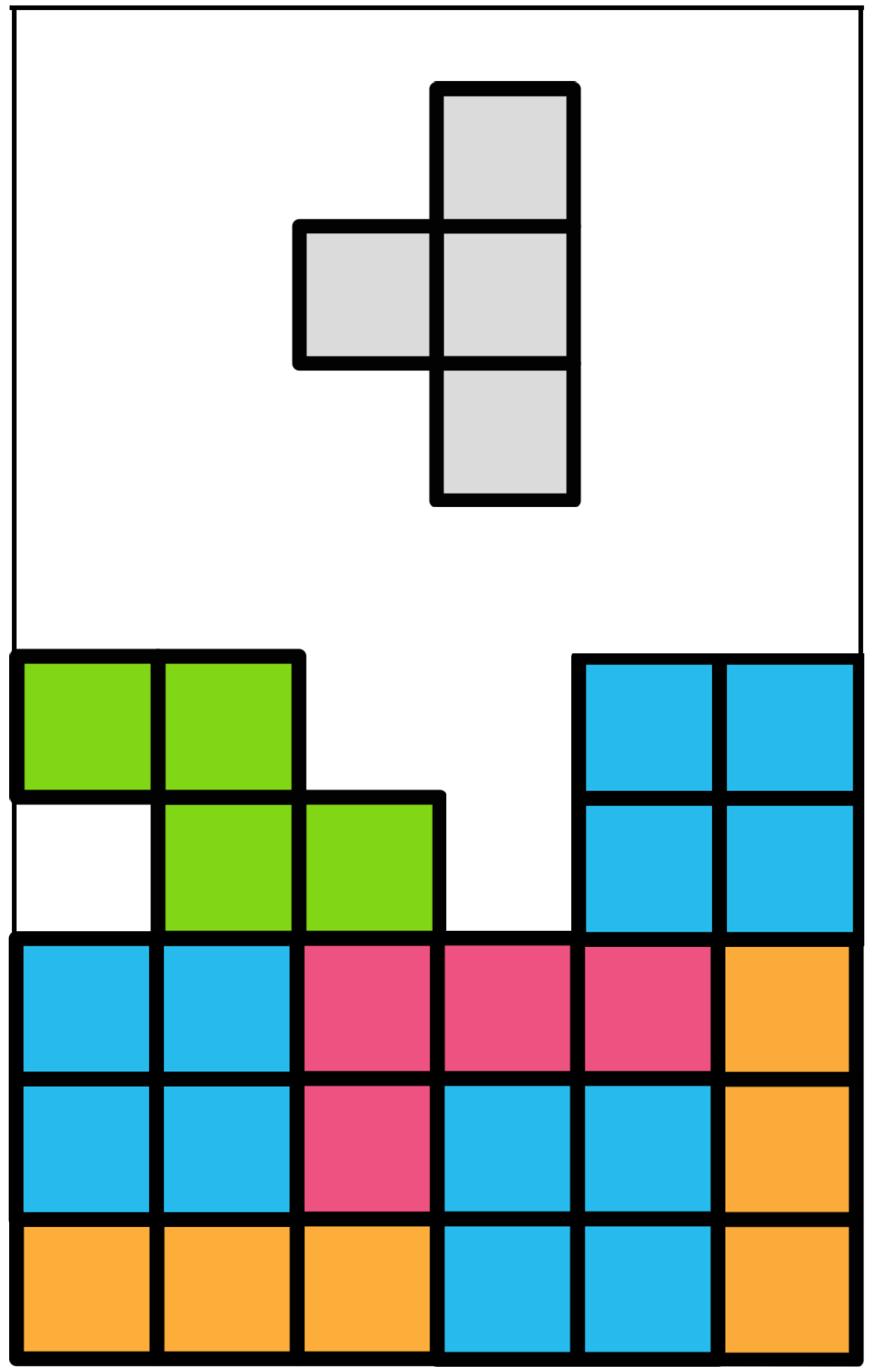
Algorithms Under Uncertainty: Robust Methods and Decision-Theoretic Evaluation

