteq C$ s.t. $0 < \sum_{C'} \bar{x}_i < 1$
- create two child nodes by setting either $\sum_{C'} x_i = 0$ or $\sum_{C'} x_i = 1$

- enforced by fixing the variable values
- leads to more balanced search trees

SOS1 branching in a facility location problem

choose a warehouse depending on its size/cost:

$$\begin{aligned} \text{COST} &= 100x_1 + 180x_2 + 320x_3 + 450x_4 + 600x_5 \\ \text{SIZE} &= 10x_1 + 20x_2 + 40x_3 + 60x_4 + 80x_5 \\ \text{(SOS1)} &: x_1 + x_2 + x_3 + x_4 + x_5 = 1 \end{aligned}$$

- let $\bar{x}_1 = 0.35$ and $\bar{x}_5 = 0.65$ in the LP solution then SIZE$= 55.5$
- choose $C' = \{1,2,3\}$ in order to model SIZE≤ 40 or SIZE≥ 60

bounding: the dual simplex algorithm

- primal-dual problem pair: $\min_x\{c^\top x \,|\, Ax = b, x \geq 0\} = \max_u\{u^\top b \,|\, A^\top u \leq c\}$
- primal-dual basic solutions: $x = (x_B, x_N)$ with $x_N = 0,\ x_B = A_B^{-1}b$ and $u^\top = c_B^\top A_B^{-1}$,
- primal basic feasible solutions are the extreme points of polyhedron $P = \{x \geq 0 \mid Ax \geq b\}$
- if both are feasible ($x_B \geq 0$ and $c^\top - u^\top A \geq 0$) then both are optimal ($u^T b = c_B^\top x_B = cx$)
- primal simplex algorithm: iterate over bases, maintain primal feasibility, stop when achieving dual feasibility
- dual simplex algorithm: iterate over bases, maintain dual feasibility, stop when achieving primal feasibility
- branching $\Longrightarrow$ updating $b \quad \Longrightarrow \quad$ the dual basic solution remains feasible
- we can warm-start the dual simplex algorithm to solve the LP-relaxation at a search node with the dual basic solution of the parent node
- great impact on the running time of the LP-B&B algorithm
- convex MINLP: NLP-B&B algorithm does usually not perform well (OA-based cutting-plane algorithms are usually better) mostly because no such warm-start algorithm exists for NLP


project:
power generation

Input (noncyclic)

up/down times: minimum time Δ_t^+, Δ_t^- unit $t \in T$ may remain on or off; time $\Delta_{0it}^+, \Delta_{0it}^-$ the ith unit of type $t \in T$ has been on/off before period 0.

ramp rates: maximum power increase/decrease L_t^+, L_t^- between two consecutive periods; maximum power L_t^S when turned on; maximum power L_t^E before turned off; load L_{0it} for ith unit of type $t \in T$ before period 0.

Input (cyclic)

the status before period 0 $(\Delta_{0it}^+, \Delta_{0it}^-, L_{0it})$ are duplicated from period P-1.

Physical limits of the units: minimum up/down times and maximum ramp up/down rates

$\boxed{\text{commitment must be monitored for units individually}}$

minimum uptime

Let $P_t^+(p) = \{0 \le p' \le p \mid \sum_{k=p'}^{p-1} \Delta_k < \Delta_t^+\}$.

Show that an unit of type t cannot been turned on more than once during $P_t^+(p)$.

Show that if an unit of type t is off at time p then it has not been turned on at any time $p' \in P_t^+(p)$.

Reformulate these assertions as a linear relation between the binary variables modelling the unit status and status change at appropriate periods.

minimum uptime (noncyclic case)

If unit i of type t has been on for exactly $\Delta^+_{0it} > 0$ hours before time 0, what can you say about its status and status change at any period in

$$P^+_{it} = \{p \geq 0 \mid \sum_{k=0}^{p-1} \Delta_k < \Delta^+_t - \Delta^+_{0it}\}?$$

Fix binary variables modelling the status and status change of a unit at given periods according to this assertion.

maximum ramp

If unit i of type t starts at p ($y_{ithp} = 1$) then $l_{itp} - l_{itp-1} \leq L_{it}^{S}$

Otherwise either unit i is on at p-1 and on at p and $l_{itp} - l_{itp-1} \leq L_{it}^{+}$

or unit i is on at p-1 and off at p and $l_{itp} - l_{itp-1} < 0$

or unit i is off at p-1 and off at p and $l_{itp} - l_{itp-1} = 0$

ENCORE EN GRÈVE

the MILP way

a practical view

(4)

