



the MILP way

a practical view

Sophie Demassey 2014

Combinatorics everywhere



1

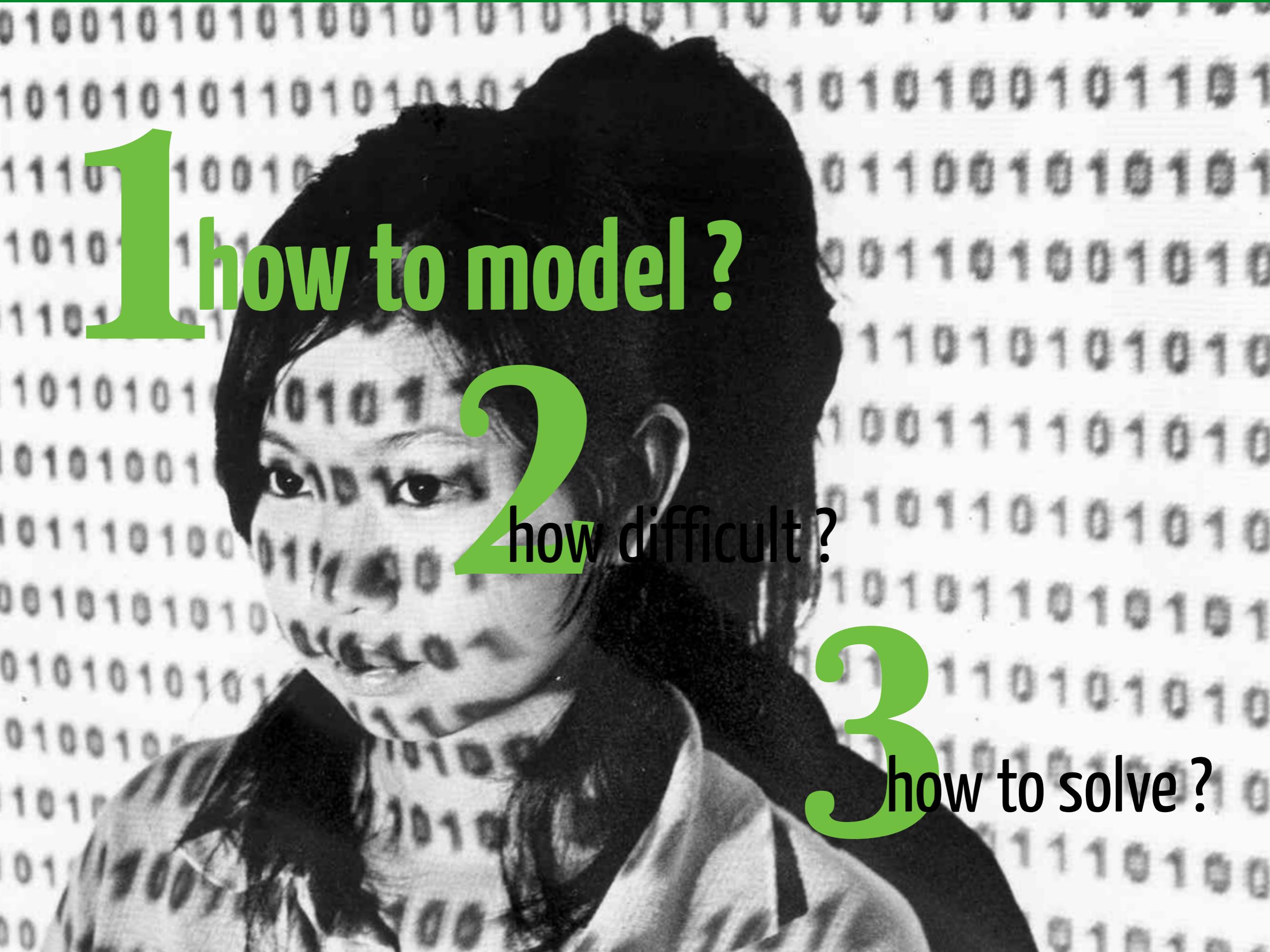
how to model ?

2

how difficult ?

3

how to solve ?

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

~~true¹ or false⁰~~

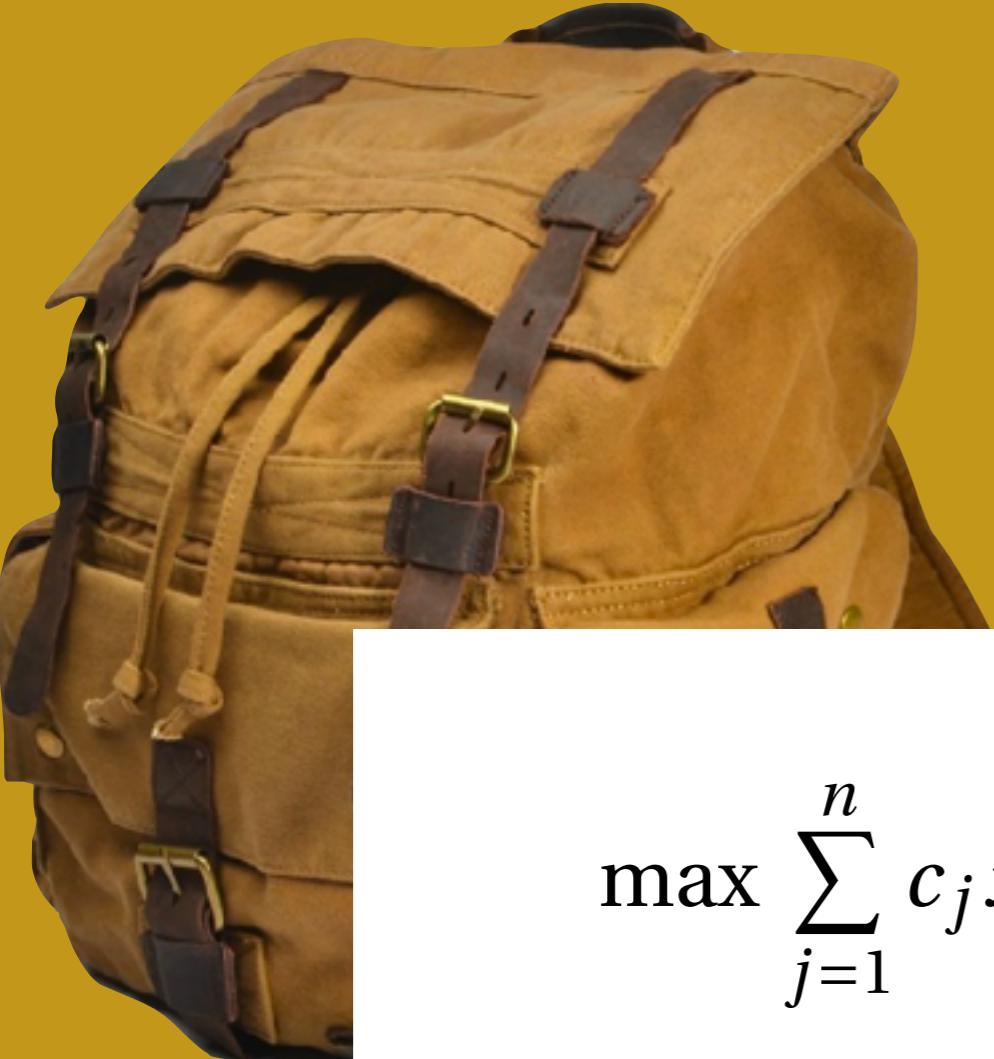
- is item j selected ? $x_j \in \{0, 1\}$
- is item j associated to item i ? $y_{ij} \in \{0, 1\}$
- is variable x greater than a ? $x \geq ay, y \in \{0, 1\}$
- is constraint c satisfied ? ...



Integer Knapsack Problem

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 .



Integer Knapsack Problem

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

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

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

x_j is item j packed ?

logic with binaries

- either x or y
$$x + y = 1$$
- if x then y
$$y \geq x$$
- if x then $f \leq a$
$$f \leq ax + M(1 - x)$$
 “big M”
- at most 1 out of n
$$x_1 + \cdots + x_n \leq 1$$
- at least k out of n
$$x_1 + \cdots + x_n \geq k$$



Uncapacitated Facility Location Problem

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 minimum (opening and service) cost assignment of customers to facilities.

Uncapacitated Facility Location Problem

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

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

$$y_{ij} \leq x_j \quad j = 1..n, i = 1..m$$

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

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

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



1||C_{max} Scheduling Problem

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

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



1 || Cmax Scheduling Problem

$$\min s_{n+1}$$

$$\text{s.t. } s_{n+1} \geq s_j + p_j \quad j = 1..n$$

$$s_j - s_i \geq Mx_{ij} + (p_i - M) \quad i, j = 1..n$$

$$x_{ij} + x_{ji} = 1 \quad i, j = 1..n; i < j$$

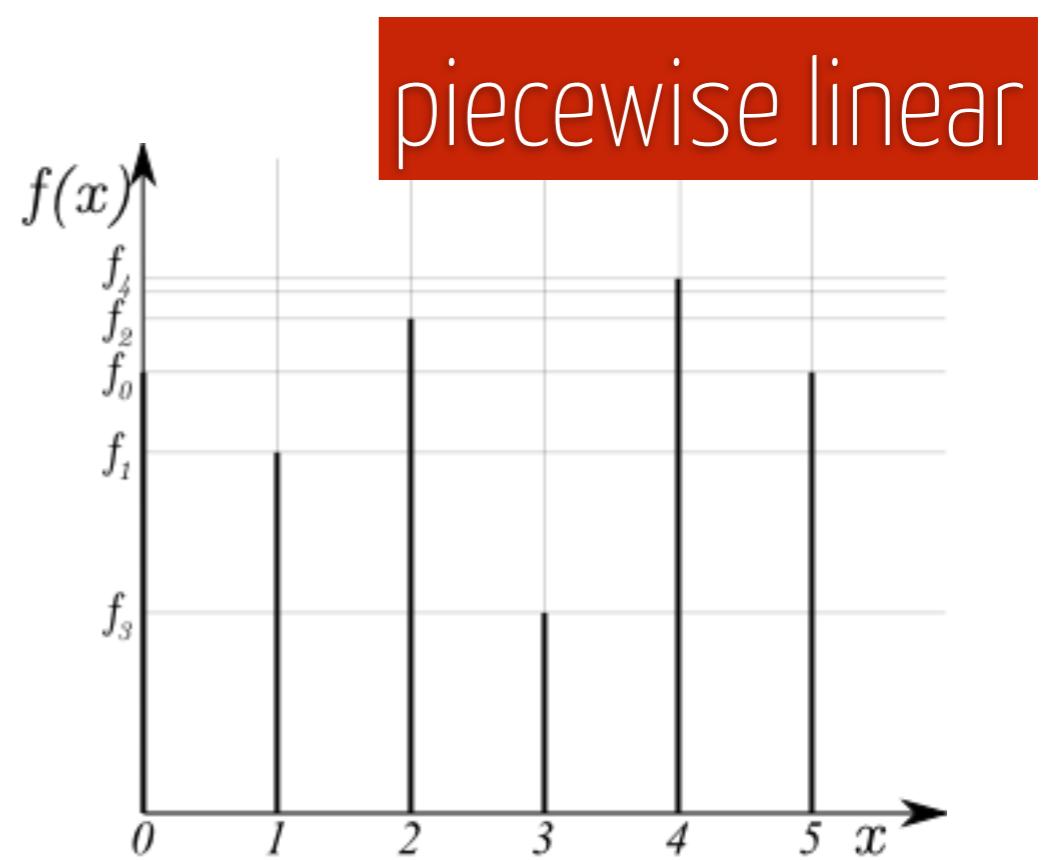
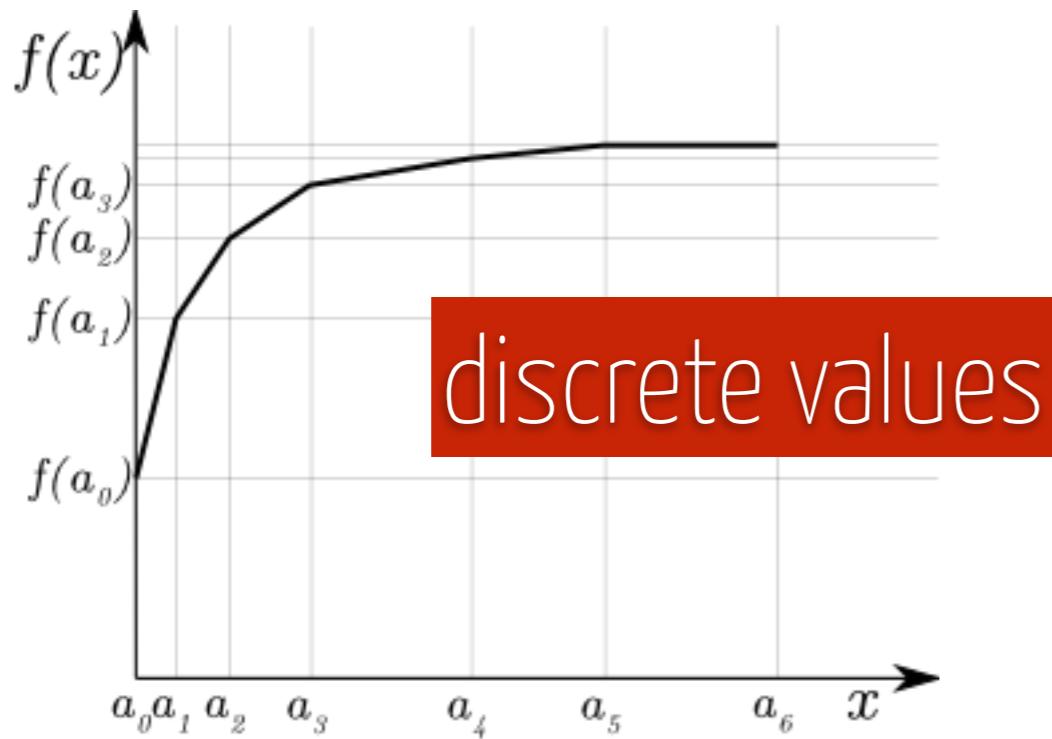
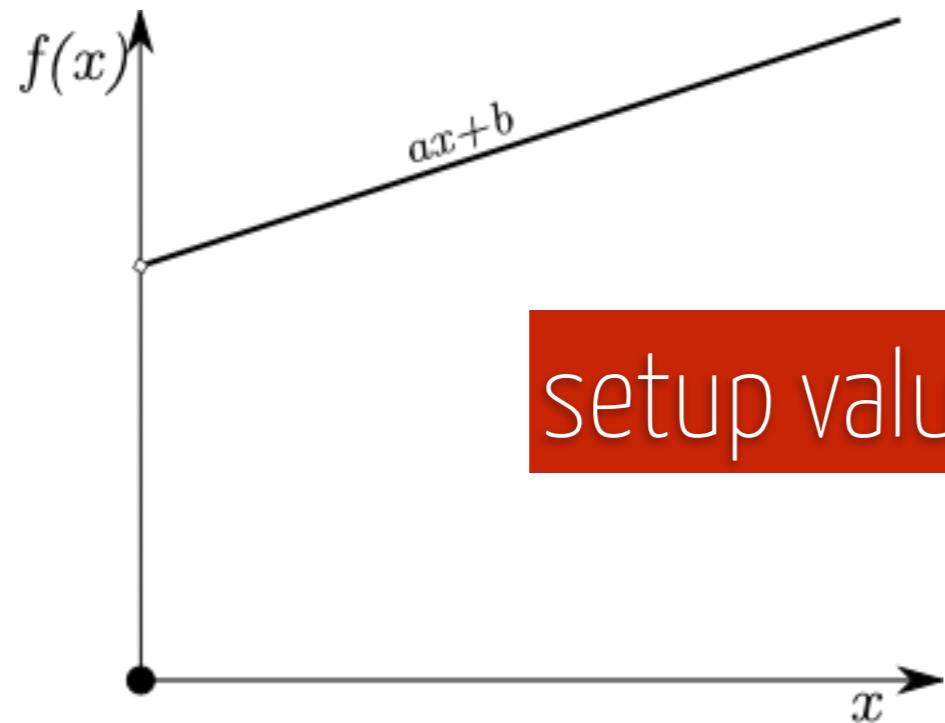
$$s_j \in \mathbb{Z}_+ \quad j = 1..n+1$$