PhD Defense Presentation

Georgii Melidi
September 25, 2025

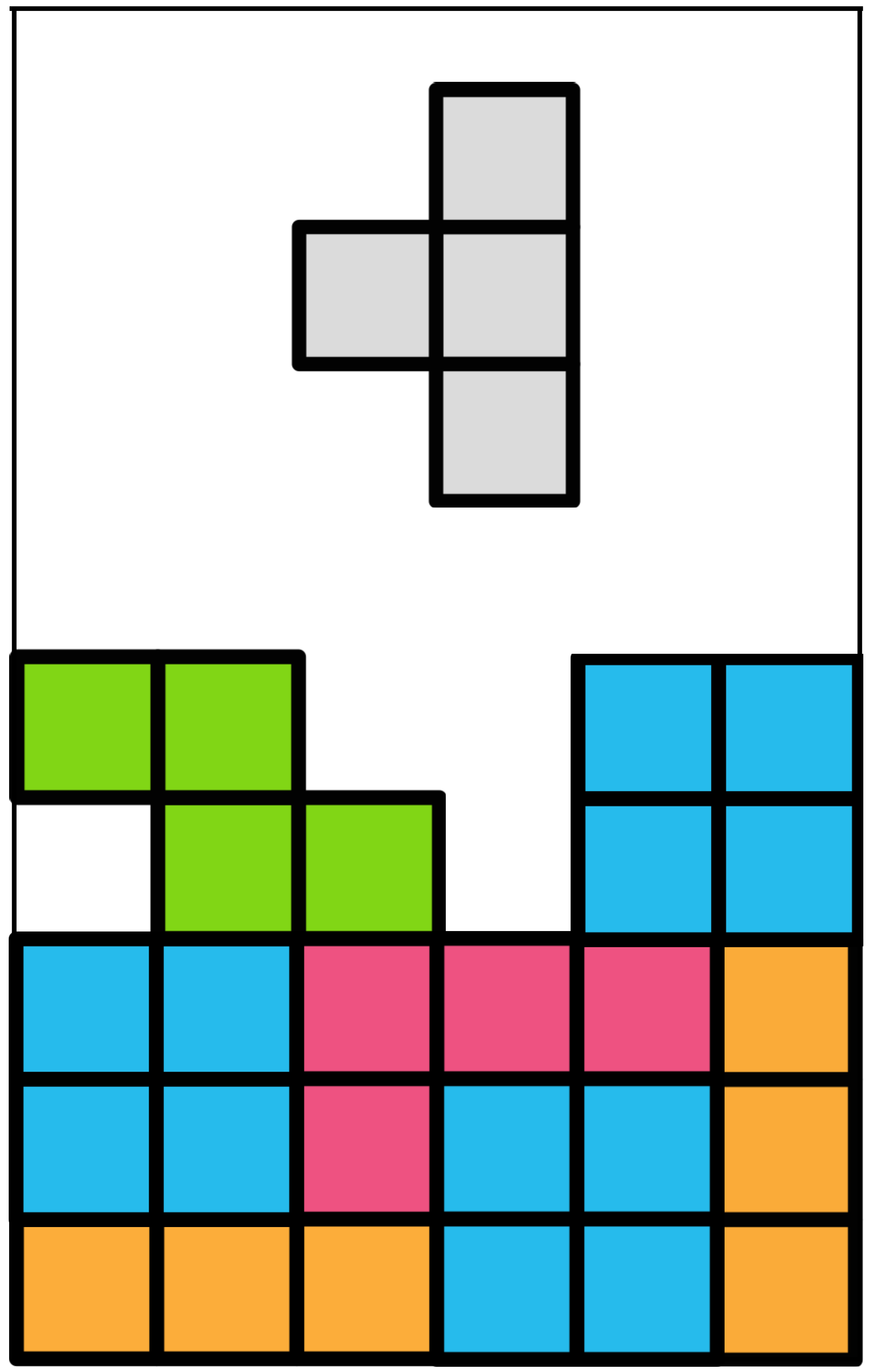
Supervisor: **Spyros Angelopoulos**
Co-Supervisor: **Christoph Dürr**

