The Voice of Optimization

Bartolomeo Stellato

joint work with Dimitris Bertsimas

MIT MANAGEMENT SLOAN SCHOOL

INFORMS Annual Meeting 2019

Inventory management

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\ \text{subject to} & x_{t+1} = x_t + u_t - d_t \\ & x_0 = x_{\text{init}} \\ & 0 \leq u_t \leq M \end{array}$$

Inventory management

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} \quad & \boxed{x_{t+1}} = x_t + u_t - d_t \\
& x_0 = x_{\text{init}} \\
& 0 \le u_t \le M
\end{aligned}
$$

Inventory

Inventory management

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} \quad & x_{t+1} = x_t + u_t - d_t \\
& x_0 = x_{\text{init}} \\
& 0 \le u_t \le M
\end{aligned}
$$

Inventory

Order

Inventory management

$$\begin{aligned}\text{minimize} \quad & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\ \text{subject to} \quad & x_{t+1} = x_t + u_t - d_t \\ & x_0 = x_{\text{init}} \\ & 0 \leq u_t \leq M\end{aligned}$$

Inventory

Order

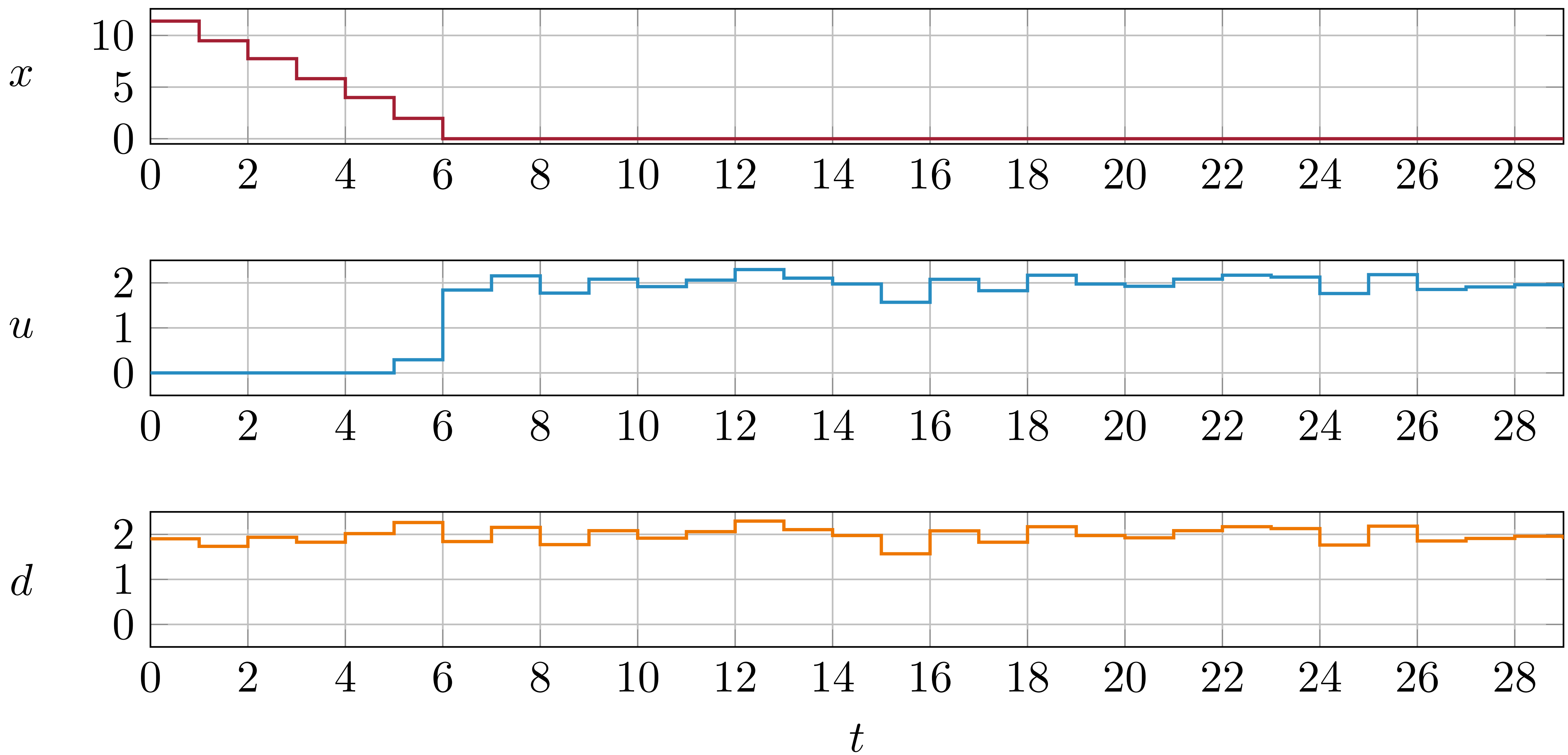
Demand

Inventory management

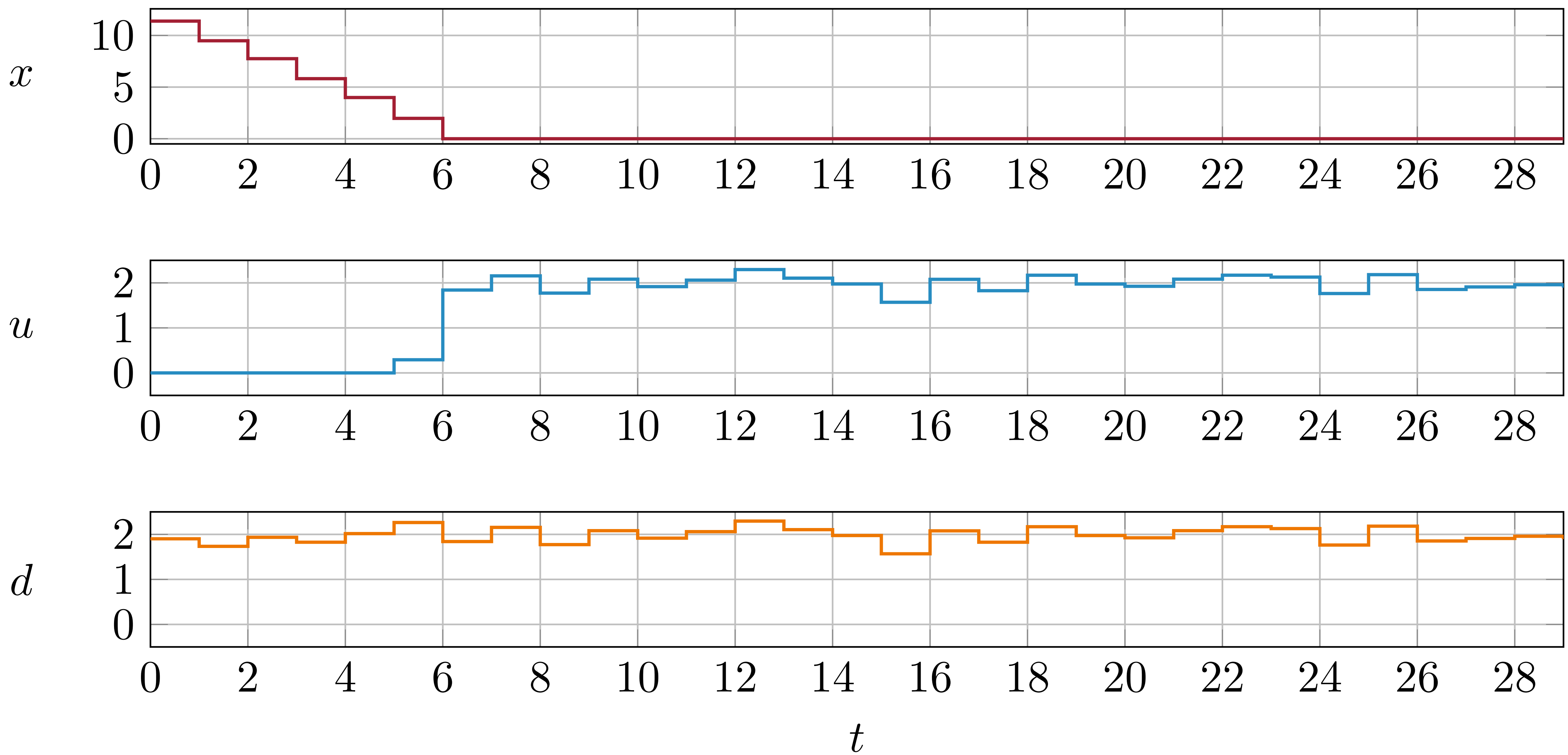
$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} \quad & x_{t+1} = x_t + u_t - d_t \\
& x_0 = x_{\text{init}} \\
& 0 \le u_t \le M
\end{aligned}
$$

Parameters

How do x and u depend on the parameters?



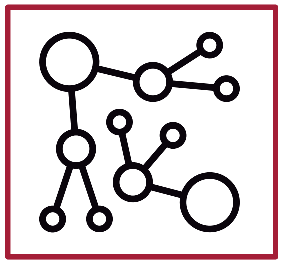
How do x and u depend on the parameters?



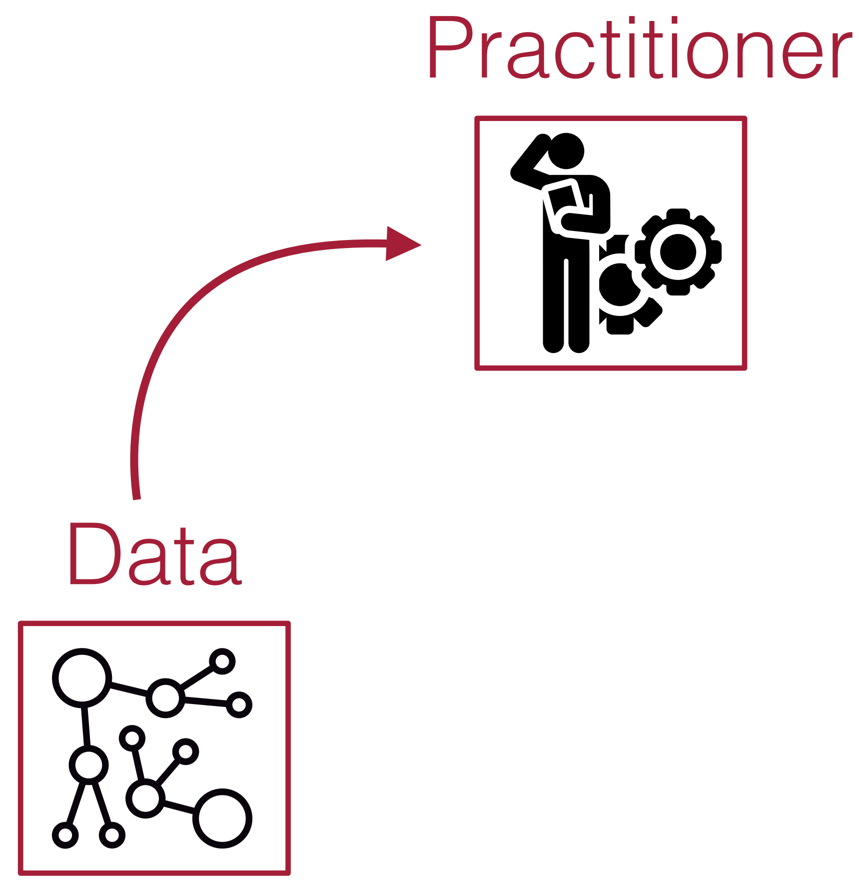
x_{init} ? d_t ?