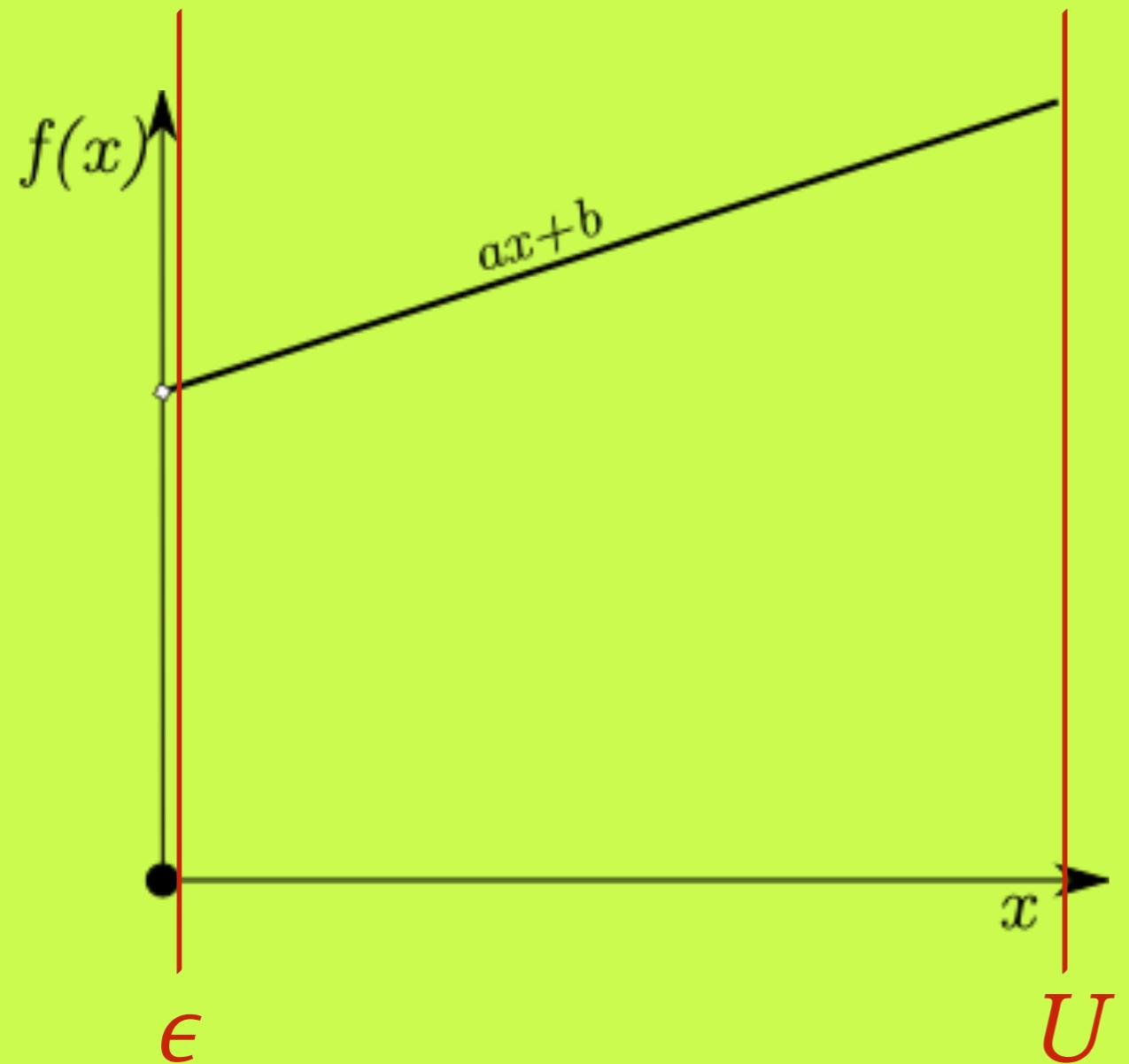
$$x_{ij} \in \{0, 1\} \quad i, j = 1..n$$

on pi for
ine
span
on the
ap

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

non-linear functions





setup value

$$f(x) = ax + b\delta$$

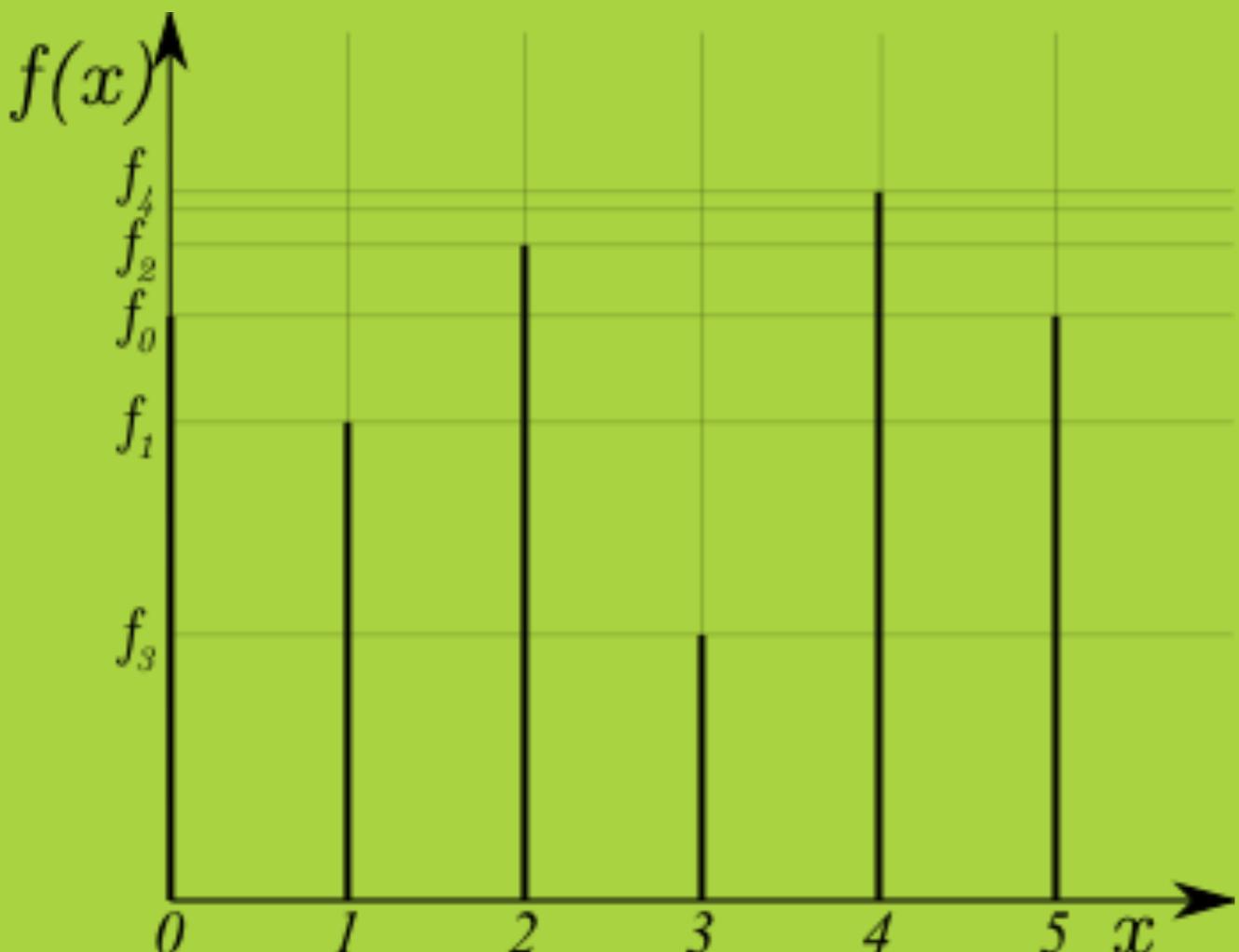
$$\epsilon\delta \leq x \leq U\delta$$

$$\delta \in \{0, 1\}$$

δ is x positive ?

Special Ordered Set of type 1:

ordered set of variables, all zero except at most one



discrete values

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

$$\sum_i i \delta_i = x$$

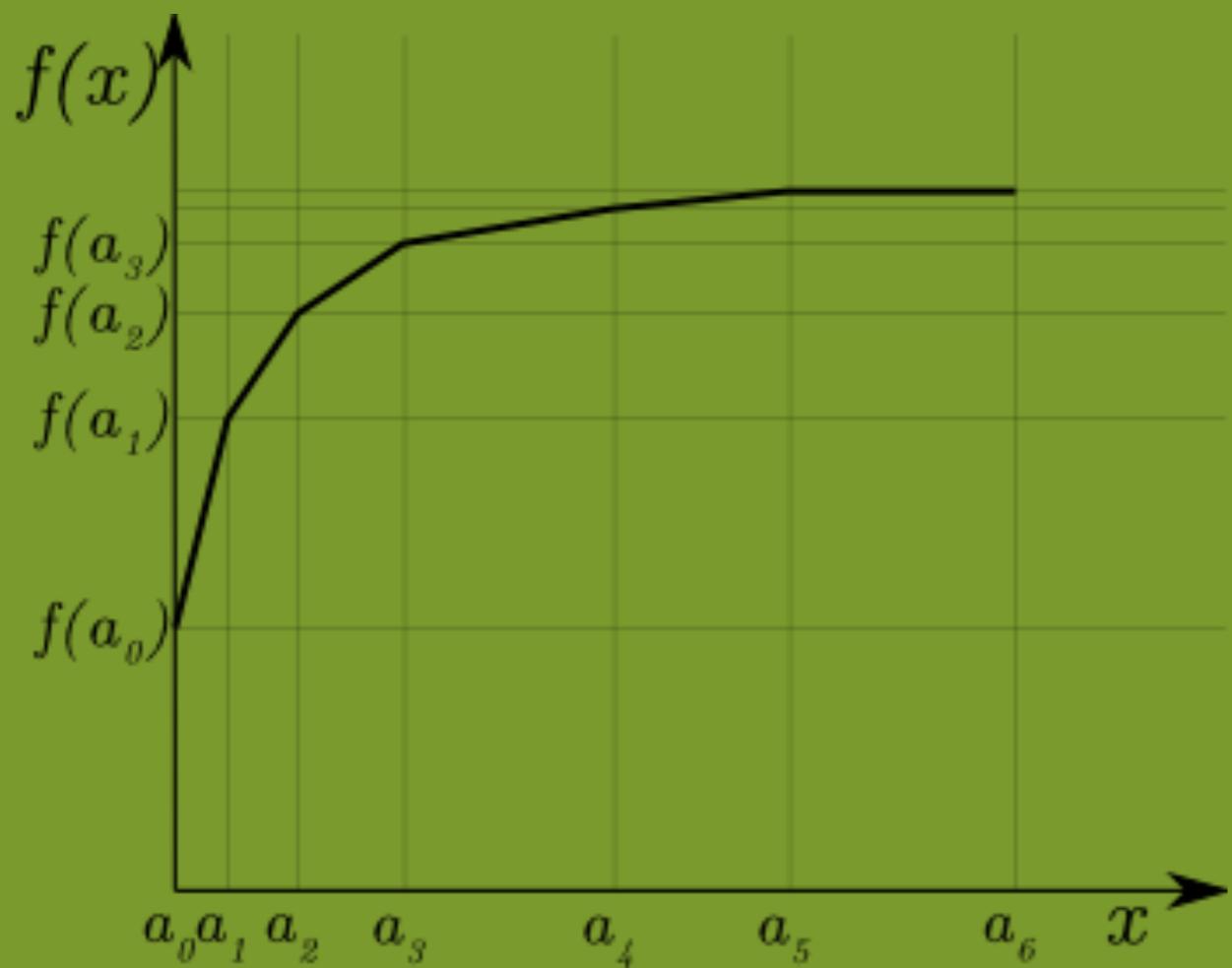
~~$$\sum_i \delta_i = 1$$~~ **SOS1(δ)**

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

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

$$f(x) = \sum_i \lambda_i f(a_i)$$

$$\sum_i a_i \lambda_i = x$$

$$\lambda_i \in [0, 1] \quad i = 0..n$$

SOS2(λ)

