Learning to Branch: Generalization Guarantees and Limits of Data-Independent Discretization

Maria-Florina Balcan[1]
Travis Dick[2]
Tuomas Sandholm[1,3,4,5]
Ellen Vitercik[6]

[1]Carnegie Mellon University, United States
[2]Google, United States
[3]Strategic Machine, Inc., United States
[4]Strategy Robot, Inc., United States
[5]Optimized Markets, Inc., United States
[6]Stanford University, United States

JACM, 2024

Presented by: Hao Su on June 10, 2026

Meet the Authors

Maria-Florina Balcan is the Cadence Design Systems Professor of Computer Science at Carnegie Mellon University, working on machine learning, theoretical computer science, and algorithmic game theory.

Travis Dick is a research scientist at Google and received his PhD in Computer Science from Carnegie Mellon University under the supervision of Maria-Florina Balcan.

Tuomas Sandholm is the Angel Jordan University Professor of Computer Science at Carnegie Mellon University, studying the intersection of artificial intelligence, economics, and operations research.

Ellen Vitercik is an Assistant Professor at Stanford University whose research spans machine learning theory, algorithm design, and the interface between economics and computation.

Outline

Overview

Main Contribution

The first generalization guarantee for data-driven tuning of variable-selection rules in branch-and-bound for IP.

We are going to check in this presentation

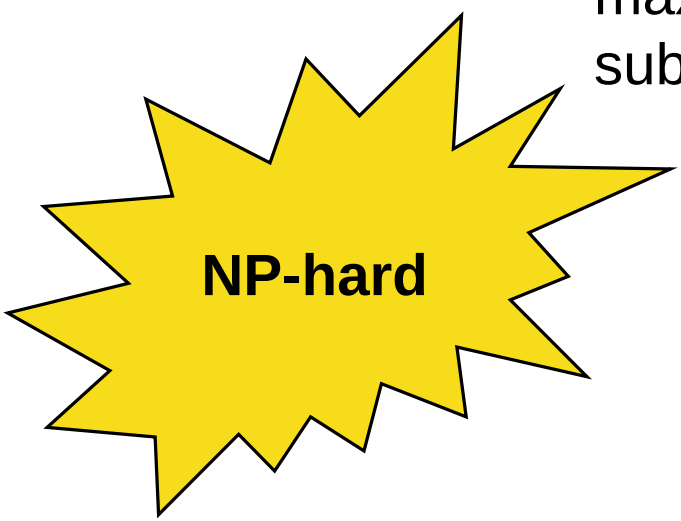
- What this sentence means.
- Why is this meaningful.
- Why is this hard & this paper's approach.
- Extensions & Experiments.

Integer Programs (IPs)

$$\begin{array}{ll} \text{maximize} & \boldsymbol{c} \cdot \boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \{0,1\}^n \end{array}$$

2

Integer Programs (IPs)

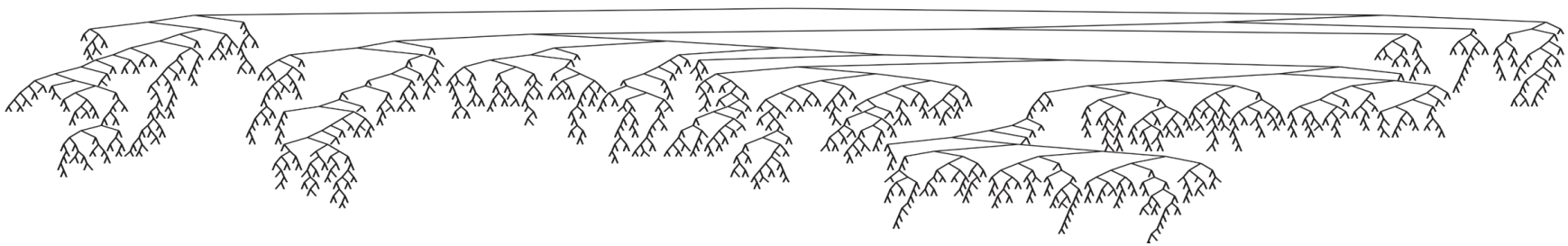


maximize $\boldsymbol{c} \cdot \boldsymbol{x}$
subject to $A\boldsymbol{x} = \boldsymbol{b}$
$\boldsymbol{x} \in \{0,1\}^n$

Branch and Bound (B&B)

- Most widely-used algorithm for IP-solving (CPLEX, Gurobi)
- Recursively partitions search space to find an optimal solution
  - Organizes partition as a tree
- **Many** parameters
  - CPLEX has a 221-page manual describing 135 parameters

    "You may need to experiment."



7

$$
\begin{array}{ll}
\max & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\
\text{s.t.} & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\
& \boldsymbol{x} \in \{0,1\}^7
\end{array}
$$

max	$(40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x}$
s.t.	$(40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100$
	$\boldsymbol{x} \in \{0,1\}^7$

$\left(\frac{1}{2}, 1, 0, 0, 0, 0, 1\right)$
140

17

B&B

1. Choose leaf of tree

$$\begin{array}{ll} \max & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{array}$$

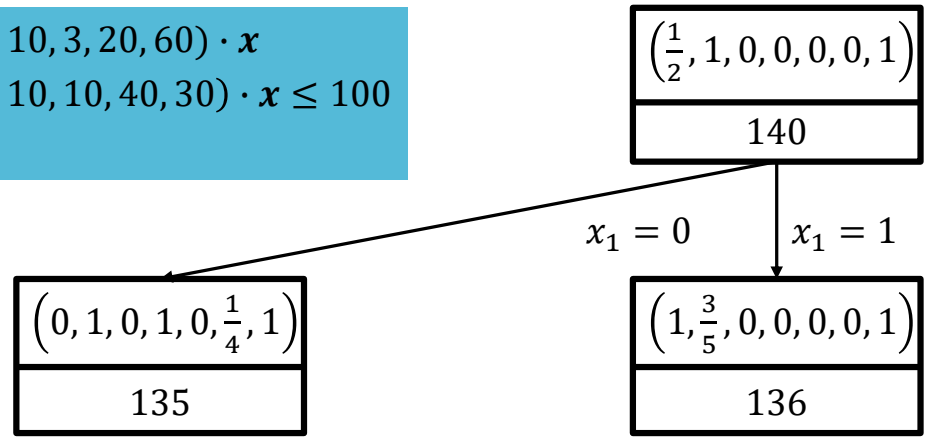
$\left(\frac{1}{2}, 1, 0, 0, 0, 0, 1\right)$
140

18

B&B

1. Choose leaf of tree
2. **Branch** on a variable

$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$

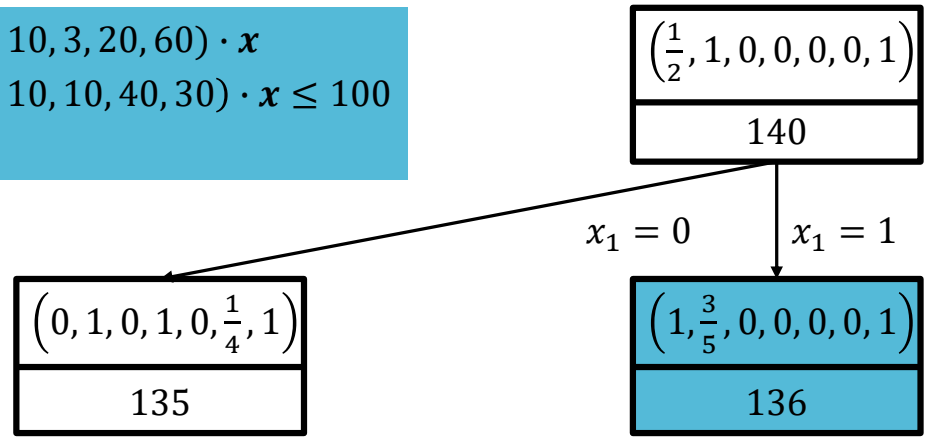


B&B

1. Choose leaf of tree

2. Branch on a variable

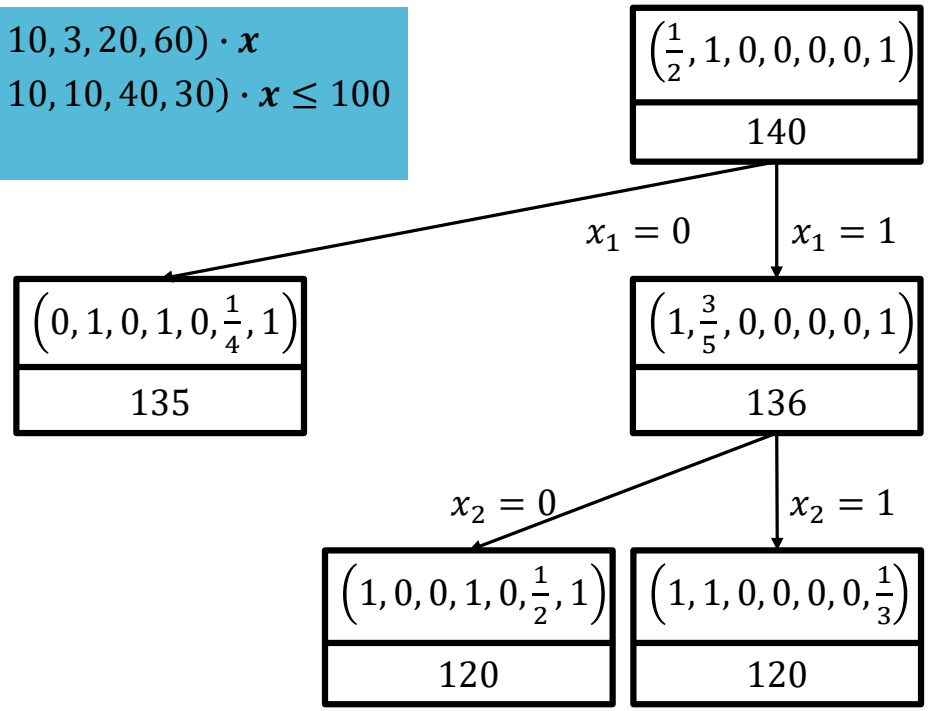
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. **Branch** on a variable

