UCA - Master Hydroprotech - 2024

Mathematical optimization: introduction and application to water management

Sophie Demassey
https://sofdem.github.io/

water management ?

- collect, treat, distribute, value water as a commodity

ex: design and operate wastewater networks under normal or extreme conditions

- mobilize water in processes as a resource with limited availability

ex: withdraw water for cooling or cleaning while preserving water source quality

- preserve a biotope, or deal with a natural hazard involving water

ex: adapt landscape to flood resilience

decision & management

accuracy

Operational
effective process

Tactical
system design

Strategic
long-term planning

time

operational research: models and algorithms

prospective: data and scenarios

this class

- overview of prescriptive tools in decision support
- focus on mathematical optimization and discrete decision
- model and solve mathematical programs
- selected applications in water management

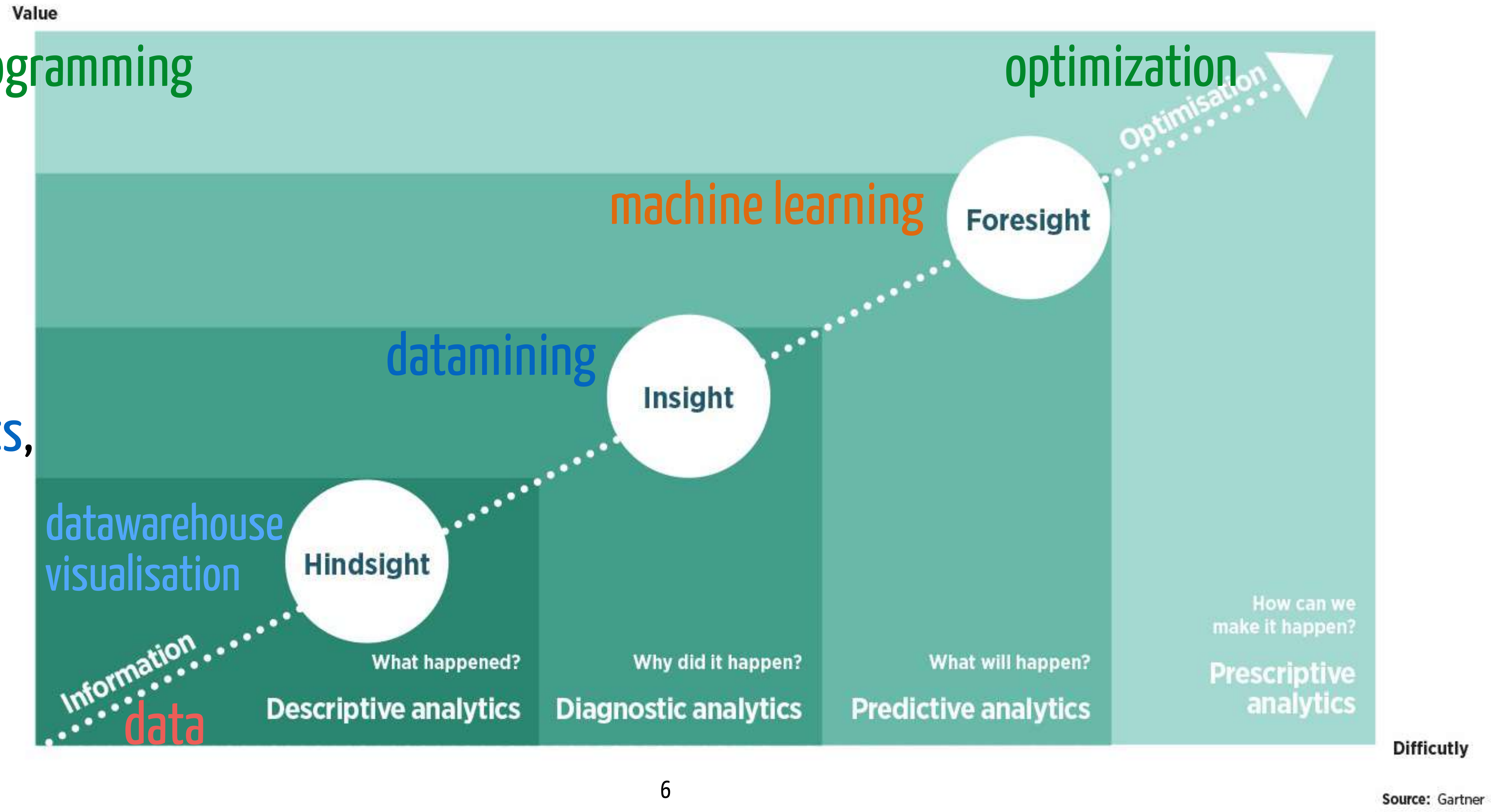
prescriptive tools in decision support

decision support

from WWII:
mathematical programming

in the 2010s:
AI, deep learning

in the 2000s:
business analytics,
big data



decide = optimize

Decision Making

identify possible alternatives, attach a quantitative score,
search an alternative with the highest score

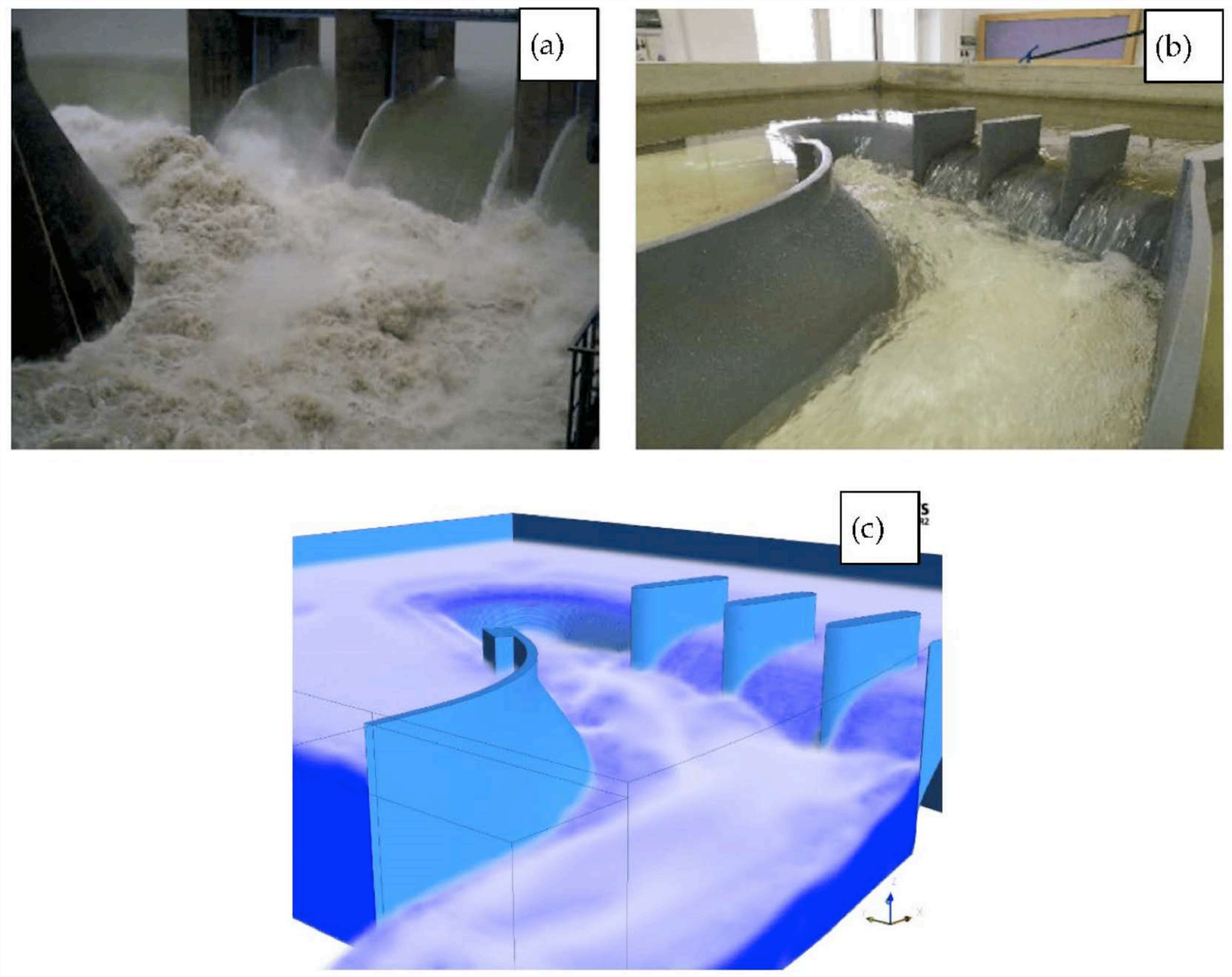
Optimization

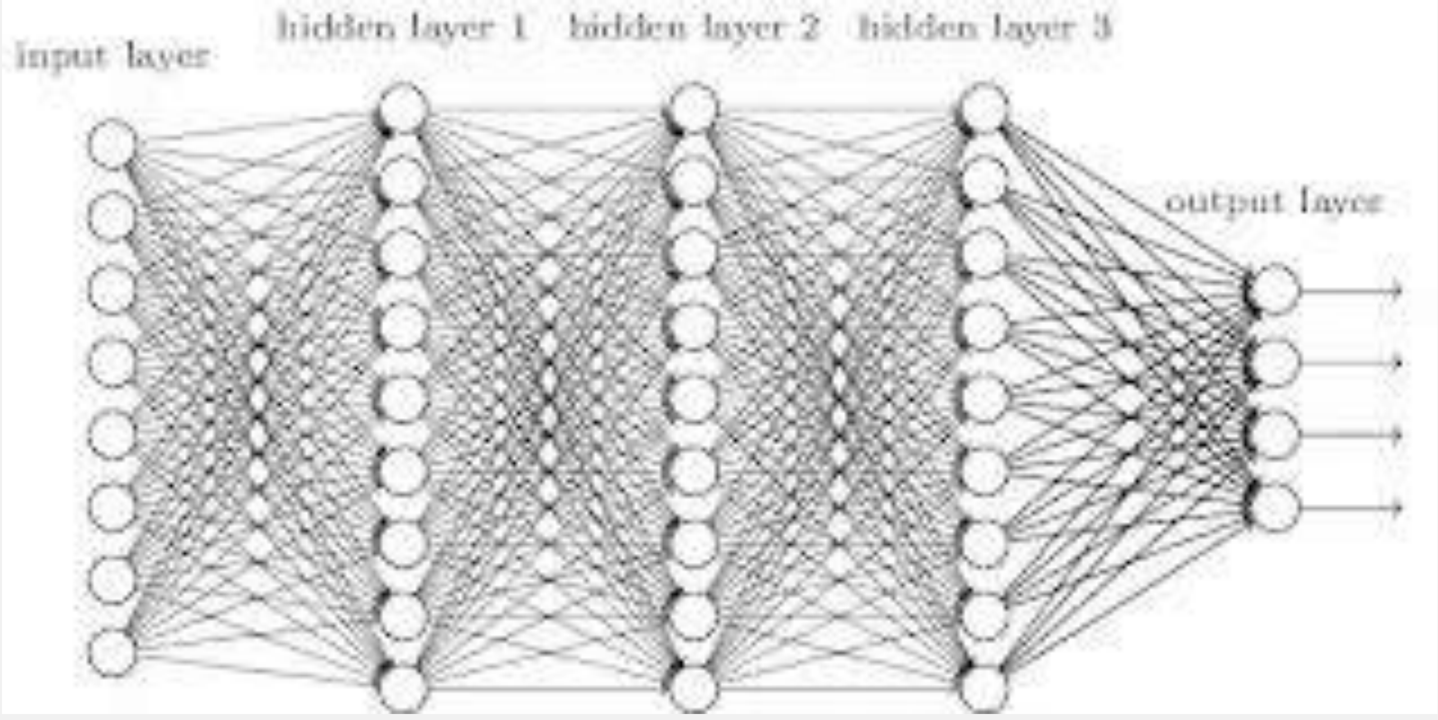
model : describe the feasible solutions
objective: a mapping from solutions to scores
optimize : compute a feasible solution of maximum score