Sophie Demassey 2023

CMA Centre de Mathématiques Appliquées

modern solvers

Simplex

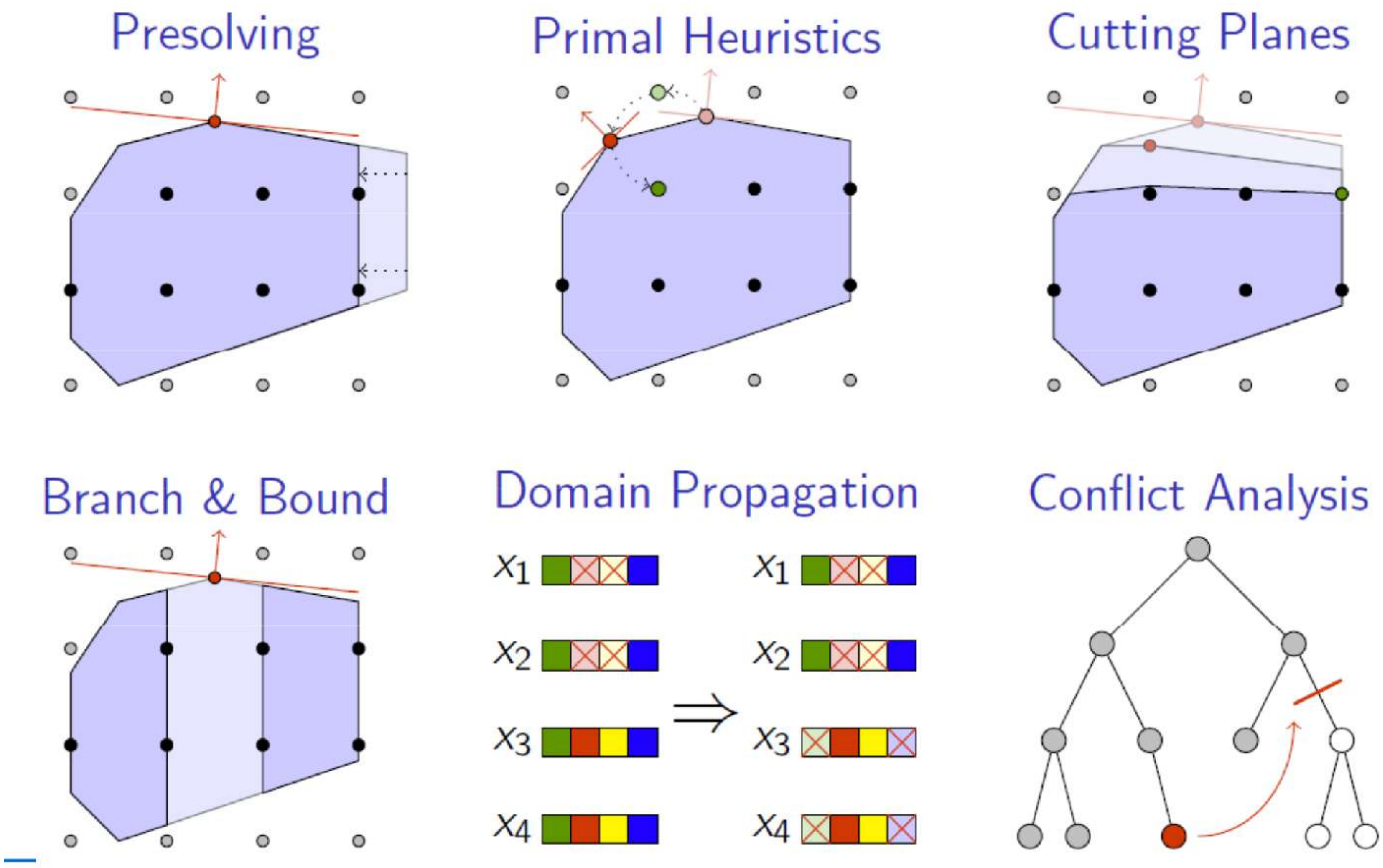
var branching

Preprocessing

Branch & Cut

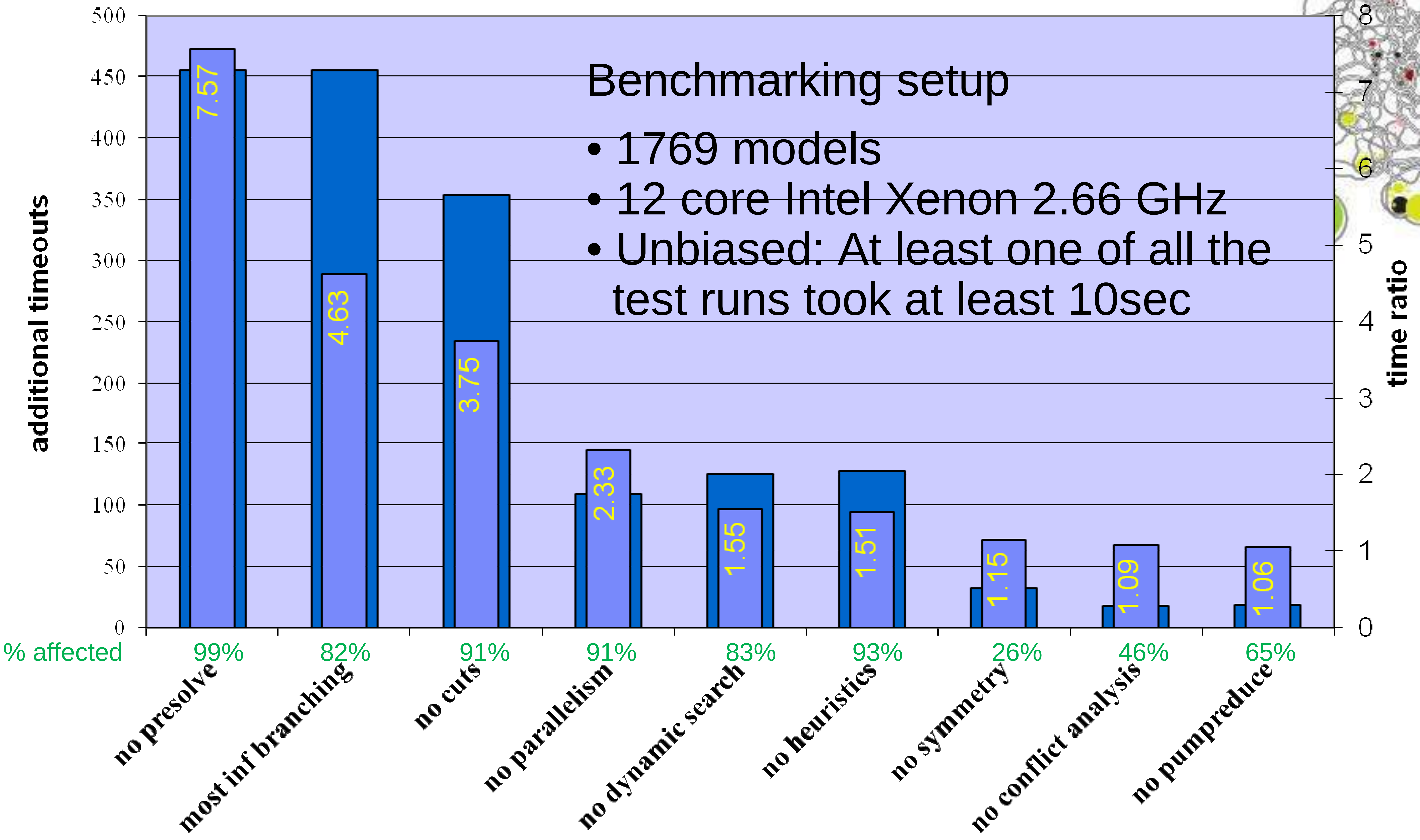
Heuristics

Parallelism



Slide from Martin Grötschel Co@W Berlin 2015

Component Impact CPLEX 12.5 Summary

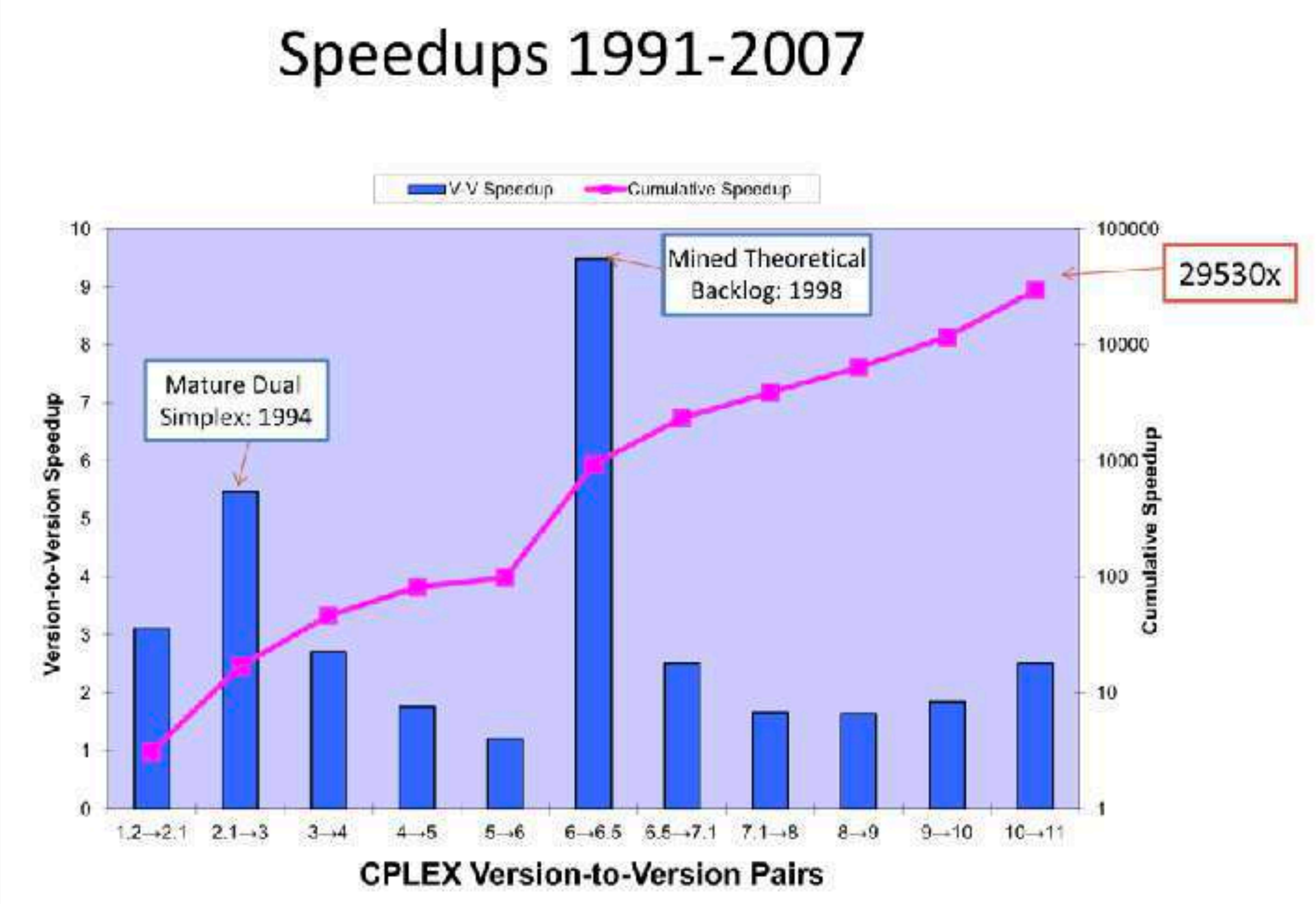


MIP Evolution, Cplex numbers

- Bob Bixby (Gurobi) & Tobias Achterberg (IBM) performed the following interesting experiment comparing Cplex versions from Cplex 1.2 (the first one with MIP capability) up to Cplex 11.0.
- 1,734 MIP instances, time limit of 30,000 CPU seconds, computing times as geometric means normalized wrt Cplex 11.0 (equivalent if within 10%).