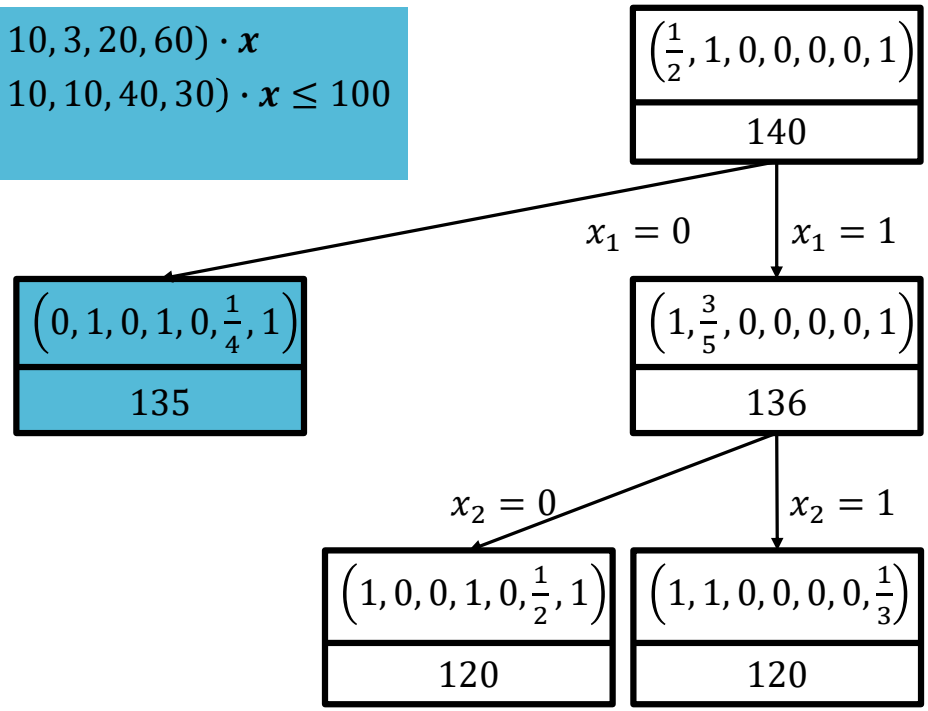
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. **Choose leaf of tree**
2. Branch on a variable

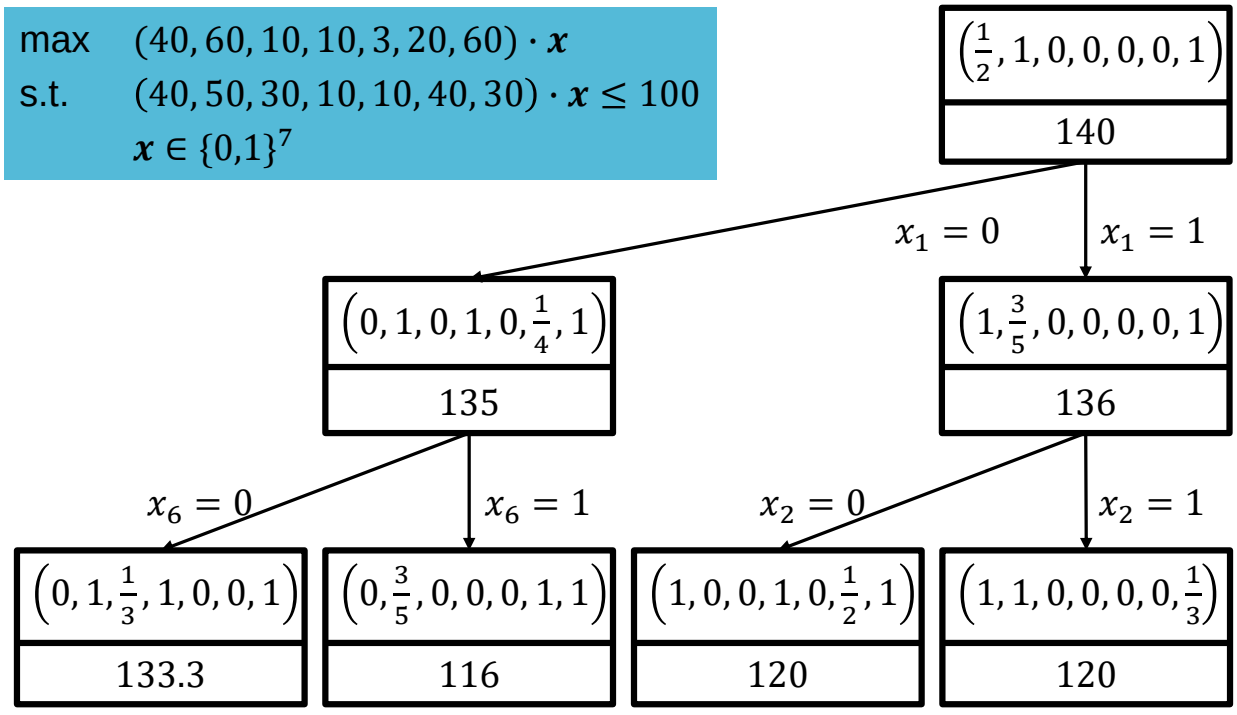
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. **Branch** on a variable