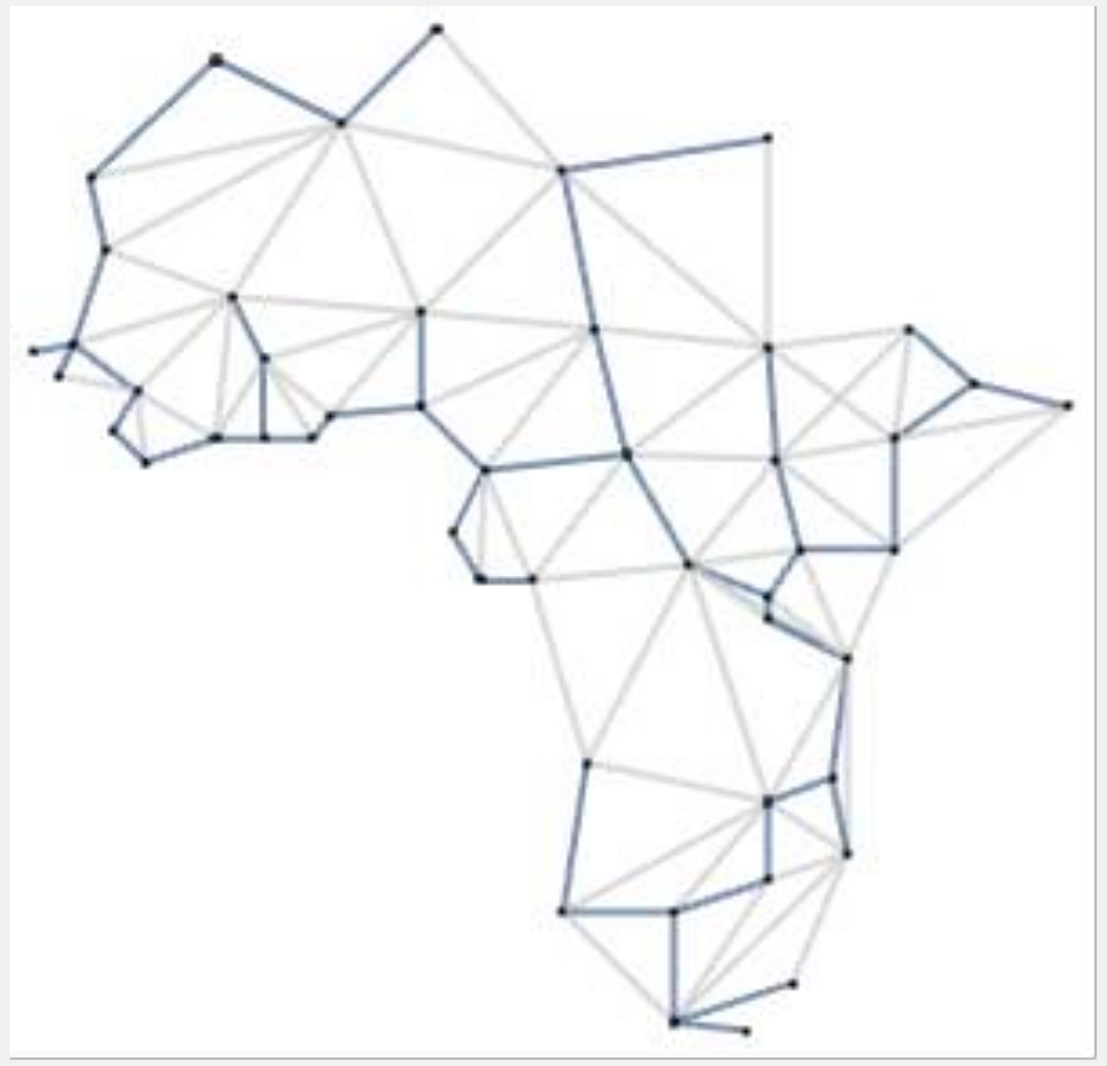
models

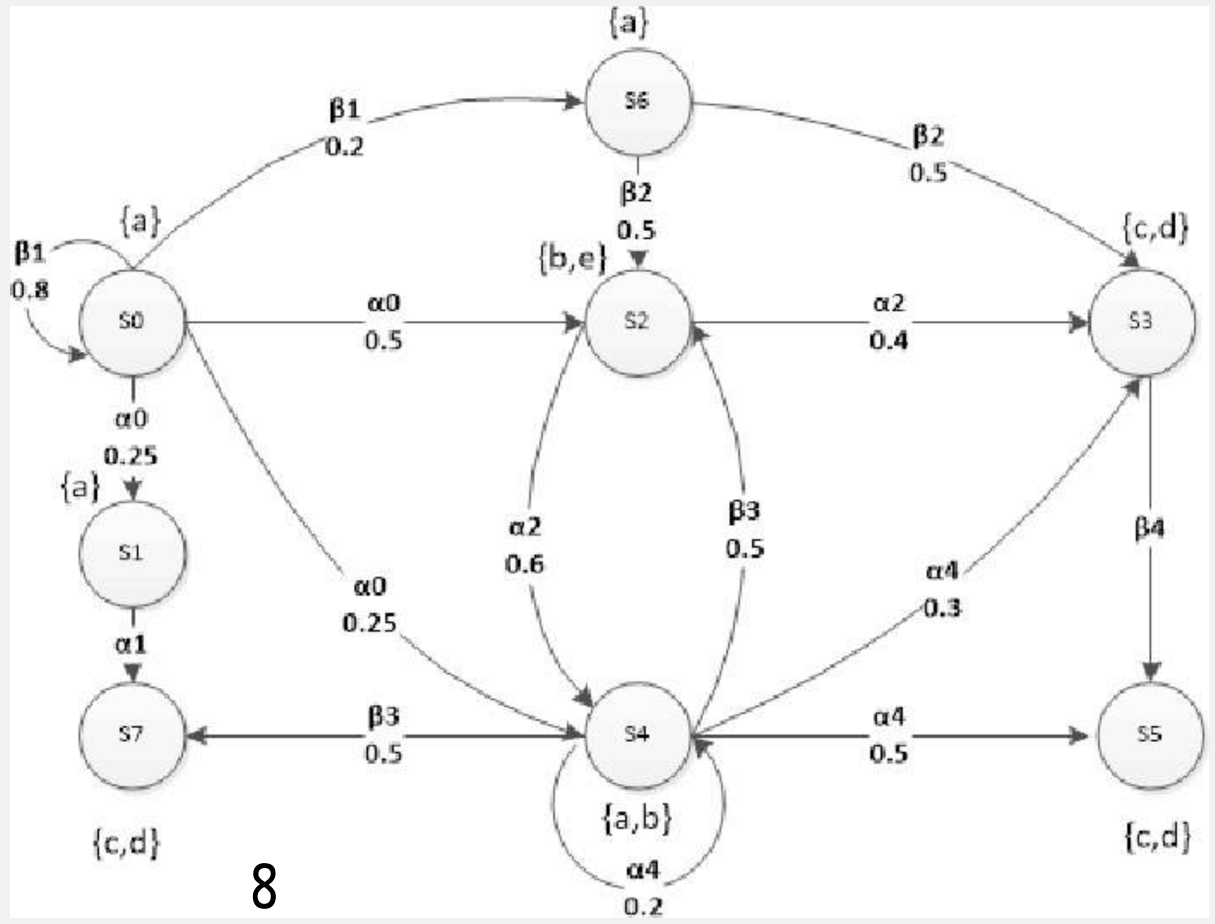
physical and virtual/numerical models

simulators: *imperative* "how"









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

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

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

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

$$\frac{\Delta; \Gamma, \alpha}{\Delta; \Gamma, \alpha_1 \mid \Delta; \Gamma, \alpha_2}\ (\alpha\text{-}\Gamma) \qquad \frac{\Delta; \Gamma, \beta}{\Delta; \Gamma, \top \mid \Delta; \Gamma, \beta_1, \beta_2}\ (\beta\text{-}\Gamma)$$

$$\frac{\Delta; \Gamma, \neg\neg A}{\Delta; \Gamma, A}\ (\neg\neg\text{-}\Gamma) \qquad \frac{\Delta; \Gamma, A, \neg A}{\Delta; \Gamma, \top}\ (\top\text{-}\Gamma)$$