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

$$x_i = 5$$

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

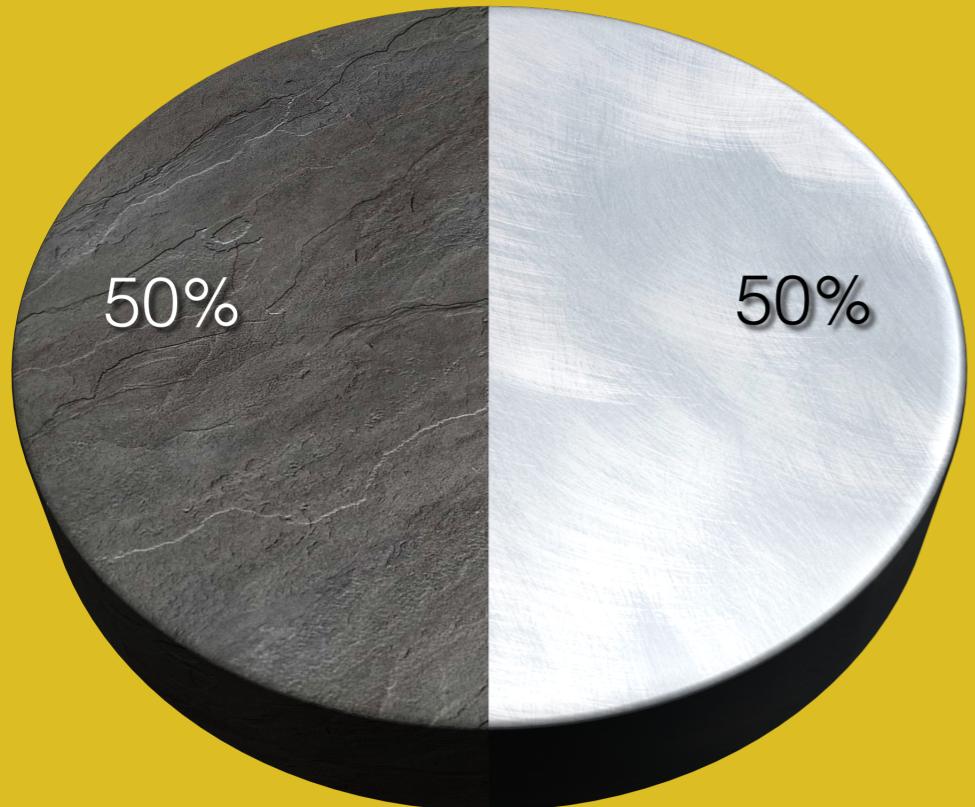
$$\approx \delta_{i5} = 1$$

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

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

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

Interlude 1



Market Split Problem

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.

Interlude 1

Market Split Problem

$$\min \sum_{j=1}^m s_j^+ + s_j^-$$

$$\text{s.t. } \sum_{i=1}^n a_{ij} x_i + s_j^+ - s_j^- = \frac{d_j}{2} \quad j = 1..m$$

$$x_i \in \{0, 1\}$$

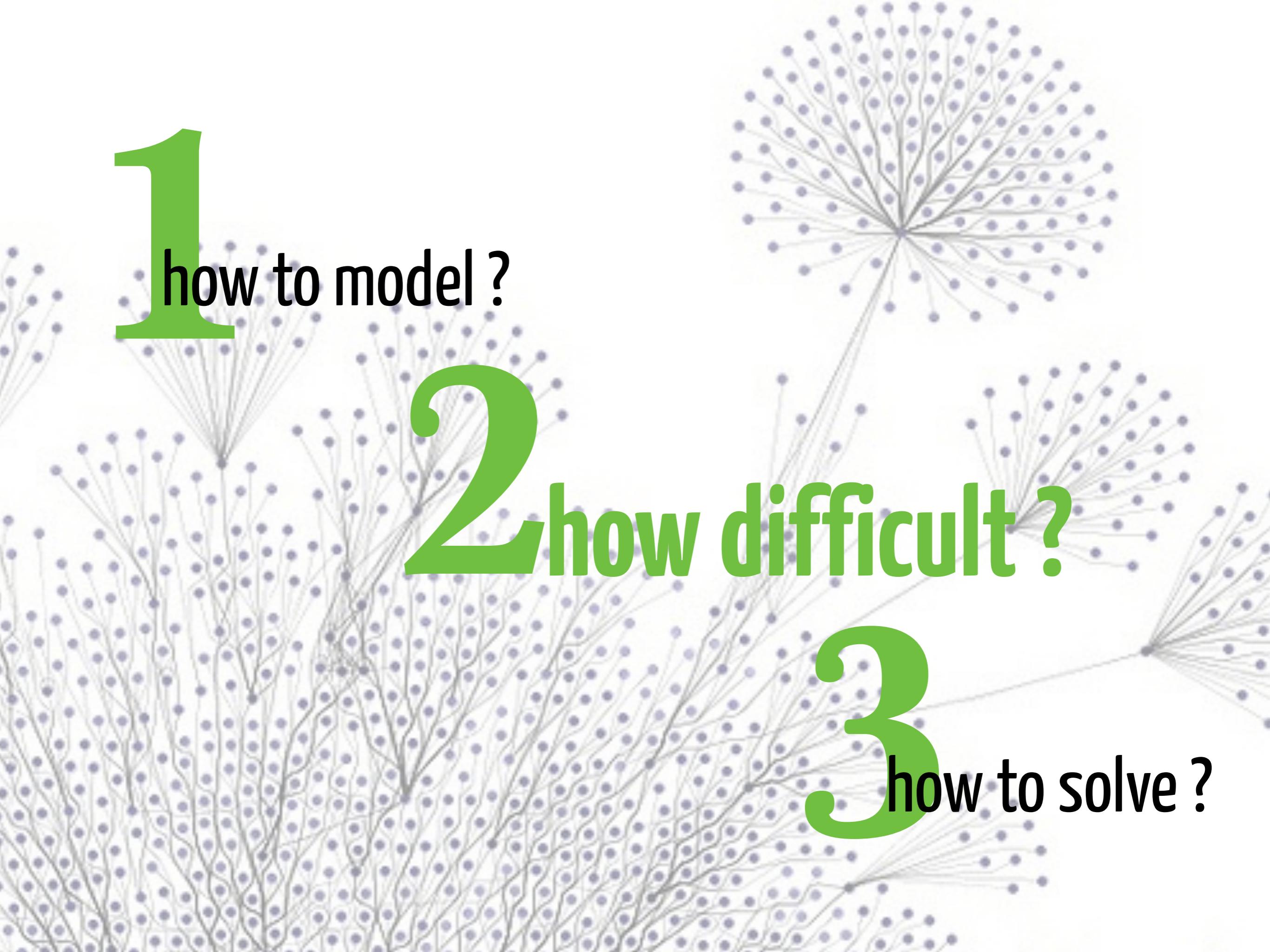
$$i = 1..n$$

$$s_j^+ \geq 0, s_j^- \geq 0$$

$$j = 1..m$$

x_i is retailer i assigned to division 1 ?

s_j gap to the 50% split goal for product j

A faint, light-gray network diagram serves as the background for the entire slide. It consists of numerous small, dark purple circular nodes connected by thin, gray lines that branch out like tree roots or dendrites.

1

how to model ?

2

how difficult?

3

how to solve ?

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

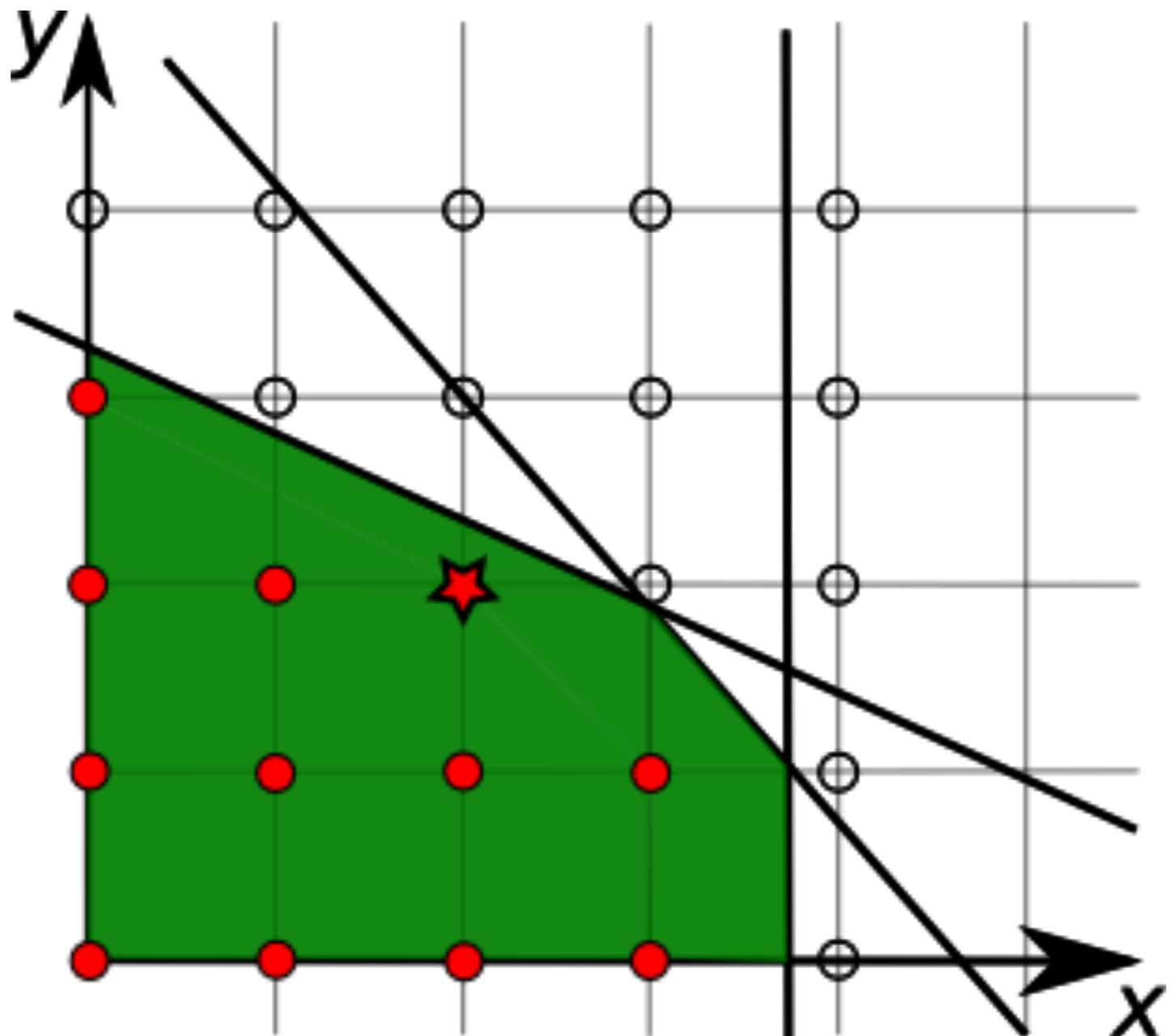
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)

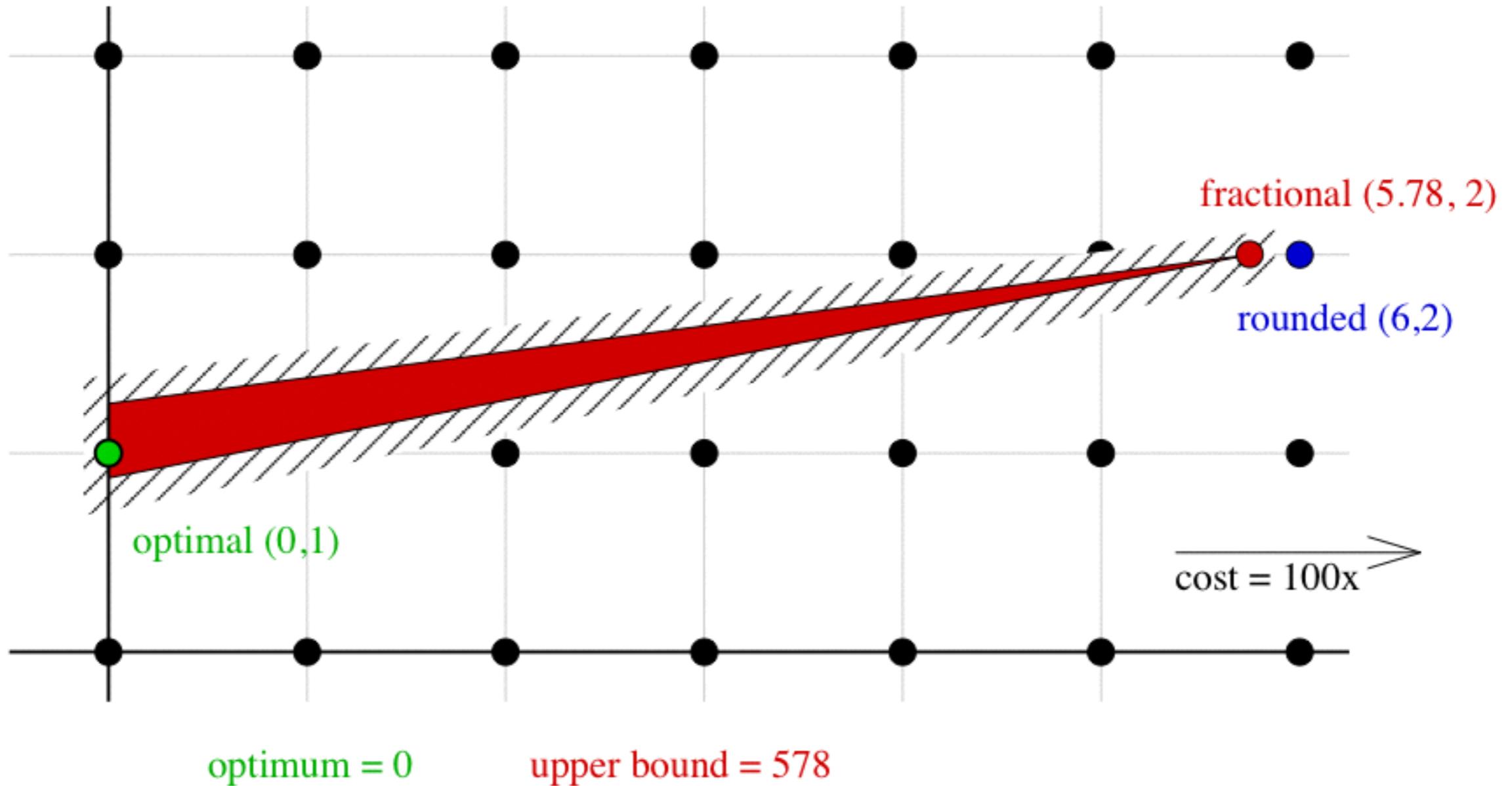
Best objective 1.00000000000e+00, best bound 1.00000000000e+00, gap 0.0%

Optimal objective: 1

LP \neq ILP



round LP \neq ILP



“ILP is NP-hard: I can't solve it !”