The decision-making workflow

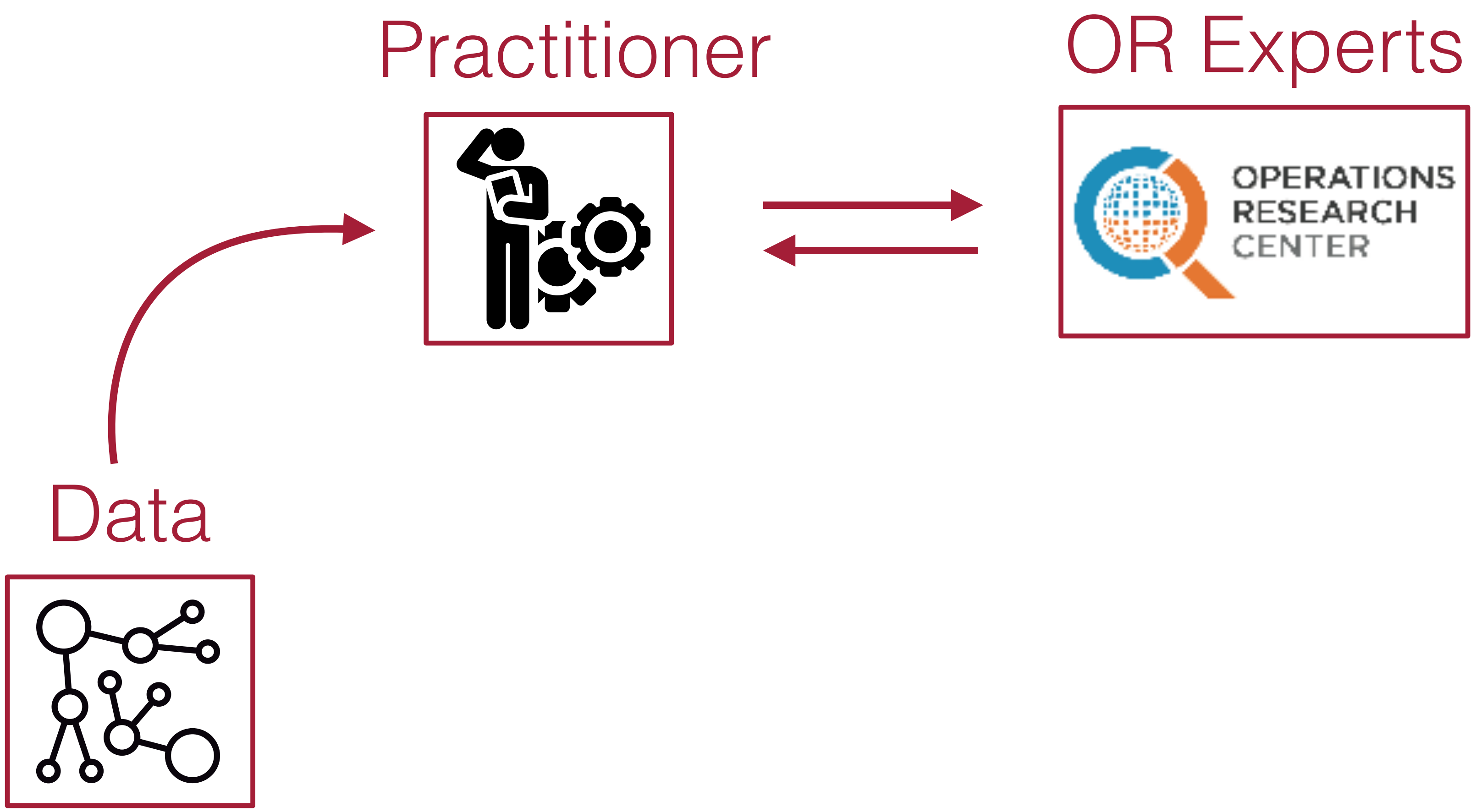
Data



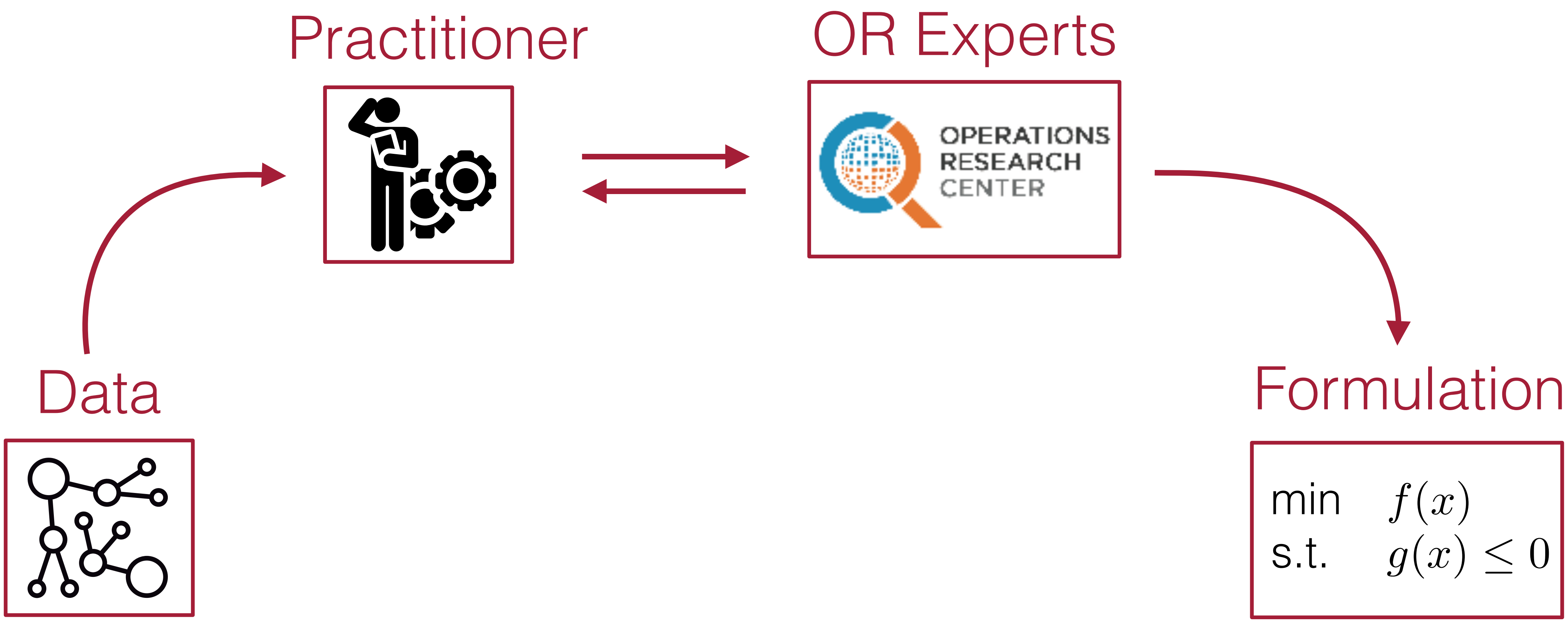
The decision-making workflow



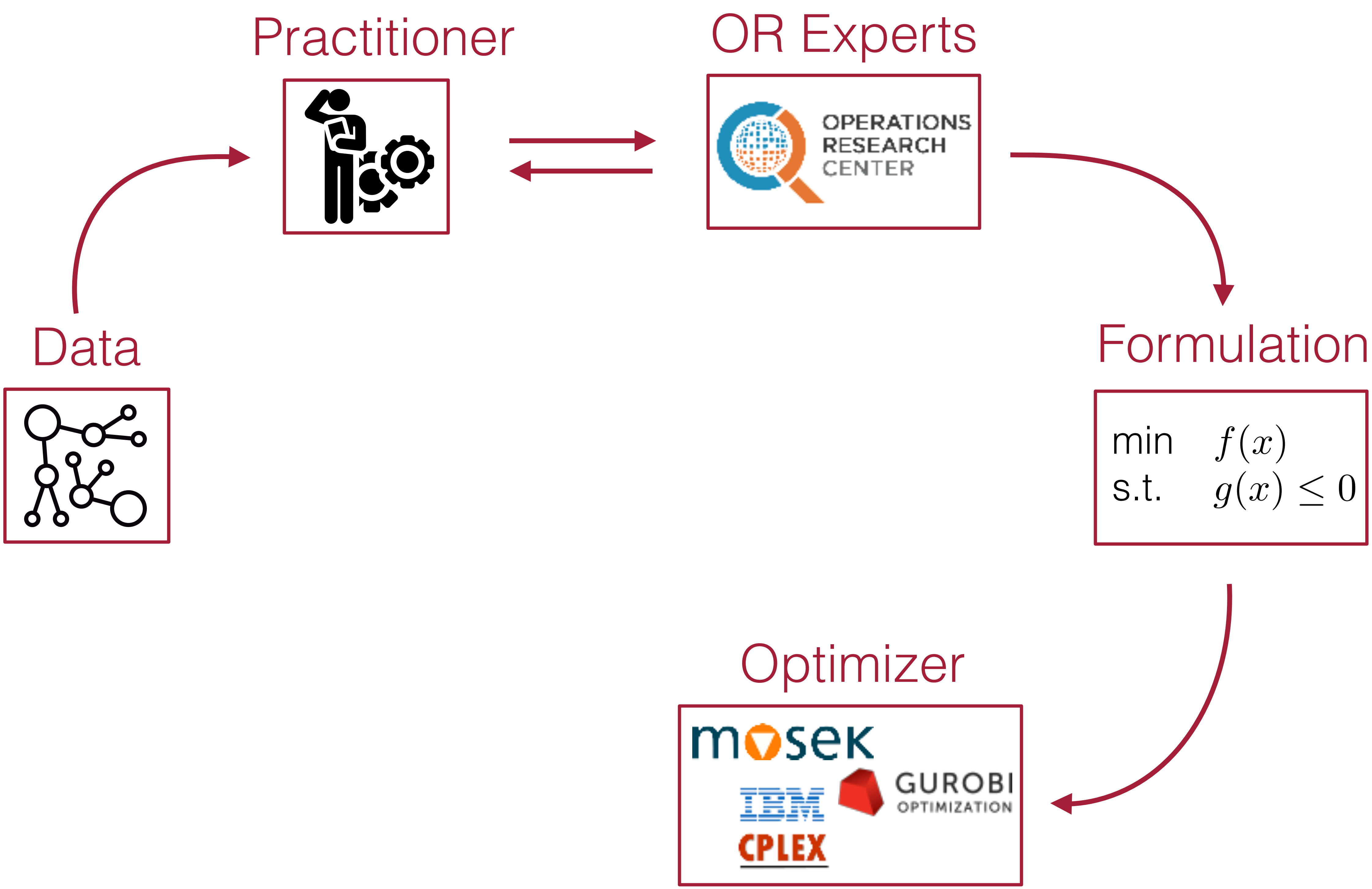
The decision-making workflow



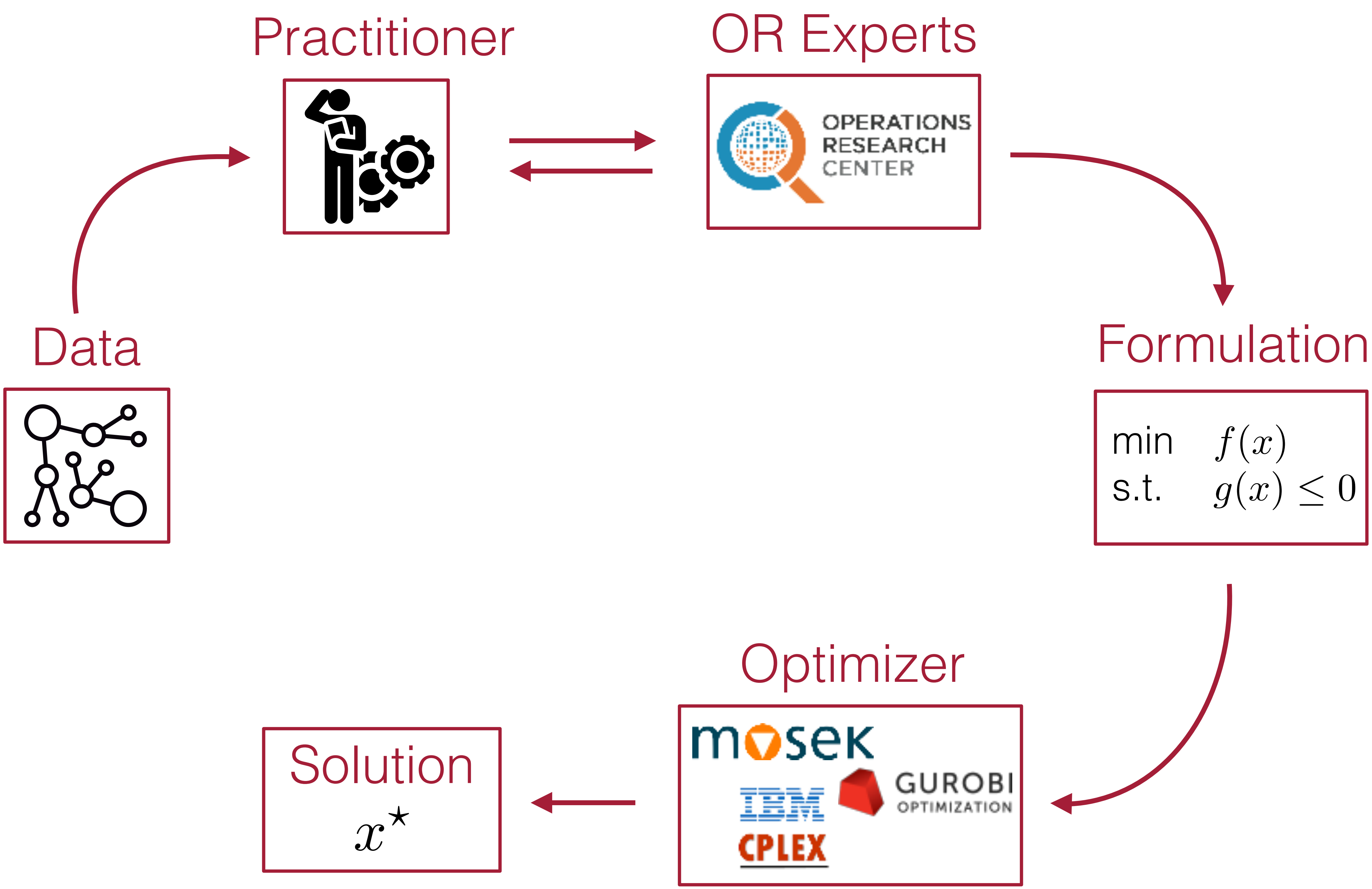
The decision-making workflow



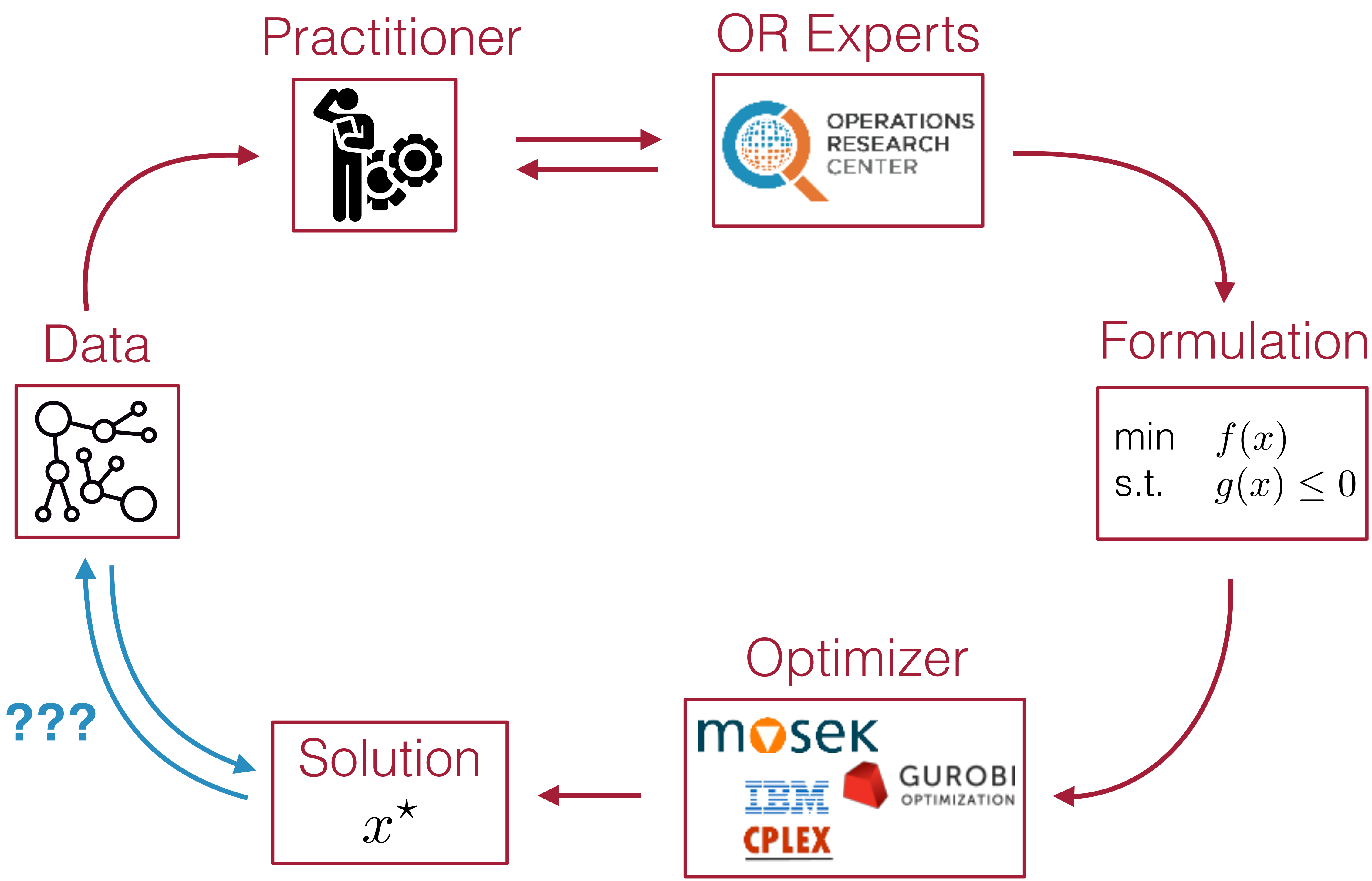
The decision-making workflow



The decision-making workflow



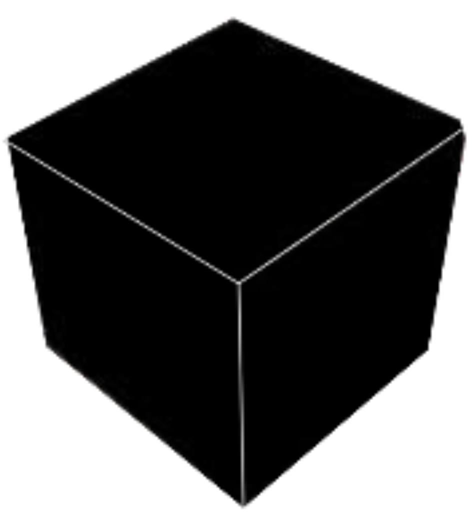
The decision-making workflow



Traditional optimization

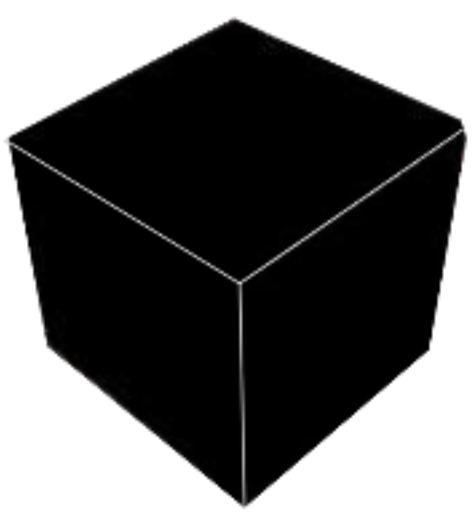
Traditional optimization

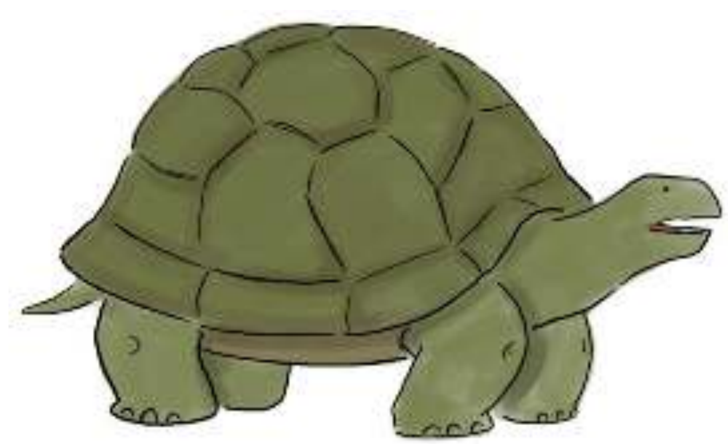
$$\theta \longrightarrow \boxed{\text{Optimization}} \longrightarrow x^\star$$