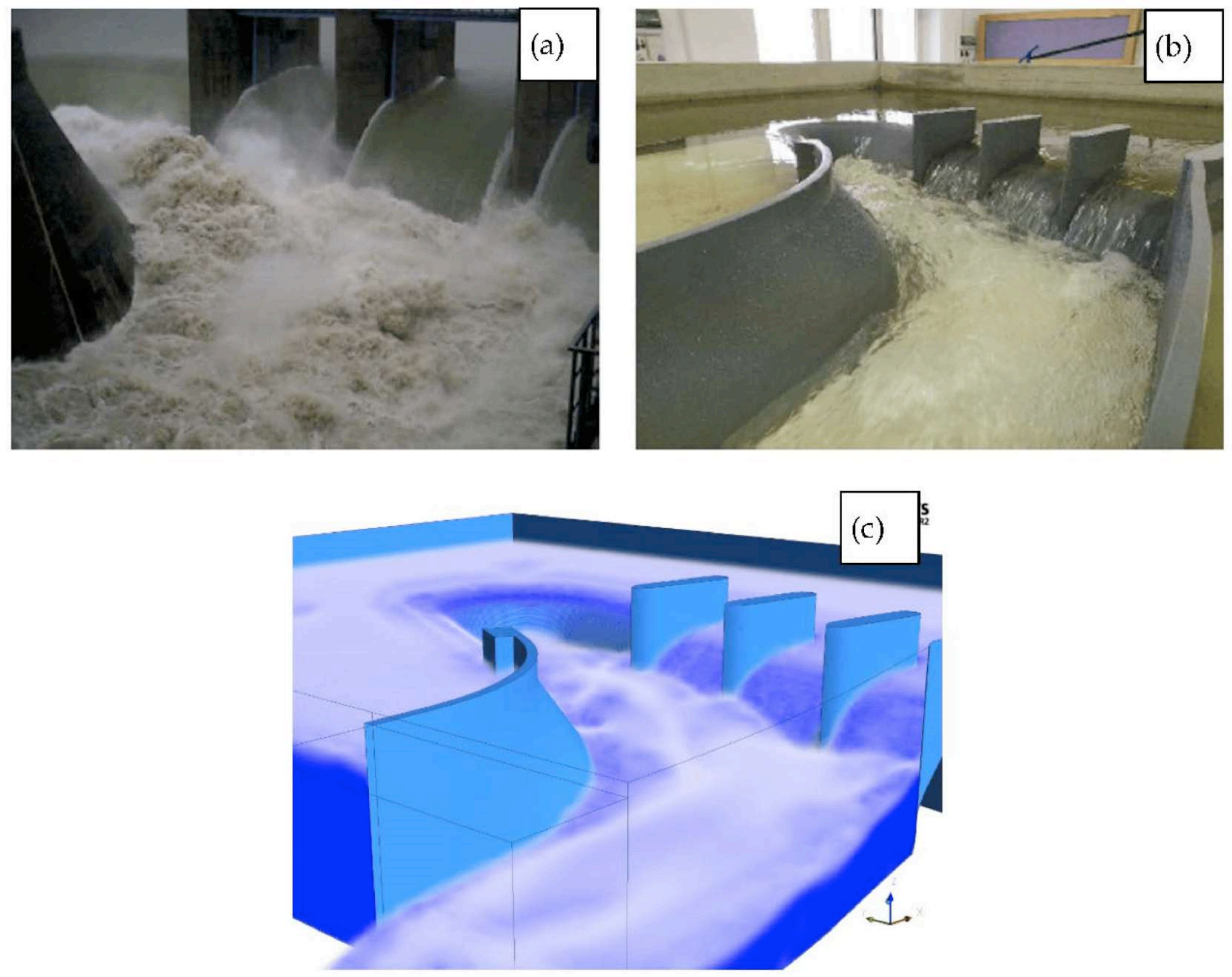
$$\frac{\Delta, \alpha; \Gamma}{\Delta, \alpha_1, \alpha_2; \Gamma}\ (\alpha\text{-}\Delta) \qquad \frac{\Delta, \beta; \Gamma}{\Delta, \beta_1; \Gamma \mid \Delta, \beta_2; \Gamma}\ (\beta\text{-}\Delta)$$

$$\frac{\Delta, A, \neg A; \Gamma}{\Delta, \blacksquare; \Gamma}\ (\blacksquare) \qquad \frac{\Delta, A; \Gamma, A}{\Delta, A; \Gamma, \top}\ (\top\text{-intr})$$

conceptual models

formulation: *declarative* "what"

models



created by experts

maybe reinforced

automatically from data

(machine learning)

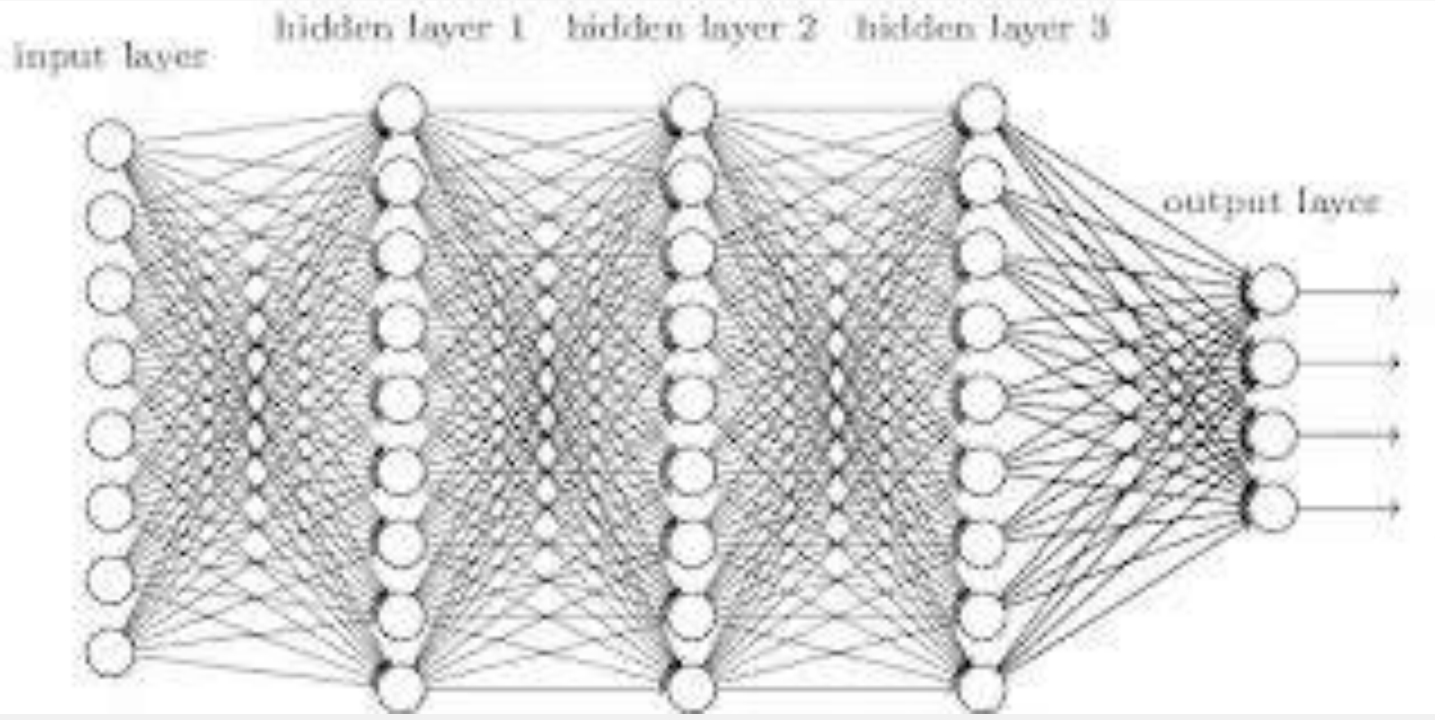
$$\min \sum_{k=1}^{K} \sum_{j=1}^{n} d_{jk}$$

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

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

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

35	-50	73787	28083	1439	2240	2746	3687	5293	2740
48	101	758353	383745	201999	62107	36293	130536	57243	25354
57	-5	2E+06	129350	61236	17084	11488	62462	49960	33932
53	-8	1E+06	354328	37102	88881	45307	99603	44790	29749
66	73	2E+06	176766	59352	26157	15054	33669	33782	31750
69	130	635191	122446	90107	65072	36230	53019	62938	59307
61	-2	161098	12119	1963	809	1277	3186	3266	2518
69	17	492796	120998	63697	68242	10769	88403	73756	22676
69	-59	82048	116131	47317	26197	41642	28866	32551	41810
38	-14	757165	186196	3242	3841	18854	43021	46799	11928
48	72	667513	141854	75050	16234	45926	34496	74875	31839
34	121	165360	42119	3158	6256	7270	19462	10984	8148
30	-52	737665	84275	2235	38748	21705	28343	80927	24735
48	69	577024	179555	25937	21604	43790	44432	69203	25586
43	44	234964	80944	7991	32568	11828	63682	41013	18147
40	144	671467	133227	7142	14300	23373	65591	47860	16501
40	25	33290	14729	10980	10407	14316	37112	60378	36306
44	-138	1E+06	268122	67285	36996	10083	80826	90350	26345
53	25	756442	97387	59785	32241	16122	53157	44602	14331
66	124	87109	7200	4987	2817	2130	11413	9422	4922
53	43	802937	83858	3229	7895	27104	17791	18901	12191
51	35	157687	283941	27259	47930	15132	38238	31599	10121
50	119	613031	758257	246386	132015	26636	83848	94542	56127