1||Cmax Scheduling Problem

$$\min s_{n+1} = p_1 + \dots + p_n$$

$$\text{s.t. } s_{n+1} \geq s_j + p_j \quad j = 1..n$$

$$s_j - s_i \geq Mx_{ij} + (p_i - M) \quad i, j = 1..n$$

$$x_{ij} + x_{ji} = 1 \quad i, j = 1..n; i < j$$

$$s_j \in \mathbb{Z}_+ \geq 0 \quad j = 1..n+1$$

$$x_{ij} \in \{0, 1\} \quad i, j = 1..n$$

on p_i for
ine
span
on the
ap



Capacitated Transhipment Problem

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 minimum cost integer flow to satisfy the demand

Capacitated Transhipment Problem

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

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

$$x_{ij} \leq h_{ij}$$

$$x_{ij} \in \mathbb{Z}_+ \geq 0$$

$(i, j) \in A$

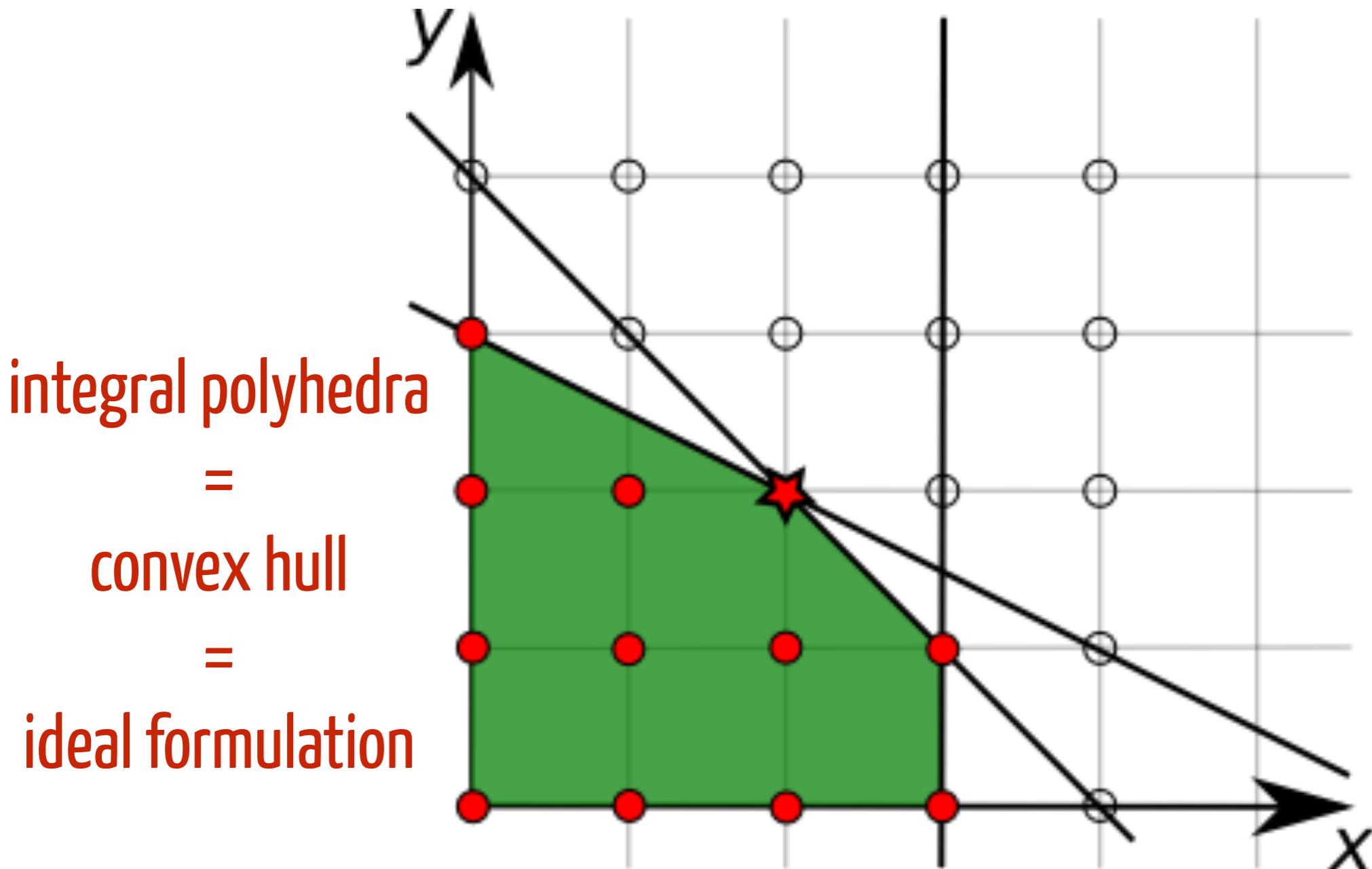
$(i, j) \in A$

demand or
 b_i ,
flow cost

integer
demand

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

LP = ILP



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

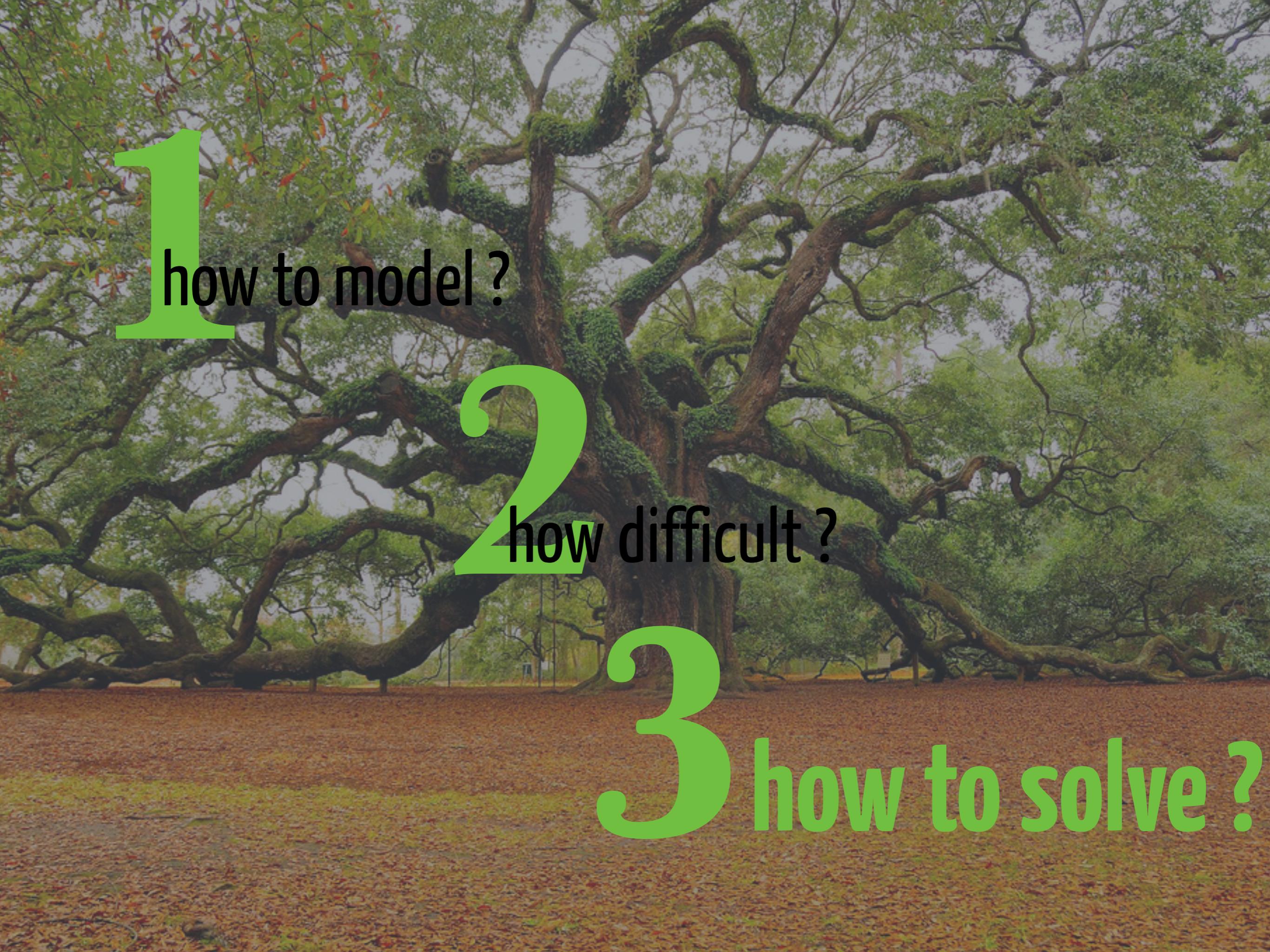
A is TU $\iff A^t$ is TU $\iff (A, I_m)$ is TU

where A^t is the transpose matrix, I_m the identitiy matrix

Interlude²

Show that the **Transhipment ILP** is **ideal**

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



1

how to model ?

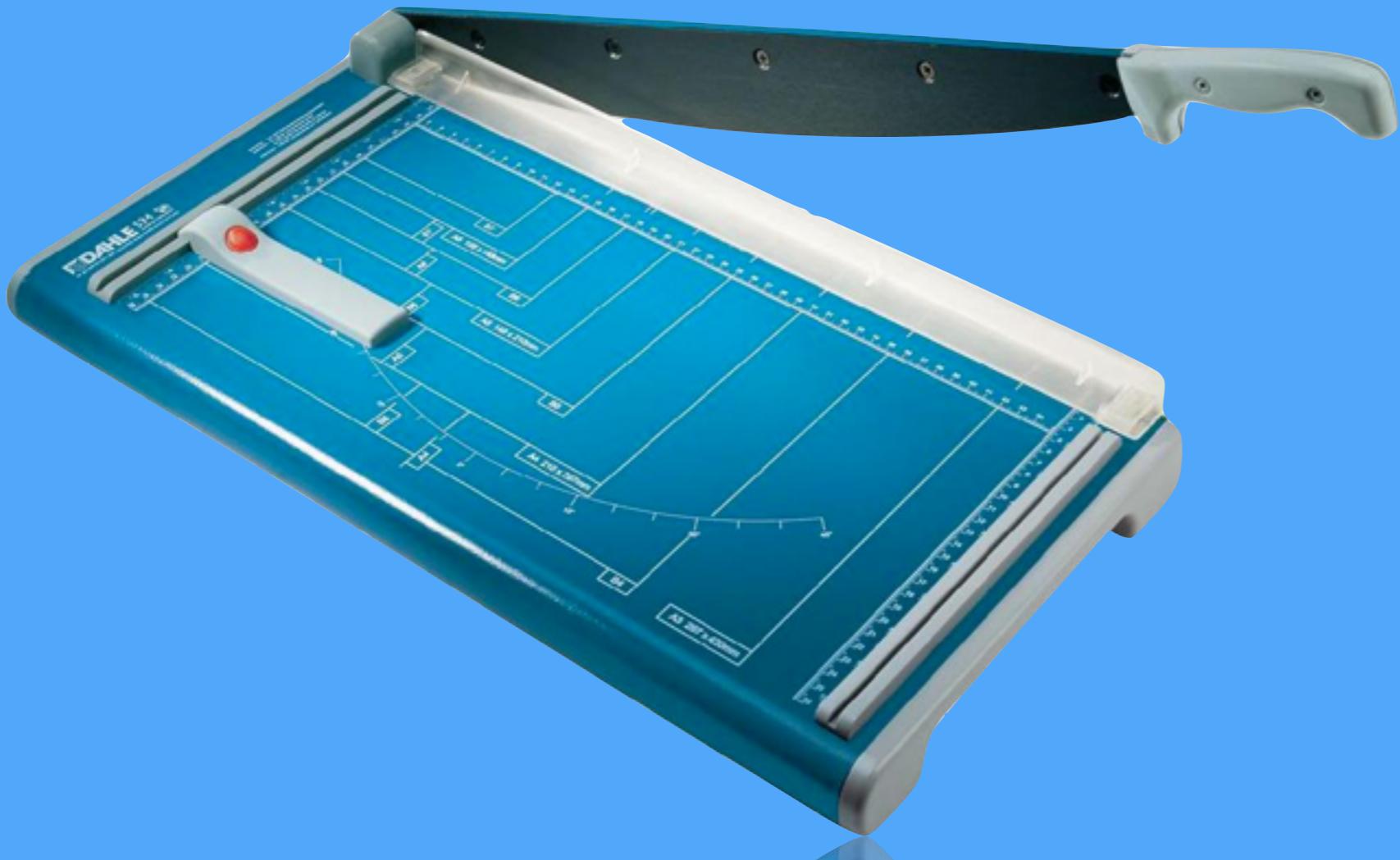
2

how difficult ?