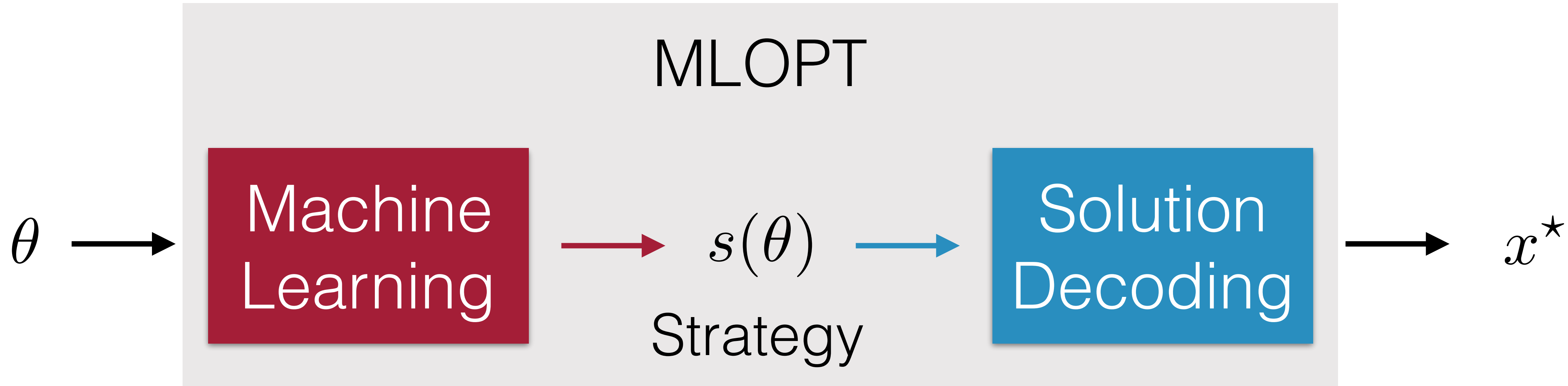
Traditional optimization

$\theta \longrightarrow$ Optimization $\longrightarrow x^\star$

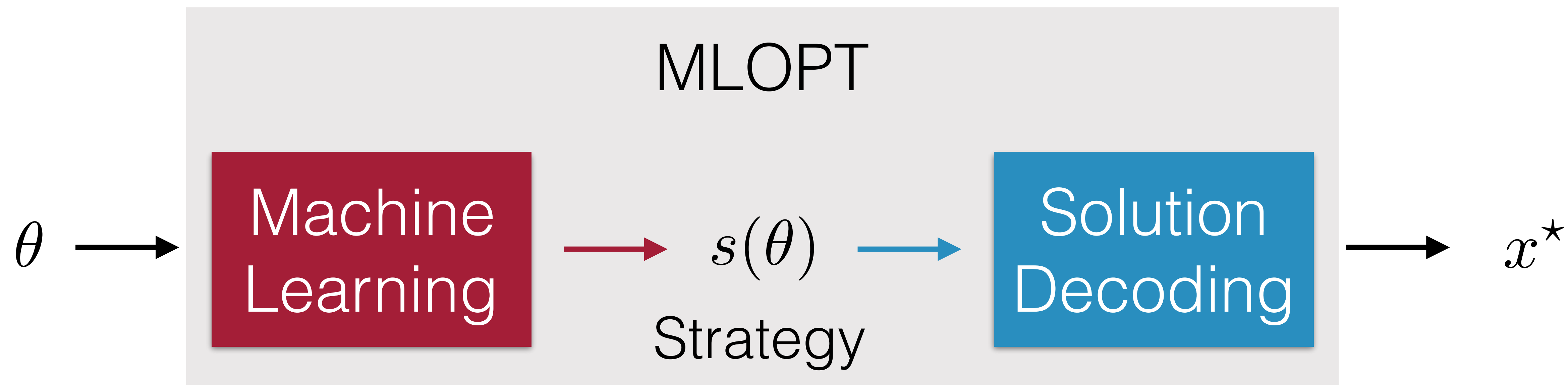


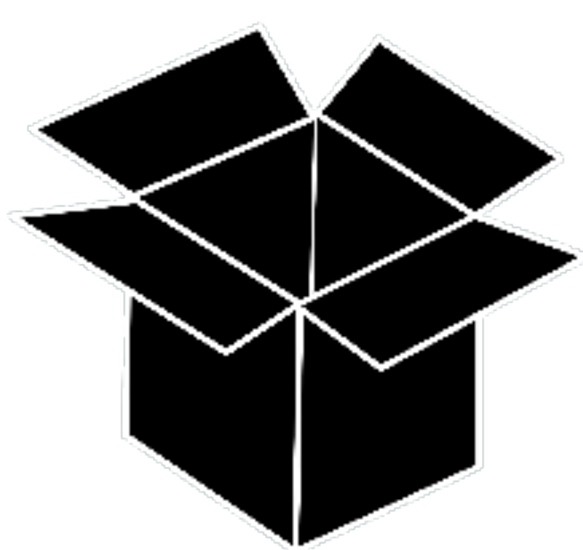


Machine Learning Optimizer

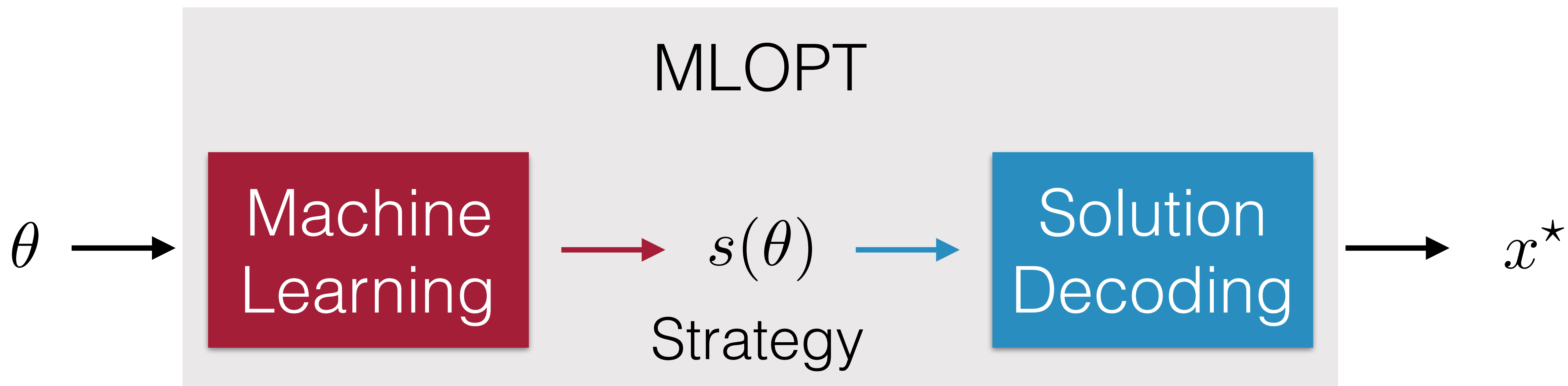


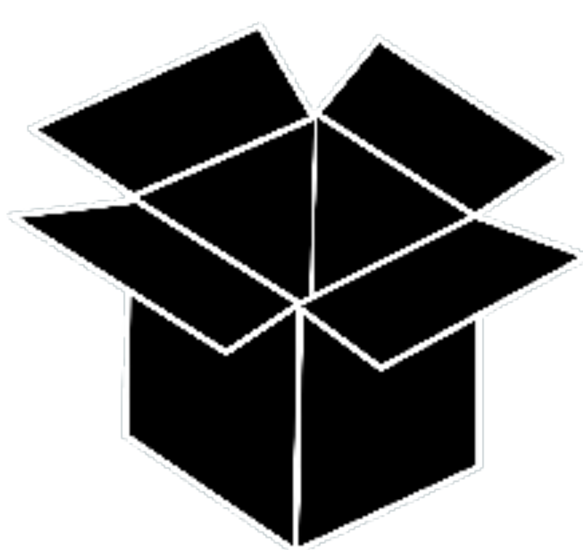
Machine Learning Optimizer



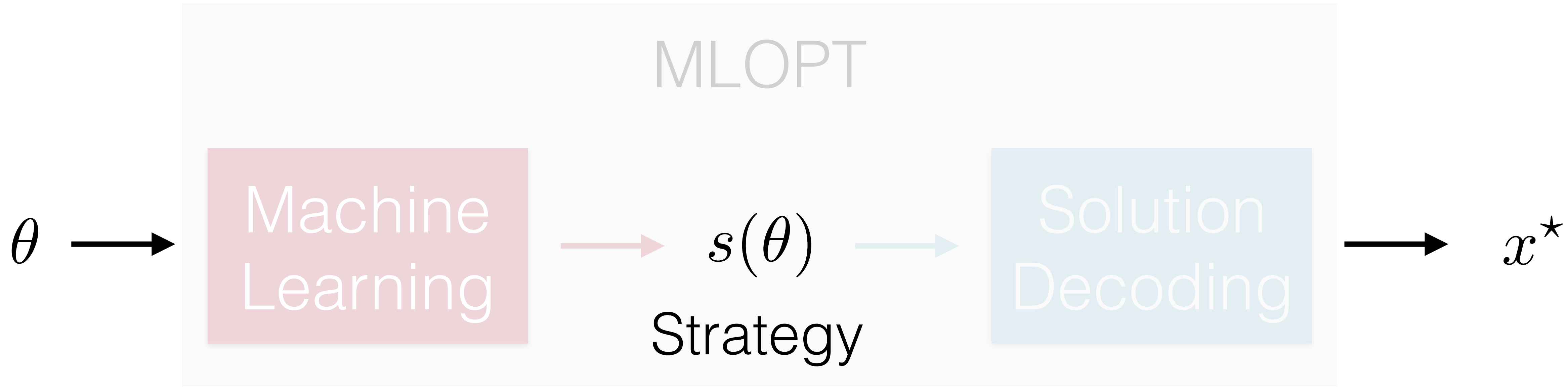


Machine Learning Optimizer



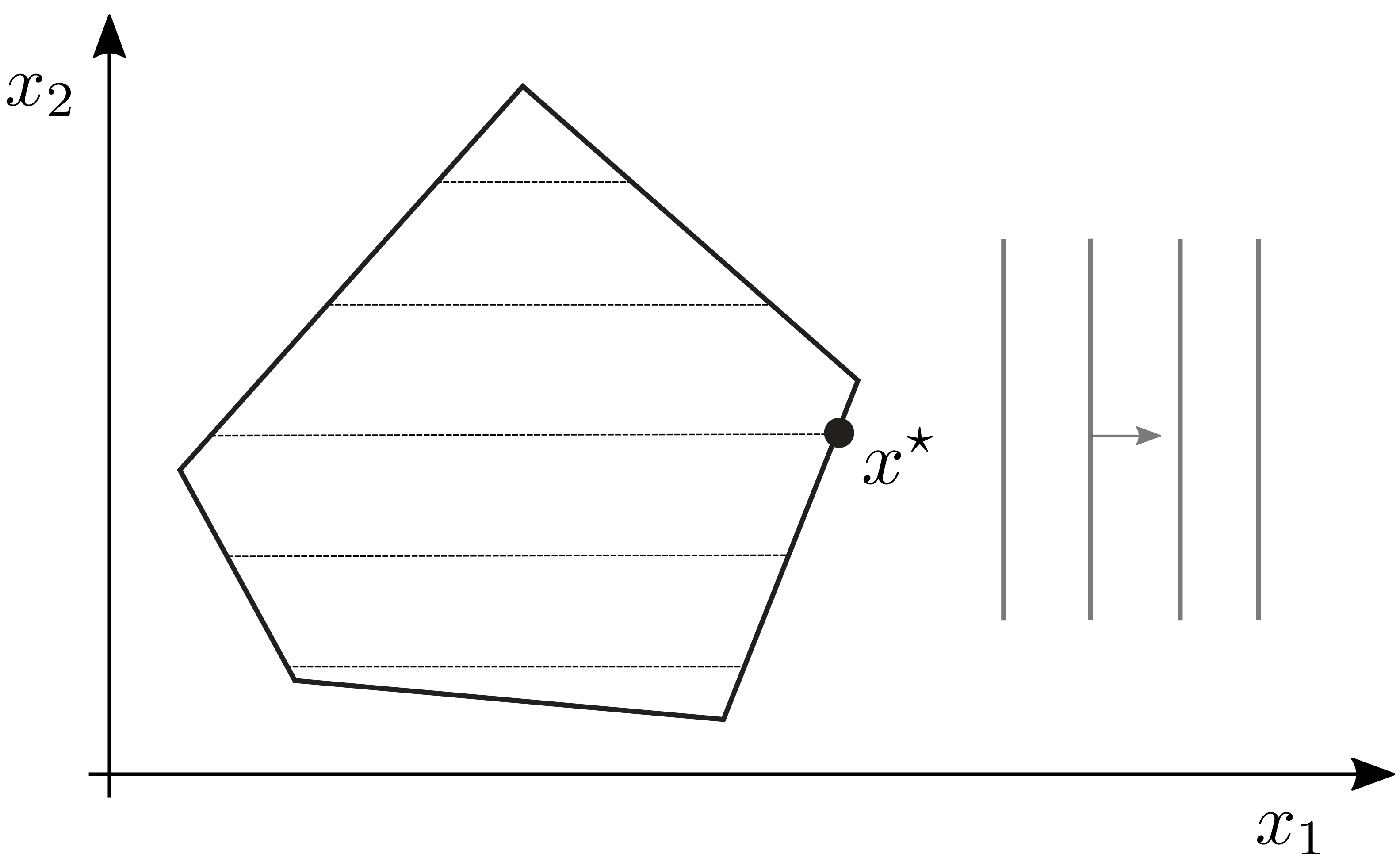


What is a strategy?