Cplex versions	year	better	worse	time
11.0	2007	0	0	1.00
10.0	2005	201	650	1.91
9.0	2003	142	793	2.73
8.0	2002	117	856	3.56
7.1	2001	63	930	4.59
6.5	1999	71	997	7.47
6.0	1998	55	1060	21.30
5.0	1997	45	1069	22.57
4.0	1995	37	1089	26.29
3.0	1994	34	1107	34.63
2.1	1993	13	1137	56.16
1.2	1991	17	1132	67.90

From Andrea Lodi's MIP course (Wien 2012)



From Robert Bixby (1000x MIP Tricks 2012)

CPLEX 20.1

Search in IBM ILOG CPLEX Optimization Studio 20.1.0

Here are descriptions of the cuts implemented in CPLEX.
exception of **implied bound cuts**. Implied bound cuts ca

- Boolean Quadric Polytope (BQP) cuts
- Clique cuts
- Cover cuts
- Disjunctive cuts
- Flow cover cuts
- Flow path cuts
- Gomory fractional cuts
- Generalized upper bound (GUB) cover cuts
- Implied bound cuts: global and local
- Lift-and-project cuts
- Mixed integer rounding (MIR) cuts
- Multi-commodity flow (MCF) cuts
- Reformulation Linearization Technique (RLT) cuts
- Zero-half cuts

GUROBI 7.5 - 10.0

CliqueCuts	Clique cut generation
CoverCuts	Cover cut generation
FlowCoverCuts	Flow cover cut generation
FlowPathCuts	Flow path cut generation
GUBCoverCuts	GUB cover cut generation
ImpliedCuts	Implied bound cut generation
MIPSepCuts	MIP separation cut generation
MIRCuts	MIR cut generation
StrongCGCuts	Strong-CG cut generation
ModKCuts	Mod-k cut generation
NetworkCuts	Network cut generation
ProjImpliedCuts	Projected implied bound cut generation
SubMIPCuts	Sub-MIP cut generation
ZeroHalfCuts	Zero-half cut generation
InfProofCuts	Infeasibility proof cut generation

Preprocessing

reduce size

remove redundancies $x+y\leq 3$, *binaries*

substitute variables $x+y-z=0$

fix variables by duality $c_j\geq 0,\ A_j\geq 0 \Rightarrow x=x_{min}$

fix variables by probing $x=1$ *infeas* $\Rightarrow x=0$

strengthen LP relaxation

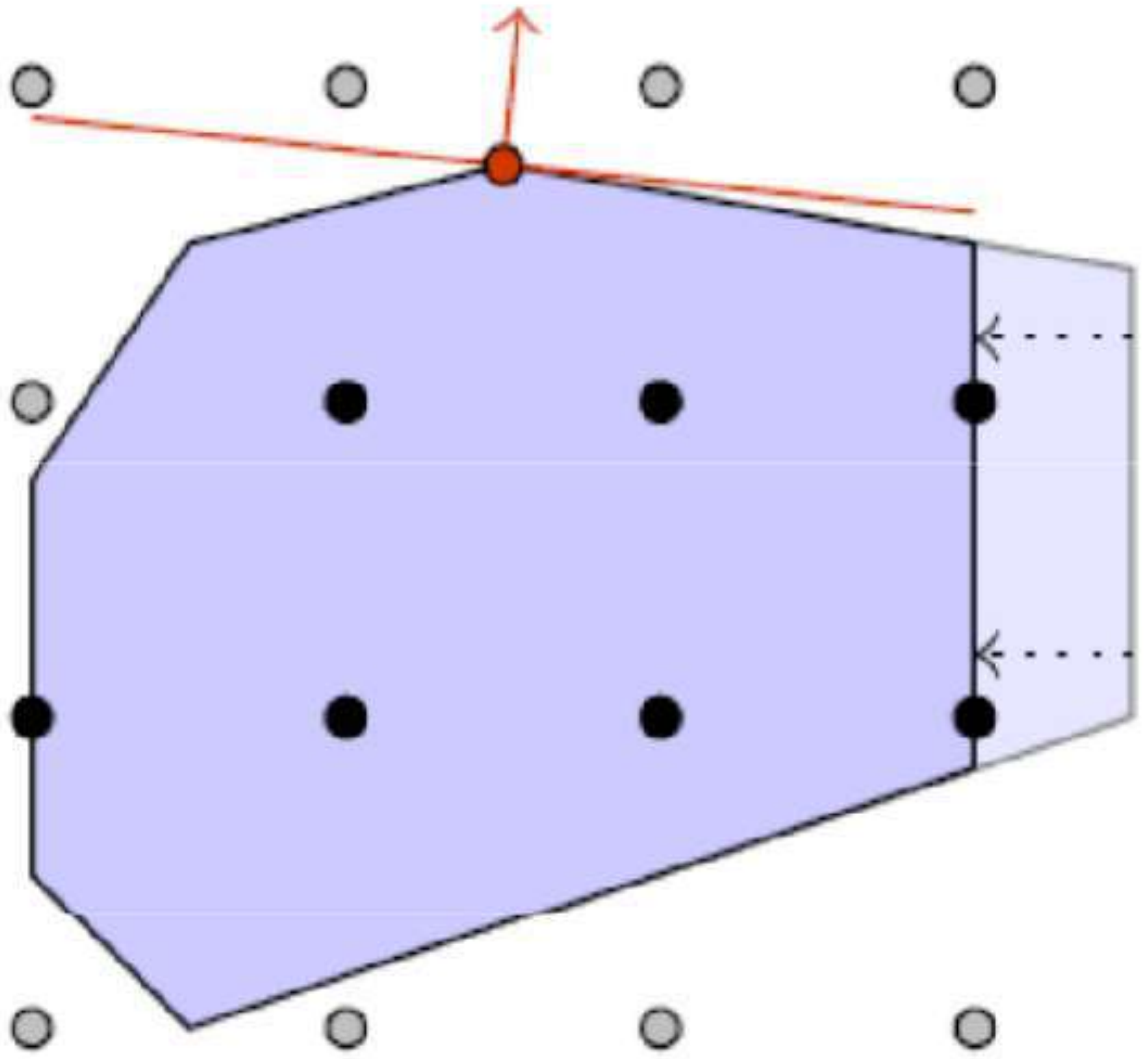
adjust bounds $2x+y\leq 1$, *binaries* $\Rightarrow x=0$