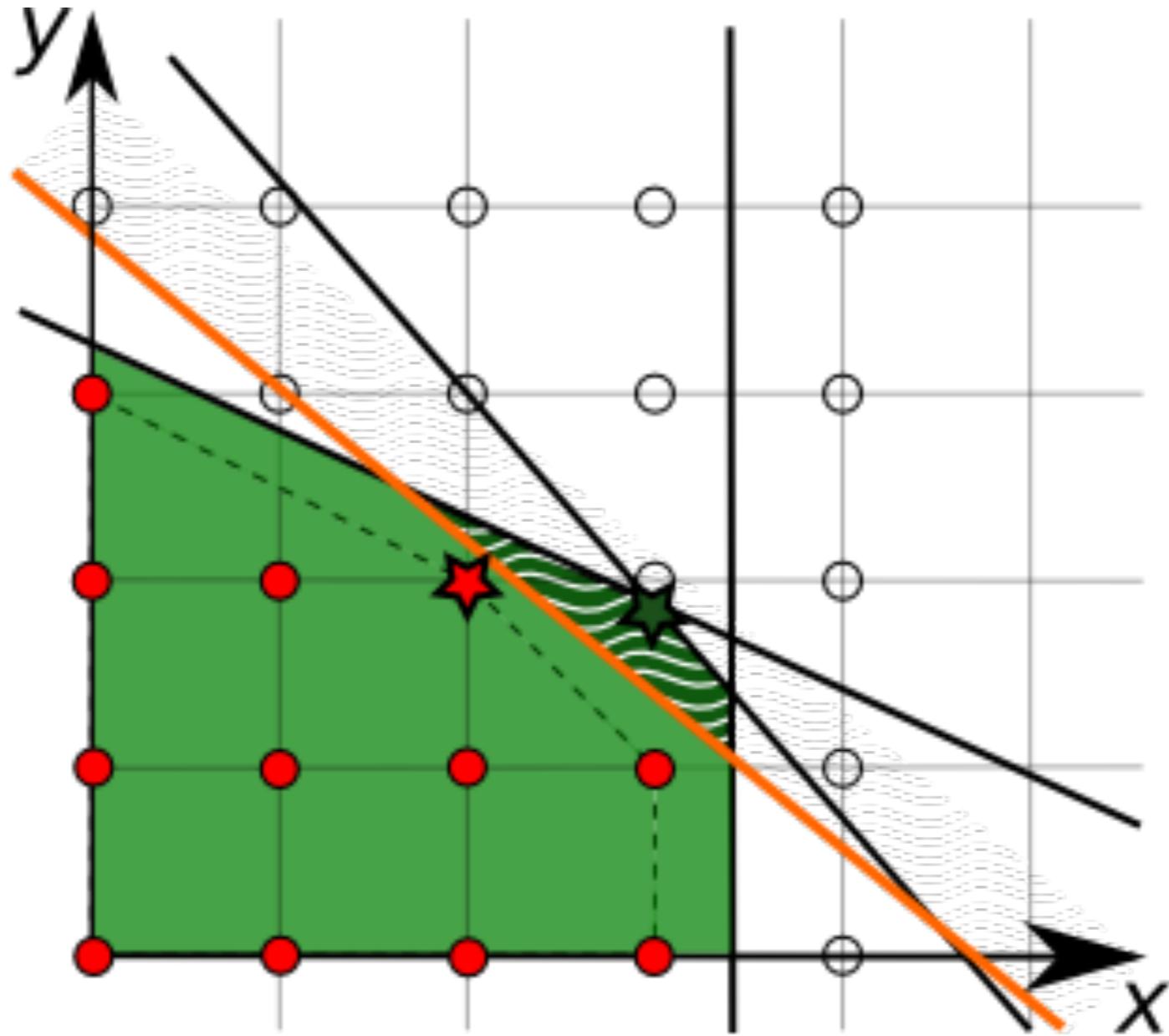
3

how to solve ?

- 1 Cuts** compute an ideal formulation and solve the LP
- 2 Branch&Bound** enumerate solutions implicitly
- 3 modern Branch&Cut** mix up+presolve +heuristics
- 4 decomposition methods** (Branch&Price, Lagrangian, Benders)



Cutting Plane Algorithm



Cut valid inequality that separates the LP solution

Farkas Lemma any cut is a linear combination of the constraints

cutting plane algorithm

1. solve the LP relaxation (P), get x^*
2. if x^* is integral, STOP
3. find a cut for (P, x^*) from a template T

templates

generic

Gomory Mixed Integer, Mixed Integer Rounding, Split, Chvátal-Gomory

structural

clique, cover, flow cover, zero half

problem-specific

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

ex 1 Mixed Integer Rounding

Combining constraints, then rounding leads to valid inequalities.

Let $u \in \mathbb{R}_+^m$, then the following inequalities are valid for (P) :

- surrogate: $\sum_{j=1}^m u_j a_{ij} x_i \leq \sum_{j=1}^m u_j b_j$ (since $u \geq 0$)
- round off: $\sum_{j=1}^m \lfloor u_j a_{ij} \rfloor x_i \leq \sum_{j=1}^m u_j b_j$ (since $\lfloor u_j a_{ij} \rfloor \leq u_j a_{ij}$ and $x \geq 0$)
- Chvátal-Gomory: $\sum_{j=1}^m \lfloor u_j a_{ij} \rfloor x_i \leq \lfloor \sum_{j=1}^m u_j b_j \rfloor$ (since $e \in \mathbb{Z}$ and $e \leq f$ implies that $e \leq \lfloor f \rfloor$)
- CG inequalities form a generic class of valid inequalities: they apply to any IP

ex 2 cover

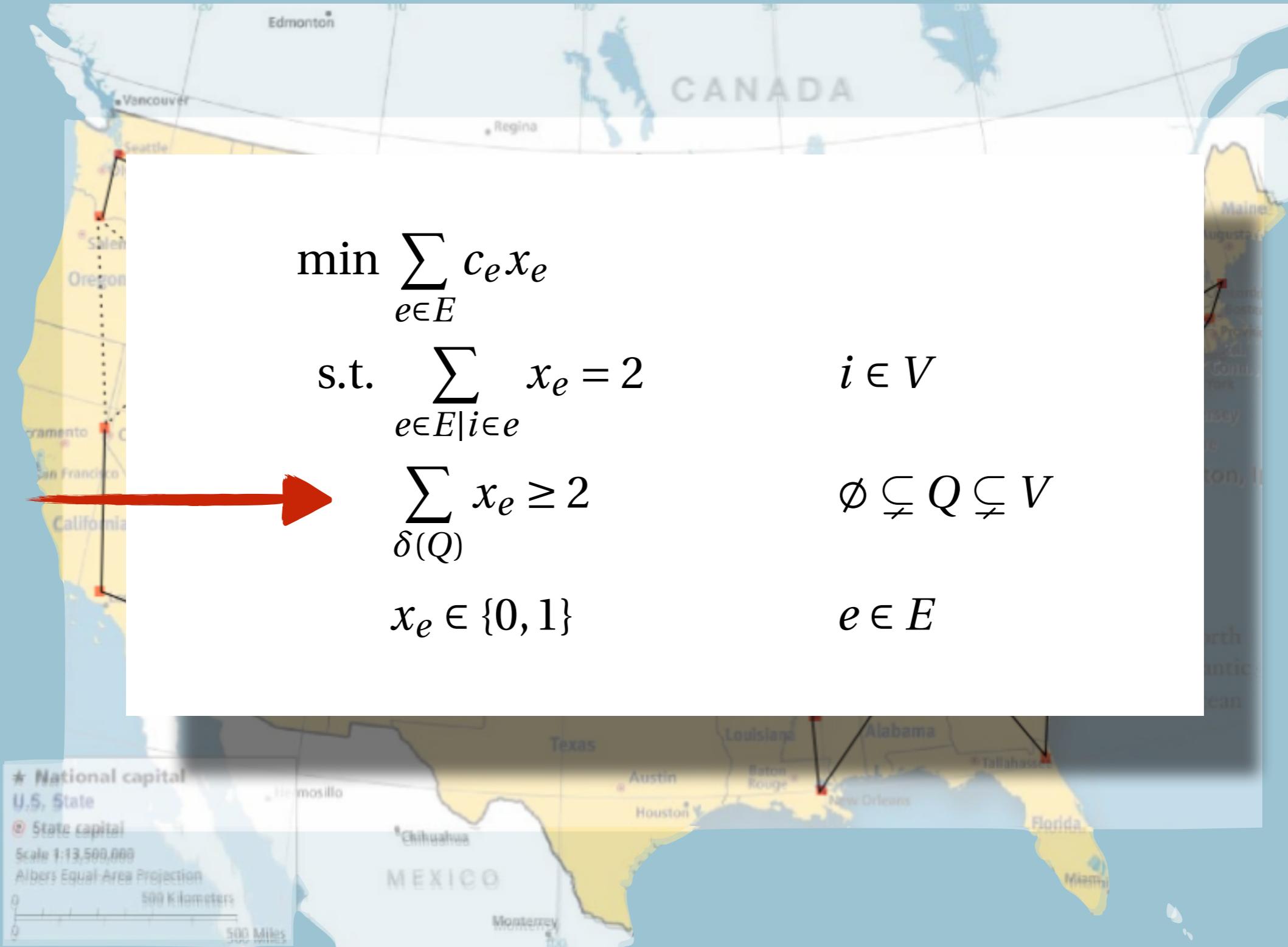
Cover inequalities

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

- (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

The procedure to get this last equality is called *lifting*

ex 3 Subtour for TSP



ex 3 Subtour for TSP



limits

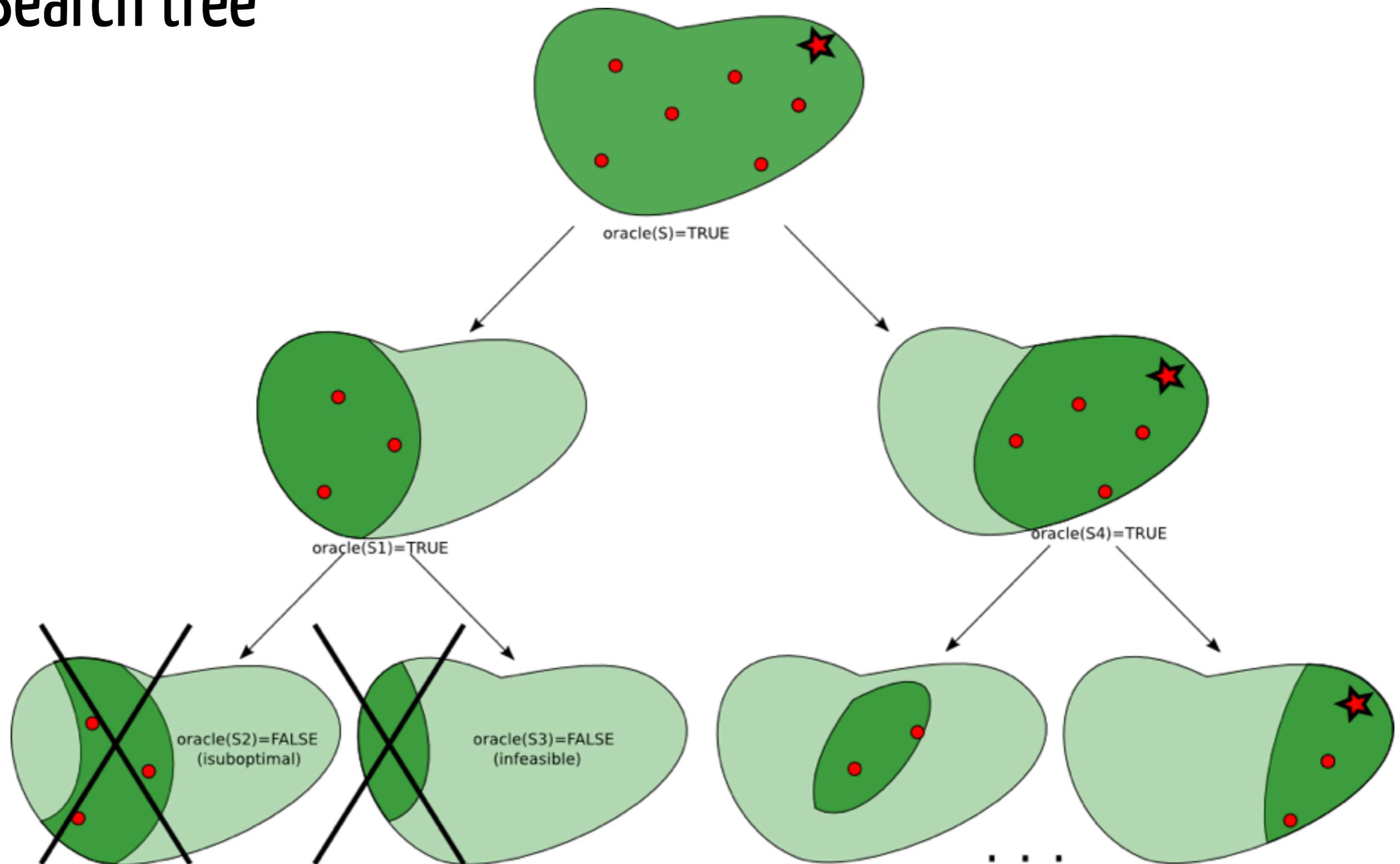
depending on the cut families

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

Branch and Bound



Search tree

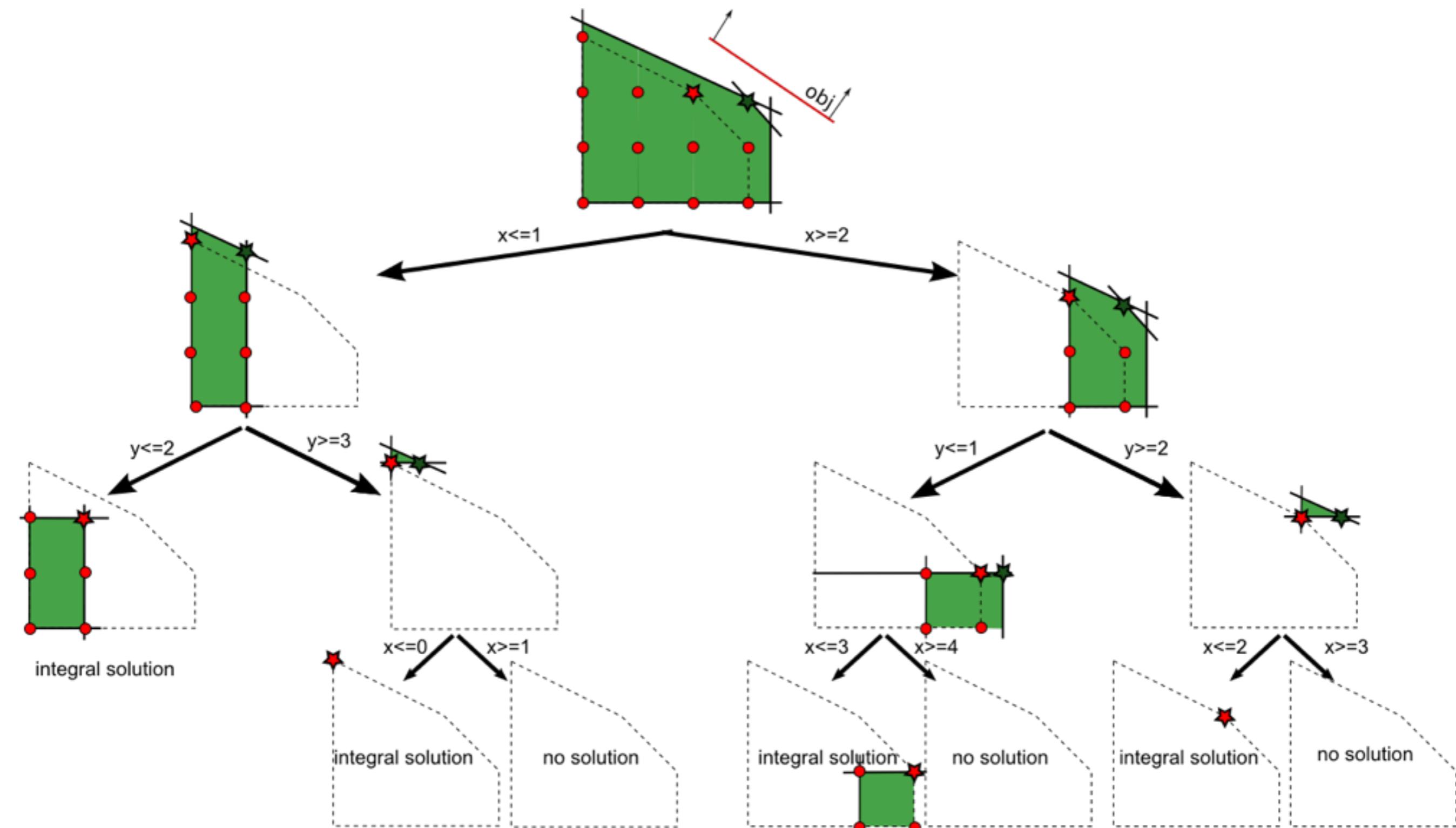


LP-based B&B

oracle(S) = FALSE iff either:

- LP is infeasible
- the fractional solution \bar{x}
is not better than the
incumbent x^*
- \bar{x} is integer (update x^*)

then prune node S



branching

node selection

which order to visit nodes ?