Strategies for Mixed-Integer Optimization

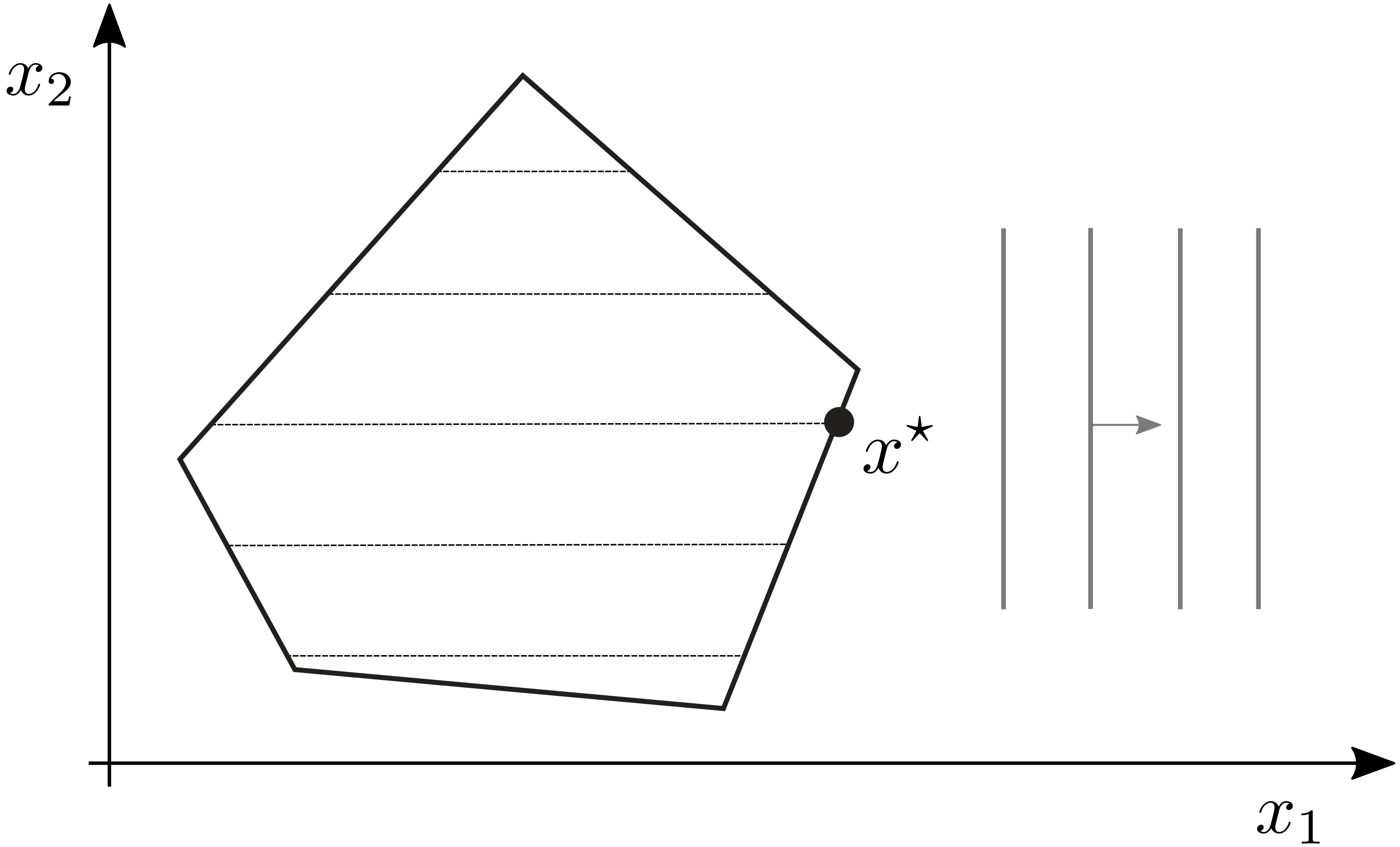
$$
\begin{array}{ll}
\text{minimize} & c(\theta)^T x \\
\text{subject to} & A(\theta)x \leq b(\theta) \\
& x_{\mathcal{I}} \in \mathbf{Z}^d
\end{array}
$$



Strategies for Mixed-Integer Optimization

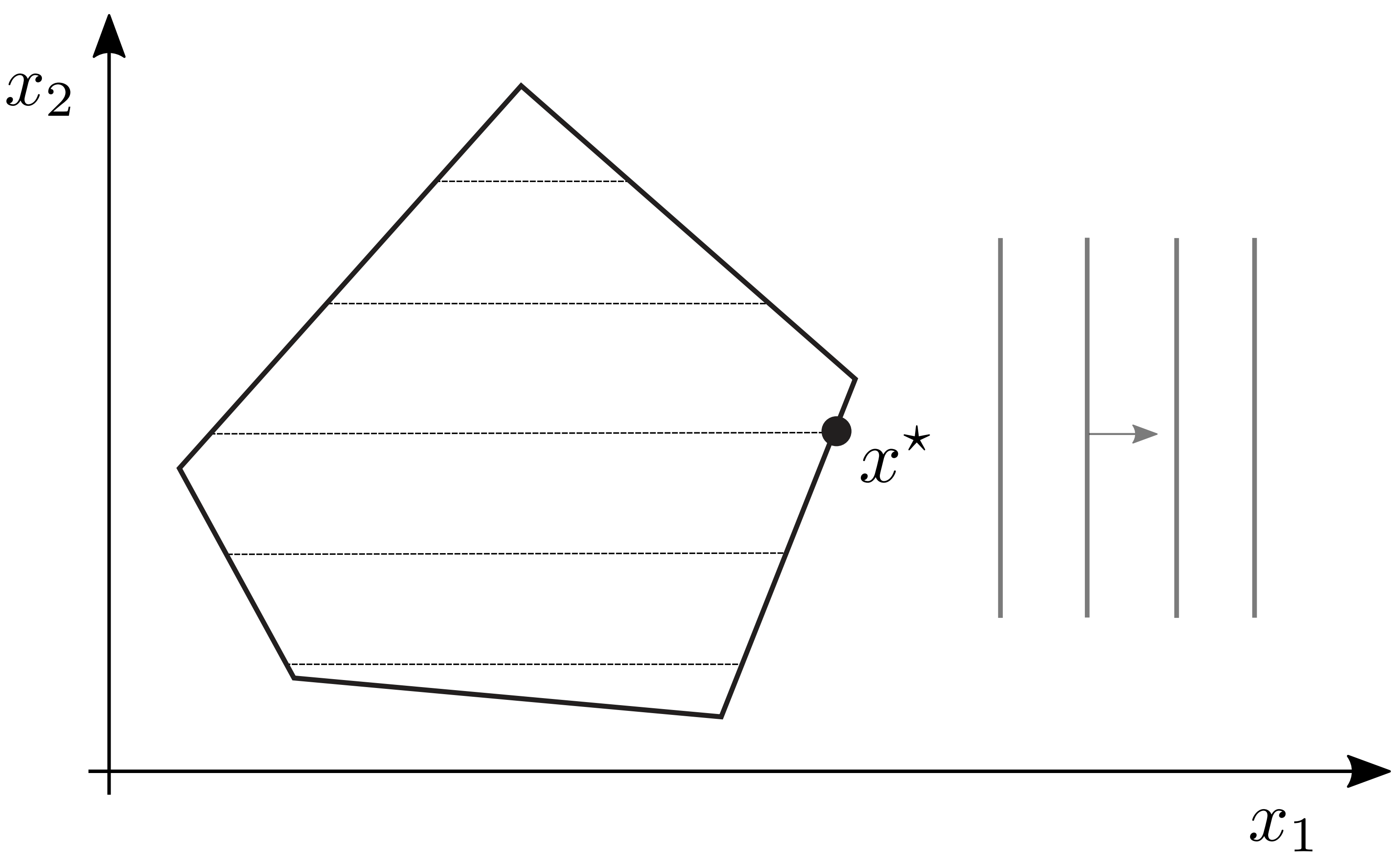
$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A(\theta)x \leq b(\theta) \\ & x_{\mathcal{I}} \in \mathbf{Z}^d \end{array}$$

Parameters



Strategies for Mixed-Integer Optimization

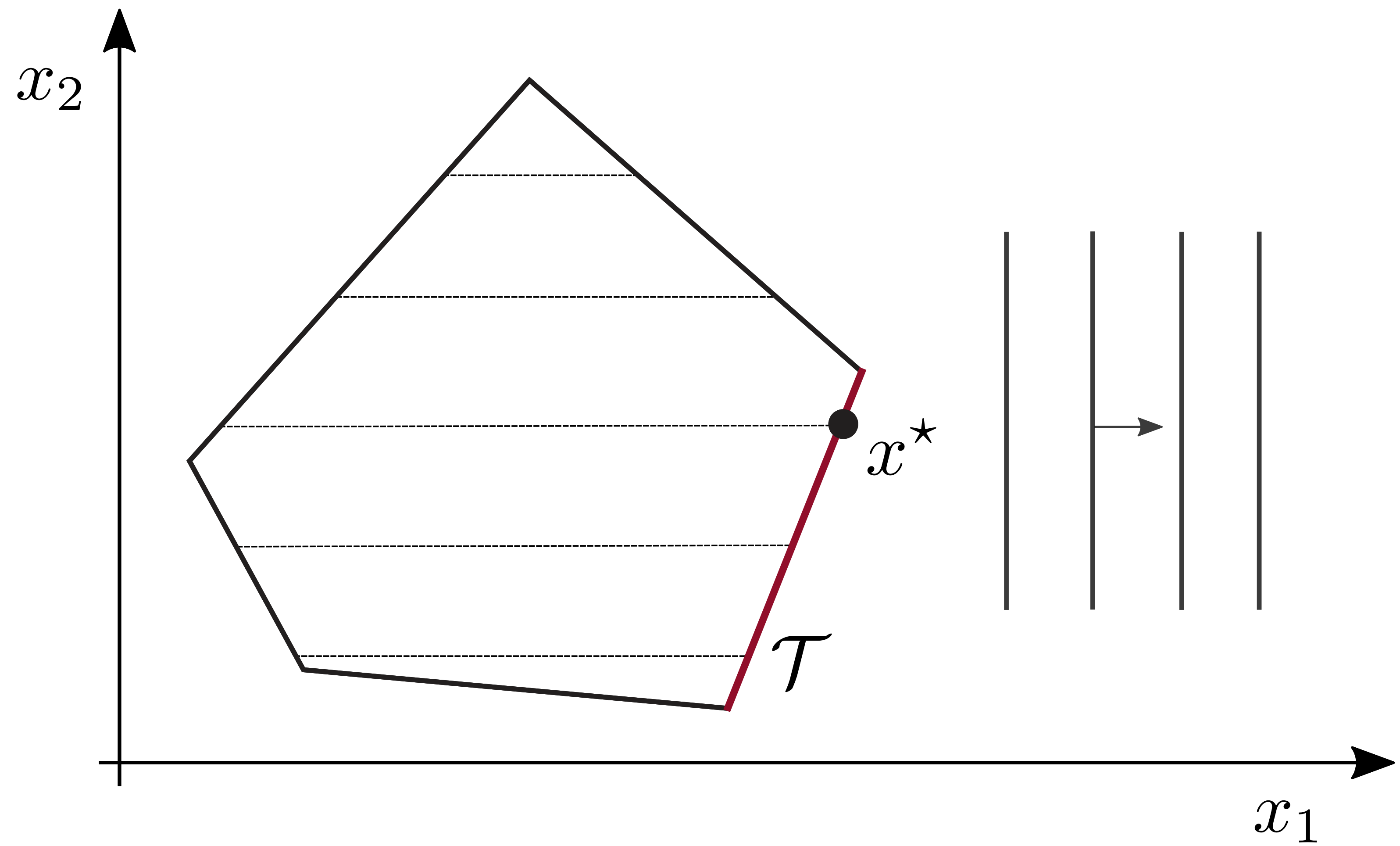
$$s(\theta) = (\mathcal{T}(\theta), x^\star_{\mathcal{I}}(\theta))$$



Strategies for Mixed-Integer Optimization

$$s(\theta) = (\mathcal{T}(\theta), x^\star_{\mathcal{I}}(\theta))$$

Tight constraints

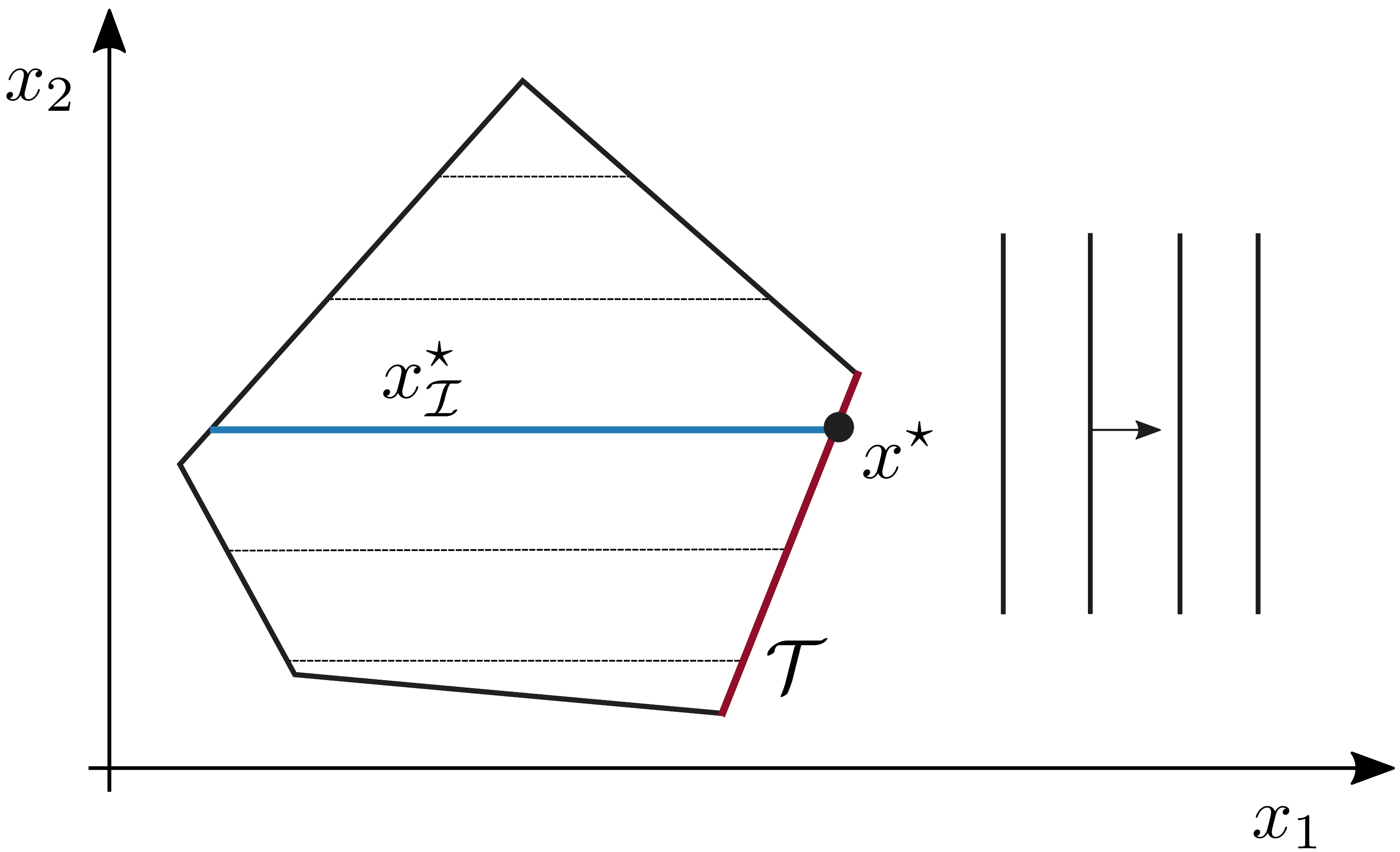


Strategies for Mixed-Integer Optimization

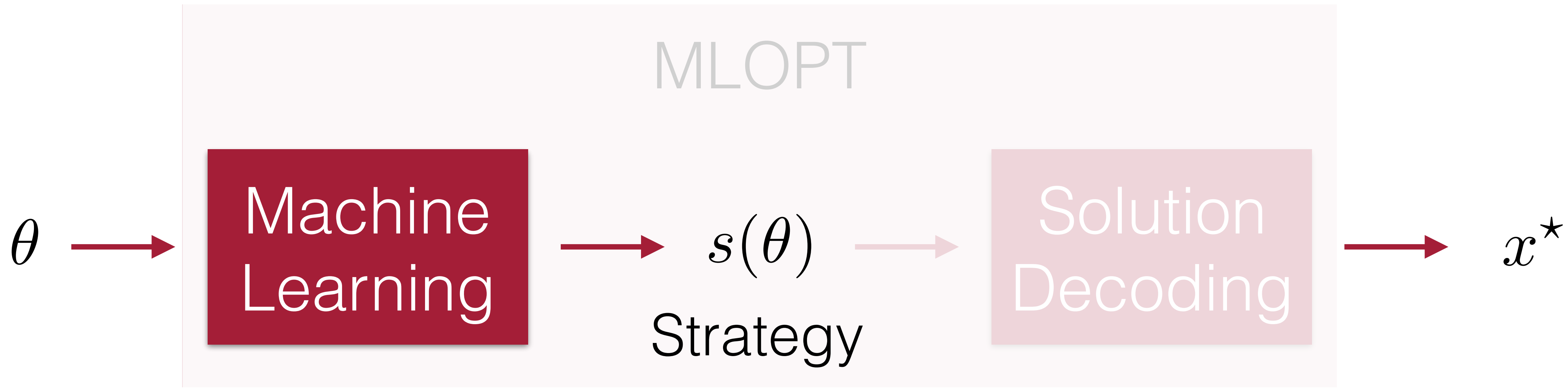
$$s(\theta) = (\mathcal{T}(\theta), x^\star_{\mathcal{I}}(\theta))$$