$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$

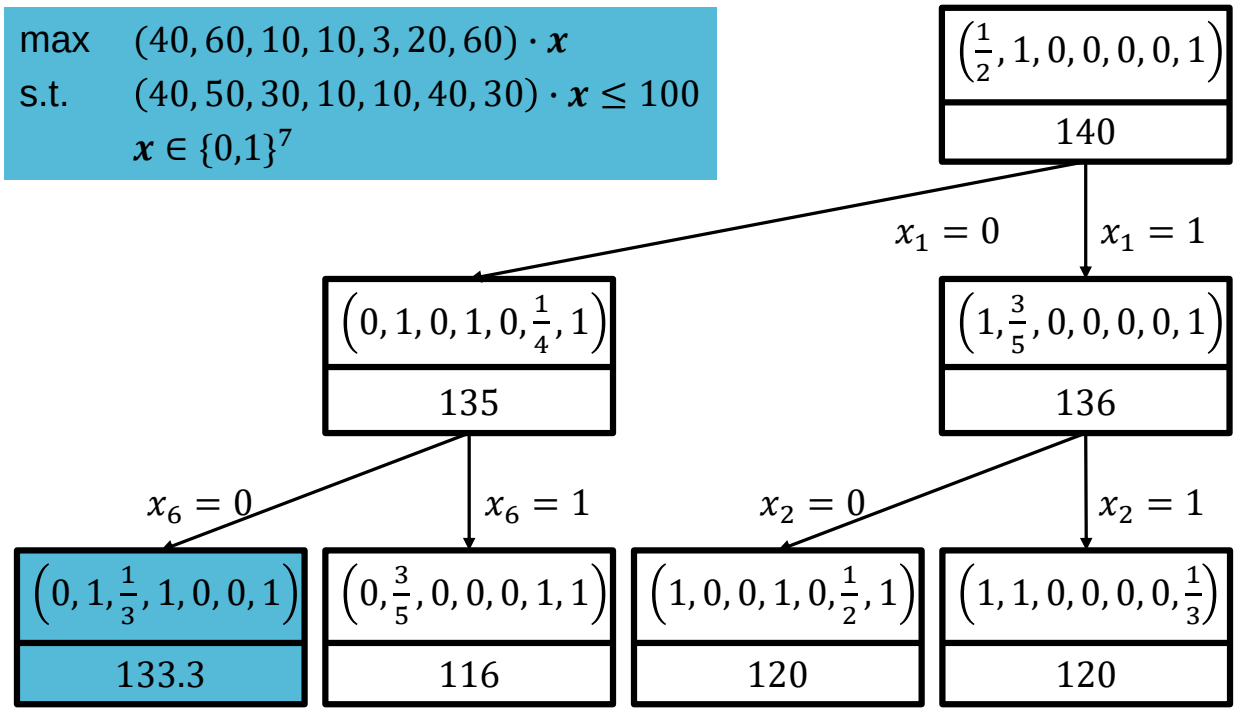


B&B

1. Choose leaf of tree

2. Branch on a variable

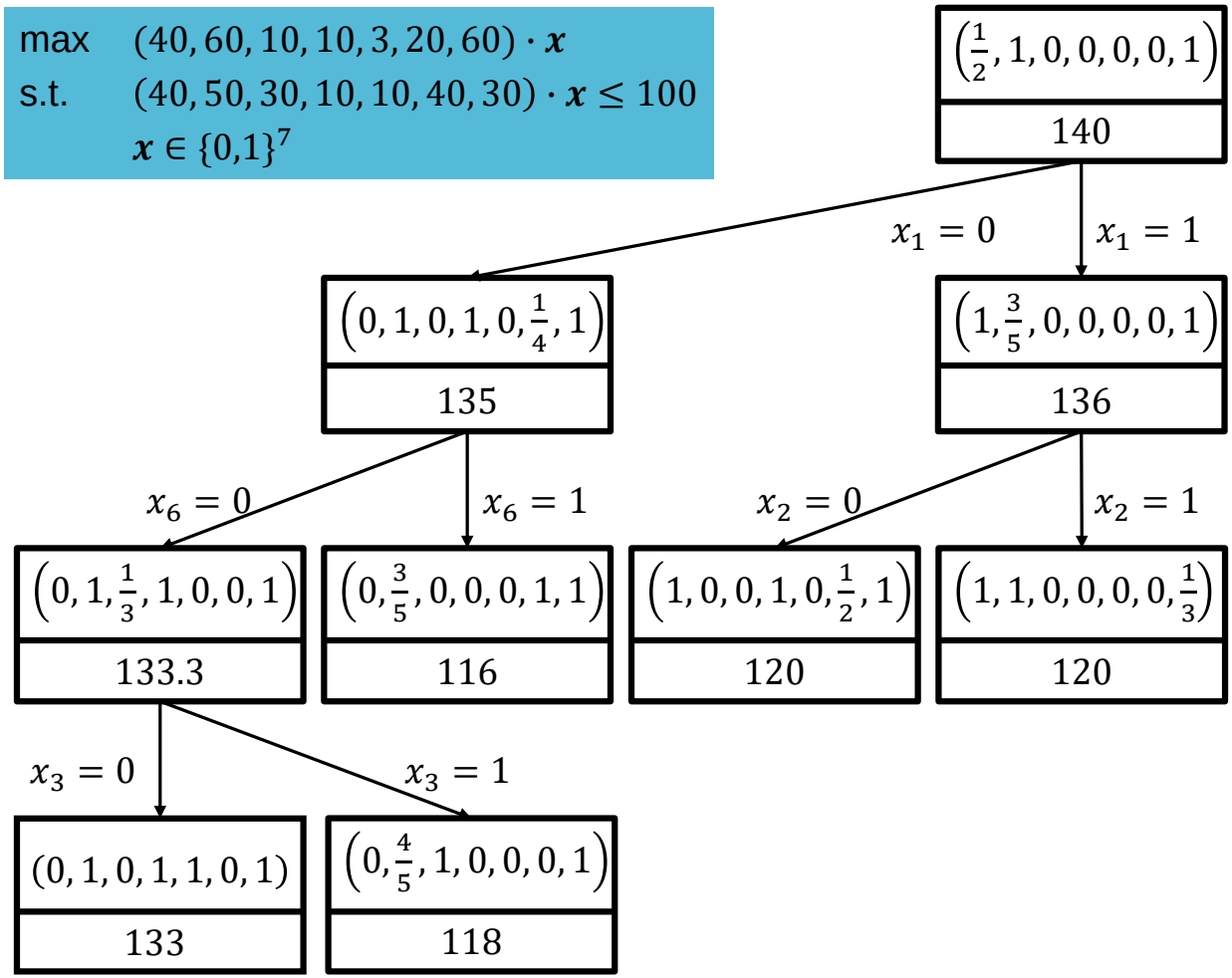
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. **Branch** on a variable

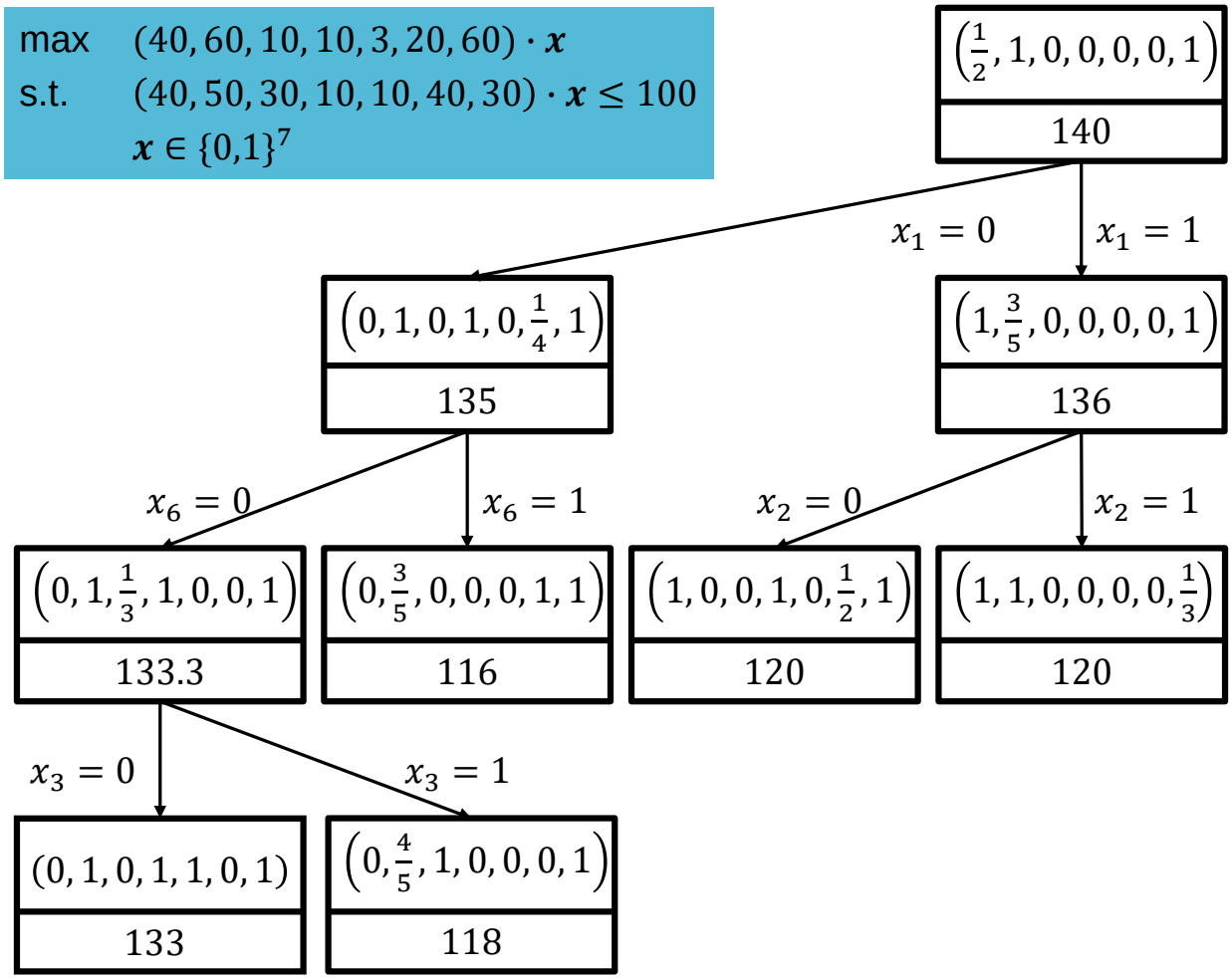
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. Branch on a variable
3. **Fathom** leaf if:
   i. LP relaxation solution is integral
   ii. LP relaxation is infeasible
   iii. LP relaxation solution isn't better than best-known integral solution

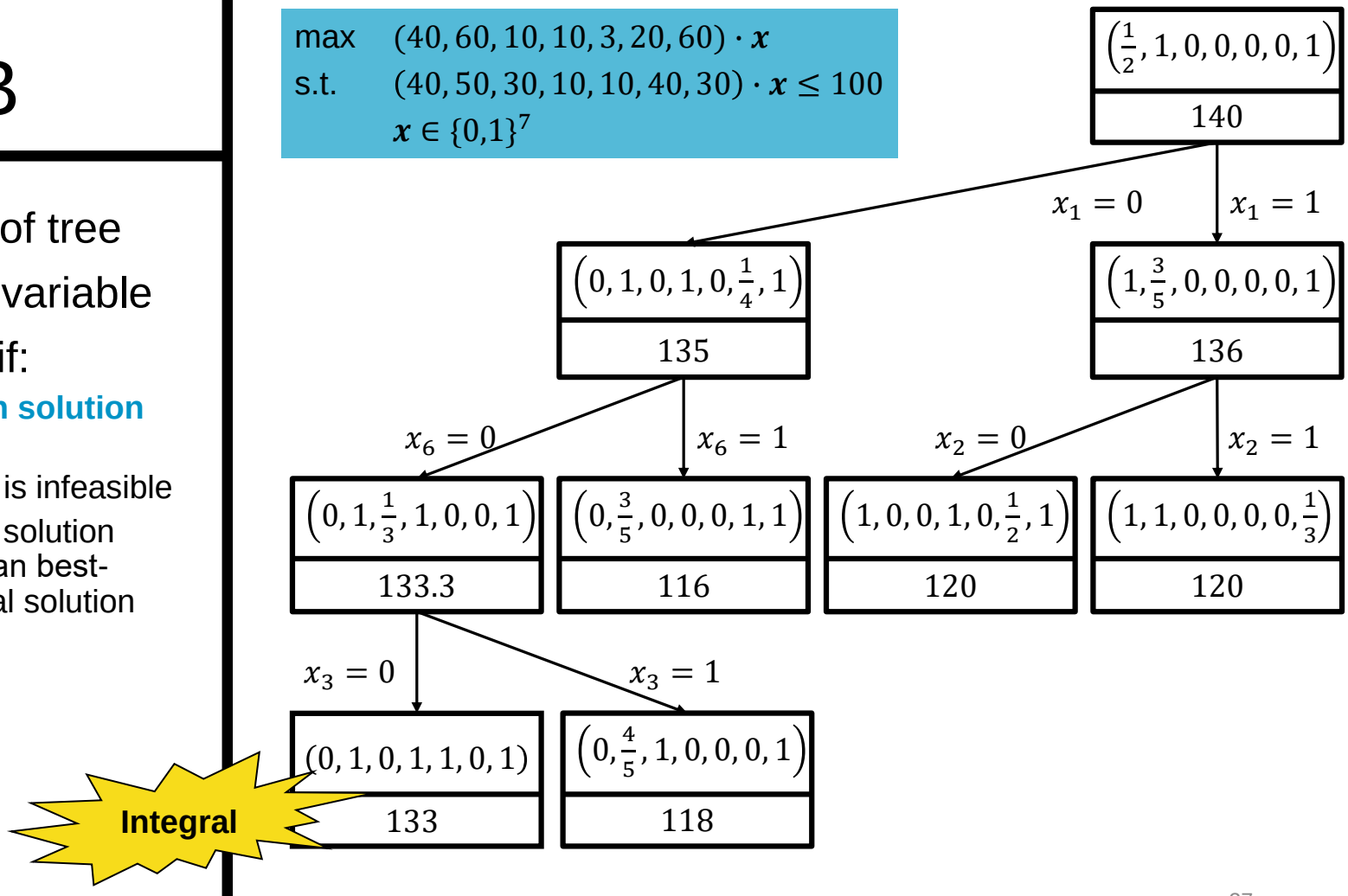
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. Branch on a variable
3. Fathom leaf if:
   i. **LP relaxation solution is integral**
   ii. LP relaxation is infeasible
   iii. LP relaxation solution isn't better than best-known integral solution