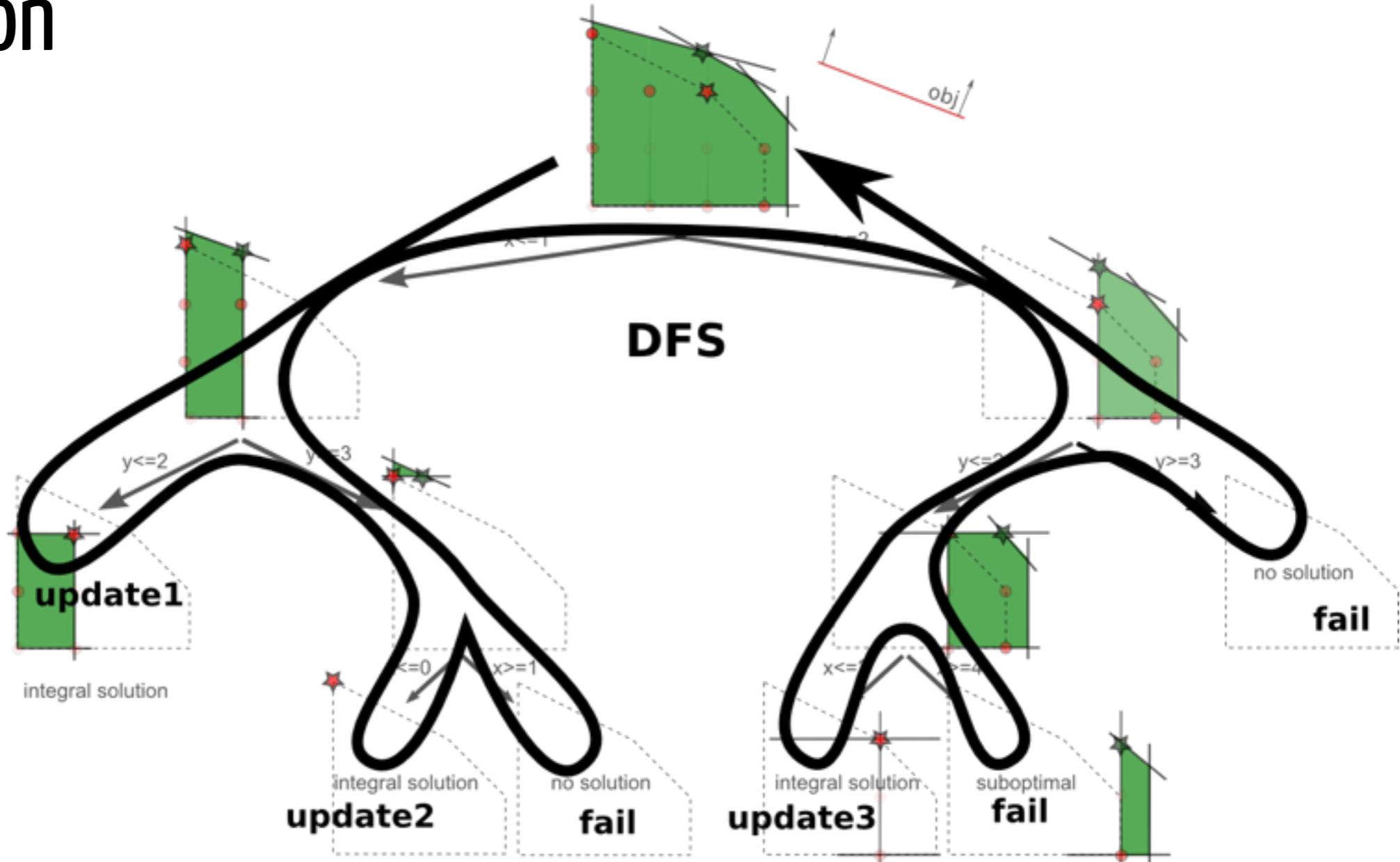
variable selection

how to separate nodes ?

constraint branching

alternative to variable branching

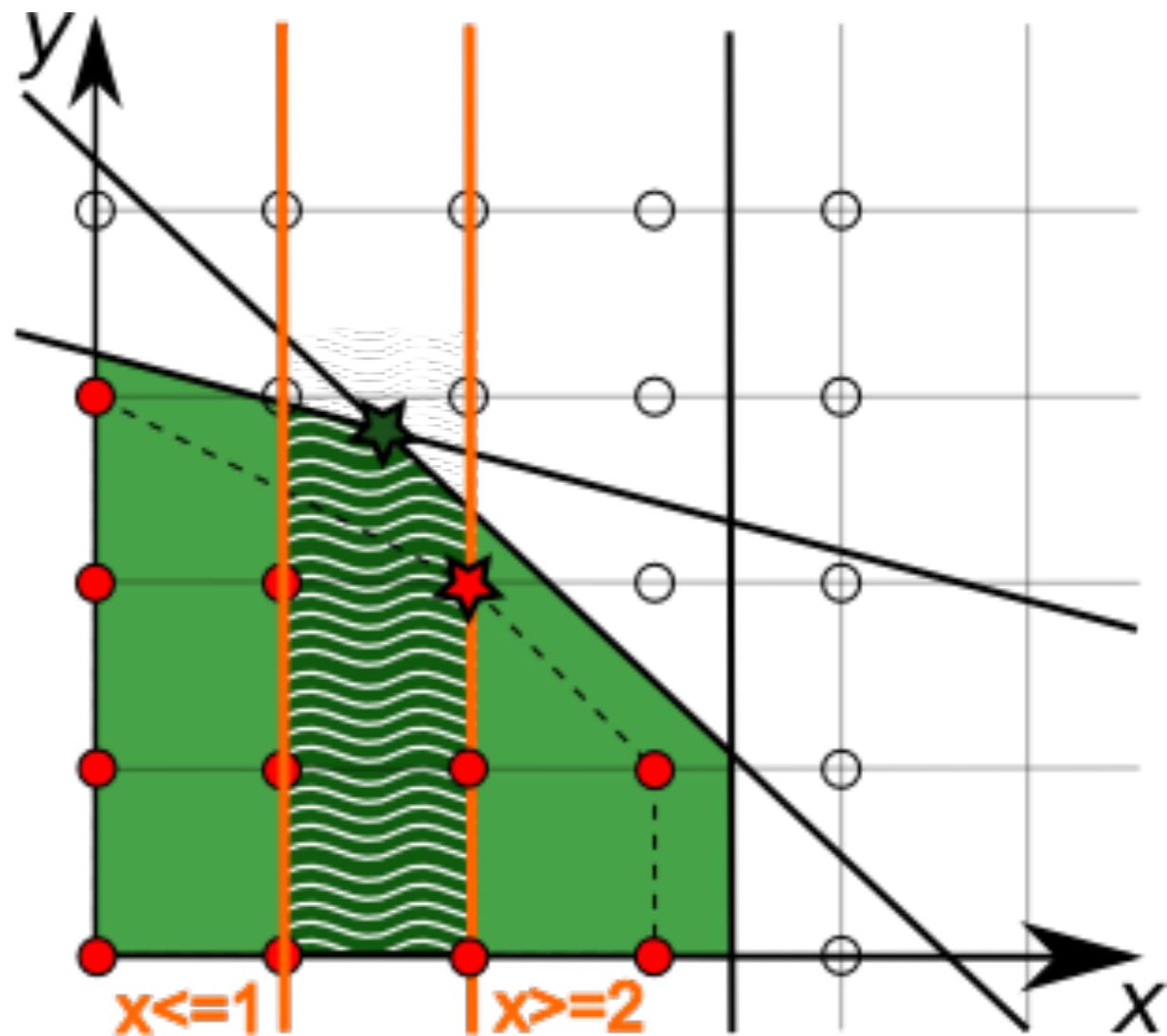
node selection



Best Bound First Search explore less nodes, manages larger trees

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

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:

$$\text{COST} = 100x_1 + 180x_2 + 320x_3 + 450x_4 + 600x_5$$

$$\text{SIZE} = 10x_1 + 20x_2 + 40x_3 + 60x_4 + 80x_5$$

$$(\text{SOS1}) : x_1 + x_2 + x_3 + x_4 + x_5 = 1$$

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

modern solvers



Simplex
var branching

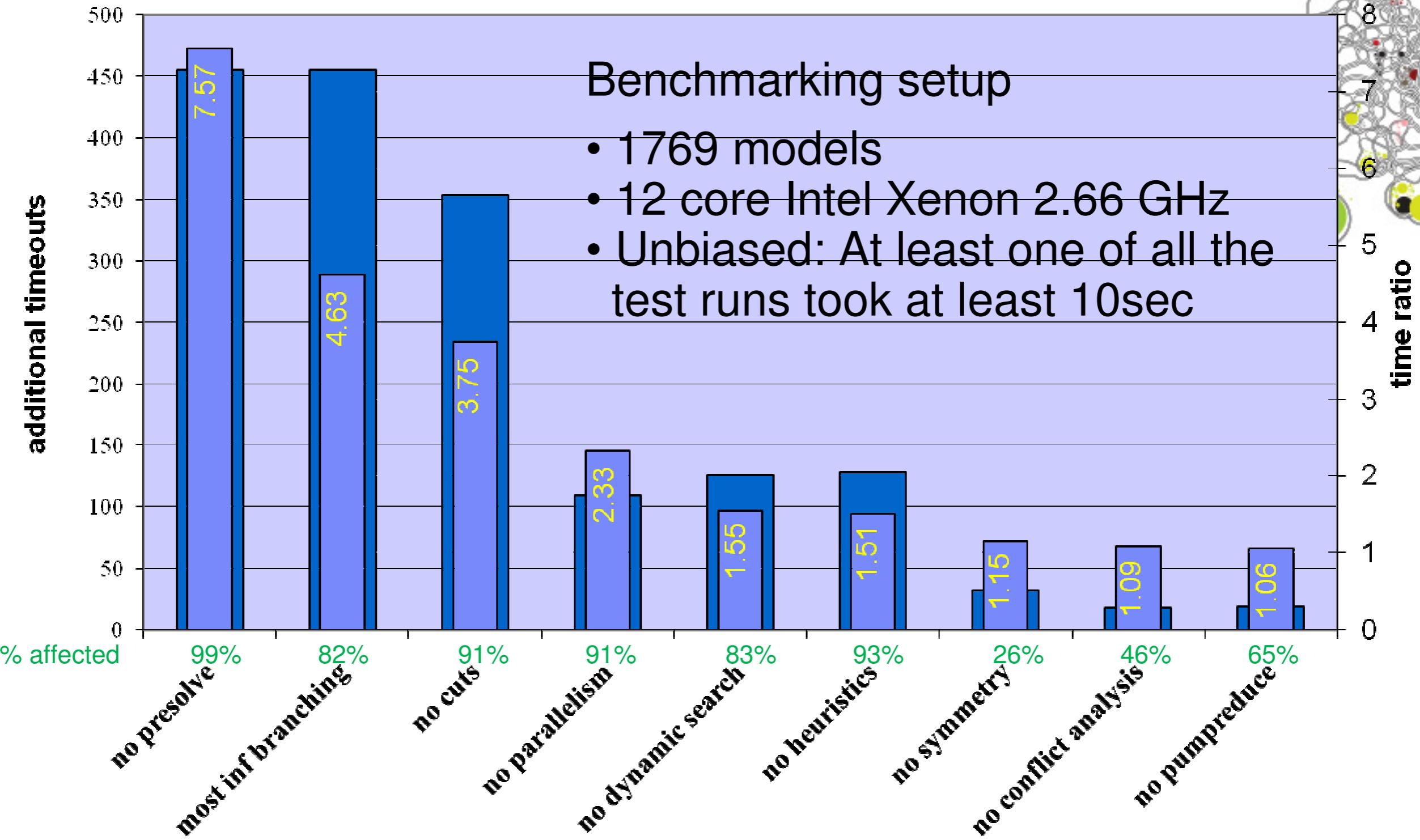
Preprocessing

Branch & Cut

Heuristics

Parallelism

Component Impact CPLEX 12.5 Summary



CPLEX 11

ILOG CPLEX generates its cuts in such a way that they subsequent cuts, ILOG CPLEX may add a constraint to

- [Clique Cuts](#)
- [Cover Cuts](#)
- [Disjunctive Cuts](#)
- [Flow Cover Cuts](#)
- [Flow Path Cuts](#)
- [Gomory Fractional Cuts](#)
- [Generalized Upper Bound \(GUB\) Cover Cuts](#)
- [Implied Bound Cuts](#)
- [Mixed Integer Rounding \(MIR\) Cuts](#)
- [Adding Cuts and Re-Optimizing](#)
- [Counting Cuts](#)
- [Parameters Affecting Cuts](#)

GUROBI 5.6

Parameter name	Purpose
Cuts	Global cut generation control
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
ModKCuts	Mod-k cut generation
NetworkCuts	Network cut generation
SubMIPCuts	Sub-MIP cut generation
ZeroHalfCuts	Zero-half cut generation
CutAggPasses	Constraint aggregation passes performed
CutPasses	Root cutting plane pass limit
GomoryPasses	Root Gomory cut pass limit