Optimization under uncertainty

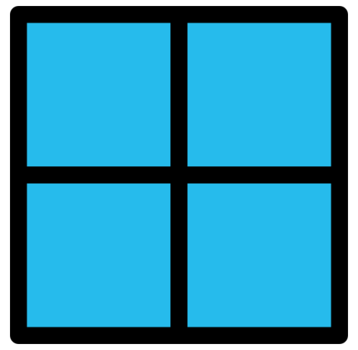


pieces arrive
online

Optimization under uncertainty



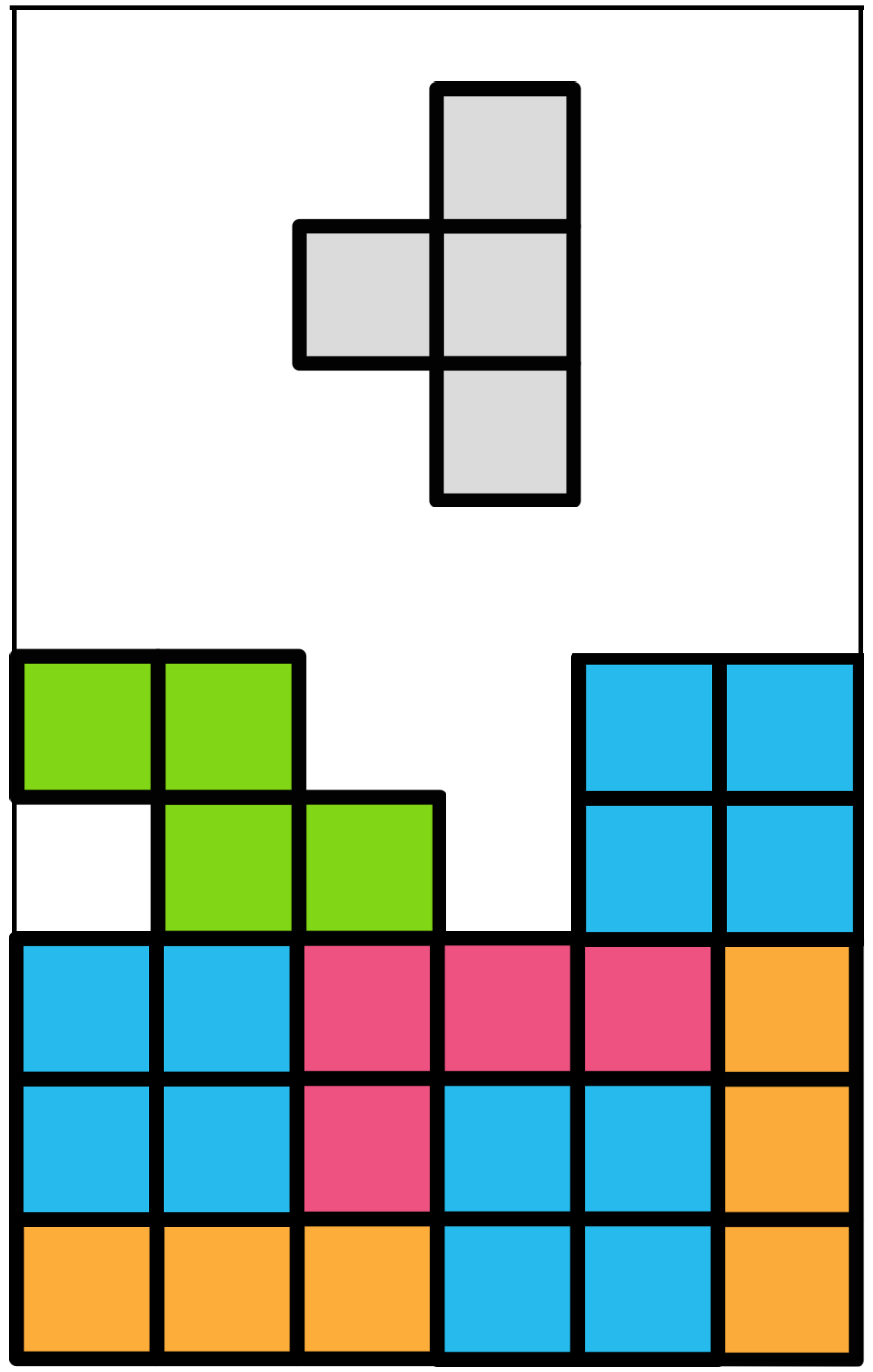