lift coefficients $2x-y\leq 1$, *binaries* $\Rightarrow x-y\leq 1$

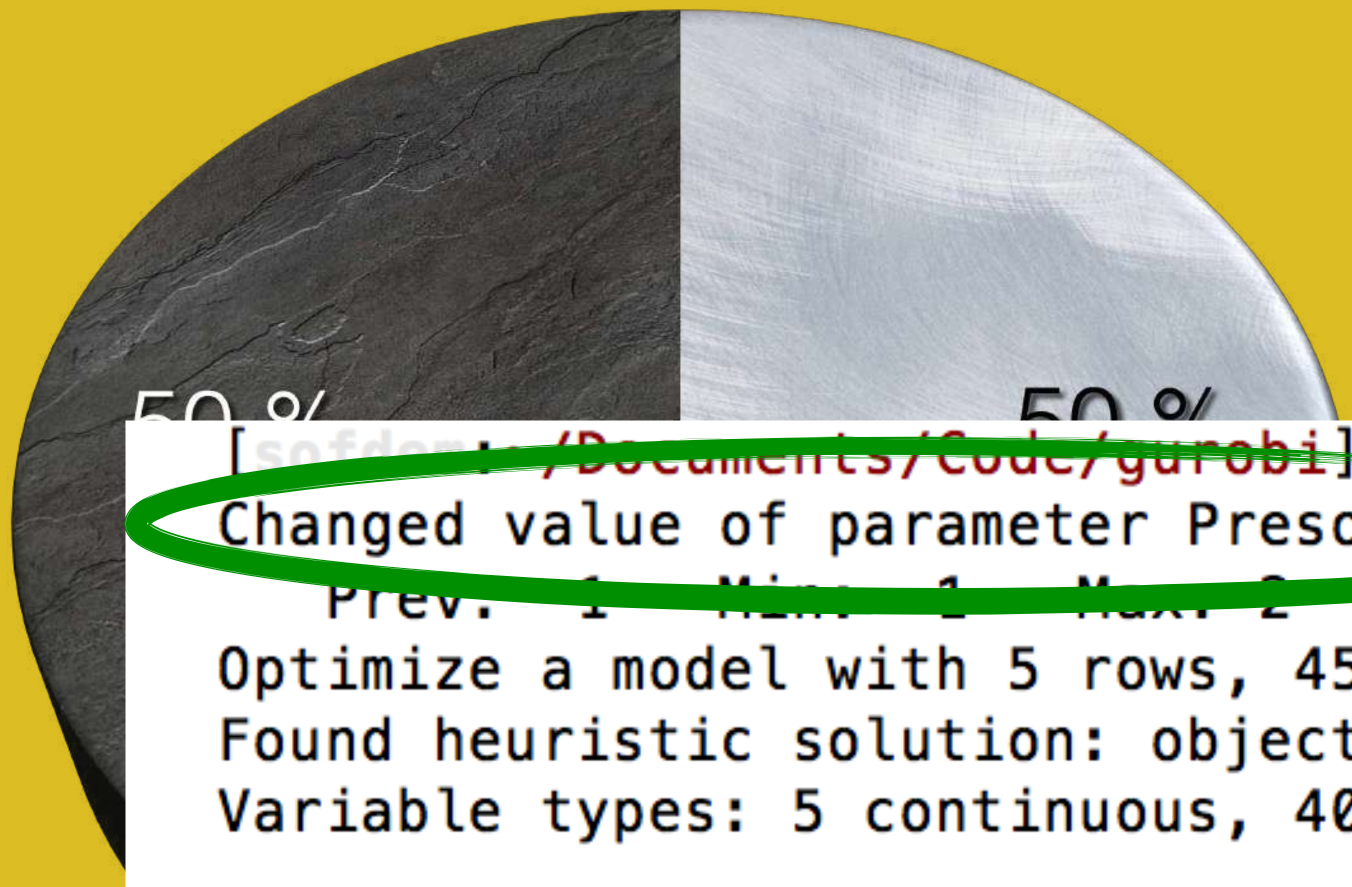
identify/exploit properties

detect implied integer $3x+y=7$, x *int* $\Rightarrow y$ *int*

build the conflict graph
detect disconnected components
remove symmetries

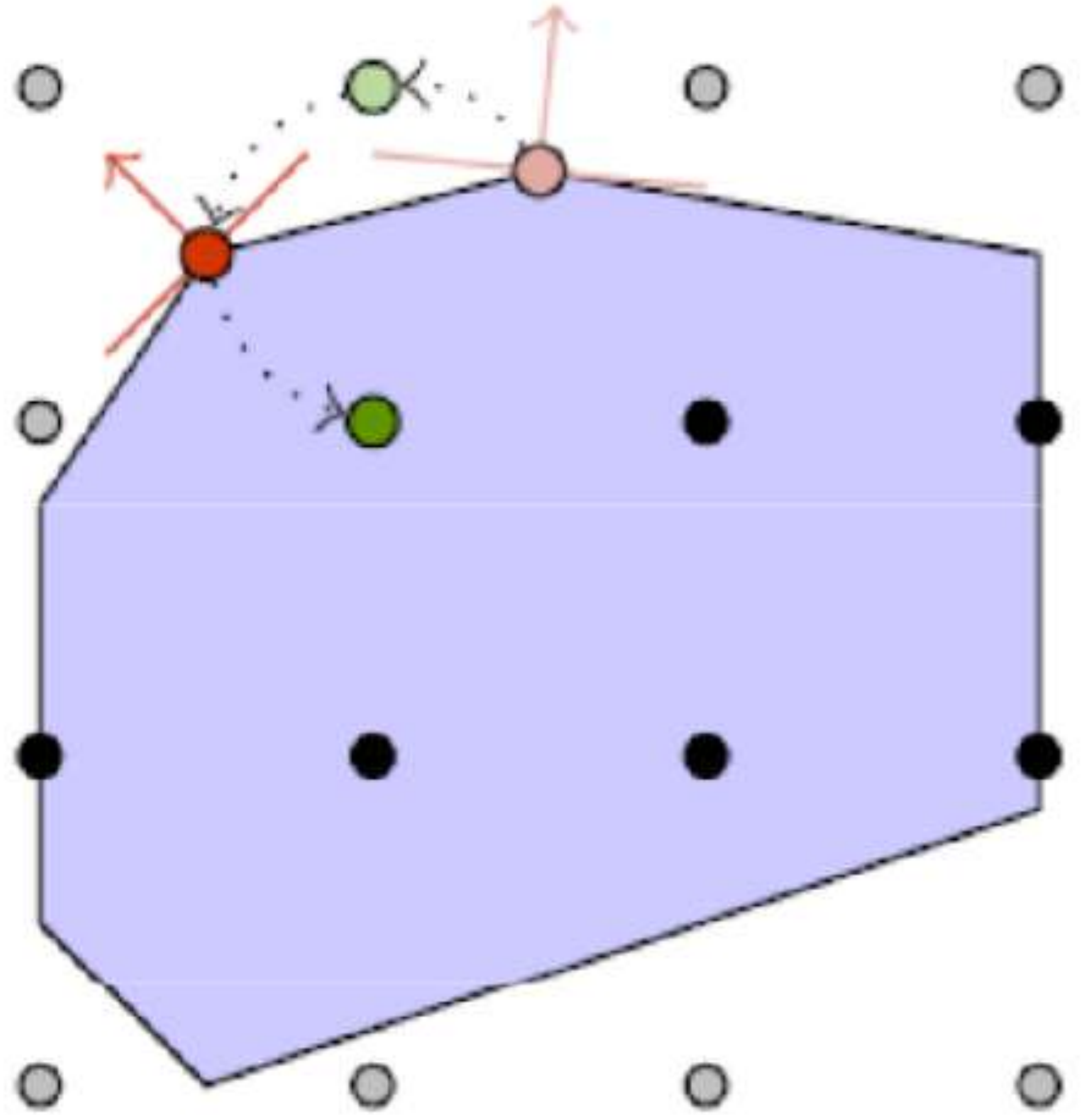


MIPLIB markshare_5_0




```
[sofdem:~/Documents/Code/gurobi]$ gurobi.sh mymip.py markshare_5_0.mps.gz
Changed value of parameter Presolve to 0
   Prev: -1  Min: -1  Max: 2  Default: -1
Optimize a model with 5 rows, 45 columns and 203 nonzeros
Found heuristic solution: objective 5335
Variable types: 5 continuous, 40 integer (40 binary)

Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0     5 5335.00000    0.00000   100%     -    0s

*62706364 28044              38       1.0000000    0.00000   100%   2.1 1241s

Explored 233848403 nodes (460515864 simplex iterations) in 3883.56 seconds
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 1.000000000000e+00, best bound 1.000000000000e+00, gap 0.0%
Optimal objective: 1
```

```
[sofdem:~/Documents/Code/gurobi]$ gurobi.sh mymip.py markshare_5_0.mps.gz
Optimize a model with 5 rows, 45 columns and 203 nonzeros
Found heuristic solution: objective 5335
Presolve time: 0.00s
Presolved: 5 rows, 45 columns, 203 nonzeros
Variable types: 0 continuous, 45 integer (40 binary)

Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0    5 5335.00000    0.00000   100%     -    0s
H    0     0                     320.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
H    0     0                     239.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    5  239.00000    0.00000   100%     -    0s
*   36     0              29      96.0000000    0.00000   100%   2.7    0s
*   99    32              34      58.0000000    0.00000   100%   2.1    0s
H  506   214                      53.0000000    0.00000   100%   1.9    0s
H30682   442                       1.0000000    1.00000  0.00%   2.1    0s

Cutting planes:
  Cover: 26

Explored 30682 nodes (65348 simplex iterations) in 0.70 seconds
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 1.000000000000e+00, best bound 1.000000000000e+00, gap 0.0%
Optimal objective: 1
```



rounding LP solution
diving at some nodes
local search in the incumbent neighbourhood
e.g.: feasibility pump, RINS

Primal Heuristics

accelerate the search a little
appeal to the practitioner a lot

limits

- highly heuristic (branching decisions, cut generation)
- floating-point errors and optimality tolerance (0.01%)
- generic features
- less effective on general integers (ex: scheduling)
- hard to model (and solve) non-linear structures
- NP-hard

how to tune modern solvers

play with Gurobi

```
Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0     5 5335.00000    0.00000   100%     -    0s
H    0     0                      320.0000000    0.00000   100%     -    0s
     0     0    0.00000    0     6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0     5  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0     6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0     5  320.00000    0.00000   100%     -    0s
H    0     0                      239.0000000    0.00000   100%     -    0s
     0     0    0.00000    0     5  239.00000    0.00000   100%     -    0s
*   36     0              29      96.0000000    0.00000   100%   2.7    0s
*   99    32              34      58.0000000    0.00000   100%   2.1    0s
H  506   214                      53.0000000    0.00000   100%   1.9    0s
H30682   442                       1.0000000    1.00000  0.00%   2.1    0s
```

use as a heuristic

set a time limit

MIPFocus=1

ImproveStartGap=0.1

```
Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0    5 5335.00000    0.00000   100%     -    0s
H    0     0                     320.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
H    0     0                     239.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    5  239.00000    0.00000   100%     -    0s
*   36     0              29      96.0000000    0.00000   100%   2.7    0s
*   99    32              34      58.0000000    0.00000   100%   2.1    0s
H  506   214                      53.0000000    0.00000   100%   1.9    0s
H30682   442                       1.0000000    1.00000  0.00%   2.1    0s
```

change the LP solver

if nbIteration(node) ≥ nbIteration(root)/2

`NodeMethod=2`

```
Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0    5 5335.00000    0.00000   100%     -    0s
H    0     0                      320.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
H    0     0                      239.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    5  239.00000    0.00000   100%     -    0s
*   36     0              29      96.0000000    0.00000   100%   2.7    0s
*   99    32              34      58.0000000    0.00000   100%   2.1    0s
H  506   214                      53.0000000    0.00000   100%   1.9    0s
H30682   442                       1.0000000    1.00000  0.00%   2.1    0s
```