Preprocessing

remove redundancies

turn variables to constants

strengthen bounds and coefficients

build the conflict graph

fix variables by probing

remove symmetries

(cut generation)

MIPLIB markshare_5_0

```
[root@... /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
-----------	-------	----	-----------	---------	------	-----	-------

Explorod 233848403 nodes (460515864 simplex iterations) in 3883.5 seconds

Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)

Best objective 1.0000000000e+00, best bound 1.0000000000e+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 Expl	Unexpl	Current Node Obj	Depth	IntInf	Objective Incumbent	Bounds BestBd	Gap	Work It/Node	Time
H	0	0	0.00000	0	5	5335.00000	0.00000	100%	-	0s
	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
H	30682	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

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 Expl	Unexpl	Current Node Obj	Depth	IntInf	Objective Incumbent	Bounds BestBd	Gap	Work It/Node	Time
H	0	0	0.00000	0	5	5335.00000	0.00000	100%	-	0s
	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	Expl	Unexpl	Current Node Obj	Depth	IntInf	Objective Bounds Incumbent	BestBd	Gap	Work It/Node	Time
H	0	0	0	0.00000	0	5	5335.00000	0.00000	100%	-	0s
	0	0	0			320.000000	0.00000	100%	-	0s	
	0	0	0	0.00000	0	6	320.00000	0.00000	100%	-	0s
	0	0	0	0.00000	0	5	320.00000	0.00000	100%	-	0s
	0	0	0	0.00000	0	6	320.00000	0.00000	100%	-	0s
	0	0	0	0.00000	0	5	320.00000	0.00000	100%	-	0s
H	0	0	0			239.000000	0.00000	100%	-	0s	
	0	0	0	0.00000	0	5	239.00000	0.00000	100%	-	0s
*	36	0		29		96.000000	0.00000	100%	2.7	0s	
*	99	32		34		58.000000	0.00000	100%	2.1	0s	
H	506	214				53.000000	0.00000	100%	1.9	0s	
H30682		442				1.000000	1.00000	0.00%	2.1	0s	

change the LP solver

if $\text{nblIteration}(\text{node}) \geq \text{nblIteration}(\text{root})/2$

NodeMethod=2

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

	Nodes Expl	Unexpl	Current Node Obj	Depth	IntInf	Objective Incumbent	Bounds BestBd	Gap	Work It/Node	Time
H	0	0	0.00000	0	5	5335.00000	0.00000	100%	-	0s
	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

supply 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 Expl	Unexpl	Current Node Obj	Depth	IntInf	Objective Incumbent	Bounds BestBd	Gap	Work It/Node	Time
H	0	0	0.00000	0	5	5335.00000	0.00000	100%	-	0s
	0	0			320.000000	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			239.000000	0.00000	100%	-	0s	
	0	0	0.00000	0	5	239.00000	0.00000	100%	-	0s
*	36	0		29	96.000000	0.00000	100%	2.7	0s	
*	99	32		34	58.000000	0.00000	100%	2.1	0s	
H	506	214			53.000000	0.00000	100%	1.9	0s	
H30682		442			1.000000	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/>

[/documentation/5.6/reference-manual/mip_models](#)

[resources/seminars-and-videos/overview](#)

[/documentation/5.6/reference-manual/parameter_tuning_tool](#)

you know your problem better
than your solver

**Improve
your
MODEL**

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

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

$$\sum_{i=1}^m y_{ij} \leq mx_j \quad j = 1..n$$

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

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

14 hours

Facilitated
ty Location
roblem

Input: n facility locations, m

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

2 seconds

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