Prediction
about next item



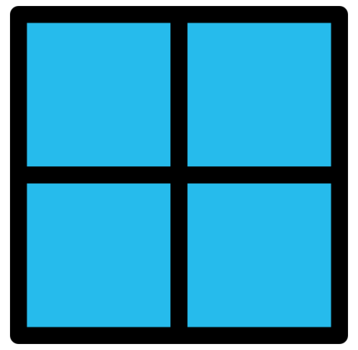
pieces arrive
online

Optimization under uncertainty

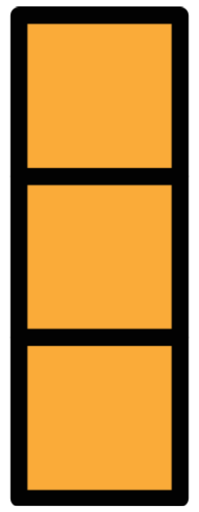
pieces arrive
online



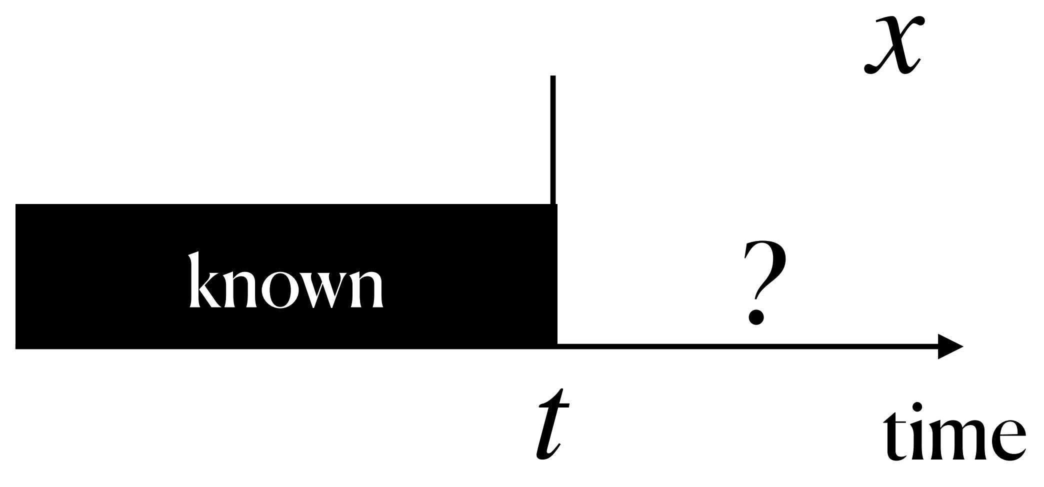
Prediction
about next item



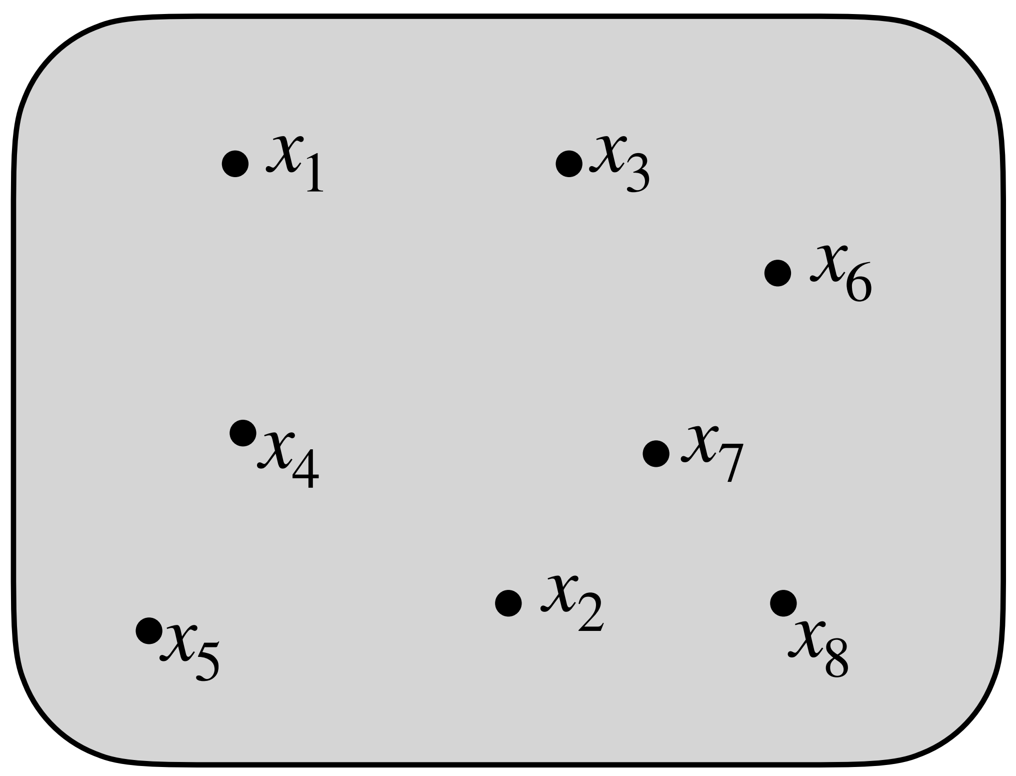