accuracy & approximation

Decision Making

Mathematical Optimization

$$\min_{x \in \mathbb{R}^n} f(x) : g_i(x) = 0 \ \forall i = 1,\ldots,m$$

concrete problem $\longrightarrow$ abstract model

$\neq$ solve

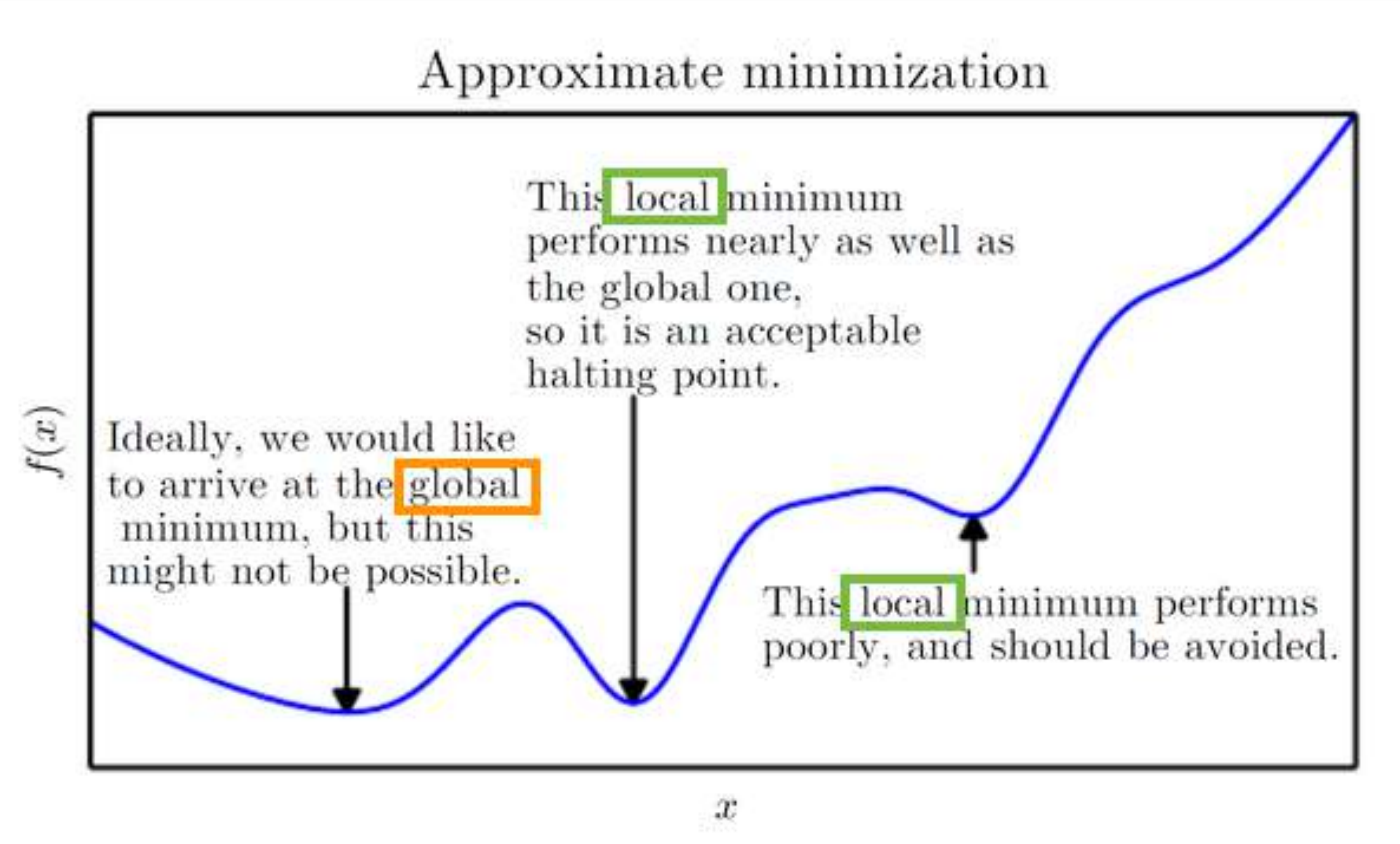
practical decision $\longleftarrow$ (optimal) solution

solving, optimizing: find a solution to a model not to a problem

‘solve’

solve a model not a problem

- imprecise (truncated) and uncertain (forecast) data
- approximate dynamics and simplified (soften) constraints
- conceptual objective



solve a model ?

- solution may be infeasible or feasible within a tolerance gap
- solution may be sub-optimal or optimal within a tolerance gap
- solution may not be provably optimal, neither globally nor locally
- theoretic ≠ practical optimality guarantees: high complexity, slow convergence, limited time

optimize

Model describes the system behavior

Simulation evaluates behavior and score for one given input decision

Optimization search the input decision leading to the highest score

different problems, different needs ⟶ many algorithms

operational research

different classes of algorithms:

- local/global, exact/heuristic
- deterministic/stochastic
- generic/specific

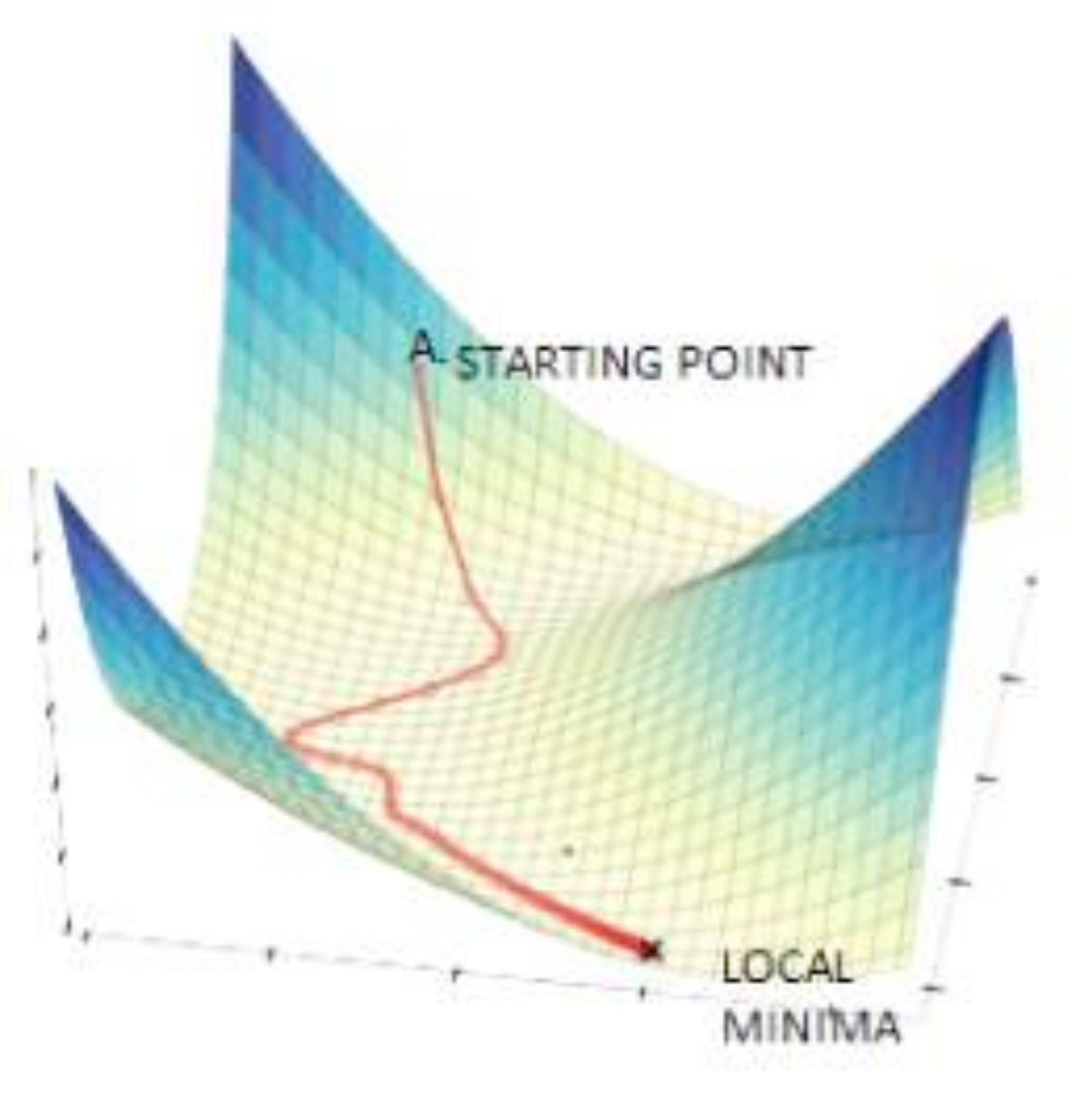
2 main principles:

- generate & test
- divide & conquer

generate & test: principle

black-box or numerical methods:

1. select a candidate decision
2. simulate/evaluate feasibility and score
3. stop or iterate

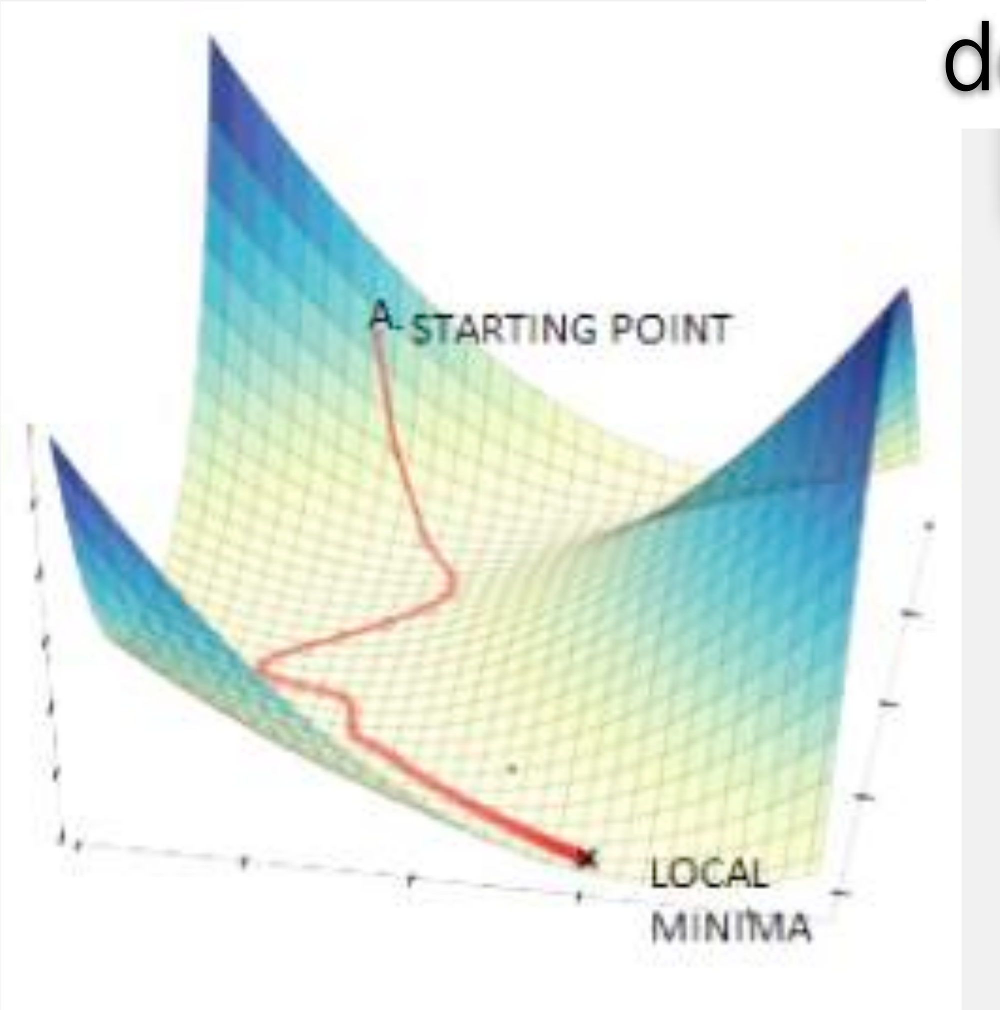


which candidates to evaluate ?