$$y_{ij} \leq x_j \quad j = 1..n, i = 1..m$$

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

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



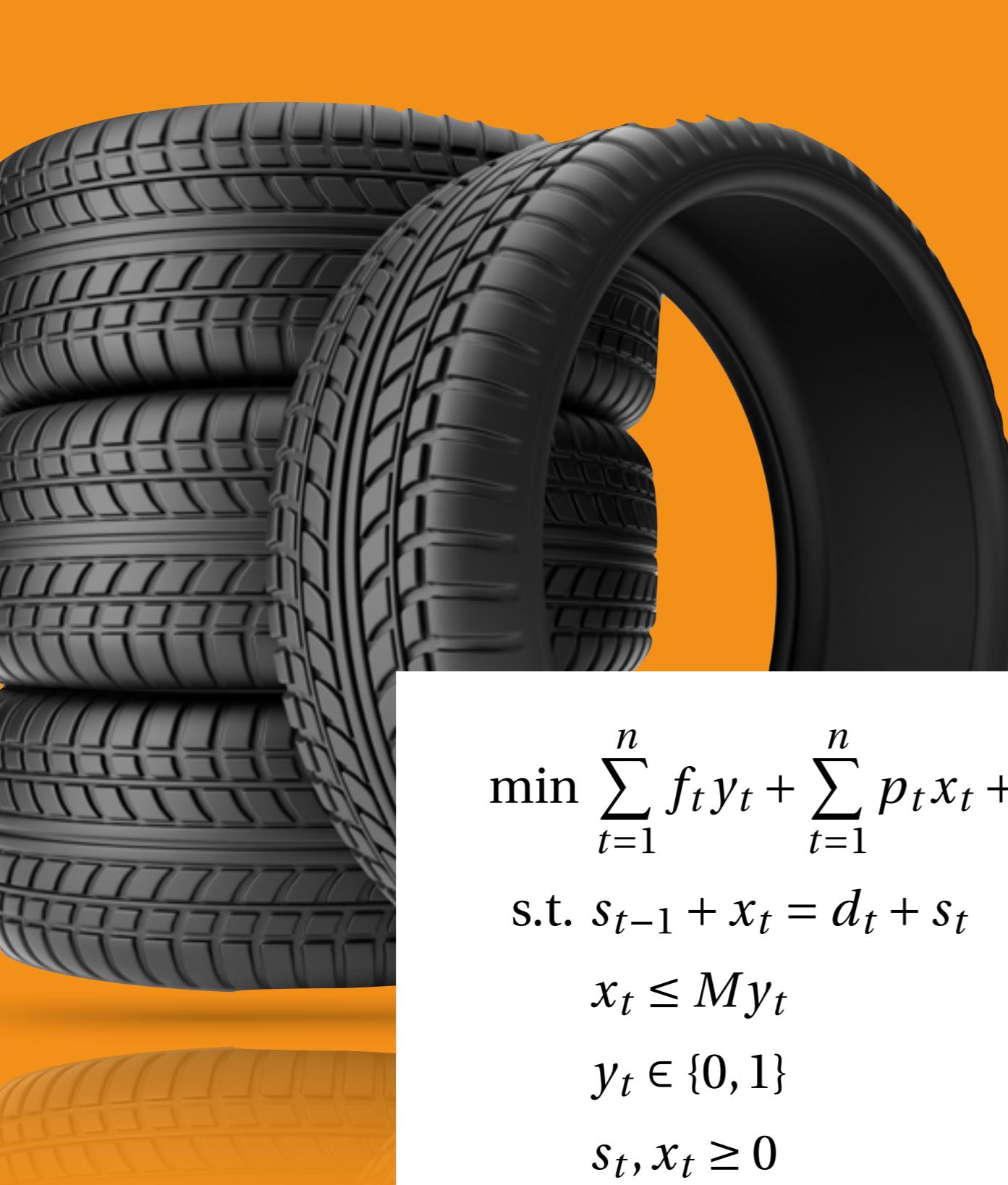
m=n=40



Uncapacitated Lot Sizing Problem

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

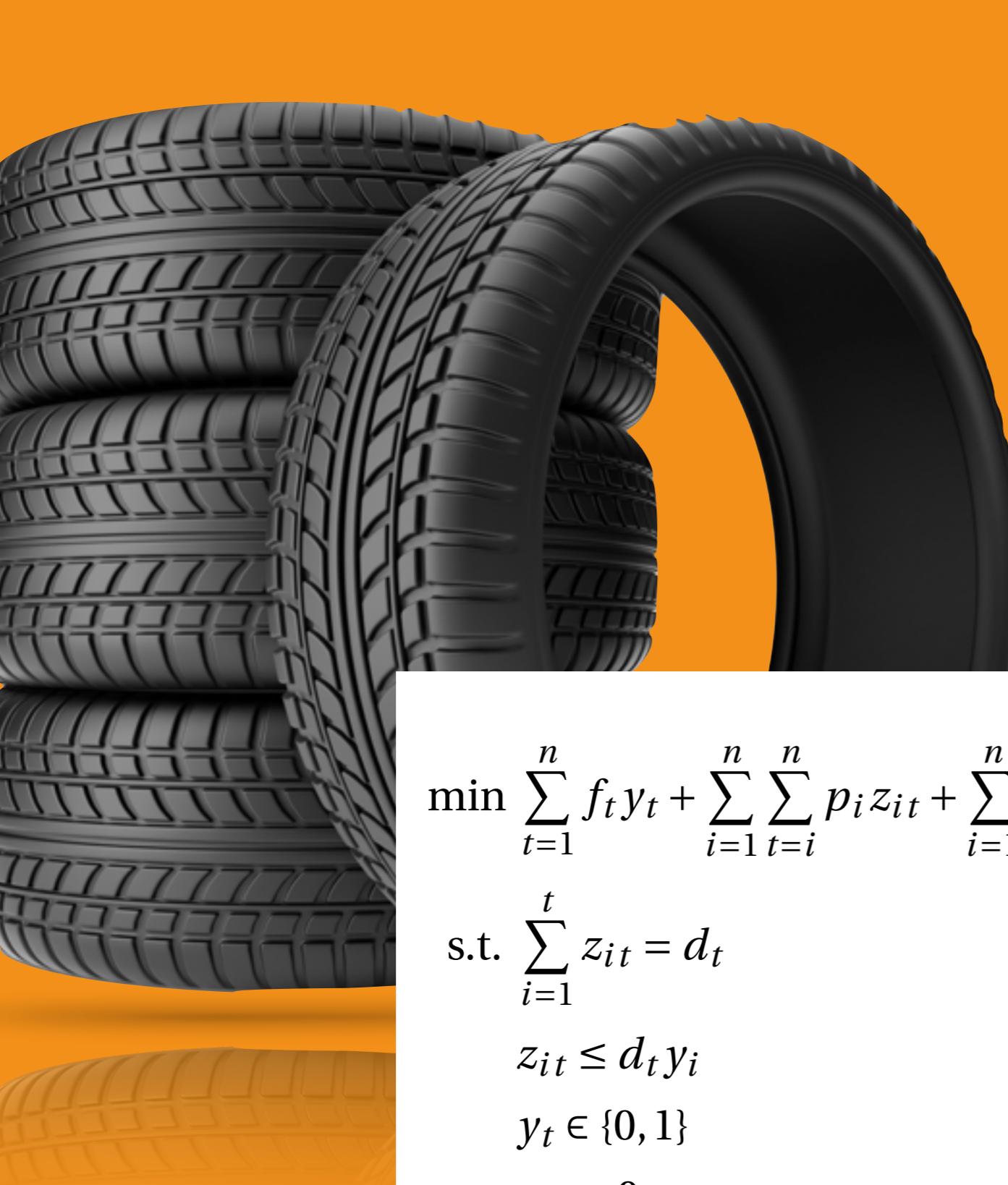
Output a minimum (production and storage) cost production plan to satisfy the demand



Uncapacitated Lot Sizing Problem

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

production
st p_t , unit
or each
n and
an to



Uncapacitated Lot Sizing Problem

$$\min \sum_{t=1}^n f_t y_t + \sum_{i=1}^n \sum_{t=i}^n p_i z_{it} + \sum_{i=1}^n \sum_{t=i+1}^n \sum_{j=i}^{t-1} h_j z_{it}$$

$$\text{s.t. } \sum_{i=1}^t z_{it} = d_t \quad t = 1..n$$

$$z_{it} \leq d_t y_i \quad i = 1..n; t = i..n$$

$$y_t \in \{0, 1\} \quad t = 1..n$$

$$z_{it} \geq 0 \quad i = 1..n; t = i..n$$

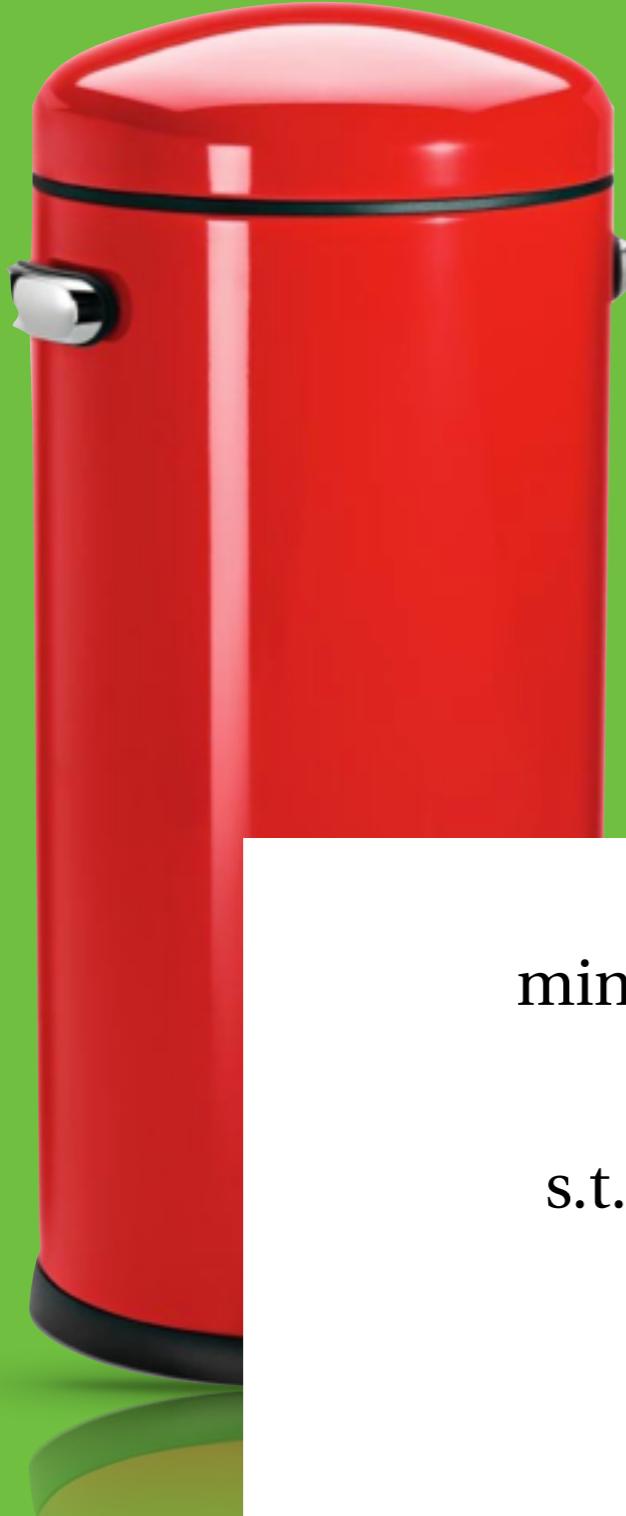
LP=ILP

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



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 minimum number of containers



Bin Packing Problem

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

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

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

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

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

items,
containers,
item j
all items in
containers



Bin Packing Problem

$$\min \sum_{s \in \mathcal{S}} x_s$$

$$\text{s.t. } \sum_{s \in \mathcal{S}} a_{js} x_s = 1 \quad j = 1..n$$

$$x_s \in \{0, 1\}$$

$$s \in \mathcal{S}$$

items,
containers,
item j
all items in
containers

delayed column generation

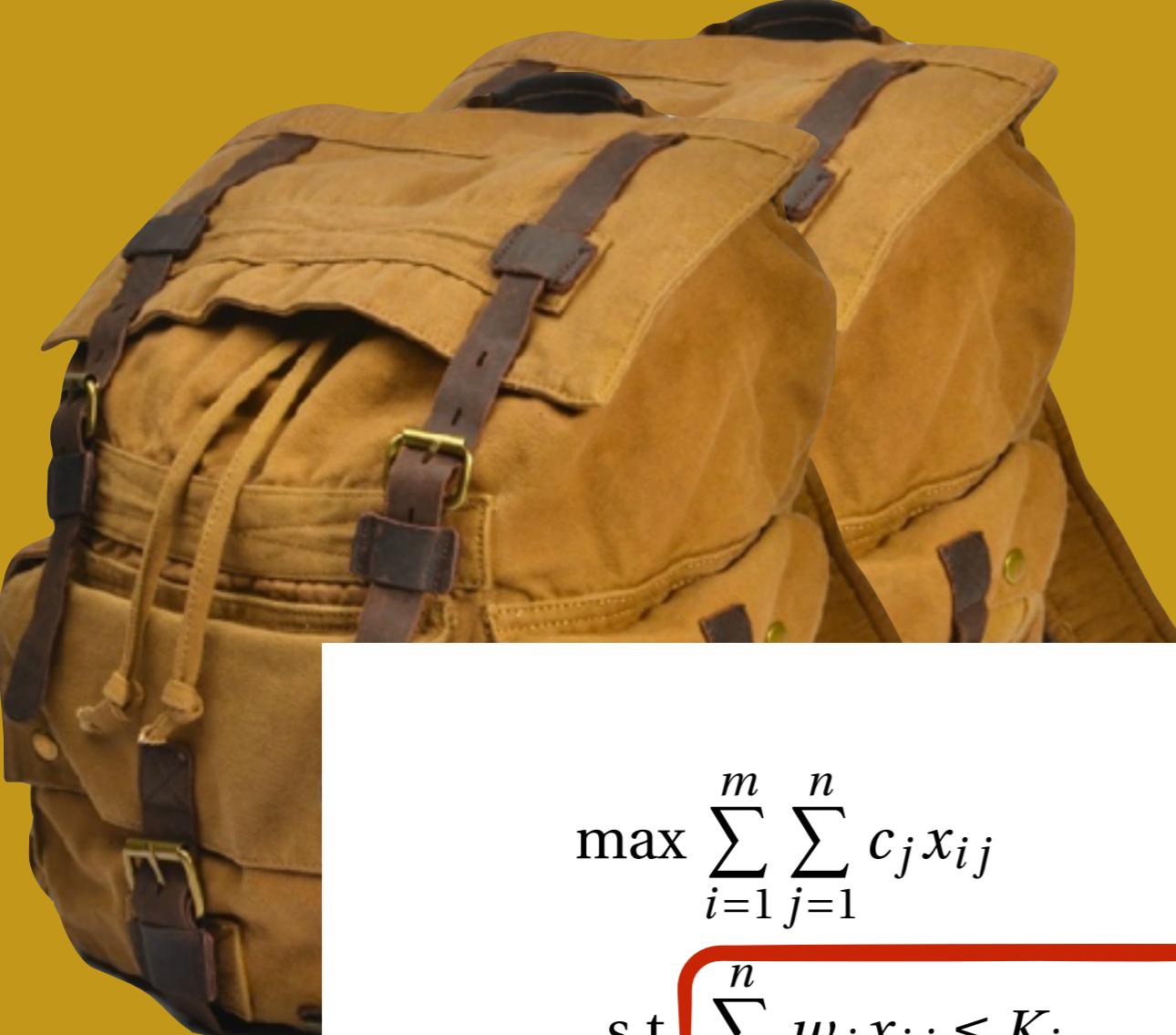
↙ all the possible arrangements of items in a bin



Multi 0-1 Knapsack Problem

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.



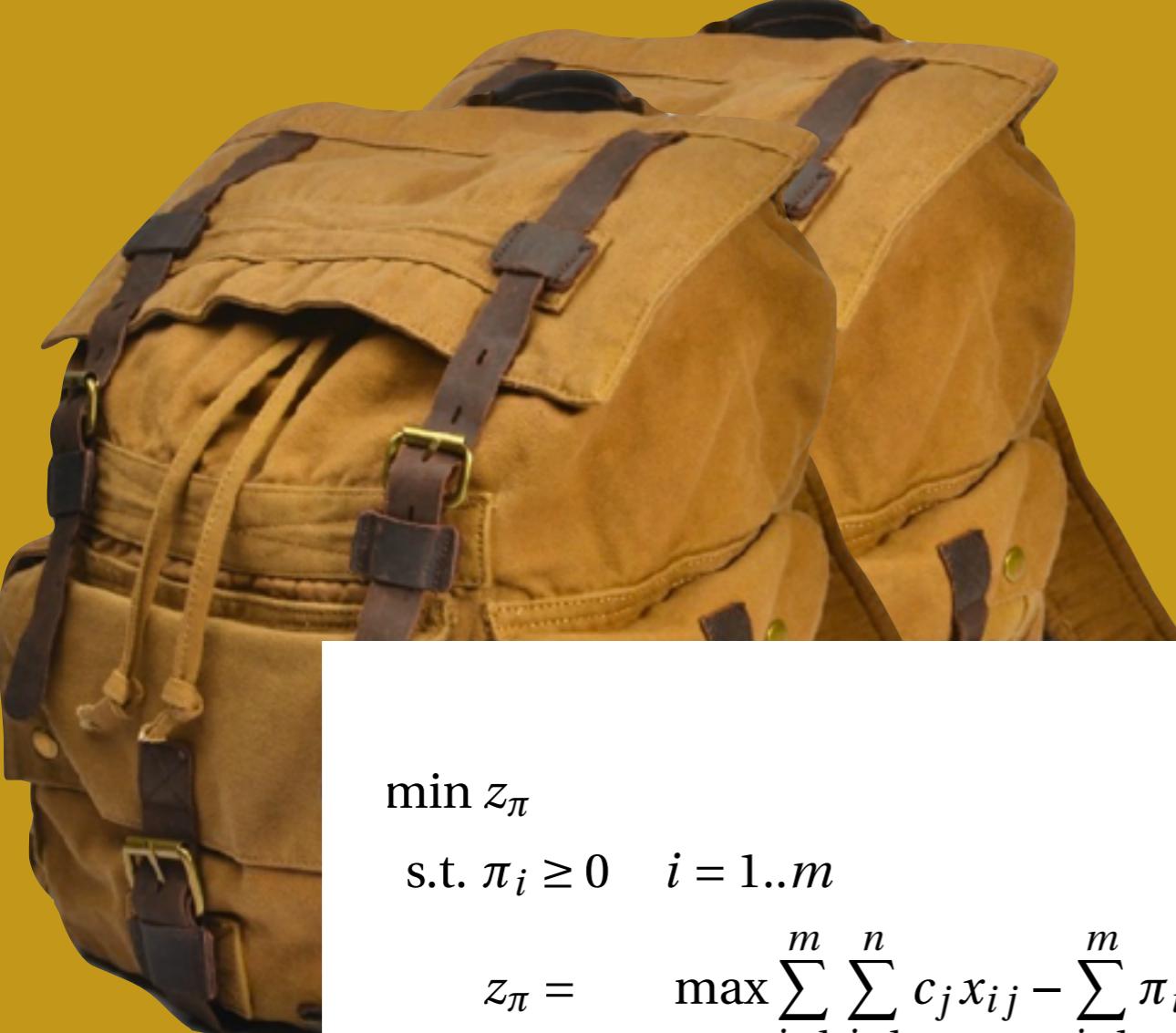
Multi 0-1 Knapsack Problem

$$\max \sum_{i=1}^m \sum_{j=1}^n c_j x_{ij}$$

$$\text{s.t. } \sum_{j=1}^n w_j x_{ij} \leq K_i \quad i = 1..m$$

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

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



Multi 0-1 Knapsack Problem

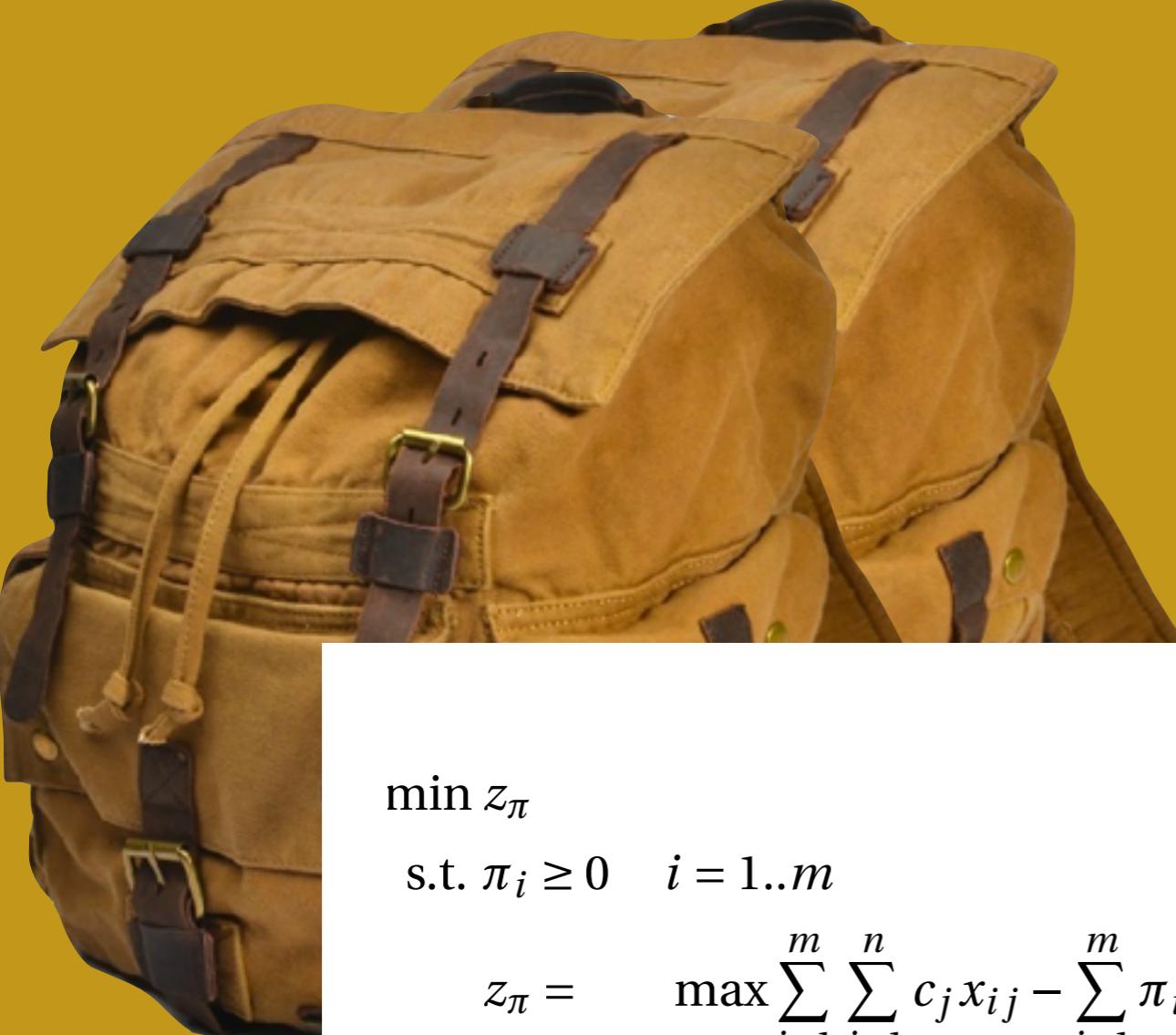
$$\min z_{\pi}$$

$$\text{s.t. } \pi_i \geq 0 \quad i = 1..m$$

$$z_{\pi} = \max \sum_{i=1}^m \sum_{j=1}^n c_j x_{ij} - \sum_{i=1}^m \pi_i (\sum_{j=1}^n w_j x_{ij} - K_i)$$

$$\text{s.t. } \sum_{i=1}^m x_{ij} \leq 1 \quad j = 1..n$$

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



Multi 0-1 Knapsack Problem

$$\min z_{\pi}$$

$$\text{s.t. } \pi_i \geq 0 \quad i = 1..m$$

$$z_{\pi} = \max \sum_{i=1}^m \sum_{j=1}^n c_j x_{ij} - \sum_{i=1}^m \pi_i (\sum_{j=1}^n w_j x_{ij} - K_i)$$

$$\text{s.t. } \sum_{i=1}^m x_{ij} \leq 1 \quad j = 1..n$$

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

lagrangian relaxation

if all else fails

non-linear programming

if all else fails

constraint programming

non-linear programming

if all else fails

constraint programming

non-linear programming

if all else fails

SAT

constraint programming

non-linear programming

if all else fails

SAT

metaheuristics

- Achterberg** T., Berthold T., Hendel G. (2012) Rounding and propagation **heuristics** for MIP. In OR Proceedings 2011, 71-76.
- 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.

Wolsey L. (1998) Integer Programming. Wiley