Next item



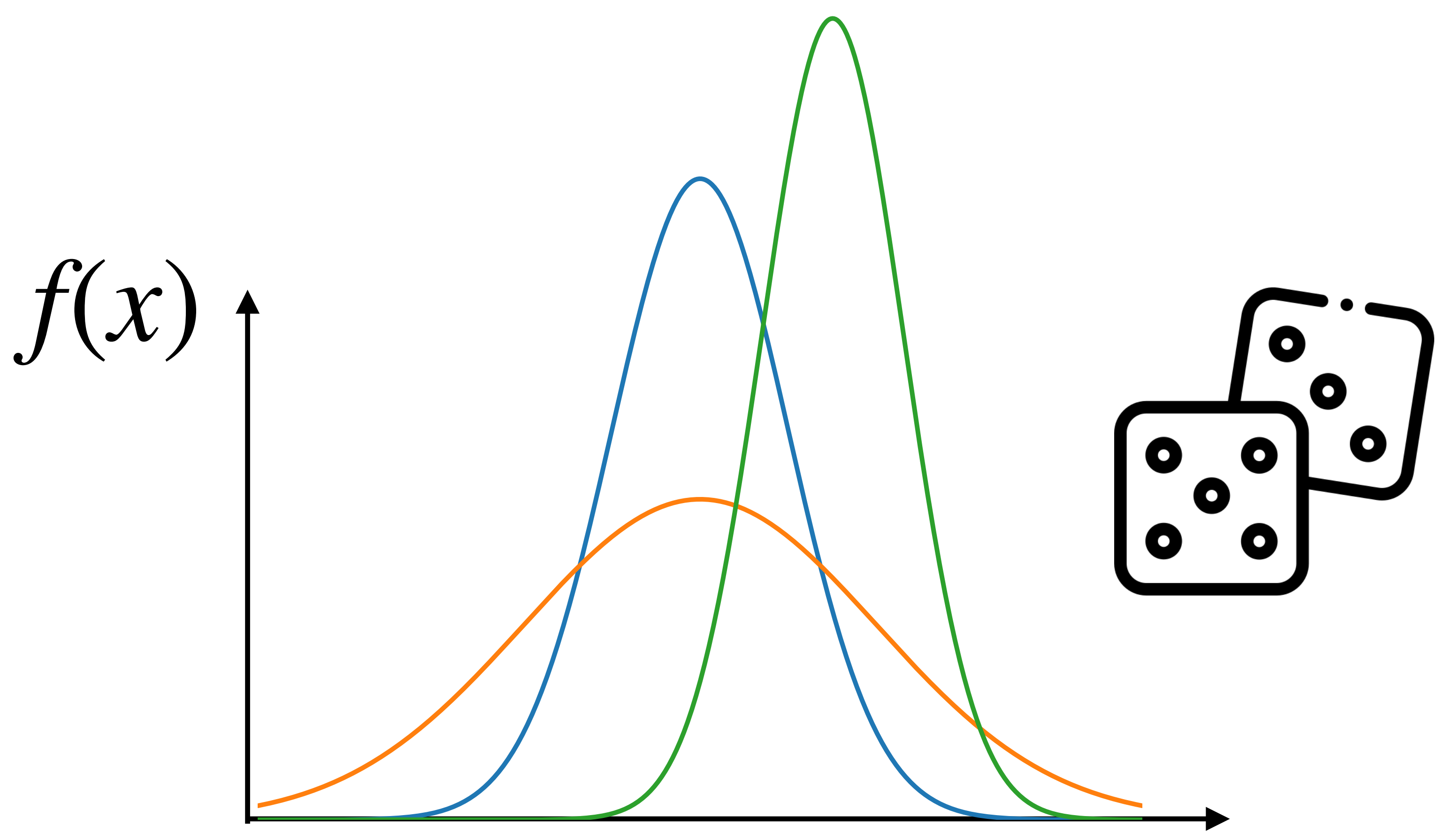
Optimization under uncertainty



online input

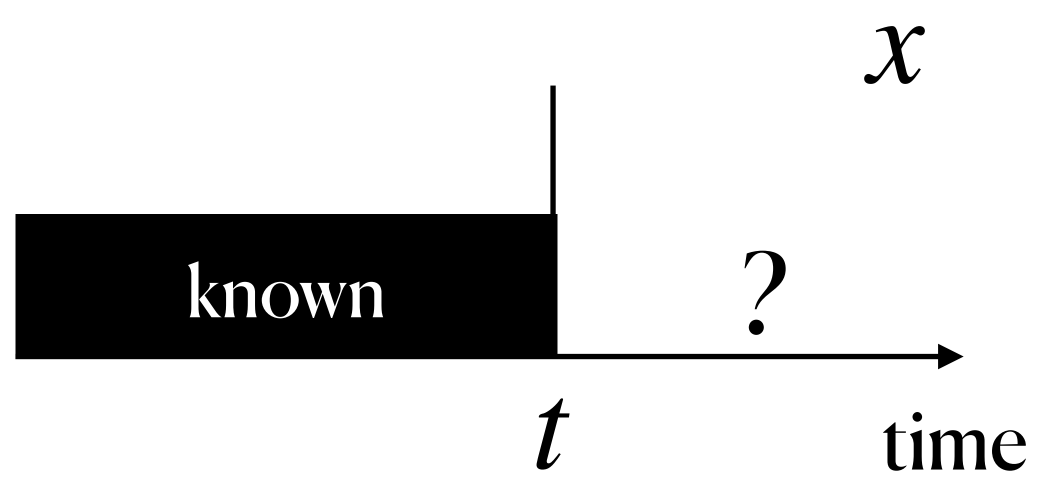


uncertainty set