init with a feasible solution

if built-in heuristics fail

`PumpPasses,MinRelNodes,ZeroObjNodes`

`model.read('initSol.mst')`

`model.cbSetSolution(vars, newSol)`

```
Root relaxation: objective 0.000000e+00, 15 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0    5 5335.00000    0.00000   100%     -    0s
H    0     0                     320.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    6  320.00000    0.00000   100%     -    0s
     0     0    0.00000    0    5  320.00000    0.00000   100%     -    0s
H    0     0                     239.0000000    0.00000   100%     -    0s
     0     0    0.00000    0    5  239.00000    0.00000   100%     -    0s
*   36     0              29      96.0000000    0.00000   100%   2.7    0s
*   99    32              34      58.0000000    0.00000   100%   2.1    0s
H  506   214                      53.0000000    0.00000   100%   1.9    0s
H30682   442                       1.0000000    1.00000  0.00%   2.1    0s
```

tighten the model

if the bound stagnates

`Cuts=3`

`Presolve=3`

`model.cbCut(lhs, sense, rhs)`

http://www.gurobi.com/

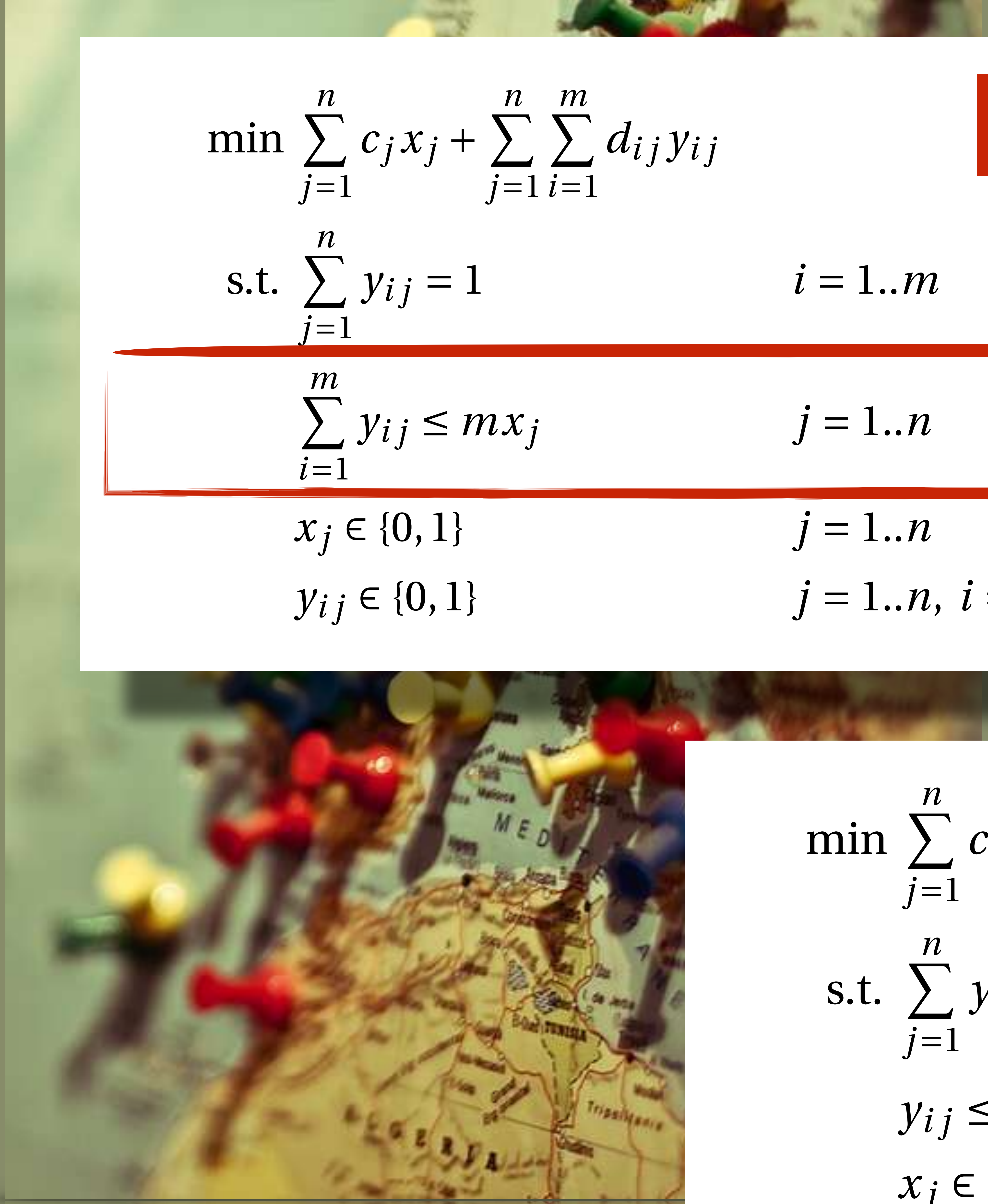
GUROBI
OPTIMIZATION

/documentation/current/refman/index.html

/resource-center/

you know your problem better
than your solver does

tighten the model



14 hours

$$\min \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n} \sum_{i=1}^{m} d_{ij} y_{ij}$$

$$\text{s.t.} \sum_{j=1}^{n} y_{ij} = 1 \qquad i = 1..m$$

$$\boxed{\sum_{i=1}^{m} y_{ij} \le m x_j \qquad j = 1..n}$$

$$x_j \in \{0,1\} \qquad j = 1..n$$

$$y_{ij} \in \{0,1\} \qquad j = 1..n,\ i = 1..m$$

apacitated

ty Location

roblem

Input n facility locations, m

2 seconds

$$\min \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n} \sum_{i=1}^{m} d_{ij} y_{ij}$$

$$\text{s.t.} \sum_{j=1}^{n} y_{ij} = 1 \qquad i = 1..m$$

$$y_{ij} \le x_j \qquad j = 1..n,\ i = 1..m$$

$$x_j \in \{0,1\} \qquad j = 1..n$$

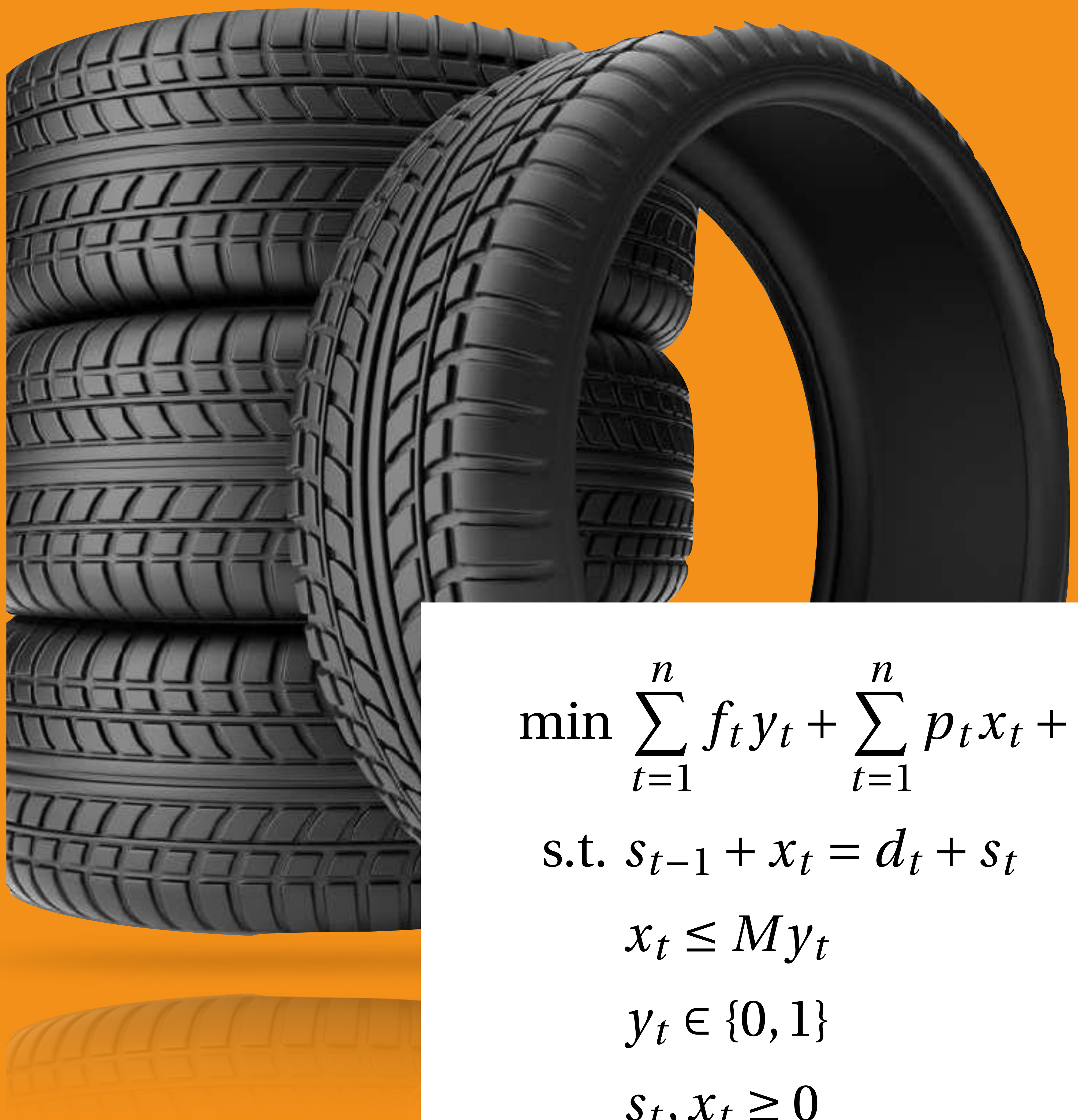
$$y_{ij} \in \{0,1\} \qquad j = 1..n,\ i = 1..m$$