Tight constraints

Integer values



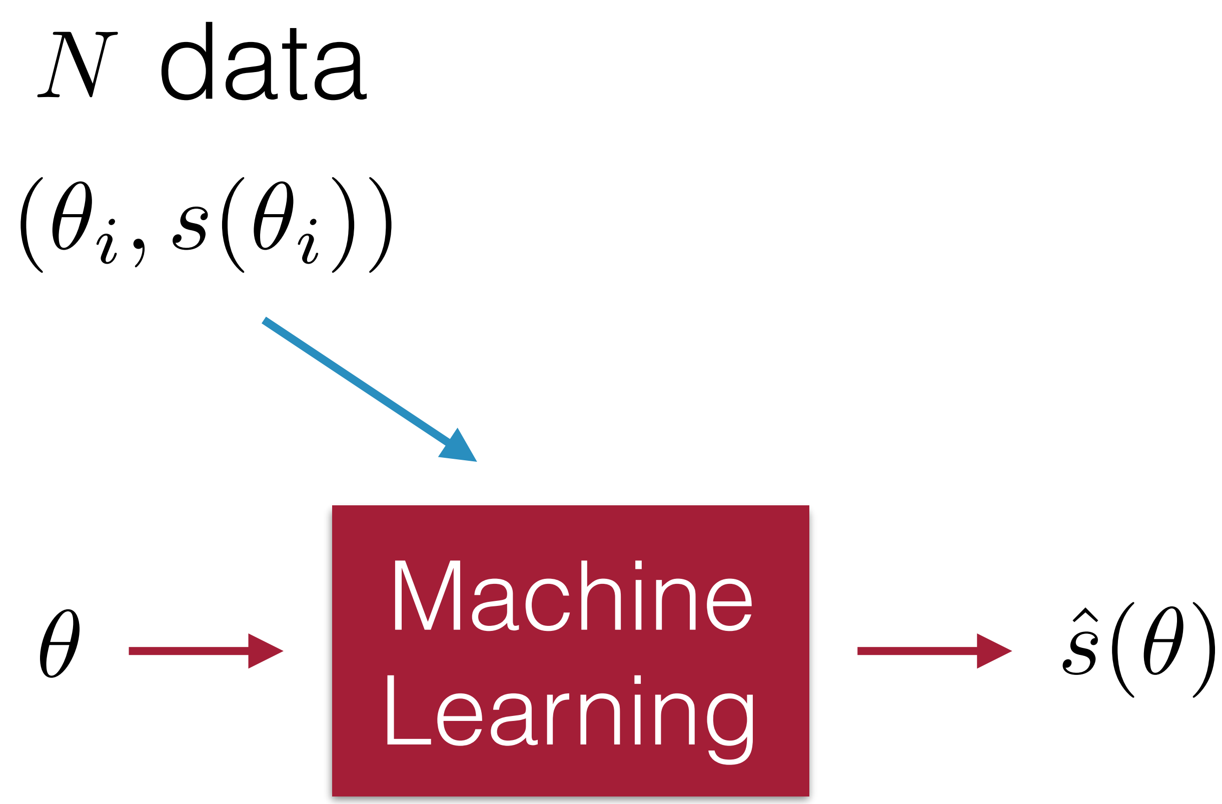
Learning the strategies



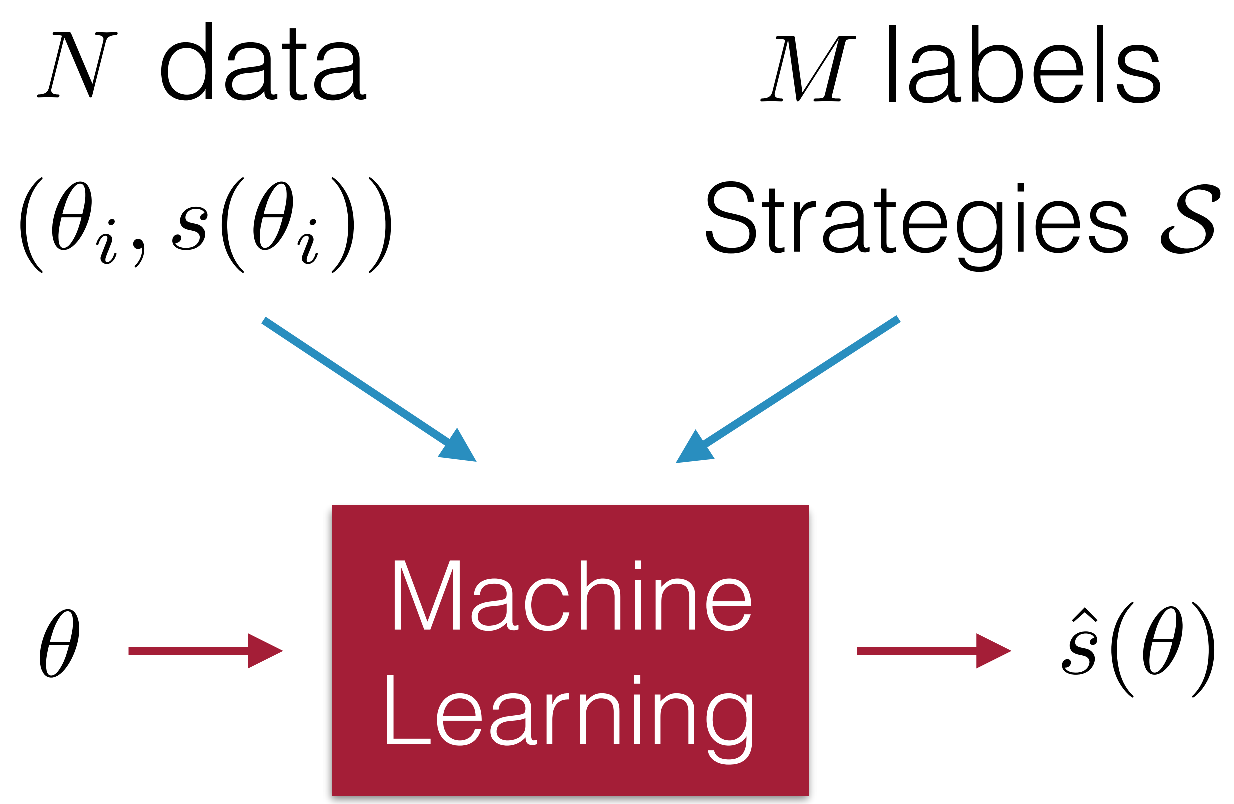
Selecting strategies is a classification problem

$\theta \longrightarrow$ **Machine Learning** $\longrightarrow \hat{s}(\theta)$

Selecting strategies is a classification problem



Selecting strategies is a classification problem



Selecting strategies is a classification problem

N data

$(\theta_i, s(\theta_i))$

M labels

Strategies $\mathcal{S}$

$\theta \longrightarrow$ **Machine Learning** $\longrightarrow \hat{s}(\theta)$

Optimal Trees

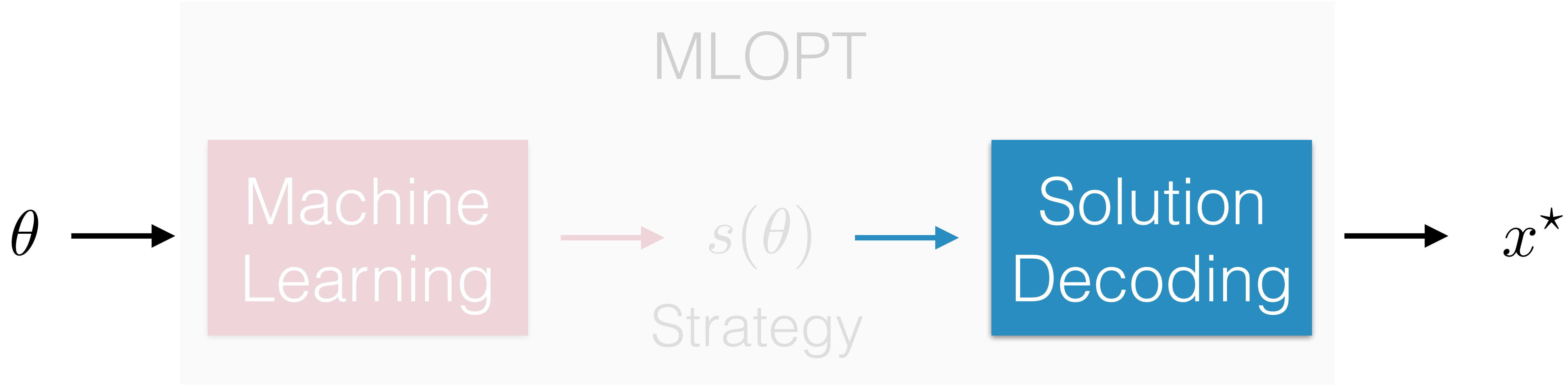
Interpretable AI

[Bertsimas and Dunn (2017)]

Neural Networks

PyTorch

Computing the solution from the optimal strategy



Solution decoding is just a linear system

$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A(\theta)x \leq b(\theta) \\ & x_{\mathcal{I}} \in \mathbf{Z}^d \end{array}$$

Solution decoding is just a linear system

$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A(\theta)x \leq b(\theta) \\ & x_{\mathcal{I}} \in \mathbf{Z}^d \end{array}$$

$$\Big\downarrow \ \text{Strategy} \quad s(\theta)$$

$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A_i(\theta)x = b_i(\theta) \quad \forall i \in \mathcal{T}(\theta) \\ & x_{\mathcal{I}} = x^\star_{\mathcal{I}}(\theta) \end{array}$$

Solution decoding is just a linear system

$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A(\theta)x \leq b(\theta) \\ & x_{\mathcal{I}} \in \mathbf{Z}^d \end{array}$$

$$\Big\downarrow \; \text{Strategy} \quad s(\theta)$$

$$\begin{array}{ll} \text{minimize} & c(\theta)^T x \\ \text{subject to} & A_i(\theta)x = b_i(\theta) \quad \forall i \in \boxed{\mathcal{T}(\theta)} \\ & x_{\mathcal{I}} = \boxed{x^\star_{\mathcal{I}}(\theta)} \end{array}$$

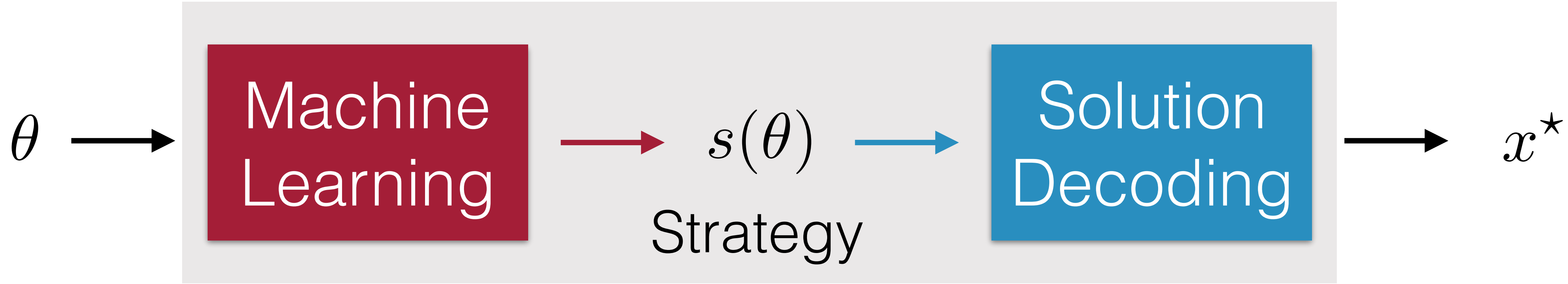
$$\Big\downarrow$$

KKT Linear System

MLOPT: Machine Learning Optimizer

Offline Learning

Online Optimization



Inventory management

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{t=0}^{T-1} h(x_t) + o(u_t) \\ \text{subject to} & x_{t+1} = x_t + u_t - d_t \\ & x_0 = x_{\text{init}} \\ & 0 \leq u_t \leq M \end{array}$$

Inventory management

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\ \text{subject to} & x_{t+1} = x_t + u_t - d_t \\ & x_0 = x_{\text{init}} \\ & 0 \leq u_t \leq M \end{array}$$

Inventory

Inventory management

$$
\begin{array}{ll}
\text{minimize} & \displaystyle\sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} & \boxed{x_{t+1}} = x_t + \boxed{u_t} - d_t \\
& x_0 = x_{\text{init}} \\
& 0 \le u_t \le M
\end{array}
$$

Inventory

Order

Inventory management

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} \quad & \boxed{x_{t+1}} = x_t + \boxed{u_t} - \boxed{d_t} \\
& x_0 = x_{\text{init}} \\
& 0 \le u_t \le M
\end{aligned}
$$