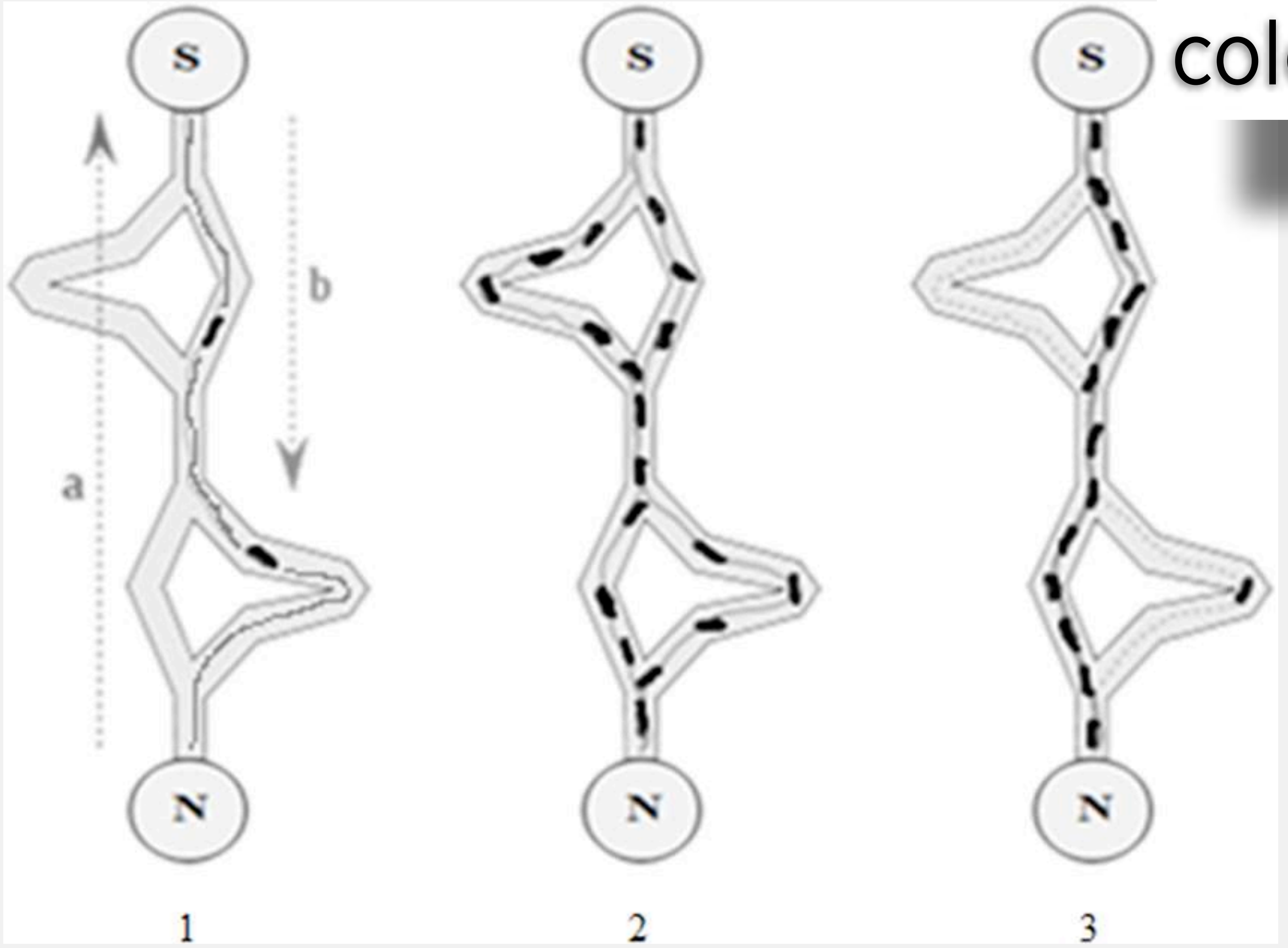
search: partial, exhaustive, exhaustive but implicit
choice: random/probabilistic or directed by distance, score, highest-order information (e.g: derivative of the objective function)

generate & test: algorithms

gradient descent



ant colony



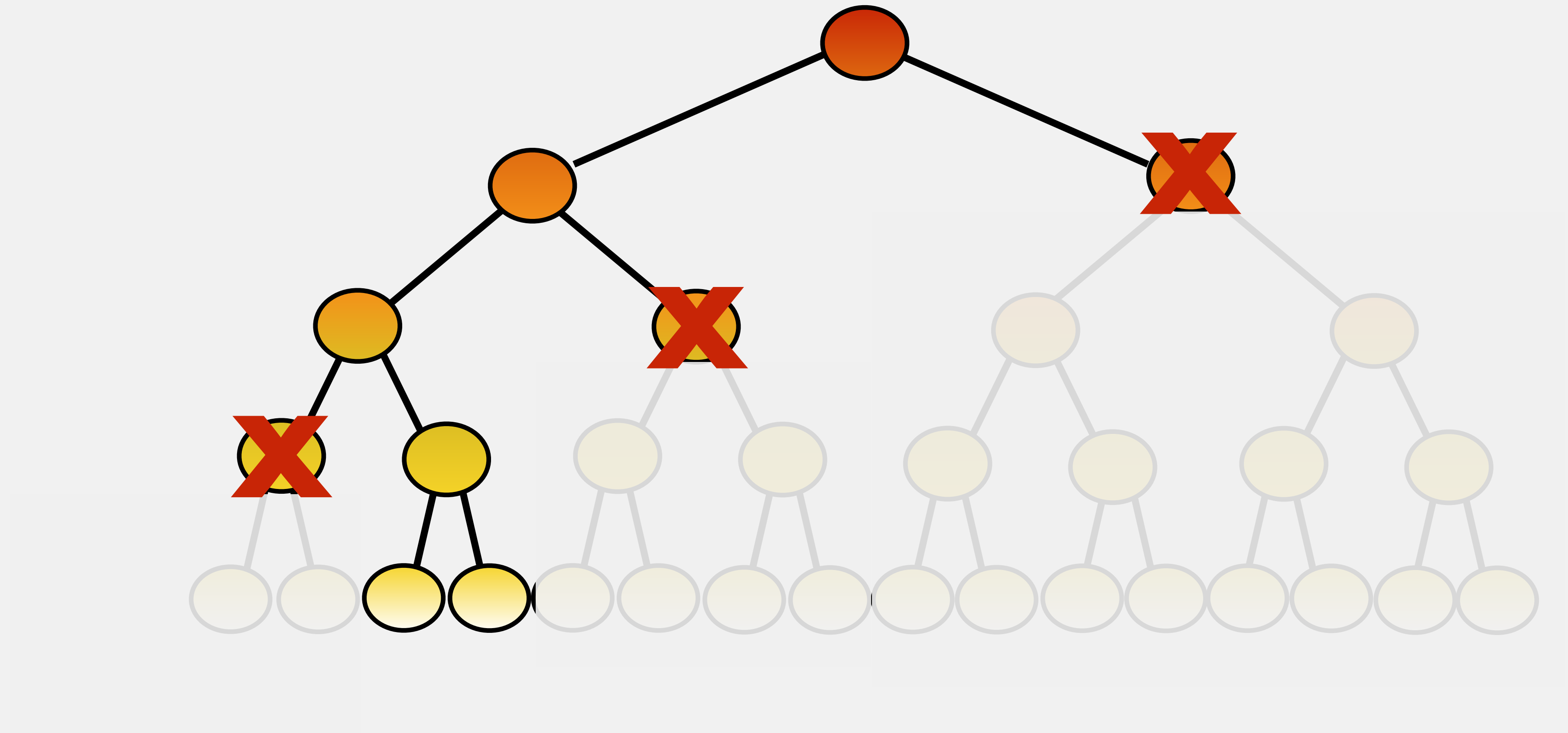
examples:

- local search: move to a neighbour candidate, the best one or in an improving direction
- may converge to a global optimum, e.g.: gradient descent in convex optimization, simplex algorithm in linear programming
- metaheuristics (evolutionary, swarm): combine candidates, use collective memory

divide & conquer: principle

divide-and-conquer:

1. separate the search space (and refine the model)
2. estimate feasibility and best score in a simpler **relaxed model**
3. **backtrack** if not better, record if full solution, or iterate



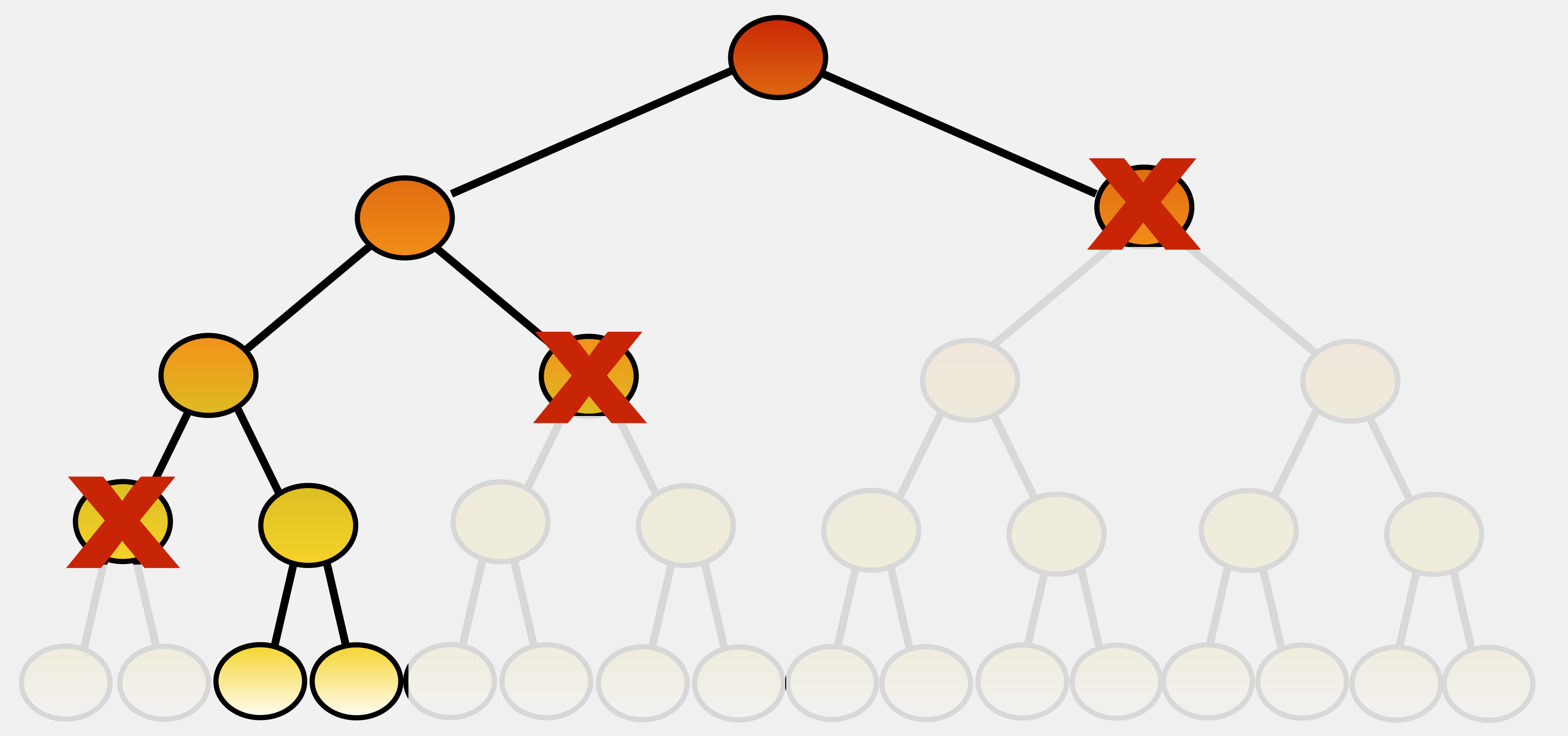
relaxation/bounding (the maximal score):