m=n=40

Uncapacitated Lot Sizing Problem

Input n time periods, fixed production cost f_t, unit production cost p_t, unit storage cost h_t, demand d_t for each period t

Output a mimimum (production and storage) cost production plan to satisfy the demand

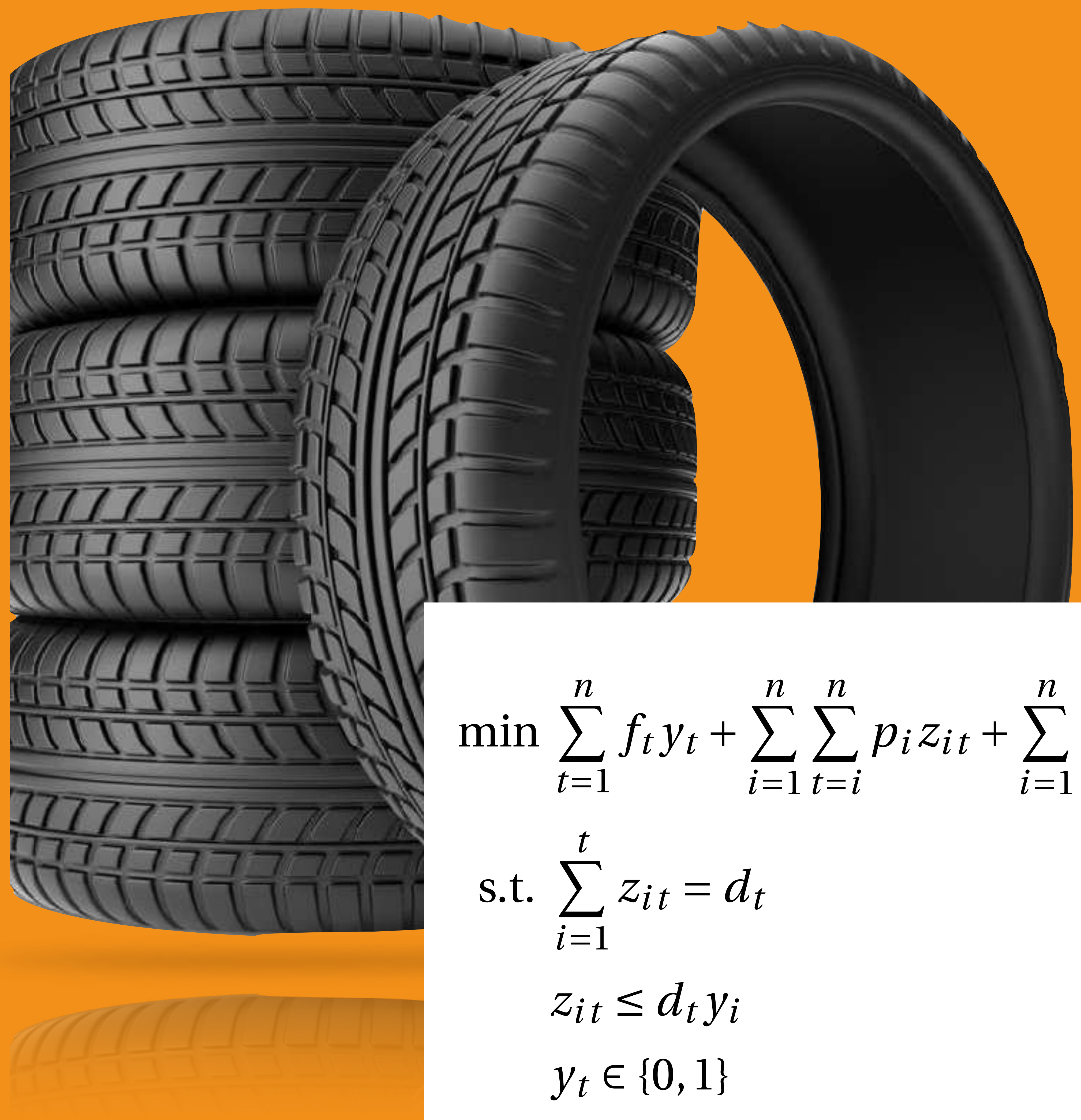


Uncapacitated Lot Sizing Problem

$$
\begin{aligned}
\min \quad & \sum_{t=1}^{n} f_t y_t + \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t && t = 1..n \\
& x_t \le M y_t && t = 1..n \\
& y_t \in \{0,1\} && t = 1..n \\
& s_t, x_t \ge 0 && t = 1,\dots,n \\
& s_0 = 0
\end{aligned}
$$

ed
roduction
h_t, demand

on and
plan to



Uncapacitated Lot Sizing Problem

$$
\begin{aligned}
\min \ & \sum_{t=1}^{n} f_t y_t + \sum_{i=1}^{n} \sum_{t=i}^{n} p_i z_{it} + \sum_{i=1}^{n} \sum_{t=i+1}^{n} \sum_{j=i}^{t-1} h_j z_{it} & \\
\text{s.t.} \ & \sum_{i=1}^{t} z_{it} = d_t & t = 1..n \\
& z_{it} \le d_t y_i & i = 1..n; t = i..n \\
& y_t \in \{0,1\} & t = 1..n \\
& z_{it} \ge 0 & i = 1..n; t = i..n
\end{aligned}
$$

LP=ILP

ed
roduction
h_t, demand
on and
plan to

z_{it} production in period i to satisfy demand of period t

project:
power generation

https://colab.research.google.com/drive/19WNrTomQnD12aScfmJRxQZGdxwsL3ehc

Input (noncyclic)

up/down times: minimum time Δ_t^+, Δ_t^- unit $t \in T$ may remain on or off; time $\Delta_{0it}^+, \Delta_{0it}^-$ the ith unit of type $t \in T$ has been on/off before period 0.

ramp rates: maximum power increase/decrease L_t^+, L_t^- between two consecutive periods; maximum power L_t^S when turned on; maximum power L_t^E before turned off; load L_{0it} for ith unit of type $t \in T$ before period 0.

Input (cyclic)

the status before period 0 $(\Delta_{0it}^+, \Delta_{0it}^-, L_{0it})$ are duplicated from period P-1.

Physical limits of the units: minimum up/down times and maximum ramp up/down rates

$\boxed{\text{commitment must be monitored for units individually}}$

minimum uptime

Let $P_t^+(p) = \{0 \leq p' \leq p \mid \sum_{k=p'}^{p-1} \Delta_k < \Delta_t^+\}$.

Show that an unit of type t cannot been turned on more than once during $P_t^+(p)$.

Show that if an unit of type t is off at time p then it has not been turned on at any time $p' \in P_t^+(p)$.

Reformulate these assertions as a linear relation between the binary variables modelling the unit status and status change at appropriate periods.

minimum uptime

Let $P_t^+(p) = \{0 \le p' \le p \mid \sum_{k=p'}^{p-1} \Delta_k < \Delta_t^+\}$.