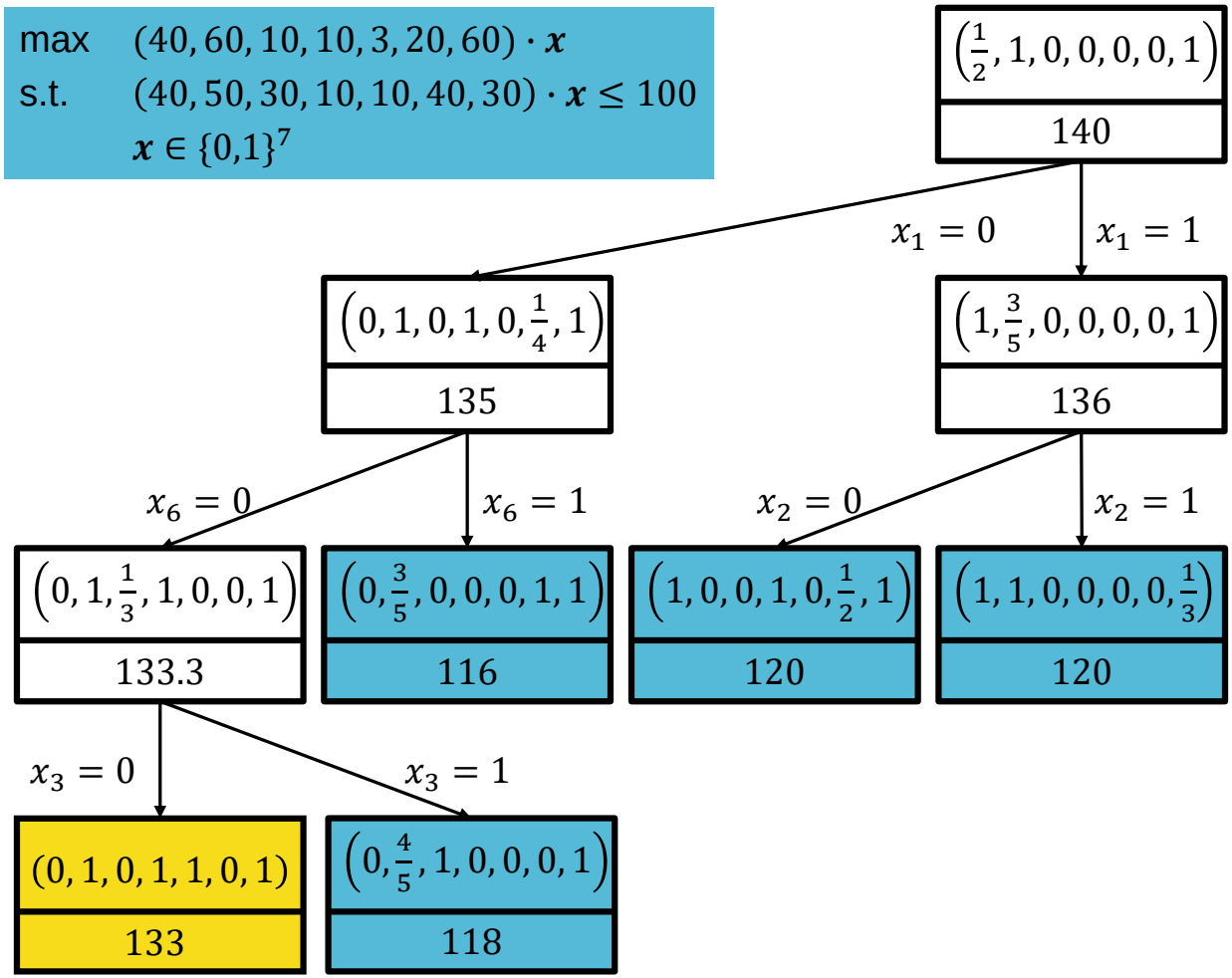
$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



B&B

1. Choose leaf of tree
2. Branch on a variable
3. Fathom leaf if:
   i. LP relaxation solution is integral
   ii. LP relaxation is infeasible
   iii. LP relaxation solution isn't better than best-known integral solution

$$\begin{aligned} \max \quad & (40, 60, 10, 10, 3, 20, 60) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & (40, 50, 30, 10, 10, 40, 30) \cdot \boldsymbol{x} \leq 100 \\ & \boldsymbol{x} \in \{0,1\}^7 \end{aligned}$$



28

B&B

1. Choose leaf of tree
2. Branch on a variable
3. Fathom leaf if:
   i. LP relaxation solution is integral
   ii. LP relaxation is infeasible
   iii. LP relaxation solution isn't better than best-known integral solution

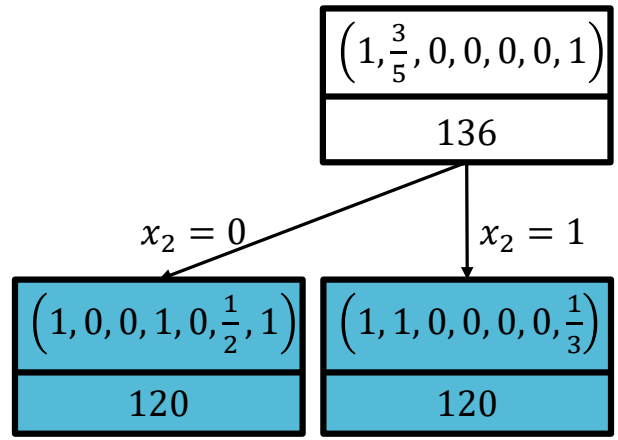
This talk: How to choose which variable?
(Assume every other aspect of B&B is fixed.)

29

Variable selection policies (VSPs)

Score-based VSP:

At leaf $\boldsymbol{Q}$, branch on variable $\boldsymbol{x_i}$ maximizing $\mathbf{score}(\boldsymbol{Q}, \boldsymbol{i})$



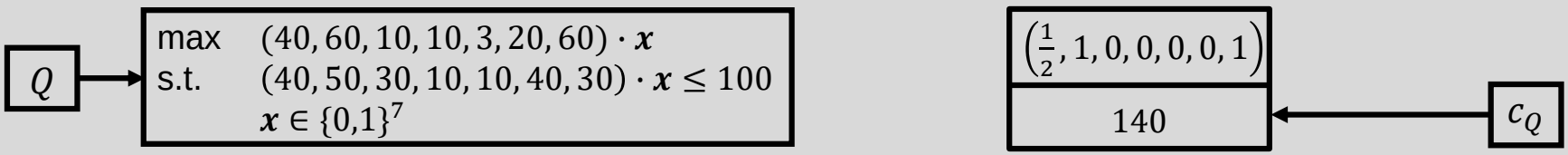
Many options! Little known about which to use when

Variable selection policies

For an IP instance Q:

- Let c_Q be the objective value of its LP relaxation

Example.



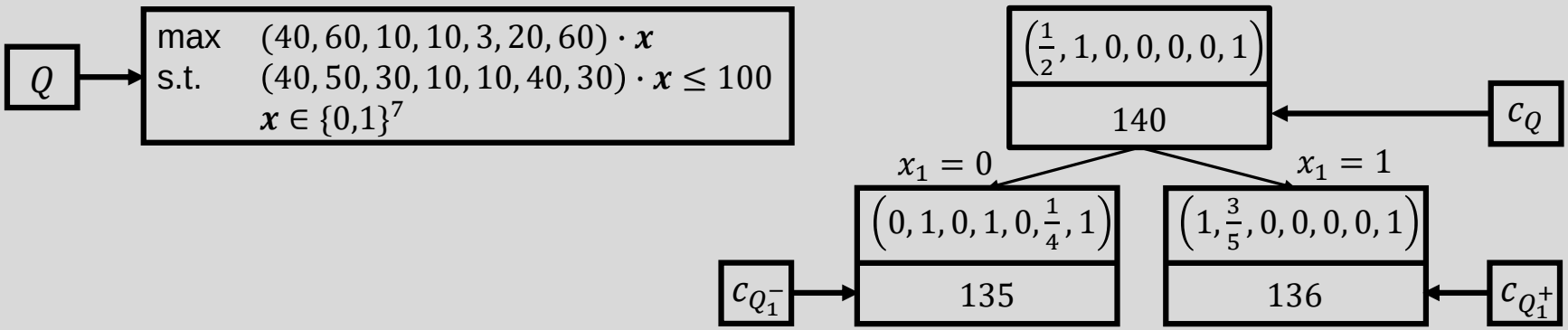
33

Variable selection policies

For an IP instance Q:

- Let c_Q be the objective value of its LP relaxation
- Let Q_i^- be Q with x_i set to 0, and let Q_i^+ be Q with x_i set to 1

Example.