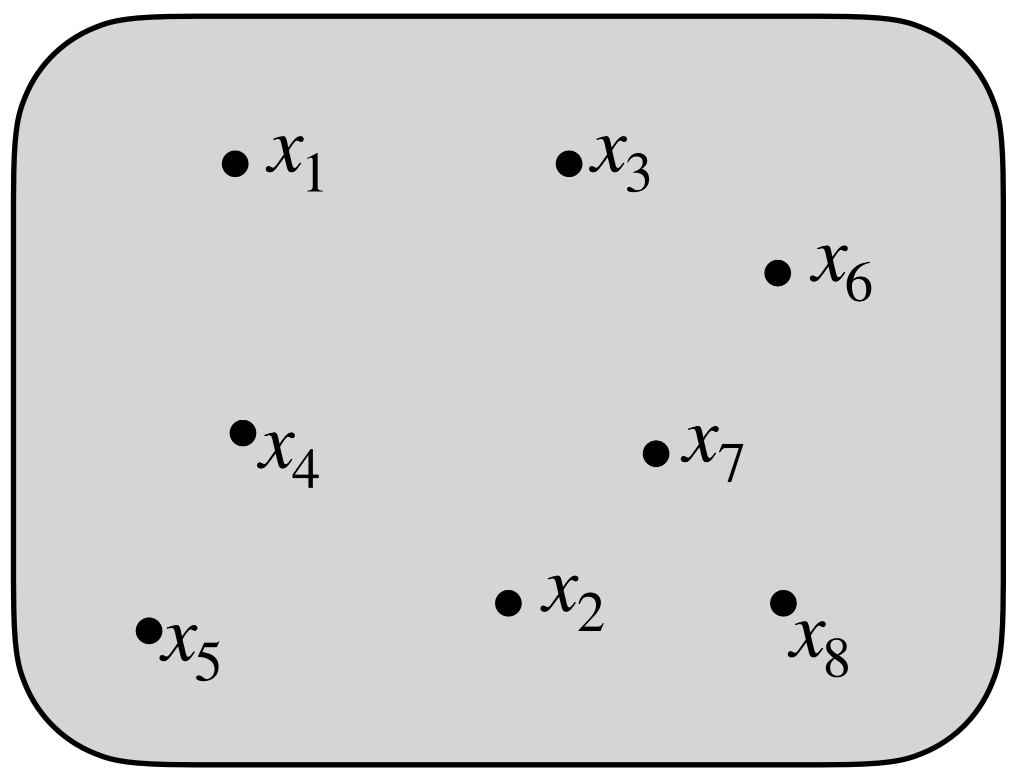


stochastic information

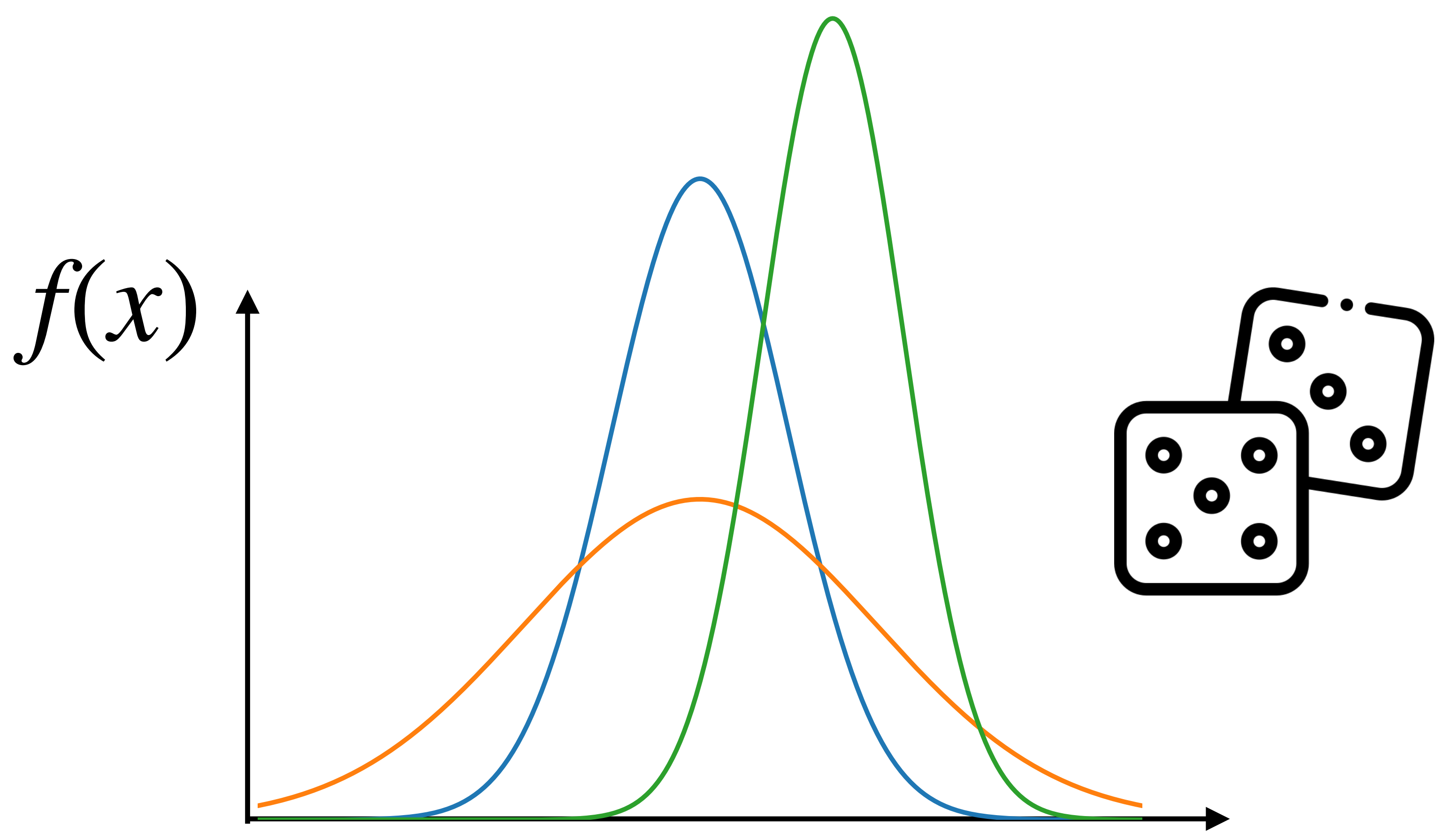
Optimization under uncertainty



online input