If an unit of type t is off at time p ($x_{itp} = 0$) then it has not been turned on at any time $p' \in P_t^+(p)$ ($\sum_{p' \in P_t^+(p)} y_{itp}^+ = 0$).

$$\sum_{p' \in P_t^+(p)} y_{itp'}^+ \le x_{itp} \; \forall i, t, p$$

minimum uptime (noncyclic case)

If unit i of type t has been on for exactly $\Delta^+_{0it} > 0$ hours before time 0, what can you say about its status and status change at any period in

$$P^+_{it} = \{p \geq 0 \mid \sum_{k=0}^{p-1} \Delta_k < \Delta^+_t - \Delta^+_{0it}\}?$$

Fix binary variables modelling the status and status change of a unit at given periods according to this assertion.

minimum uptime (noncyclic case)

If unit i of type t has been on for exactly $\Delta^+_{0it} > 0$ hours before time 0, what can you say about its status and status change at any period in

$$P^+_{it} = \{p \geq 0 \mid \sum_{k=0}^{p-1} \Delta_k < \Delta^+_t - \Delta^+_{0it}\}?$$

the unit must remain on, then it will not be turned on/off, on these periods

$$x_{itp} = 1, y^+_{itp} = 0, y^-_{itp} = 0 \;\; \forall i, t, p \in P^+_{it}$$

minimum up/down-time

$$\sum_i x_{itp} = x_{tp},\ \forall t, p$$

$$\sum_{p' \in P_t^+(p)} y^+_{itp'} \leq x_{itp}\ \forall i, t, p$$

$$x_{itp} = 1, y^+_{itp} = 0, y^-_{itp} = 0\ \forall i, t, p \in P^+_{it}$$

$$\sum_{p' \in P_t^-(p)} y^-_{itp'} \leq 1 - x_{itp}\ \forall i, t, p$$

$$x_{itp} = 0, y^+_{itp} = 0, y^-_{itp} = 0\ \forall i, t, p \in P^-_{it}$$

$$y^+_{itp} + y^-_{itp} \leq 1\ \forall i, t, p$$

$$x_{itp} - x_{itp-1} = y^+_{itp} - y^-_{itp}\ \forall i, t, p$$

$$x_{itp}, y^+_{itp}, y^-_{itp} \in \{0,1\}\ \forall i, t, p$$

minimum up/down-time

$$\sum_i x_{itp} = x_{tp},\ \forall t, p$$

$$\sum_{p' \in P_t^+(p)} y^+_{itp'} \leq x_{itp}\ \forall i, t, p$$

$$x_{itp} = 1, y^+_{itp} = 0, y^-_{itp} = 0\ \forall i, t, p \in P^+_{it}$$

$$\sum_{p' \in P_t^-(p)} y^-_{itp'} \leq 1 - x_{itp}\ \forall i, t, p$$

$$x_{itp} = 0, y^+_{itp} = 0, y^-_{itp} = 0\ \forall i, t, p \in P^-_{it}$$

$$y^+_{itp} + y^-_{itp} \leq 1\ \forall i, t, p$$

$$x_{itp} - x_{itp-1} = y^+_{itp} - y^-_{itp}\ \forall i, t, p$$

$$x_{itp}, y^+_{itp}, y^-_{itp} \in \{0,1\}\ \forall i, t, p$$

x_{itp} commit status of the ith unit of type t on period p

y_{itp} unit turned on (+) or off (-) on period p

maximum ramp

If unit i of type t starts at p ($y_{itp} = 1$) then $l_{itp} - l_{itp-1} \leq L_{it}^{S}$

Otherwise either unit i is on at p-1 and on at p and $l_{itp} - l_{itp-1} \leq L_{it}^{+}$

or unit i is on at p-1 and off at p and $l_{itp} - l_{itp-1} < 0$

or unit i is off at p-1 and off at p and $l_{itp} - l_{itp-1} = 0$

maximum ramp

If unit i of type t starts at p ($y_{itp} = 1, x_{itp-1} = 0$) then $l_{itp} - l_{itp-1} \leq L^S_{it}$

Otherwise $(y_{itp} = 0)$ either unit i is on at p-1 $(x_{itp-1} = 1)$ and on at p and $l_{itp} - l_{itp-1} \leq L^+_{it}$

or unit i is on at p-1 $(x_{itp-1} = 1)$ and off at p and $l_{itp} - l_{itp-1} < 0 \leq L^+_{it}$

or unit i is off at p-1 $(x_{itp-1} = 0)$ and off at p and $l_{itp} - l_{itp-1} = 0$

$$l_{itp} - l_{itp-1} \leq L^+_t x_{itp-1} + L^S_t y^+_{itp} \ \forall i, t, p$$

maximum ramp up/down

$$\sum_i (l_{itp} - \underline{L}_t x_{itp}) = l_{tp},\ \forall t, p$$

$$l_{itp} - l_{itp-1} \le L_t^+ x_{itp-1} + L_t^S y_{itp}^+\ \forall i, t, p$$

$$l_{itp-1} - l_{itp} \le L_t^- x_{itp} + L_t^E y_{itp}^-\ \forall i, t, p$$

$$x_{it(-1)} = 1 \text{ if } L_{0it} > 0 \text{ else } x_{it(-1)} = 0\ \forall i, t$$

$$l_{it(-1)} = L_{0it}\ \forall i, t$$

$$\underline{L}_t x_{itp} \le l_{itp} \le \overline{L}_t x_{itp} \in \{0,1\}\ \forall i, t, p$$

maximum ramp up/down

$$\sum_i (l_{itp} - \underline{L}_t x_{itp}) = l_{tp},\ \forall t, p$$

$$l_{itp} - l_{itp-1} \leq L_t^+ x_{itp-1} + L_t^S y_{itp}^+\ \forall i, t, p$$

$$l_{itp-1} - l_{itp} \leq L_t^- x_{itp} + L_t^E y_{itp}^-,\ \forall i, t, p$$

$$x_{it(-1)} = 1 \text{ if } L_{0it} > 0 \text{ else } x_{it(-1)} = 0\ \forall i, t$$

$$l_{it(-1)} = L_{0it}\ \forall i, t$$

$$\underline{L}_t x_{itp} \leq l_{itp} \leq \overline{L}_t x_{itp} \in \{0,1\}\ \forall i, t, p$$

l_{itp} load of the ith unit of type t on period p

the MILP way

a practical view

(5)

Sophie Demassey 2023

CMA Centre de Mathématiques Appliquées

decomposition methods

cut generation/branch&cut

Dantzig-Wolfe/column generation/branch&price

lagrangian relaxation

Benders decomposition

$$\min \sum_{i=1}^{n} y_i$$

$$\text{s.t.} \sum_{j=1}^{m} w_j x_{ij} \le c y_i \qquad i = 1..n$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1..m$$

$$x_{ij} \in \{0,1\} \qquad i = 1..n;\ j = 1..m$$

$$y_i \in \{0,1\} \qquad i = 1..n$$

Bin Packing Problem

Input n containers, m items, capacity c for all containers, weight w_j for each item j

Output a packing of all items in a mimimum number of containers

how to manage the exponential number of variables ?

Bin Packing Problem

$$
\begin{aligned}
\min & \sum_{s \in \mathscr{S}} x_s & \\
\text{s.t.} & \sum_{s \in \mathscr{S}} a_{js} x_s = 1 & j = 1..n \\
& x_s \in \{0, 1\} & s \in \mathscr{S}
\end{aligned}
$$

Input n containers, m items, capacity c for all containers, weight w_j for each item j

Output a packing of all items in a mimimum number of containers

Dantzig-Wolfe decomposition

$\mathscr{S}$ all the possible arrangements of items in a bin

delayed column generation for LP

$min\{c_B x_B + c_N x_N \,|\, A_B x_B + A_N x_N = b\}$ without (c_N, A_N) i.e. $x_N = 0$:

1/ solve the restricted LP with the primal simplex algorithm where the omitted columns N are implicitly non-basic variables
2/ find $j \in N$ that can profitably enter the basis $\bar{c}_j < 0$, stop if none

= dual cut generation: (cut separation = pricing problem)

$\min cx$	$\max ub$
$A_i x \geq b_i, \quad \forall i$	$uA_j \leq c_j, \quad \forall j$
$x_j \geq 0, \quad \forall j$	$u_i \geq 0, \quad \forall i$

given a basic dual solution u find j such that $\bar{c}_j = c_j - uA_j < 0$

application to Bin Packing

$\mathcal{S} \subseteq 2^m$ all the possible arrangements of items in a bin

S a feasible subset (i.e. covering all the items)

1. solve the restricted LP:

   $\min\{\sum_{s \in S} x_s \mid \sum_{s \in S} a_{js} x_s = 1\ \forall j,\ x_s \geq 0\ \forall s \in S\}$

   get the corresponding dual solution $\bar{u} \in \mathbb{R}^m$

2. look for an improving basic direction

   = some $s \in \mathcal{S} \backslash S$ with $\bar{c}_s = 1 - \sum_j a_{js} \bar{u}_j < 0$

   e.g. by solving $\max\{\sum_j a_j \bar{u}_j \mid \sum_j w_j a_j \leq K, a \in \{0,1\}^m\}$