Inventory

Order

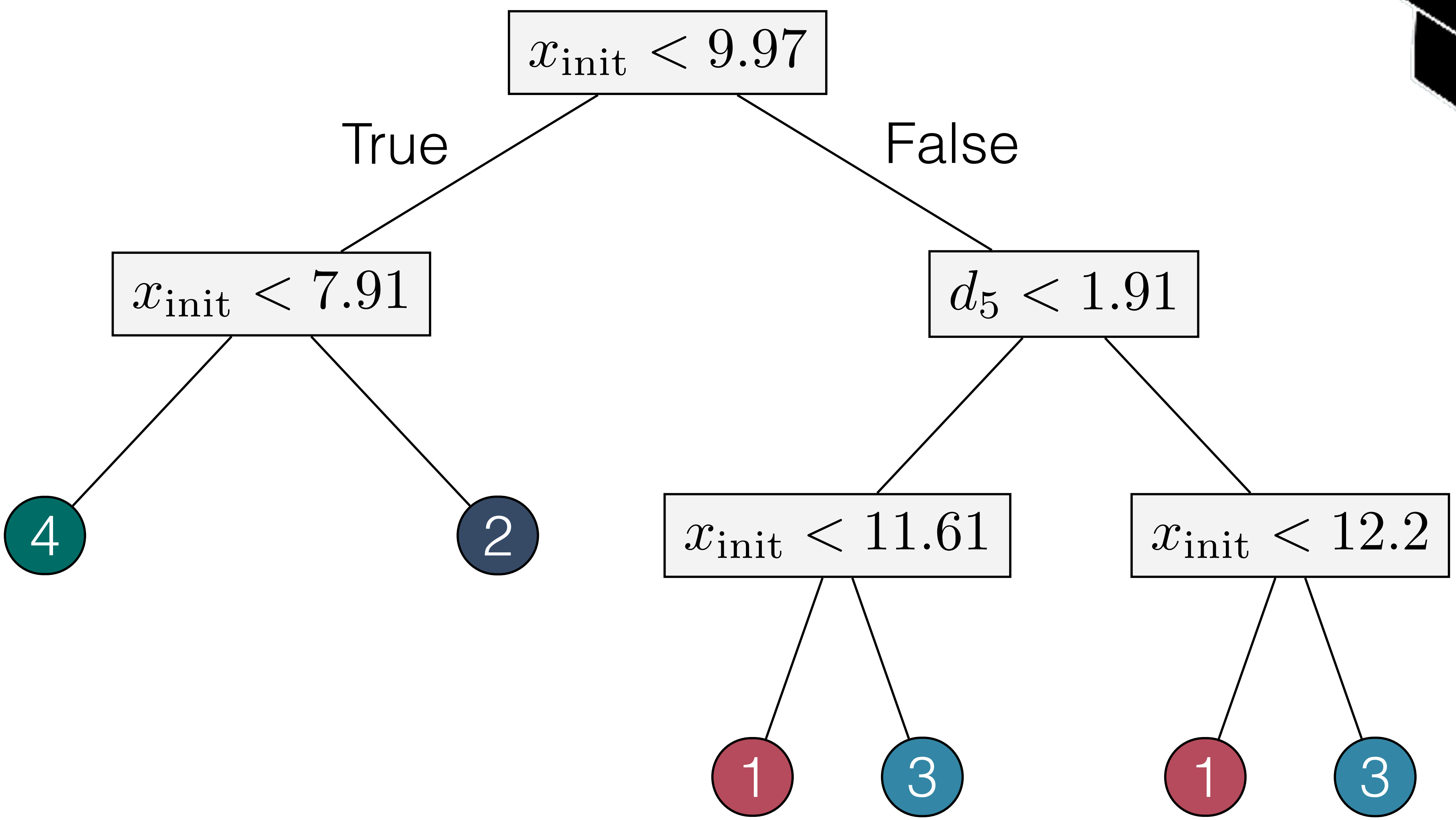
Demand

Inventory management

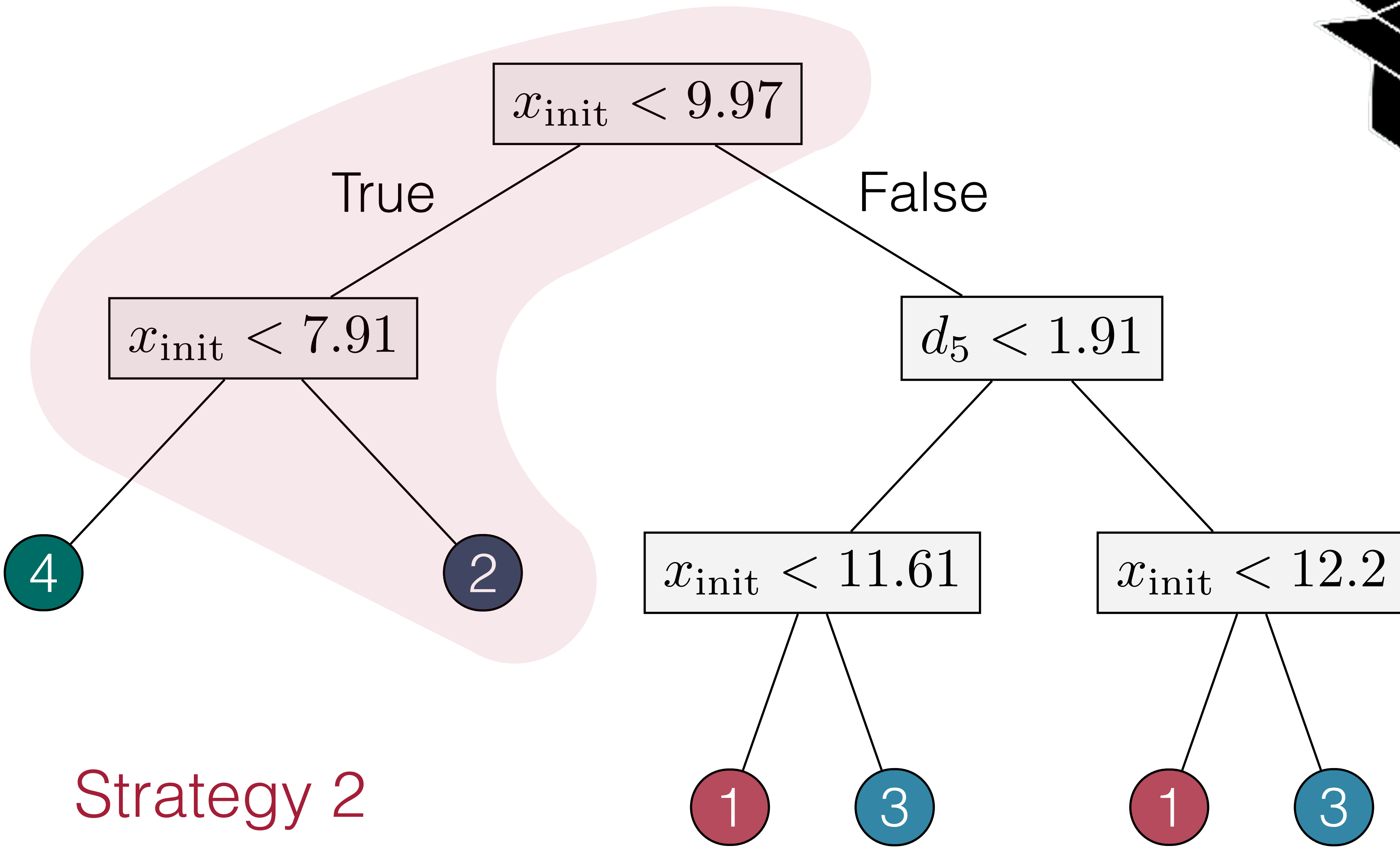
$$
\begin{array}{ll}
\text{minimize} & \sum_{t=0}^{T-1} h(x_t) + o(u_t) \\
\text{subject to} & x_{t+1} = x_t + u_t - \boxed{d_t} \\
& x_0 = \boxed{x_{\text{init}}} \\
& 0 \leq u_t \leq M
\end{array}
$$

Parameters

Strategy selection using Optimal Trees



Strategy selection using Optimal Trees



Strategy 2

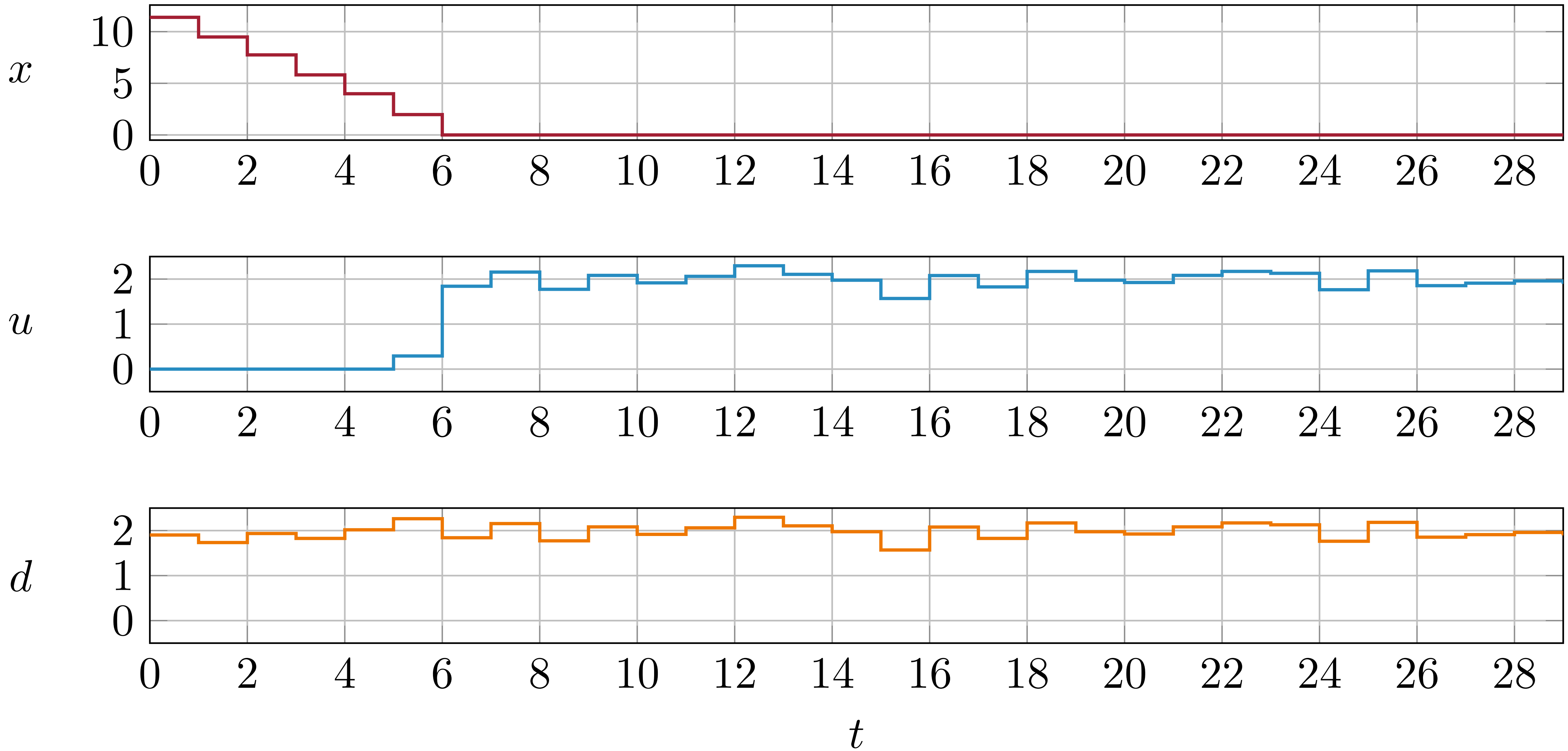
$$u_t = 0 \qquad t \le 4$$

$$0 \le u_t \le M \qquad t > 4$$

Strategies for inventory

Strategy 2

$$\begin{array}{ll} u_t = 0 & t \leq 4 \\ 0 \leq u_t \leq M & t > 4 \end{array}$$



Real-world Example

Sparse portfolio optimization

$$
\begin{array}{ll}
\text{maximize} & r^T w - \gamma w^T \Sigma w \\
\text{subject to} & \mathbf{1}^T w = 1 \\
& \mathbf{card}(w) \leq c
\end{array}
$$

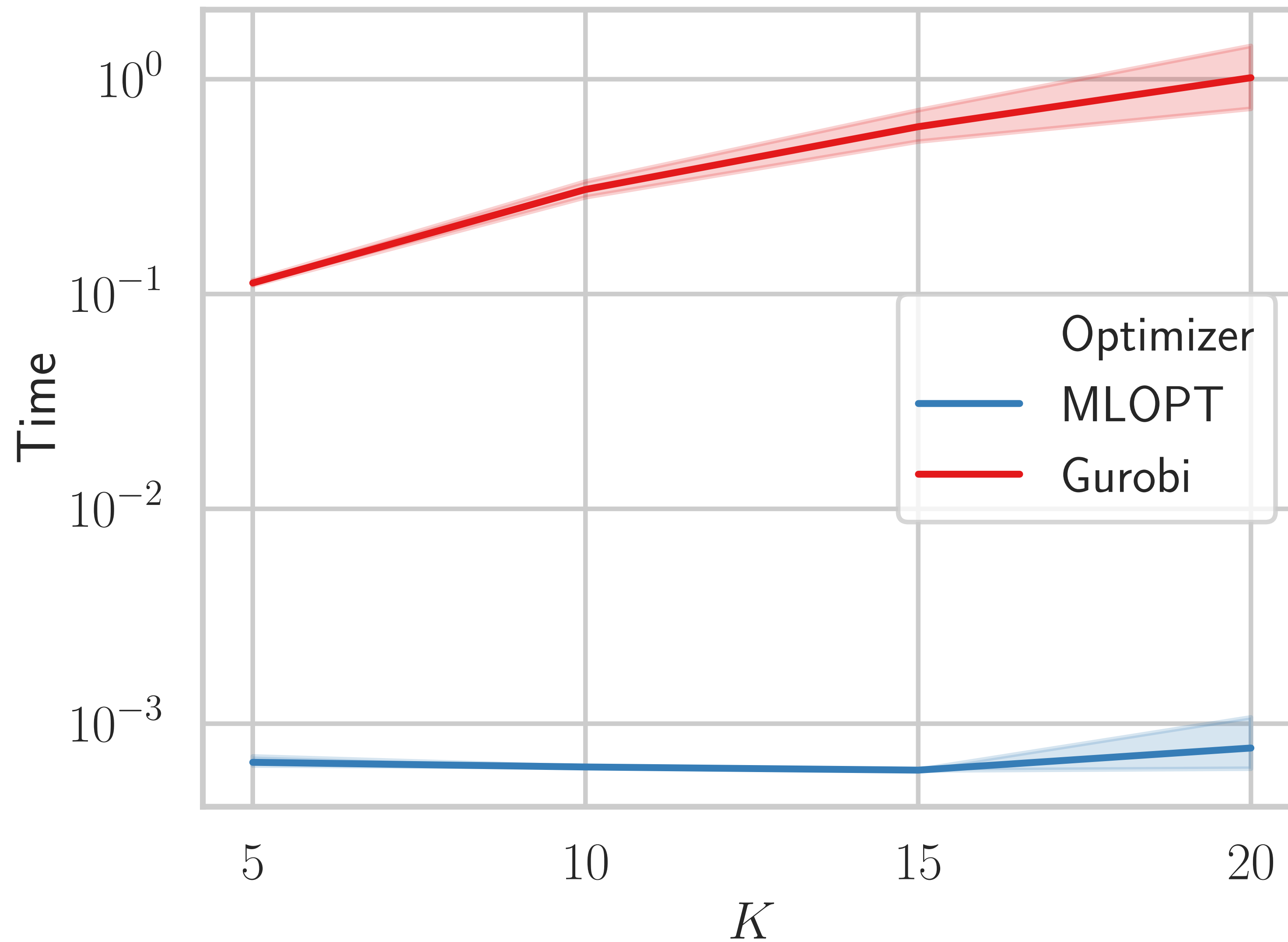
k-factor Risk Model

$$\Sigma = F \Sigma^k F^T + D$$

Parameters

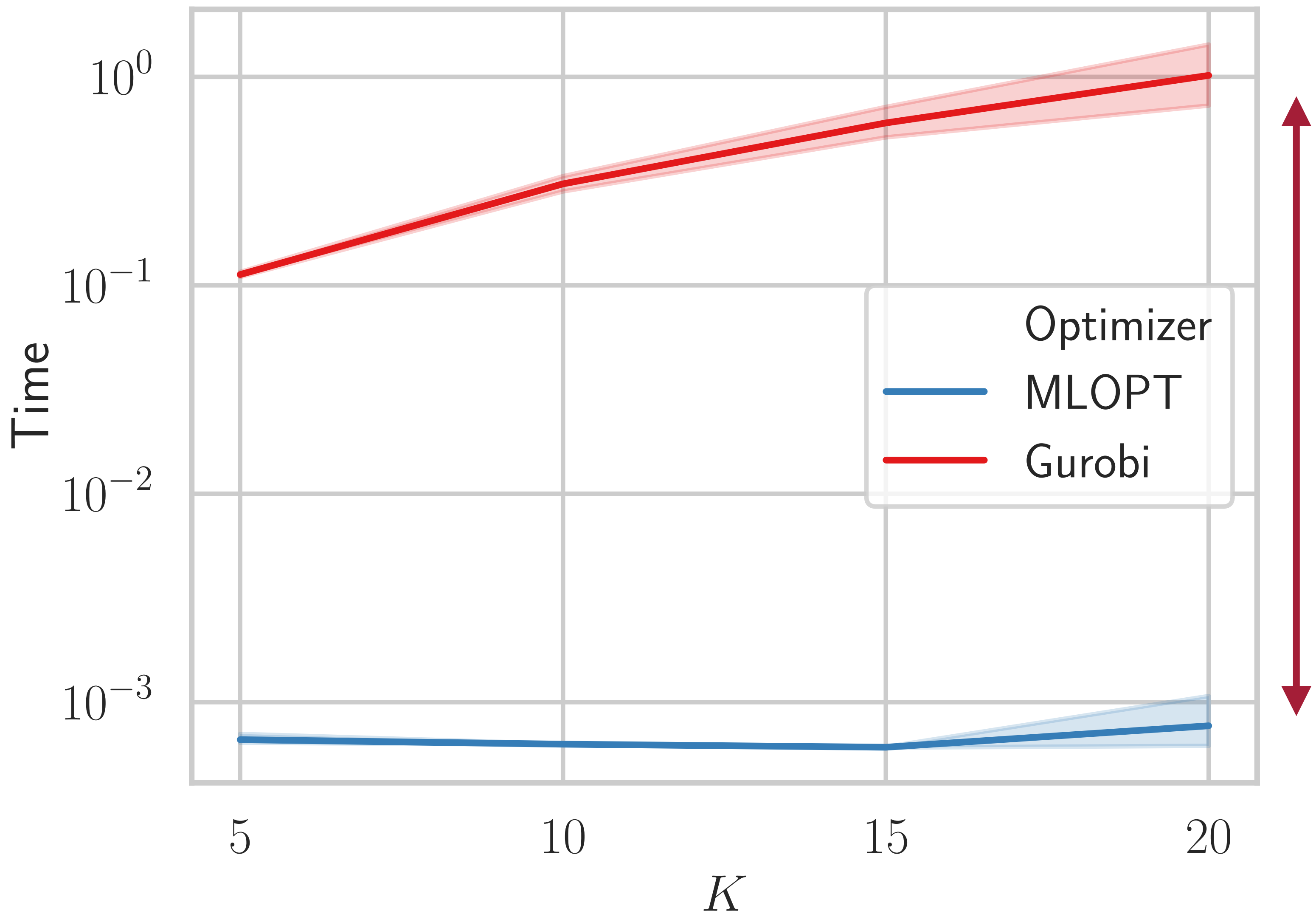
$$\theta = (r, \Sigma^k, F, D)$$

S&P100 backtesting: high speed and accuracy



K	M	acc [%]	suboptimality	infeasibility
5	1027	99.11	9.88×10^{-3}	2.25×10^{-9}
10	1083	99.21	7.01×10^{-3}	8.19×10^{-7}
15	1120	99.30	1.92×10^{-2}	2.68×10^{-7}
20	1209	99.50	3.54×10^{-3}	4.12×10^{-7}