Variable selection policies

The linear rule (parameterized by μ) [Linderoth & Savelsbergh, 1999]

Branch on variable x_i maximizing:

$$\text{score}(Q, i) = \mu \min\left\{c_Q - c_{Q_i^-}, c_Q - c_{Q_i^+}\right\} + (1-\mu)\max\left\{c_Q - c_{Q_i^-}, c_Q - c_{Q_i^+}\right\}$$

The (simplified) **product rule** [Achterberg, 2009]

Branch on variable x_i maximizing:

$$\text{score}(Q, i) = \left(c_Q - c_{Q_i^-}\right) \cdot \left(c_Q - c_{Q_i^+}\right)$$

And many more…

Variable selection policies

Given d scoring rules $\text{score}_1, \ldots, \text{score}_\text{d}$.

Goal: Learn best convex combination $\mu_1 \text{score}_1 + \cdots + \mu_d \text{score}_\text{d}$.

Our parameterized rule
Branch on variable x_i maximizing: $$\text{score}(Q, i) = \mu_1 \text{score}_1(Q, i) + \cdots + \mu_d \text{score}_\text{d}(Q, i)$$

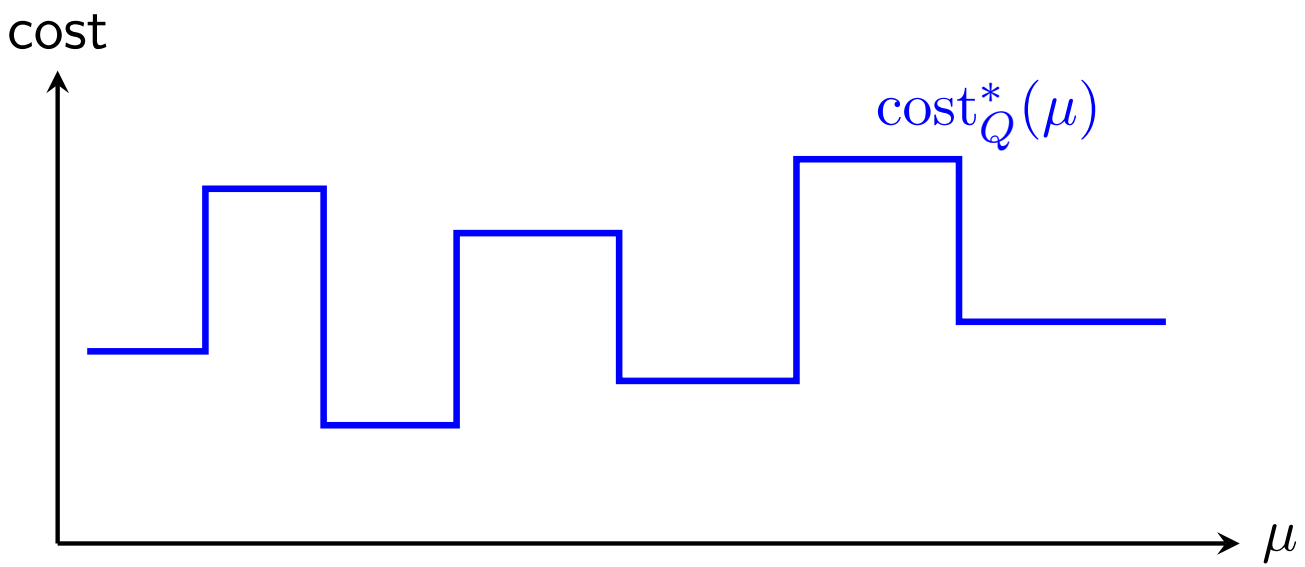
37

Generalization needs low complexity

Uniformly over all branching parameters μ,

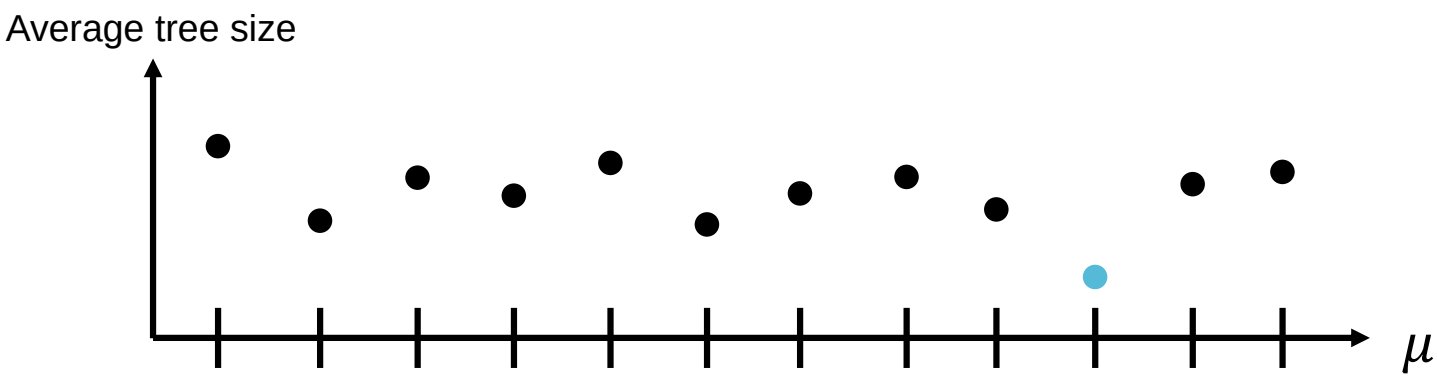
$$\left| \frac{1}{m} \sum_{i=1}^{m} \mathrm{cost}_\mu(Q_i) - \mathbb{E}_{Q \sim \mathcal{D}}[\mathrm{cost}_\mu(Q)] \right| \leq \varepsilon(m, \delta).$$

$$\varepsilon \approx \underbrace{\mathrm{complexity}(\mathcal{C})}_{\text{ability to overfit samples}} + \underbrace{\sqrt{\frac{\log(1/\delta)}{m}}}_{\text{concentration}}, \mathcal{C} = \{\mathrm{cost}_\mu : \mu \in [0, 1]\}.$$



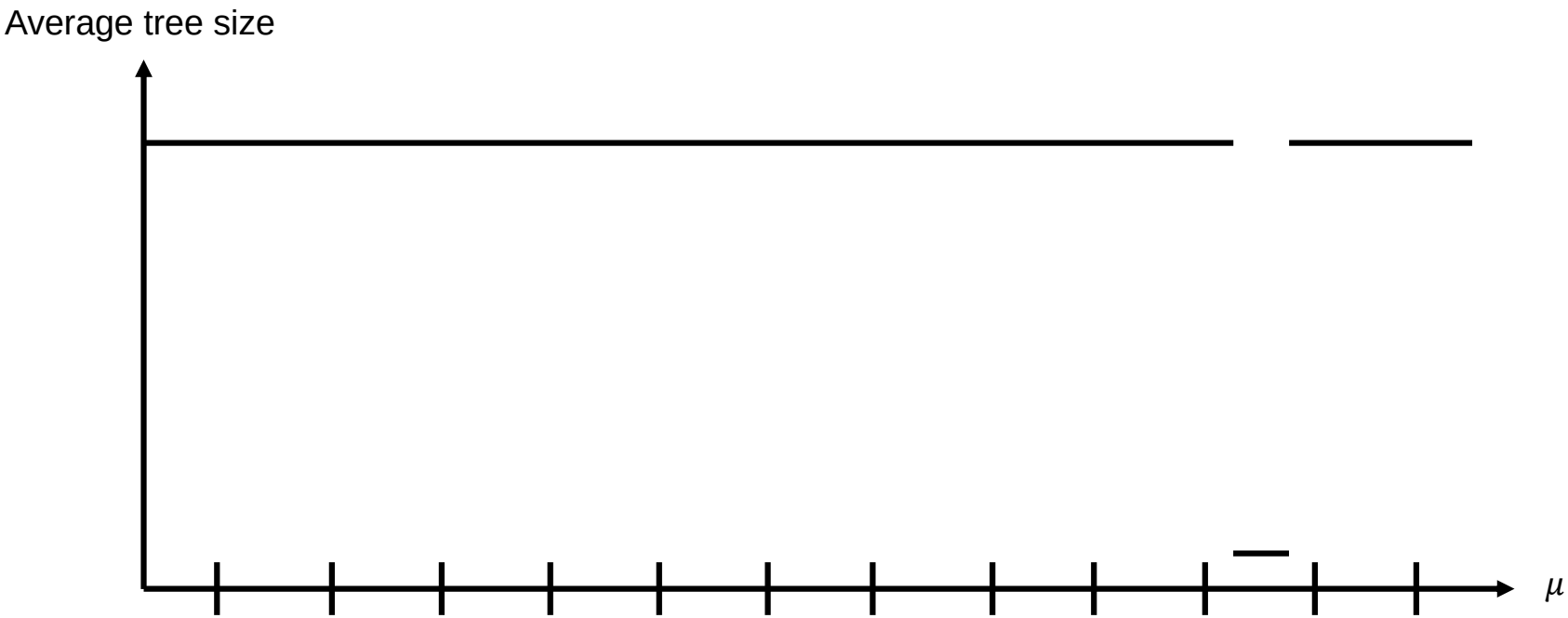
First try: Discretization

1. Discretize parameter space
2. Receive sample problems from unknown distribution
3. Find params in discretization with best average performance