3. if $\sum_j a_j^* \bar{u}_j > 1$ add column $(1, a^*)$ to S then 1 otherwise

   STOP: $(\bar{x}_S, 0)$ solves the full LP (maybe not integer)

Branch-and-Price for MILP

- branch-and-bound for ILP with large number of variables where the LP relaxation is solved by column generation
- the branching strategy should keep the search tree balanced without altering the LP relaxation structure

ex (bin packing): **branch by fixing to 0 either all** $x_s \mid \{i,j\} \subseteq s$ **or all** $x_s \mid \{i,j\} \nsubseteq s$ **for some pair of items** (i,j) **s.t.** $0 < \sum_s a_{is} a_{js} x_s^* < 1$

- the pricing problem can be seen as an optimization problem but does not need to be solved at optimality, except for the convergence proof.
- convenient decomposition method when additional constraints only appear in the pricing problem

ex (bin packing): **conflict constraint** $\sum_{j \in C} a_j \leq 1$

Multi 0-1 Knapsack Problem

$$
\begin{aligned}
&\max \sum_{i=1}^{m} \sum_{j=1}^{n} c_j x_{ij} && \\
&\text{s.t.} \sum_{j=1}^{n} w_j x_{ij} \le K_i && i = 1..m \\
&\sum_{i=1}^{m} x_{ij} \le 1 && j = 1..n \\
&x_{ij} \in \{0,1\} && j = 1..n, i = 1..m
\end{aligned}
$$

Input n items, m bins, value c_j and weight w_j for each item j, capacity K_i for each bin i.

Output a maximum value subset of items packed in the bins.

Multi 0-1 Knapsack Problem

$$
\begin{aligned}
z_u = \quad & \max \sum_{i=1}^{m}\sum_{j=1}^{n} c_j x_{ij} + \sum_{j=1}^{n} u_j\Big(1 - \sum_{i=1}^{m} x_{ij}\Big) && u \in \mathbb{R}_+^n \\
\text{s.t.} \; & \sum_{j=1}^{n} w_j x_{ij} \le K_i && i = 1..m \\
& \cancel{\sum_{i=1}^{m} x_{ij} \le 1} && \cancel{j = 1..n} \\
& x_{ij} \in \{0,1\} && j = 1..n, i = 1..m
\end{aligned}
$$

lagrangian relaxation

Input n items, m bins, value c_j and weight w_j for each item j, capacity K_i for each bin i.

Output a maximum value subset of items packed in the bins.

find the smallest upper bound

Lagrangian Relaxation

dualize the complicating or coupling constraints of an ILP:

$$
\begin{aligned}
(P): z = \max & \sum_k c_k x_k \\
& \sum_k D_k x_k \le e_k \\
& A_k x_k \le b_k, \qquad \forall k \\
& x_k \in \mathbb{Z}^p \times \mathbb{R}^n, \qquad \forall k
\end{aligned}
$$

$$
\begin{aligned}
(D): w &= \min_{u \ge 0} l(u) \\
l(u) &= ue + \sum_k z_k^u
\end{aligned}
$$

$$
\begin{aligned}
(P_u): z_u^k = \max\ & c_k x_k - u D_k x_k \\
& A_k x_k \le b_k \\
& x_k \in \mathbb{Z}^p \times \mathbb{R}^n
\end{aligned}
$$

(D) is the lagrangian dual problem

(P_u) is the lagrangian suproblem with multipliers u

strong duality may not hold if p>0, ie the dual only provides an upper bound $w \ge z$

lagrangian relaxation applied to MKP

$$(P) : z = \max \sum_i \sum_j c_j x_{ij}$$
$$\sum_j w_j x_{ij} \le K_i, \qquad \forall i$$
$$\sum_i x_{ij} \le 1, \qquad \forall j$$
$$x_{ij} \in \{0,1\}, \qquad \forall i, j$$

$$(D) : w = \min_{u \ge 0} l(u) \quad \text{with} \quad l(u) = \sum_j u_j + \sum_i z_i^u$$

$$(P_i^u) : z_i^u = \max \sum_j (c_j - u_j) x_{ij}$$
$$\sum_j w_j x_{ij} \le K_i$$
$$x_{ij} \in \{0,1\}, \forall j$$

- function l is convex and a subgradient at $u \ge 0$ is $1 - \sum_i x_i^u$ where x_i^u an optimal solution of (P_i^u) a 0-1 knapsack with altered costs
- at each iteration, for a given u, the solution x^u is KP-feasible but some items may be assigned more than once: remove the less profitable doublons to get a feasible solution
- if no doublon and if every item j with $u_j > 0$ is assigned then x^u is optimal for (P)

lagrangian relaxation: applications

- in MKP: the knapsacks subproblems share the same set of items but different capacities: helpful to speed up the solution of (P^u)
- the lagrangian dual is always at least as good as the LP relaxation
- sometimes it is not better, ex: dualize the knapsack constraints instead of the assignment constraints in MKP
- lagrangian relaxation is applied, daily and for decades, by EDF to the Unit Commitment Problem for the french electricity production: dualize the unit coupling constraints and generate independent commitment plans for each unit. It allows to take into account specific technical rules (e.g. ramping) for each unit types.
- another typical application in planning: dualize time (loosely-)coupling constraints

Benders decomposition

- typically: problems coupling binary/continuous variables

  $P : min\{cx + dy \,|\, x \in P \cap \mathbb{Z}^p, Ax + By \geq e\}$ where
  $f(x) = min\{dy \,|\, By \geq e - Ax\}$ can be dualized

- strong duality: either feasible $f(x) = max\{u(e - Ax) \,|\, uB \leq d\}$ or infeasible and it exists a ray $u \,|\, \lambda uB \leq d \,\forall \lambda, u(e - Ax) > 0$

  $P : min\{cx + z \,|\, x \in P \cap \mathbb{Z}^p, z \geq f(x)\}$

- relax $z \geq f(x)$ then at each iteration k: solve the relaxation and get solution x^k, solve the dual subproblem get u^k and generate a cut, either $z \geq u^k(e - Ax)$ if feasible or $0 \geq u^k(e - Ax)$ otherwise

- stop when lower bound $cx^k + z^k$ is equal to best upper bound $cx^j + f(x^j)$

a glimpse of MINLP

convex continuous relaxation:

- NLP-B&B: bound by solving the NLP relaxation with an interior point method
- OA algorithm: cutting-plane method with cuts as first-order approximation (LP outer approximation)
- LP-NLP B&B: a branch-and-cut with an LP relaxation with OA cuts generated at each integer node

nonconvex continuous relaxation:

- spatial B&B: branch on integer variables and on nonconvex constraints

MILP perks

performance
sophisticated algorithms

declarative
models, not algorithms

large-scale
decomposition methods

certification
primal-dual bounds

versatile
covers many problems

flexible
general-purpose solvers

logic & constraint programming

integer nonlinear programming

graph algorithms

combinatorial optimization beyond MILP

machine learning

dynamic programming

metaheuristics

Matteo Fischetti (2019) Introduction to Mathematical Optimization
Wolsey L. (1998) Integer Programming. Wiley
Bertsimas D., Tsitsiklis J. (1997) Introduction to Linear Optimization. Athena Scientific

Achterberg T., Berthold T., Hendel G. (2012) Rounding and propagation heuristics for MIP. In OR Proceedings 2011, 71-76.
Achterberg T., Bixby R., Gu Z., Rothberg E., Weninger D. (2016) Presolve reductions in MIP. ZIB-Report 16-44.
Bixby R. (2012) A brief history of LP and MIP computation. Documenta Mathematica, 107-121.
Cornuéjols G. (2008) Valid inequalities for MILP. Mathematical Programming, 112(1), 3-44.
Klotz E., Newman A. (2013) Practical guidelines for solving difficult MILP. Surveys in ORMS, 18(1), 18-32.
Linderoth J., Ralphs T. (2005) Noncommercial software for MILP. IP: theory and practice, 3, 253-303.
Linderoth J., Lodi A. (2010) MILP software. Wiley encyclopedia of ORMS.
Linderoth J., Savelsbergh M. (1999) A computational study of search strategies for MIP. INFORMS JoC, 11(2), 173-187.
Lodi A. (2010) MIP computation. In 50 Years of IP 1958-2008, 619-645.
Newman A., Weiss M. (2013) A survey of linear and mixed-integer optimization tutorials. INFORMS ToE, 14(1) 26-38.
Savelsbergh M. (1994) Preprocessing and probing techniques for MIP. ORSA Journal on Computing, 6(4), 445-454.
Vanderbeck F., Wolsey L. (2010) Reformulation and decomposition of IP. In 50 Years of IP 1958-2008, 431-502.