- certificate of optimality: zero-gap between best upper bound (UB) estimated from the relaxations and best lower bound (LB) computed on full solutions
- rely on tight but simple relaxations

divide & conquer: algorithms

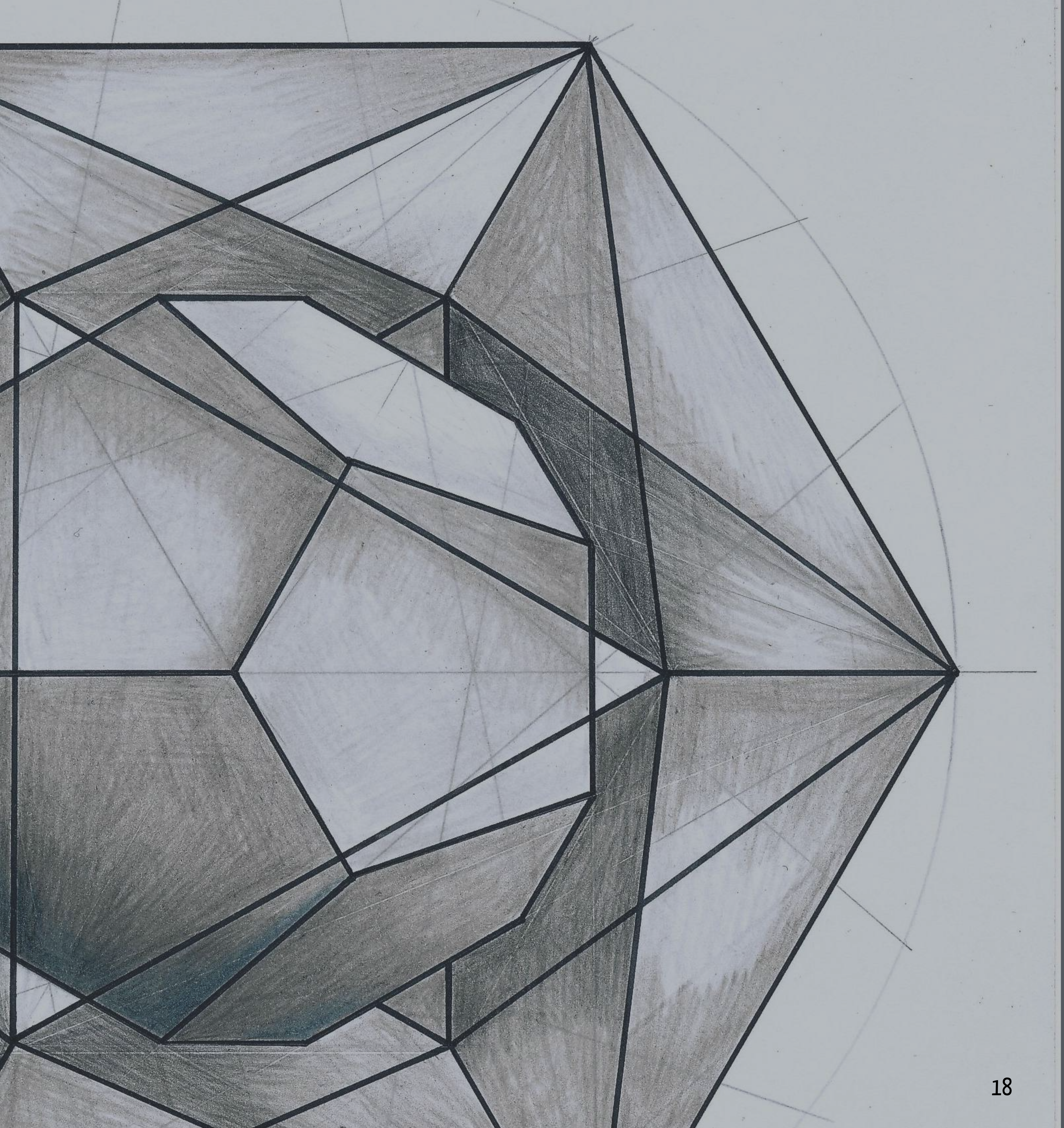
examples:

- greedy heuristic: no backtrack
- graph algorithms, dynamic programming
- backtracking methods in logic/constraint programming
- branch-and-bound in combinatorial optimization



one person's solution
should not become
another person's problem

(to keep in mind when modeling)



mathematical optimization

mathematical program

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

$f : \mathbb{R}^n \mapsto \mathbb{R}$ objective

$g : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^m$ constraints

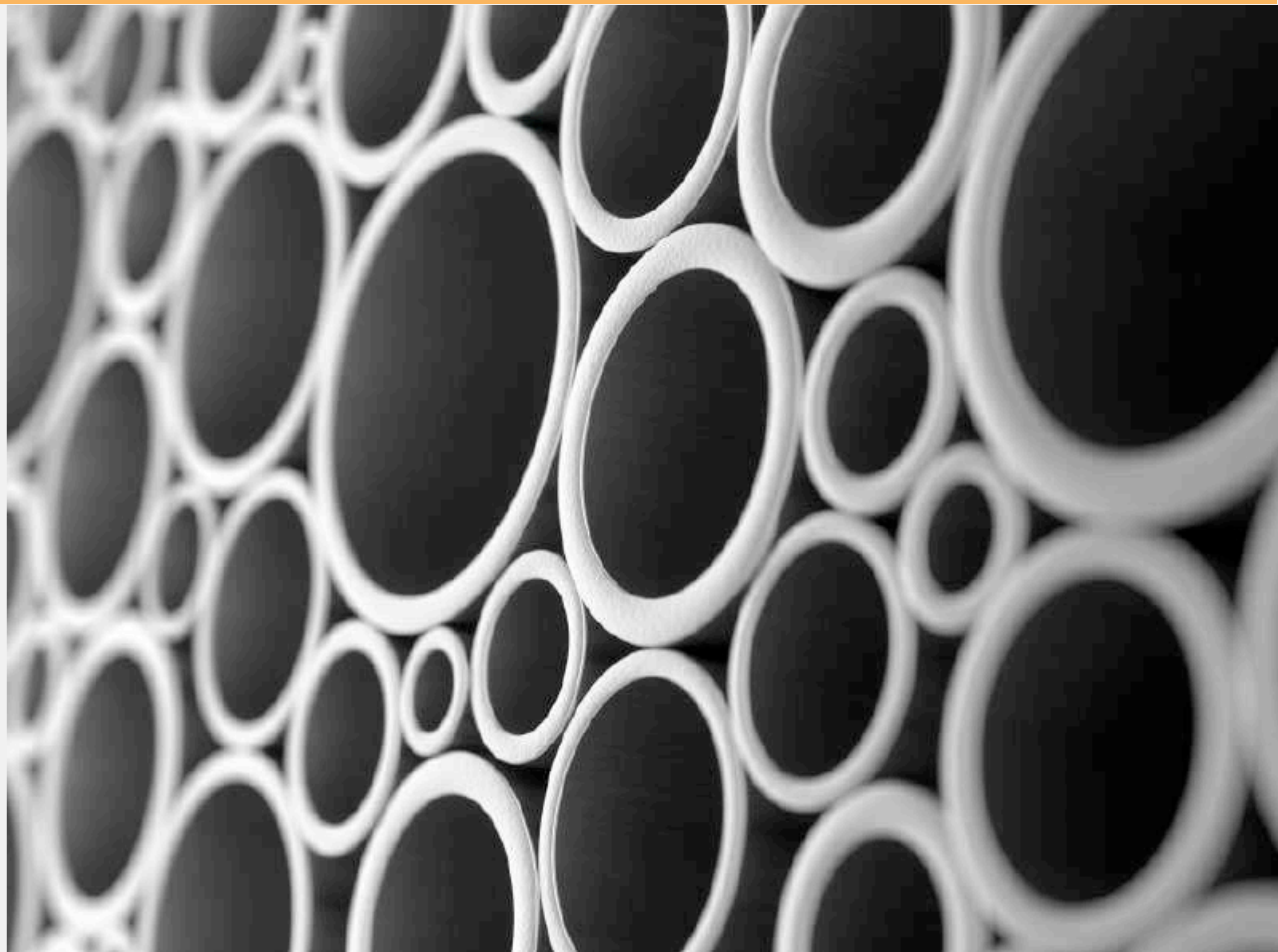
$x \in \mathbb{R}^n$ variables / solution

my first MP

Pipe Sizing

Choose the diameter of two water distribution pipes
to maximize the total rate of flow, within a budget of 180 euros, given that:

- 1st pipe: maximum diameter = 40cm, cost=3 euros/cm, avg rate=3u/cm
- 2nd pipe: maximum diameter = 60cm, cost=2 euros/cm, avg rate=5u/cm



variables: diameters for the pipes (in cm)
constraints: bounds and budget
objective: maximize flow rate

$$max\ 3x_1 + 5x_2 :$$
$$0 \le x_1 \le 40,\ 0 \le x_2 \le 60$$
$$3x_1 + 2x_2 \le 180$$