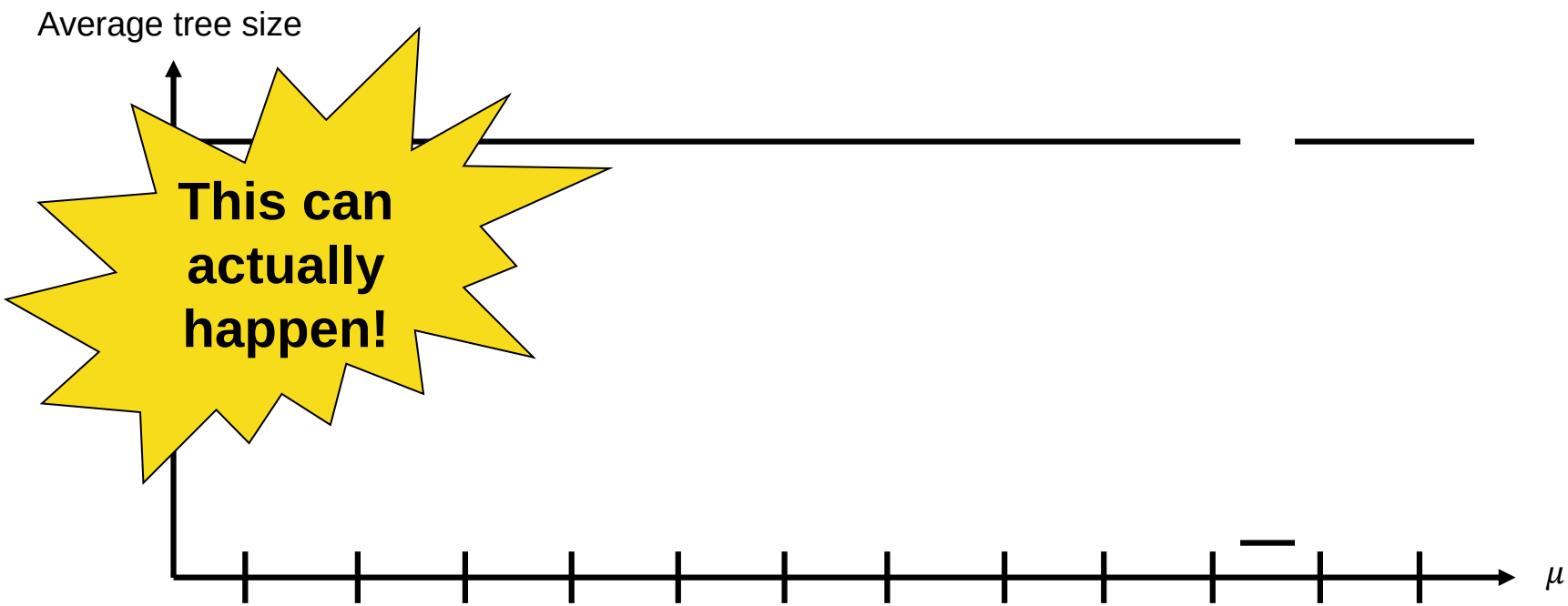
41

Discretization gone wrong



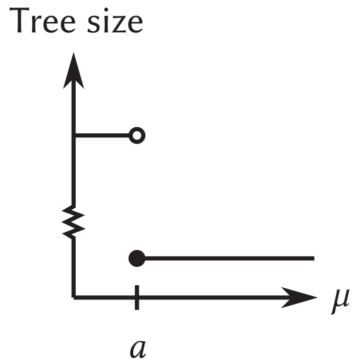
43

Discretization gone wrong

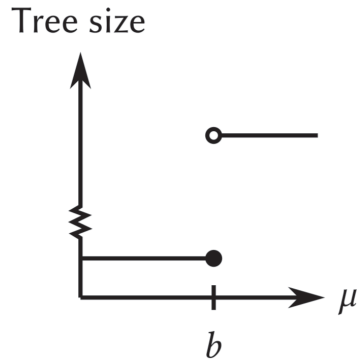


44

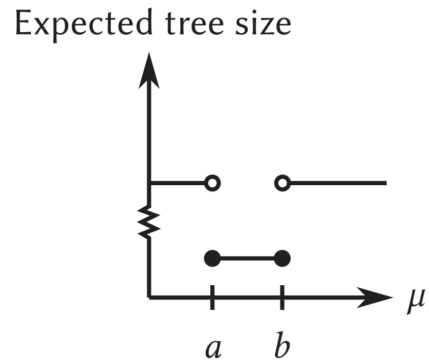
Proof Intuition: Discontinuity



(a) The tree size plot for the instance Q_a as a function of μ.

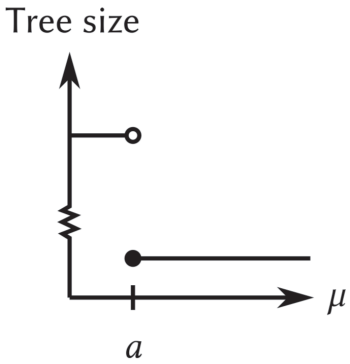


(b) The tree size plot for the instance Q_b as a function of μ.

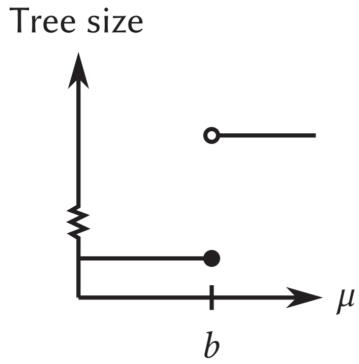


(c) The expected tree size plot under the distribution $\mathcal{D}$ as a function of μ.

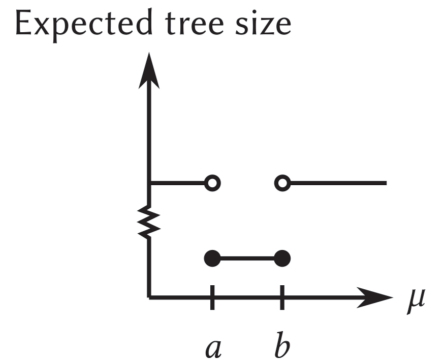
Proof Intuition: Discontinuity



(a) The tree size plot for the instance Q_a as a function of μ.



(b) The tree size plot for the instance Q_b as a function of μ.



(c) The expected tree size plot under the distribution $\mathcal{D}$ as a function of μ.

$$\begin{aligned} \max \quad & \left(1, 2, 3, 4, 5, 0, \tfrac{3}{2}, 3 - \tfrac{1}{2\mu^*}\right) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & \begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} 5 \\ 3 \end{pmatrix} \\ & \boldsymbol{x} \in \{0,1\}^8. \end{aligned}$$

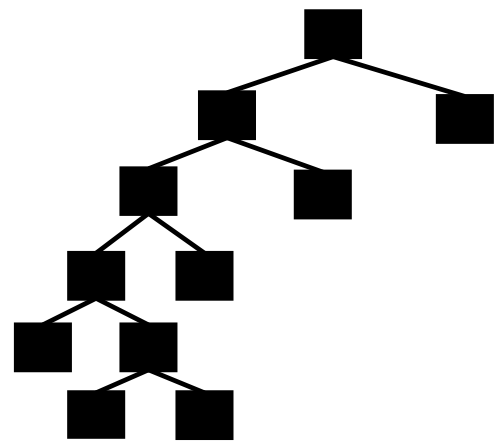
$$\begin{aligned} \max \quad & \left(1, 2, 3, 4, 5, 0, \tfrac{3}{2}, 3 - \tfrac{1}{2\mu^*}\right) \cdot \boldsymbol{x} \\ \text{s.t.} \quad & \begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} 5 \\ 3 \end{pmatrix} \\ & \boldsymbol{x} \in \{0,1\}^8. \end{aligned}$$

Simple assumption

Exists κ upper bounding the size of largest tree willing to build

Common assumption, e.g.:

- Hutter, Hoos, Leyton-Brown, Stützle, JAIR'09
- Kleinberg, Leyton-Brown, Lucier, IJCAI'17



47

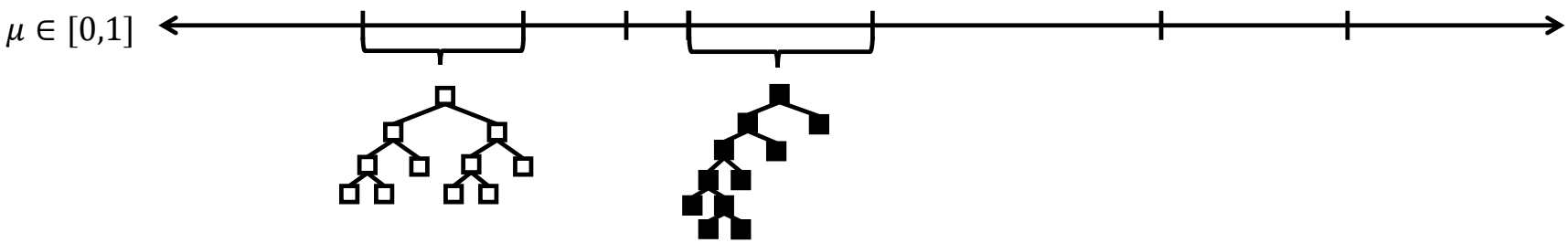
Useful lemma

Much smaller in our experiments!