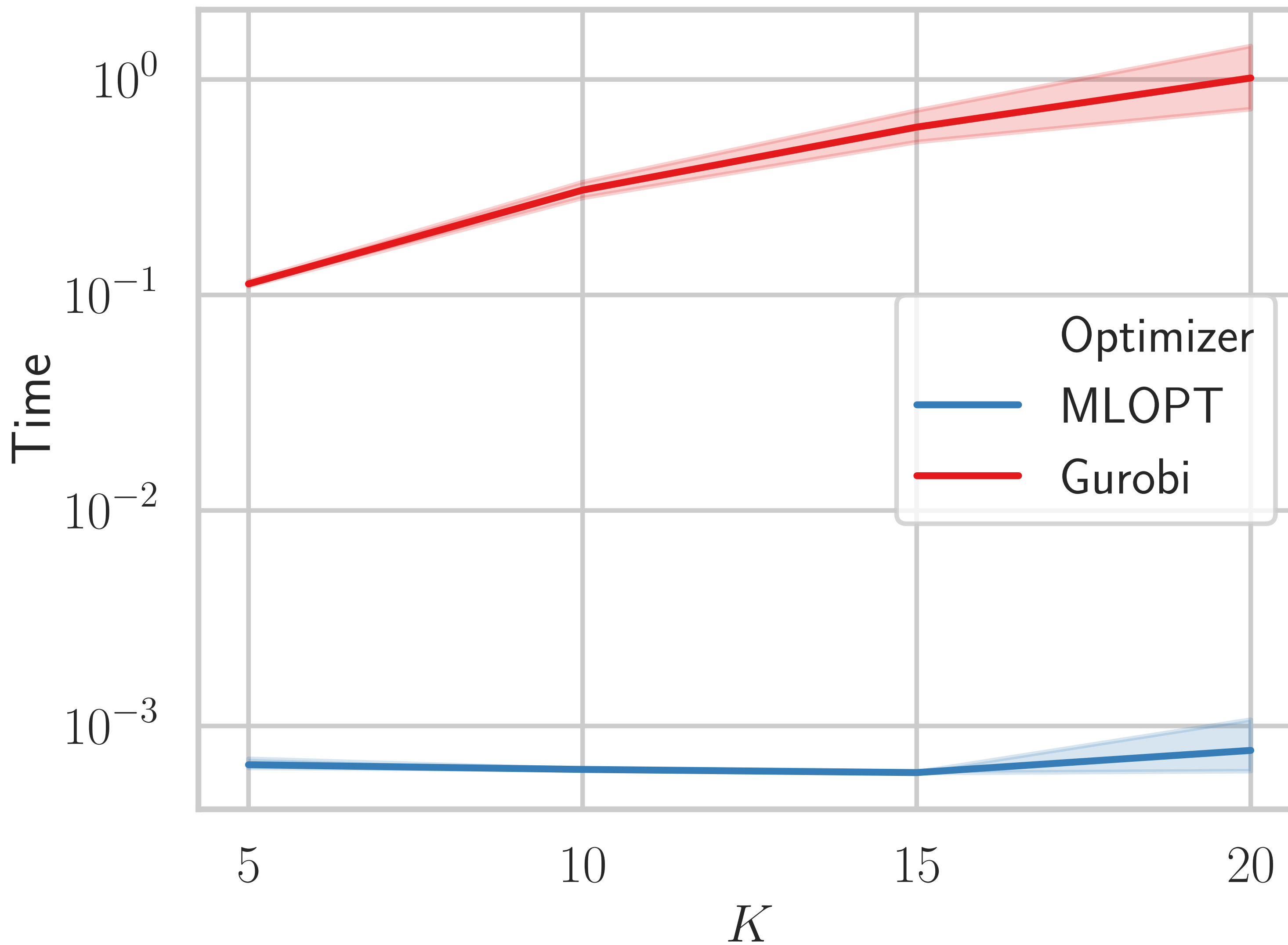
S&P100 backtesting: high speed and accuracy



1000x faster

K	M	acc [%]	suboptimality	infeasibility
5	1027	99.11	9.88×10^{-3}	2.25×10^{-9}
10	1083	99.21	7.01×10^{-3}	8.19×10^{-7}
15	1120	99.30	1.92×10^{-2}	2.68×10^{-7}
20	1209	99.50	3.54×10^{-3}	4.12×10^{-7}

S&P100 backtesting: high speed and accuracy

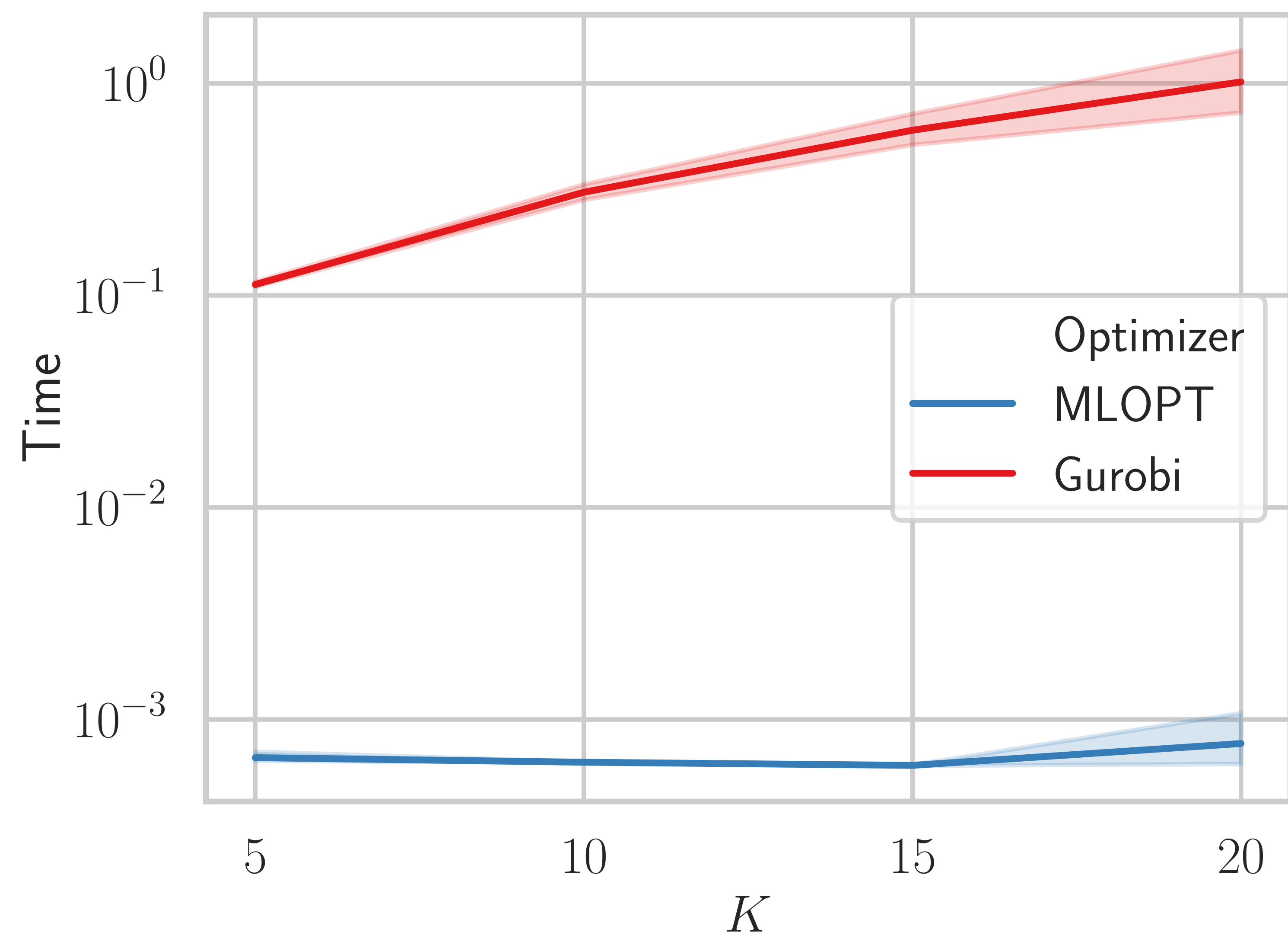


1000x faster

Low suboptimality and infeasibility

K	M	acc [%]	suboptimality	infeasibility
5	1027	99.11	9.88×10^{-3}	2.25×10^{-9}
10	1083	99.21	7.01×10^{-3}	8.19×10^{-7}
15	1120	99.30	1.92×10^{-2}	2.68×10^{-7}
20	1209	99.50	3.54×10^{-3}	4.12×10^{-7}

S&P100 backtesting: high speed and accuracy



1000x faster

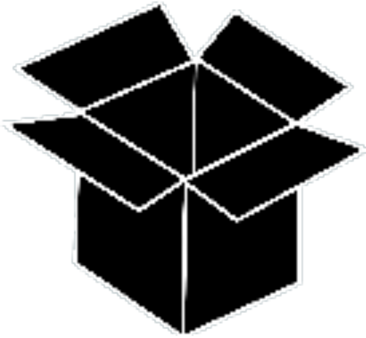
High accuracy

Low suboptimality and infeasibility

K	M	acc [%]	suboptimality	infeasibility
5	1027	99.11	9.88×10^{-3}	2.25×10^{-9}
10	1083	99.21	7.01×10^{-3}	8.19×10^{-7}
15	1120	99.30	1.92×10^{-2}	2.68×10^{-7}
20	1209	99.50	3.54×10^{-3}	4.12×10^{-7}

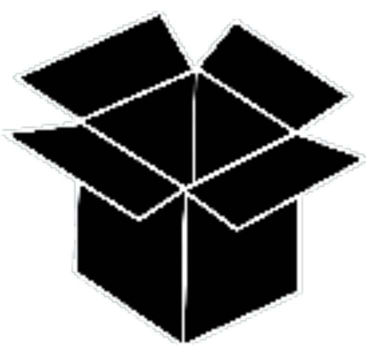
Optimization
links data
to decisions

Optimization
links data
to decisions



Understand
using ML

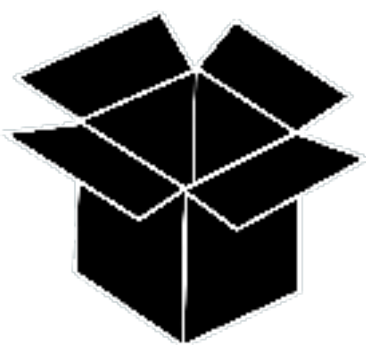
Optimization links data to decisions



Understand using ML

Solve really fast

Optimization links data to decisions



Understand using ML

Solve really fast

Bertsimas, D., Stellato, B., *The Voice of Optimization*, arXiv:1812.09991

Bertsimas, D., Stellato, B., *Online Mixed-Integer Optimization in Milliseconds*, arXiv:1907.02206