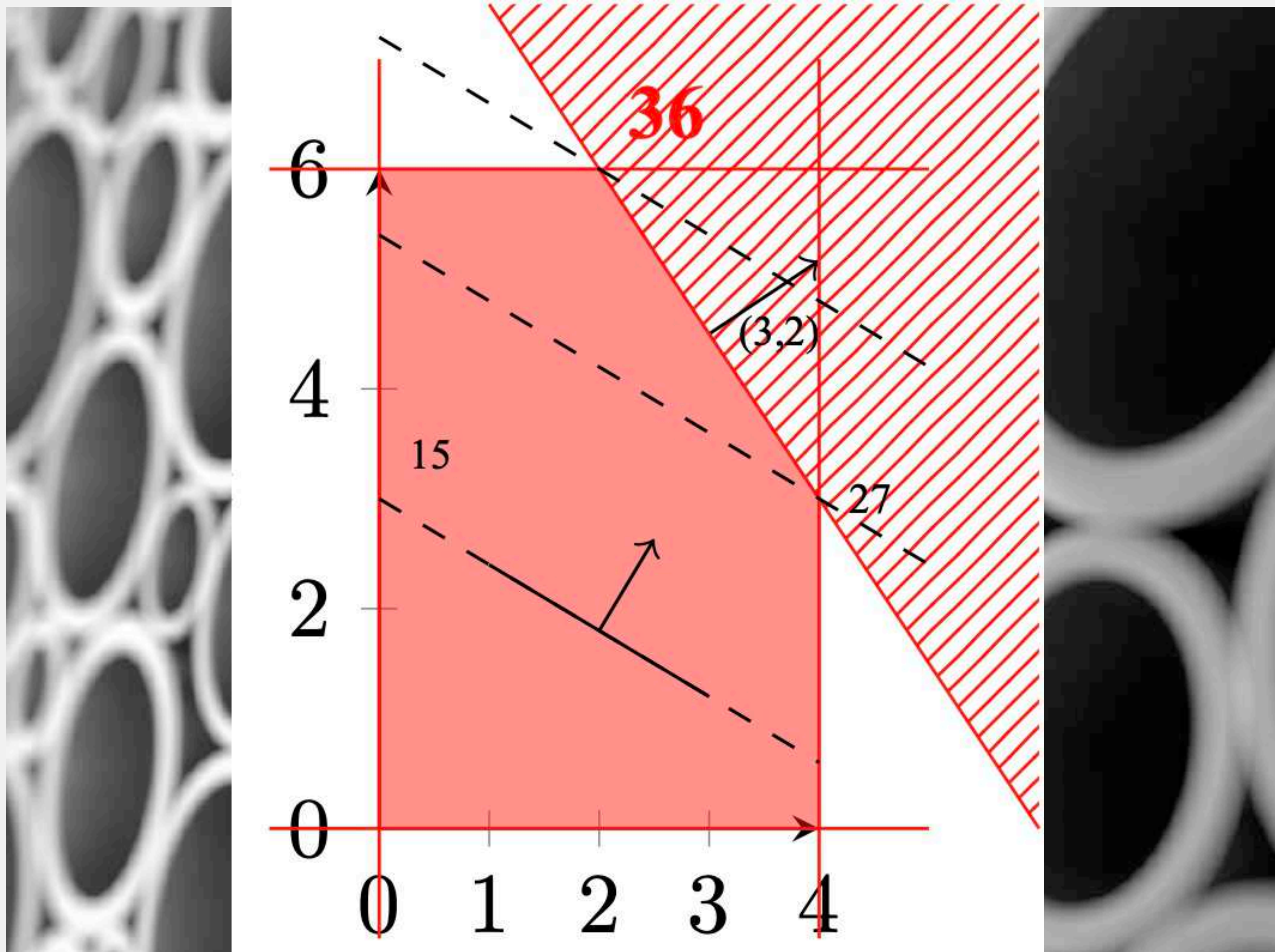
my first MP

$$\max\ 3x_1 + 5x_2 :$$
$$0 \le x_1 \le 40,\ 0 \le x_2 \le 60$$
$$3x_1 + 2x_2 \le 180$$

variables: diameters for the pipes (in cm)
constraints: bounds and budget
objective: max flow rate

linear case: graphical solution

- constraints define half-spaces in $\mathbb{R}^2$
- intersection = poyhedron = feasible solutions
- solutions of cost p: point in line $3x_1 + 5x_2 = p$
- optimal solution: corner on the highest line
- $x = (20,60)$ cost $p = 3 * 20 + 5 * 60 = 360$



mathematical program

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

well-solved classes:

f, g linear $p = 0$	linear programming
f convex, $g \equiv 0, p = 0$	unconstrained optimization
f, g smooth convex $p = 0$	convex programming
f, g linear $p > 1$	mixed integer linear programming

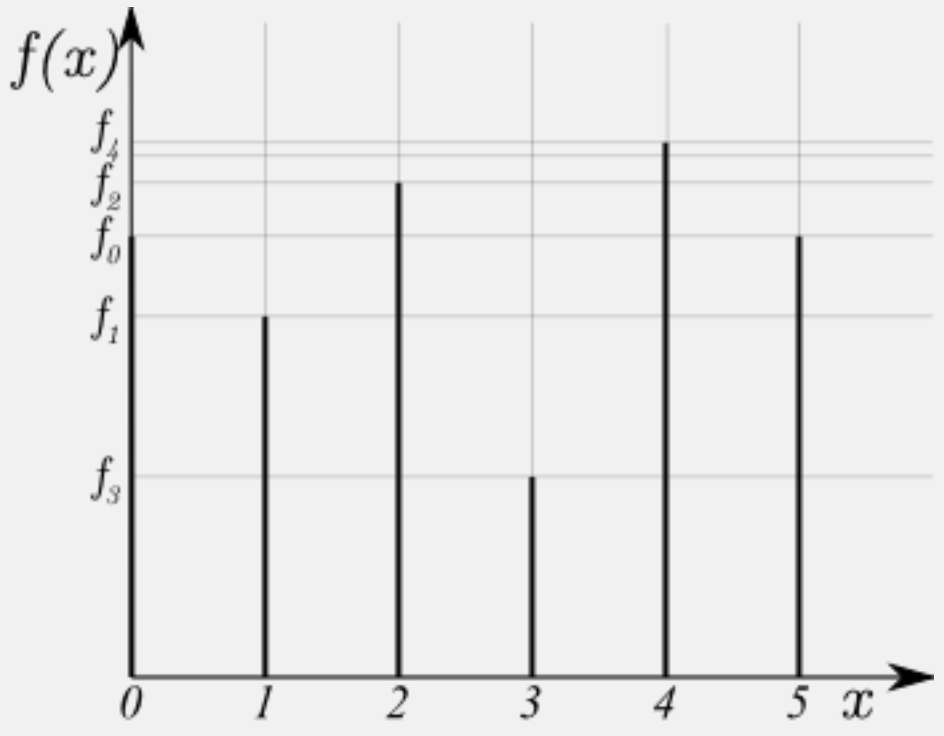
Mixed Integer Linear Program

covers discrete decisions: off/on status $x \in \{0,1\}$, operation level $l \in \{0,1,\ldots,N\}$

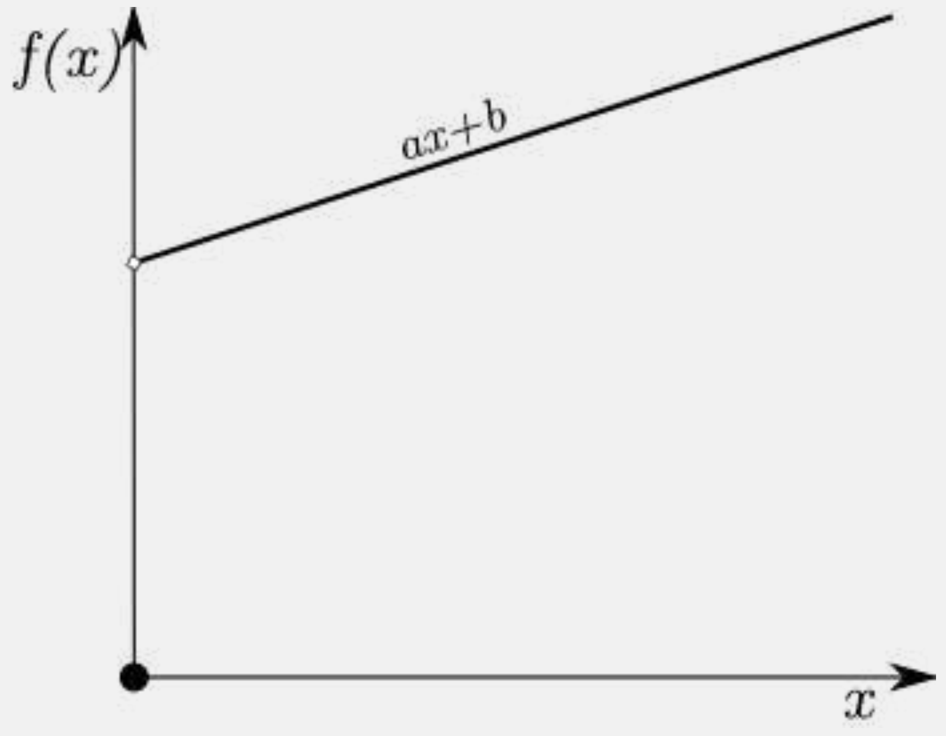
covers logical relations: $l \leq N(1-x)$ level is 0 if status is on: $x = 1 \implies l = 0$

covers nonlinear relations: $l = \sum_{i=0}^{N} i x_i,\ y = \sum_{i=0}^{N} f_i x_i,\ 1 = \sum_{i=0}^{N} x_i,\ x_i \in \{0,1\}\, \forall i \in \{0,\ldots,N\}$

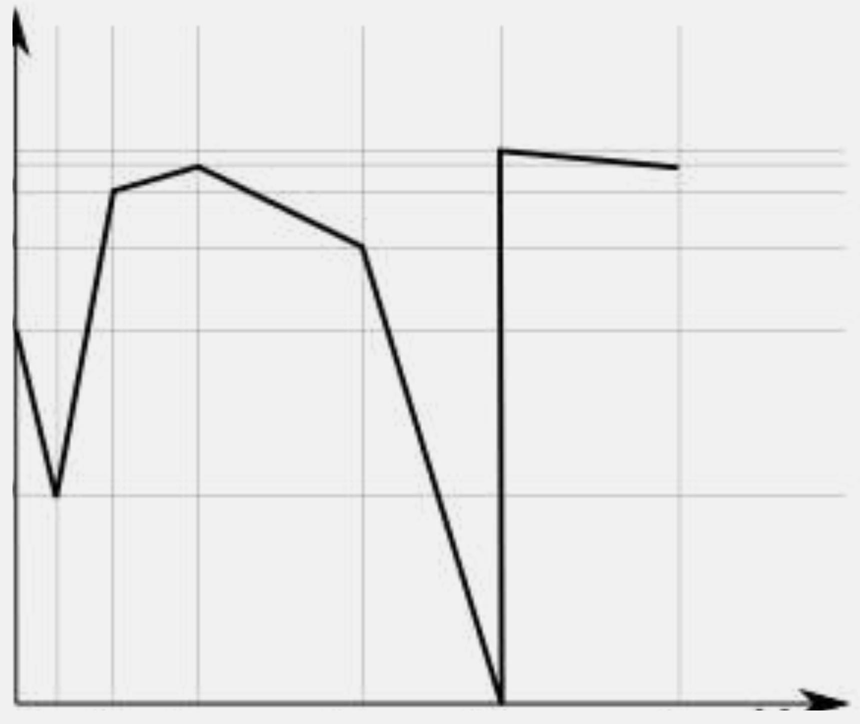
$y = f(l)$ a discrete function



discrete



setup


piecewise linear

my first MILP

Groundwater abstraction (identical pumps)