Lemma: For any two scoring rules and any IP Q,

$O\left((\#\text{ variables})^{\kappa+2}\right)$ intervals partition $[0,1]$ such that:

For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$

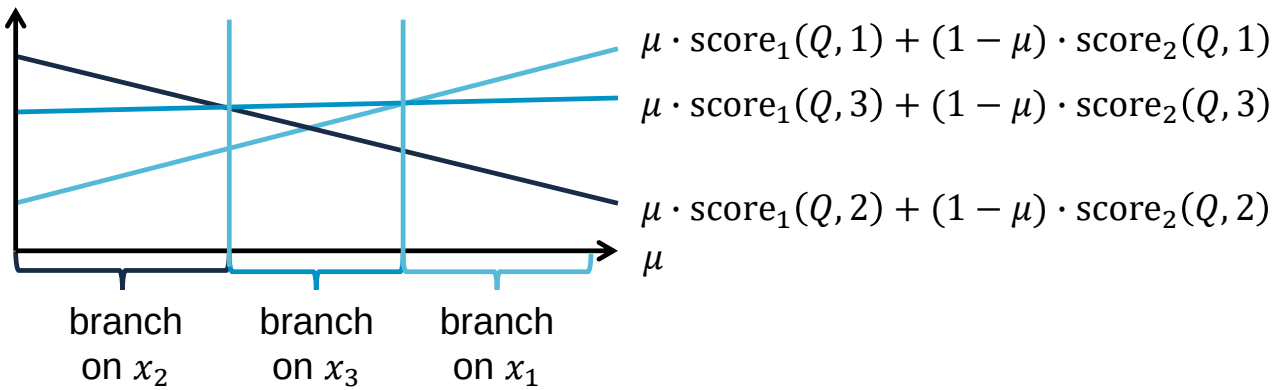


Useful lemma

Lemma: For any two scoring rules and any IP Q,

$O\big((\#\text{ variables})^{\kappa+2}\big)$ intervals partition $[0,1]$ such that:

For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$

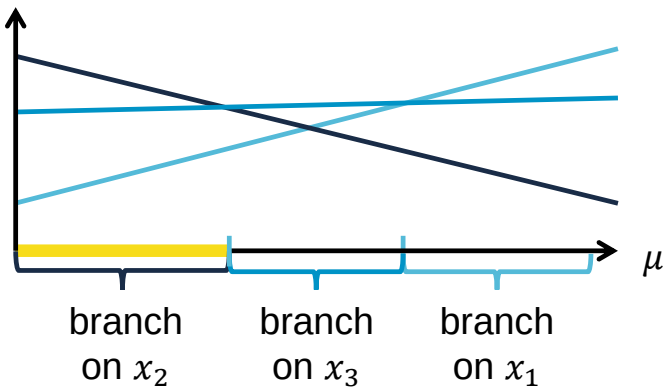


Useful lemma

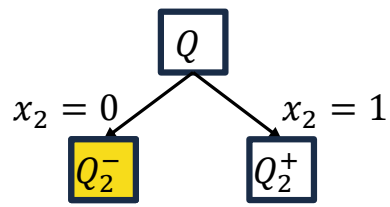
Lemma: For any two scoring rules and any IP Q,

$O\big((\#\text{ variables})^{\kappa+2}\big)$ intervals partition $[0,1]$ such that:

For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$



Any μ in yellow interval:

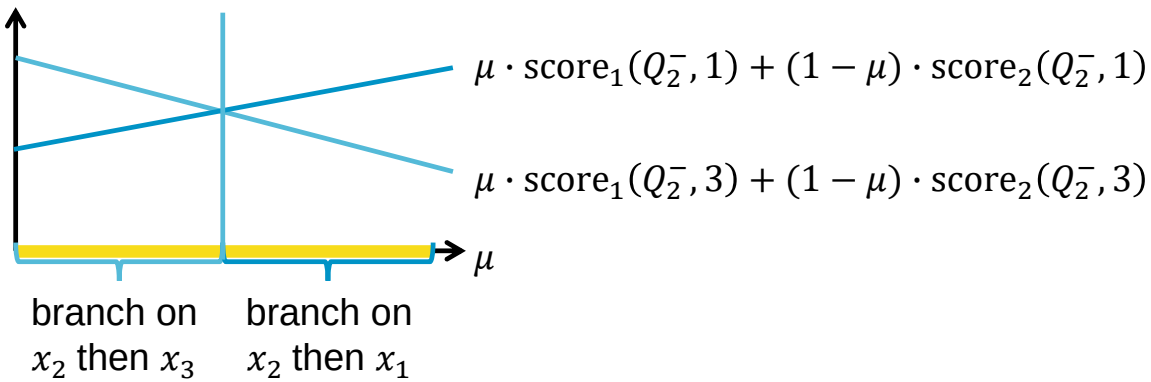


Useful lemma

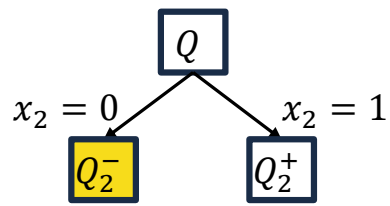
Lemma: For any two scoring rules and any IP Q,

$O\left((\#\text{ variables})^{\kappa+2}\right)$ intervals partition $[0,1]$ such that:

For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$



Any μ in yellow interval:

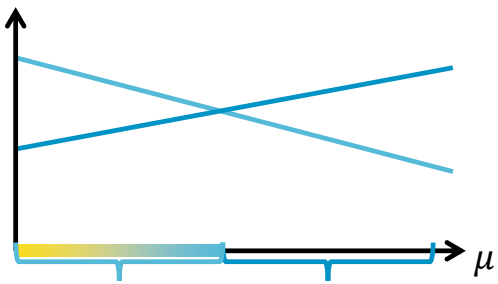


Useful lemma

Lemma: For any two scoring rules and any IP Q,

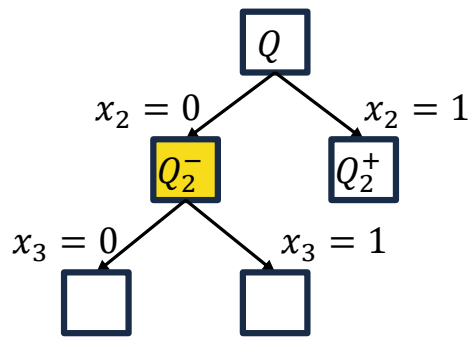
$O\big((\#\text{ variables})^{\kappa+2}\big)$ intervals partition $[0,1]$ such that:

For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$



branch on x_2 then x_3 | branch on x_2 then x_1

Any μ in blue-yellow interval:

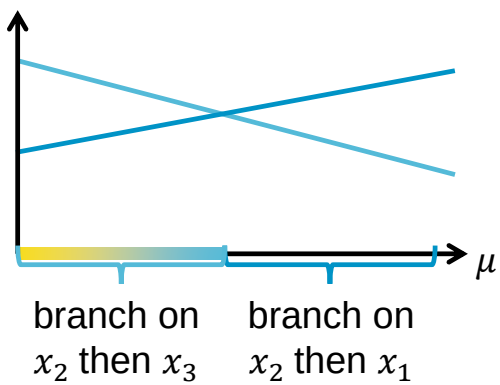


Useful lemma

Lemma: For any two scoring rules and any IP Q,

$O\big((\#\text{ variables})^{\kappa+2}\big)$ intervals partition $[0,1]$ such that:

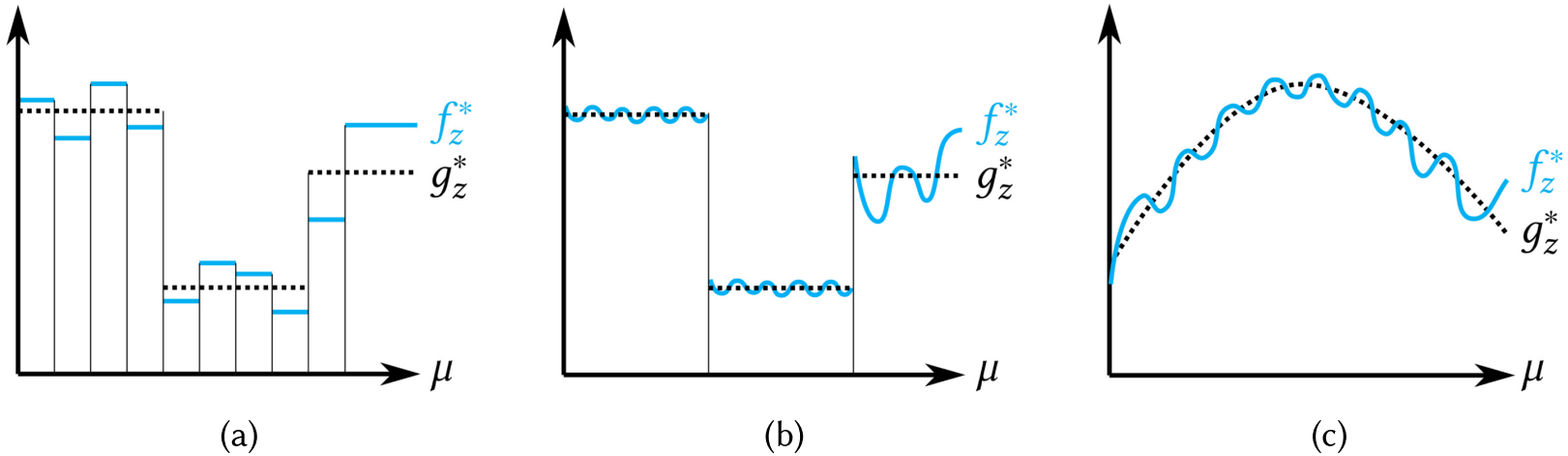
For any interval $[a,b]$, B&B builds same tree across all $\mu \in [a,b]$



Proof idea.