stellato@mit.edu

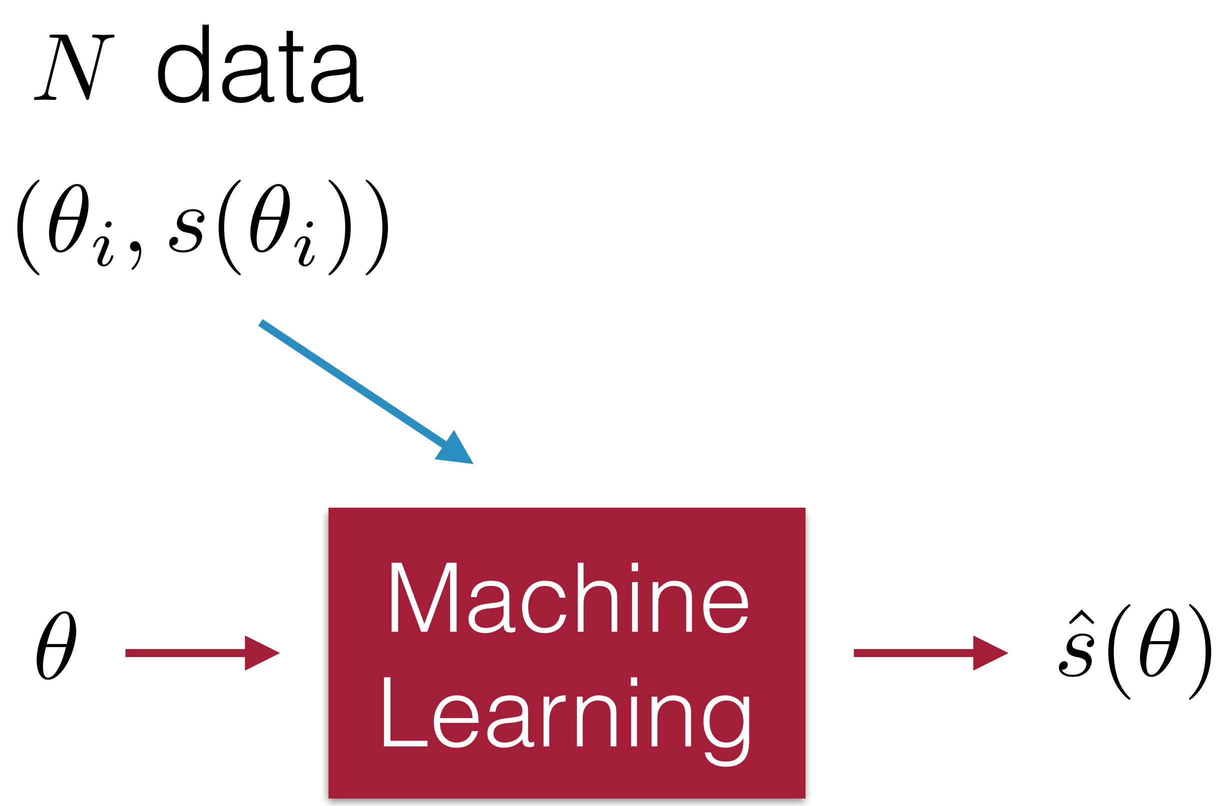
@b_stellato

Backup

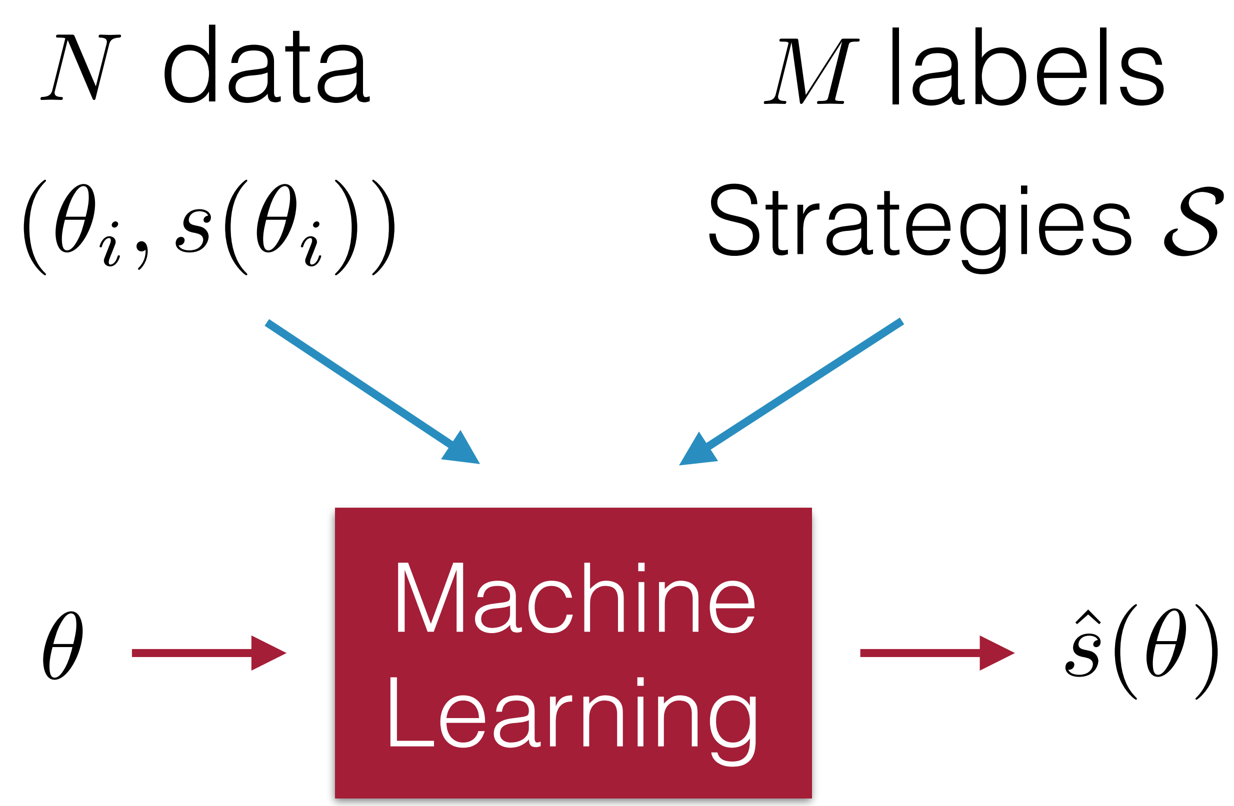
Classification problem

$\theta \longrightarrow$ **Machine Learning** $\longrightarrow \hat{s}(\theta)$

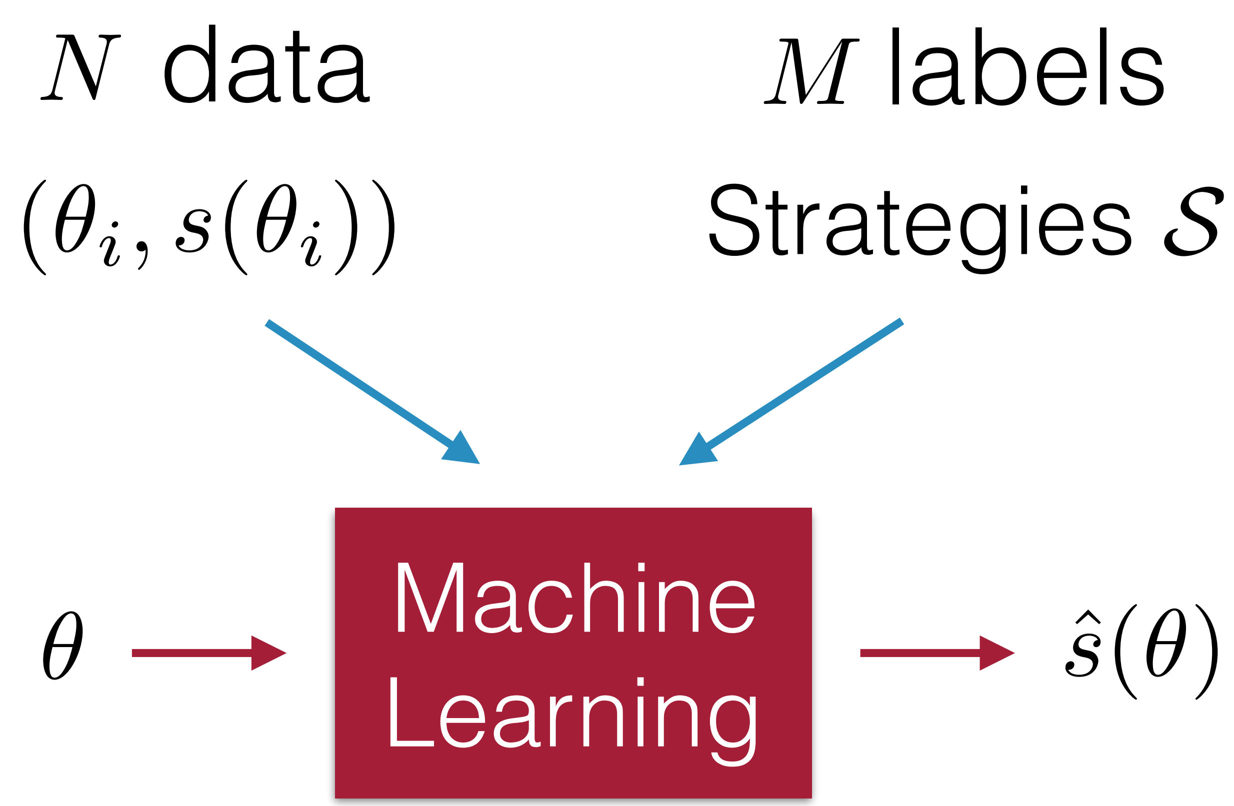
Classification problem



Classification problem

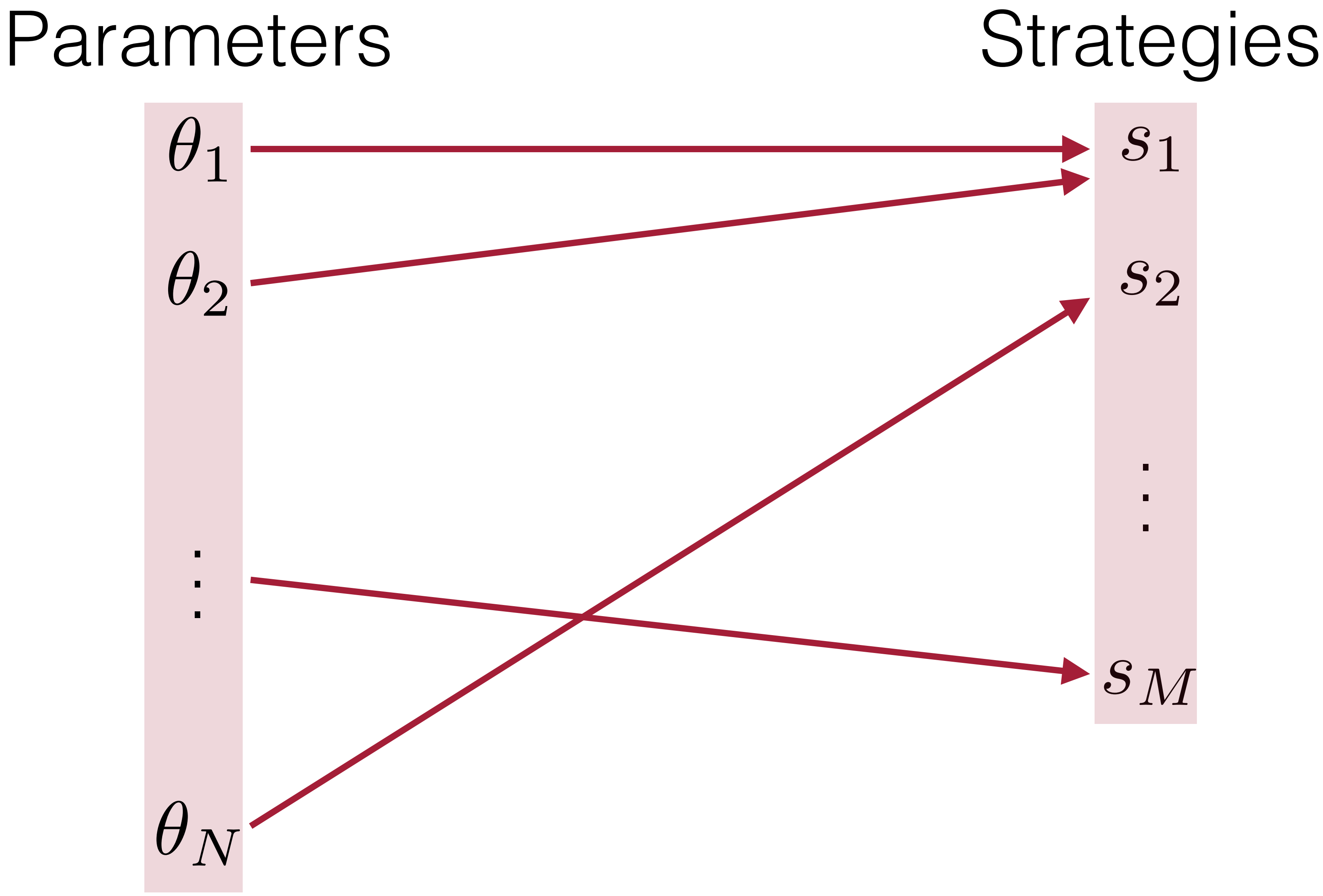


Classification problem

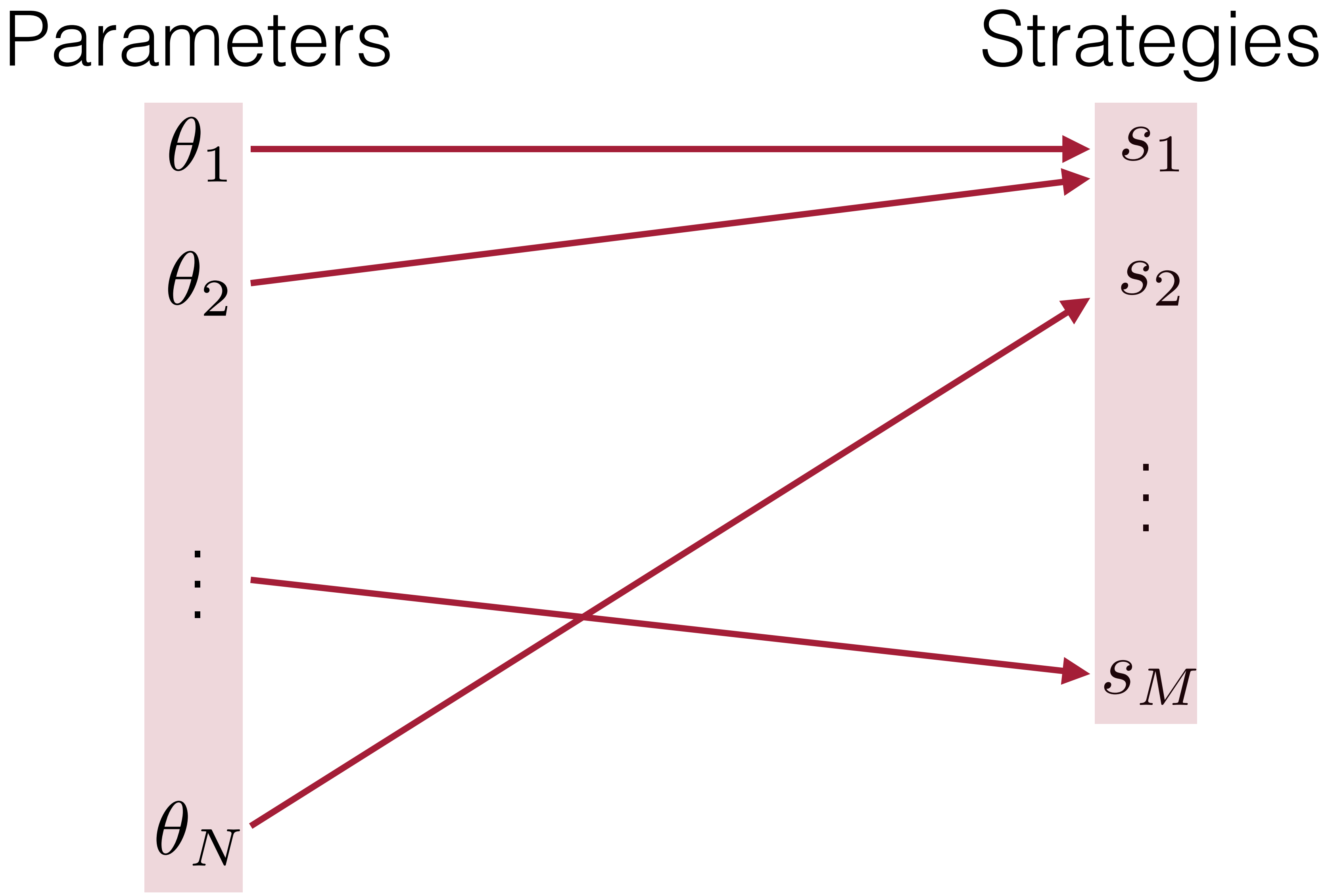


Have we seen enough data?

Will we find new strategies?



Will we find new strategies?



θ_{N+1}?

Alan Turing already knew that

Decoded the **Enigma Machine**
in
World War II

Only some words (labels) needed

Good-Turing estimator

s_1	12 times
s_2	45 times
$\vdots$	$\vdots$
s_M	2 times

Good-Turing estimator

s_1	12 times
s_2	45 times
$\vdots$	$\vdots$
s_M	2 times

Probability of unseen strategies

$$GT = \frac{N_1}{N} \approx \mathbf{P}(\theta_{N+1} \in \mathbf{R}^p \mid s(\theta_{N+1}) \notin \mathcal{S}(\Theta_N))$$

Good-Turing estimator

s_1	12 times
s_2	45 times
$\vdots$	$\vdots$
s_M	2 times

\# strategies appeared once

Probability of unseen strategies

$$GT = \frac{N_1}{N} \approx \mathbf{P}(\theta_{N+1} \in \mathbf{R}^p \mid s(\theta_{N+1}) \notin \mathcal{S}(\Theta_N))$$

Good-Turing estimator

s_1	12 times
s_2	45 times
$\vdots$	$\vdots$
s_M	2 times

$$GT = \frac{\overset{\text{\# strategies appeared once}}{N_1}}{\underset{\text{\# samples}}{N}} \approx \overbrace{\mathbf{P}(\theta_{N+1} \in \mathbf{R}^p \mid s(\theta_{N+1}) \notin \mathcal{S}(\Theta_N))}^{\text{Probability of unseen strategies}}$$

Good-Turing estimator

s_1	12 times
s_2	45 times
$\vdots$	$\vdots$
s_M	2 times

\# strategies appeared once

Probability of unseen strategies

$$GT = \frac{N_1}{N} \approx \mathbf{P}(\theta_{N+1} \in \mathbf{R}^p \mid s(\theta_{N+1}) \notin \mathcal{S}(\Theta_N))$$

\# samples

Bound

$$\mathbf{P}(\theta_{N+1} \in \mathbf{R}^p \mid s(\theta_{N+1}) \notin \mathcal{S}(\Theta_N)) \leq GT + c\sqrt{(1/N)\ln(3/\beta)}$$

Simple sampling scheme

Bound

Repeat until $GT + c\sqrt{(1/N)\ln(3/\beta)} \leq \epsilon$

1. sample θ_i
2. compute $s(\theta_i)$
3. update estimator $GT = \frac{N_1}{N}$

Sampling scheme

Algorithm 1 Strategies exploration

given $\epsilon, \beta, \Theta = \emptyset, \mathcal{S} = \emptyset, u = \infty$
for $k = 1, \ldots,$ **do**
  Sample θ_k and compute $s(\theta_k)$ ▷ Sample parameter and strategy.
  $\Theta \leftarrow \Theta \cup \{\theta_k\}$ ▷ Update set of samples.
  if $s(\theta_k) \notin \mathcal{S}$ **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{s(\theta_k)\}$ ▷ Update strategy set if new strategy found
  end if
  if $G + c\sqrt{(1/k)\ln(3/\beta)} \leq \epsilon$ **then** ▷ Break if bound less than ϵ
    break
  end if
end for
return $k, \Theta, \mathcal{S}$