Choose places to install pumps within a finite set J of candidates and minimize the global cost, given:

- installation cost c_j and average flow rate q_j of a pump at place $j \in J$
- limited total abstraction rate: lower $\underline{Q}$ and upper $\overline{Q}$ limits
- at most 3 pumps installed, no 2 pumps on places a and $b \in J$

variables: $x_j \in \{0,1\}$ number of pumps installed at $j \in J$

$$min \ \sum c_j x_j :$$

$$\underline{Q} \leq \sum q_j x_j \leq \overline{Q}, \sum x_j \leq 3$$

$$x_a + x_b \leq 1, \ x \in \{0,1\}^J$$

variant MILP

Groundwater abstraction (distinct available pumps)

Assign available pumps taken from a finite set K:

- installation cost c_k and flow rate q_k now depends on pump $k \in K$
- limited abstraction $\underline{Q}, \overline{Q}$

$$\min \sum_k \sum_j c_k x_{jk} :$$

$$\underline{Q} \leq \sum_k \sum_j q_k x_{jk} \leq \overline{Q}$$

$$\sum_j x_{jk} \leq 1 \quad \forall k \in K$$

$$\sum_k x_{jk} \leq 1 \quad \forall j \in J$$

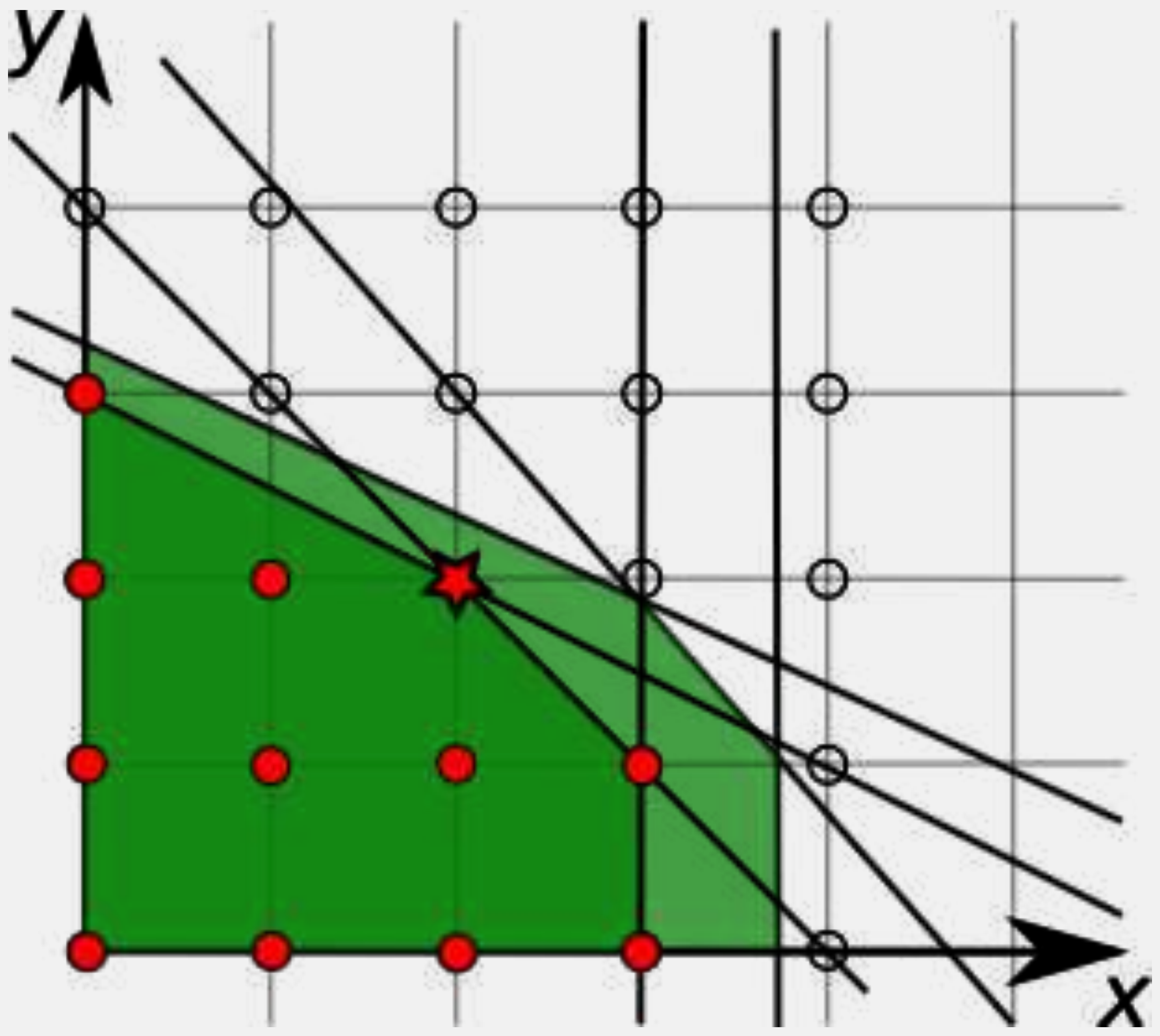
$$x_{jk} \in \{0,1\} \quad \forall j \in J, k \in K$$

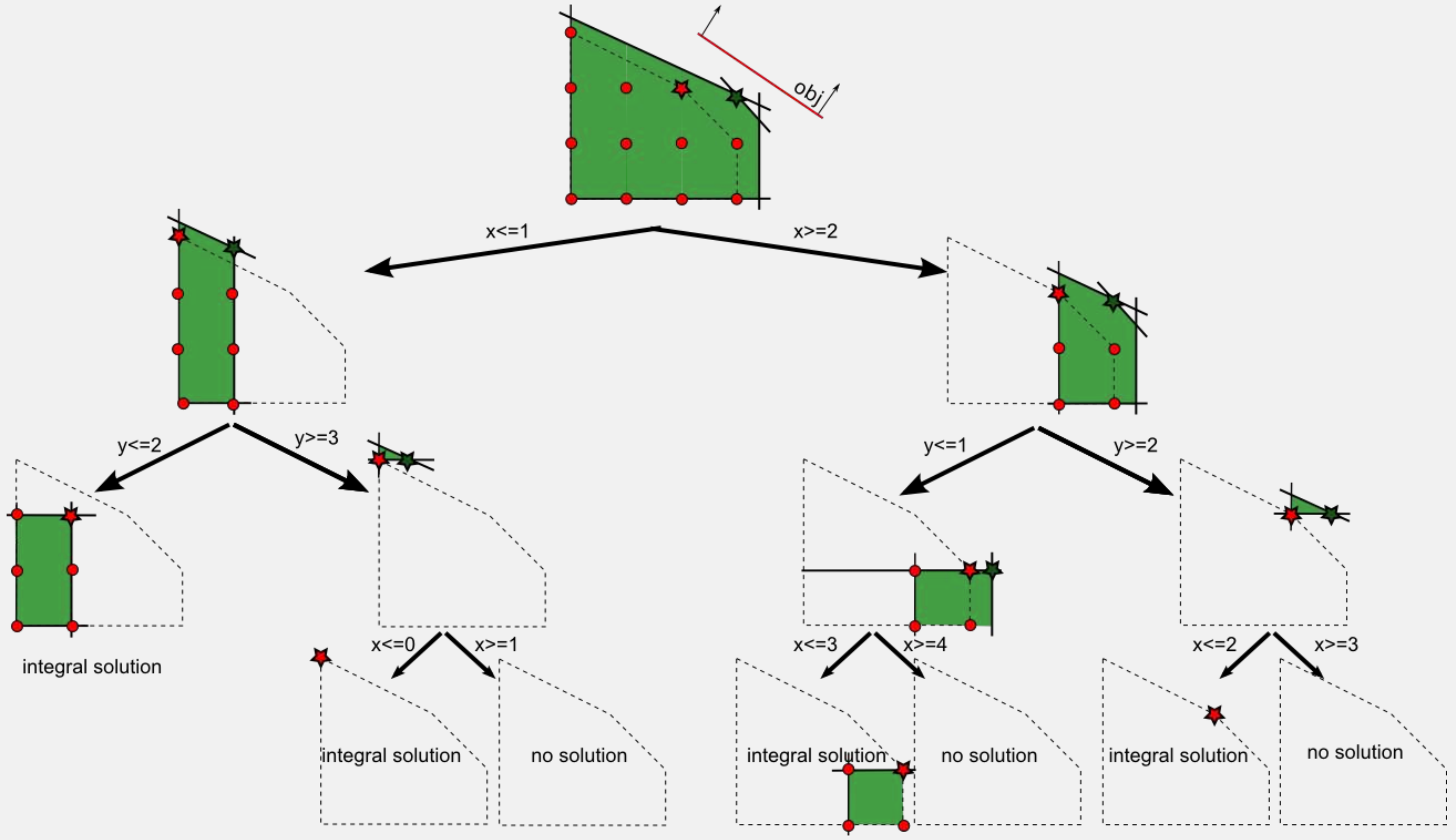
variables: $x_{jk} = 1$ if pump $k \in K$ installed at place $j \in J$

MILP algorithms

cutting-plane algorithm
branch-and-bound
branch-and-cut

- based on the LP relaxation
- evaluate, refine, iterate
- separate (on discrete variables), estimate, backtrack/iterate
- refine then estimate





declarative

equations, not algorithms

performance

sophisticated solvers

versatile

covers logic & nonlinear

optimality

primal-dual bounds

MILP perks

flexible

general-purpose format & solvers

large-scale

decomposition methods

MILP perks*

declarative
equations, not algorithms
*good model ?

performance
sophisticated solvers
*still NP-hard: scale to some extent
(or consider LP)

versatile
covers logic & nonlinear
*approximation
(or consider MINLP)

optimality
primal-dual bounds

large-scale
decomposition methods
*algorithmic challenge

flexible
general-purpose format & solvers
*generic ≠ best