- Continue dividing [0,1] into intervals s.t.: In each interval, var. selection order fixed
- Can subdivide only finite number of times
- Proof follows by induction on tree depth

Data-dependent bounds: approximation vs. complexity



Trade-off

The data-dependent bound has the form

$$\text{generalization error} \lesssim \underbrace{\|F - G\|_\infty}_{\text{approximation error}} + \underbrace{\text{complexity}(\mathcal{G}_j)}_{\text{estimation error}}.$$

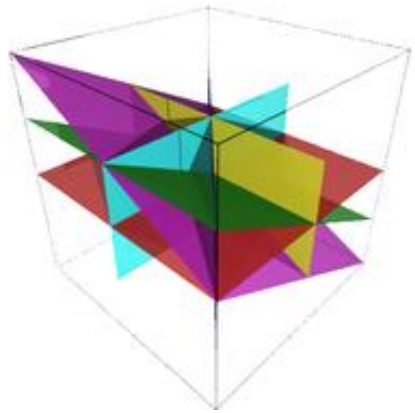
Useful lemma: higher dimensions

Lemma: For any d scoring rules and any IP,

a set $\mathcal{H}$ of $O\left((\#\text{ variables})^{\kappa+2}\right)$ hyperplanes partitions $[0,1]^d$ s.t.:

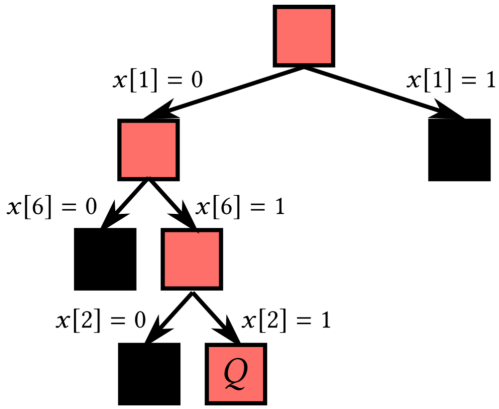
For any connected component R of $[0,1]^d \setminus \mathcal{H}$,

B&B builds the same tree across all $\boldsymbol{\mu} \in R$

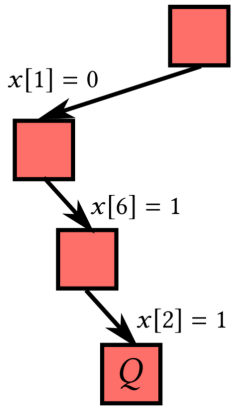


57

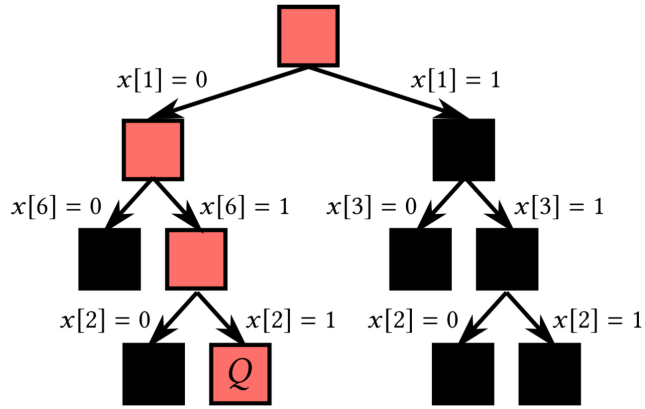
Other Extentions



(a) A B&B search tree $\mathcal{T}$.



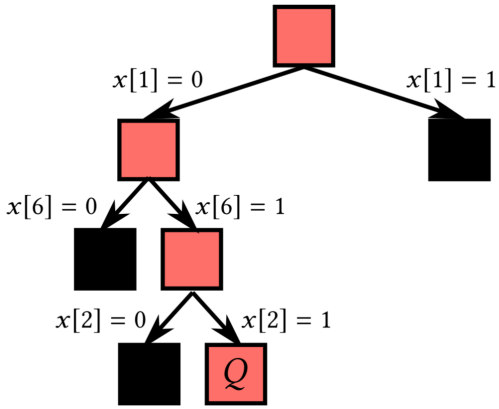
(b) The path $\mathcal{T}_Q$ from the root of the tree $\mathcal{T}$ in Figure 2a to the node labeled Q.



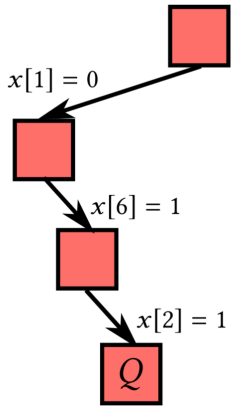
(c) Another tree $\mathcal{T}'$ that has the path $\mathcal{T}_Q$ as a rooted subtree.

Path-Wise Scoring Rule for better bounds (by n instead of κ).

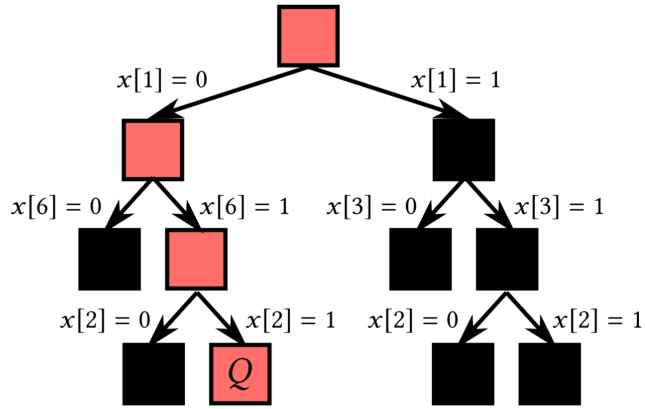
Other Extentions



(a) A B&B search tree $\mathcal{T}$.



(b) The path $\mathcal{T}_Q$ from the root of the tree $\mathcal{T}$ in Figure 2a to the node labeled Q.

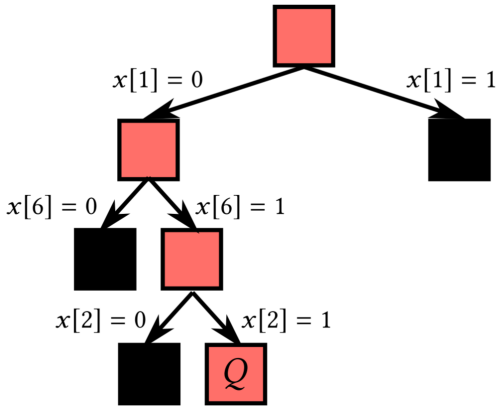


(c) Another tree $\mathcal{T}'$ that has the path $\mathcal{T}_Q$ as a rooted subtree.

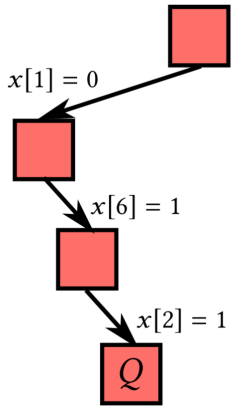
Path-Wise Scoring Rule for better bounds (by n instead of κ).

Taking log to reduce nonlinear combination of scoring rules of the form $\Pi_{i=1}^{d} \mathrm{score}_i^{\mu_i}$ to convex combinations.

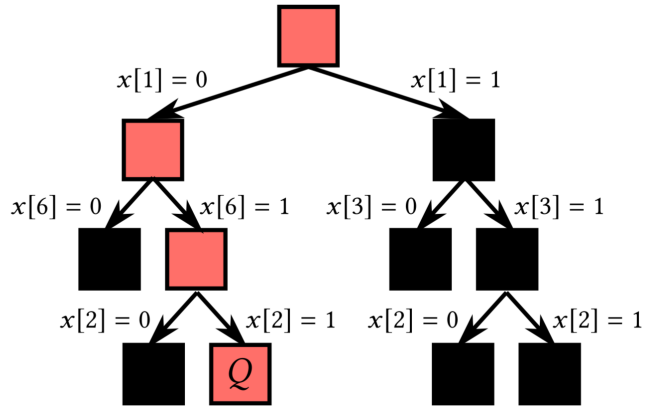
Other Extentions



(a) A B&B search tree $\mathcal{T}$.



(b) The path $\mathcal{T}_Q$ from the root of the tree $\mathcal{T}$ in Figure 2a to the node labeled Q.



(c) Another tree $\mathcal{T}'$ that has the path $\mathcal{T}_Q$ as a rooted subtree.

Path-Wise Scoring Rule for better bounds (by n instead of κ).

Taking log to reduce nonlinear combination of scoring rules of the form $\Pi_{i=1}^{d} \text{score}_i^{\mu_i}$ to convex combinations.

Constraint Satisfaction Problems.

Experiments: What is being demonstrated?

The authors tune the branching score parameter μ and measure the resulting B&B tree size on different problem distributions.

- **Main empirical observation.** The optimal μ varies substantially across domains, so there is no single universally good branching parameter.
- **Data-dependent bounds.** They also compare the worst-case generalization bound with the data-dependent bound from Section 3.3. The data-dependent bound is often much smaller, suggesting that the observed dual cost functions have low effective complexity.

Takeaways

- What IP and B&B are. Why learning is needed.
- Why discretization failed to give generalization.
- How the structural traits led to generalization.
- Data-dependent generalization bounds.
- What kind of experiments can be carried out.

Recommended to check: Rademacher Complexity.

Takeaways

- What IP and B&B are. Why learning is needed.
- Why discretization failed to give generalization.
- How the structural traits led to generalization.
- Data-dependent generalization bounds.
- What kind of experiments can be carried out.

Recommended to check: Rademacher Complexity.

Questions?