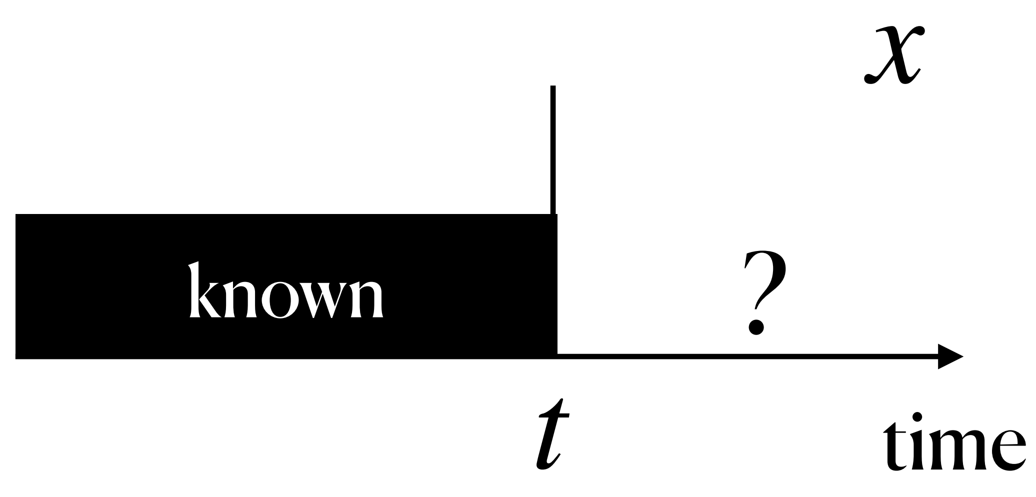
uncertainty set



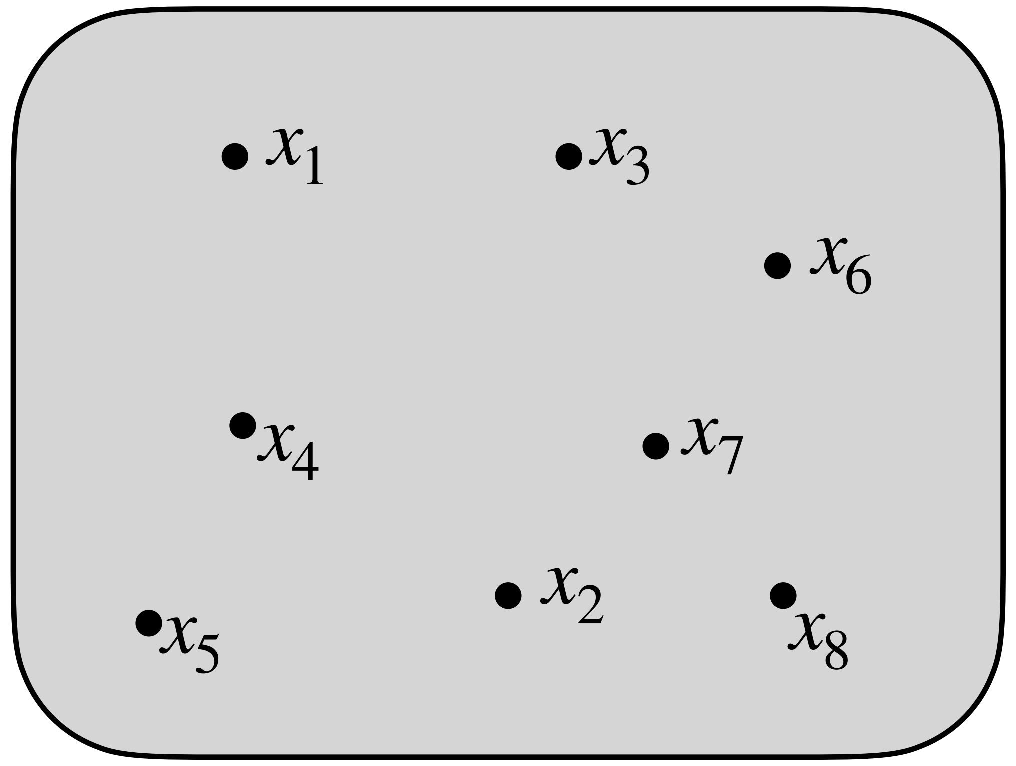
stochastic information

+ predictions (e.g., ML-based)

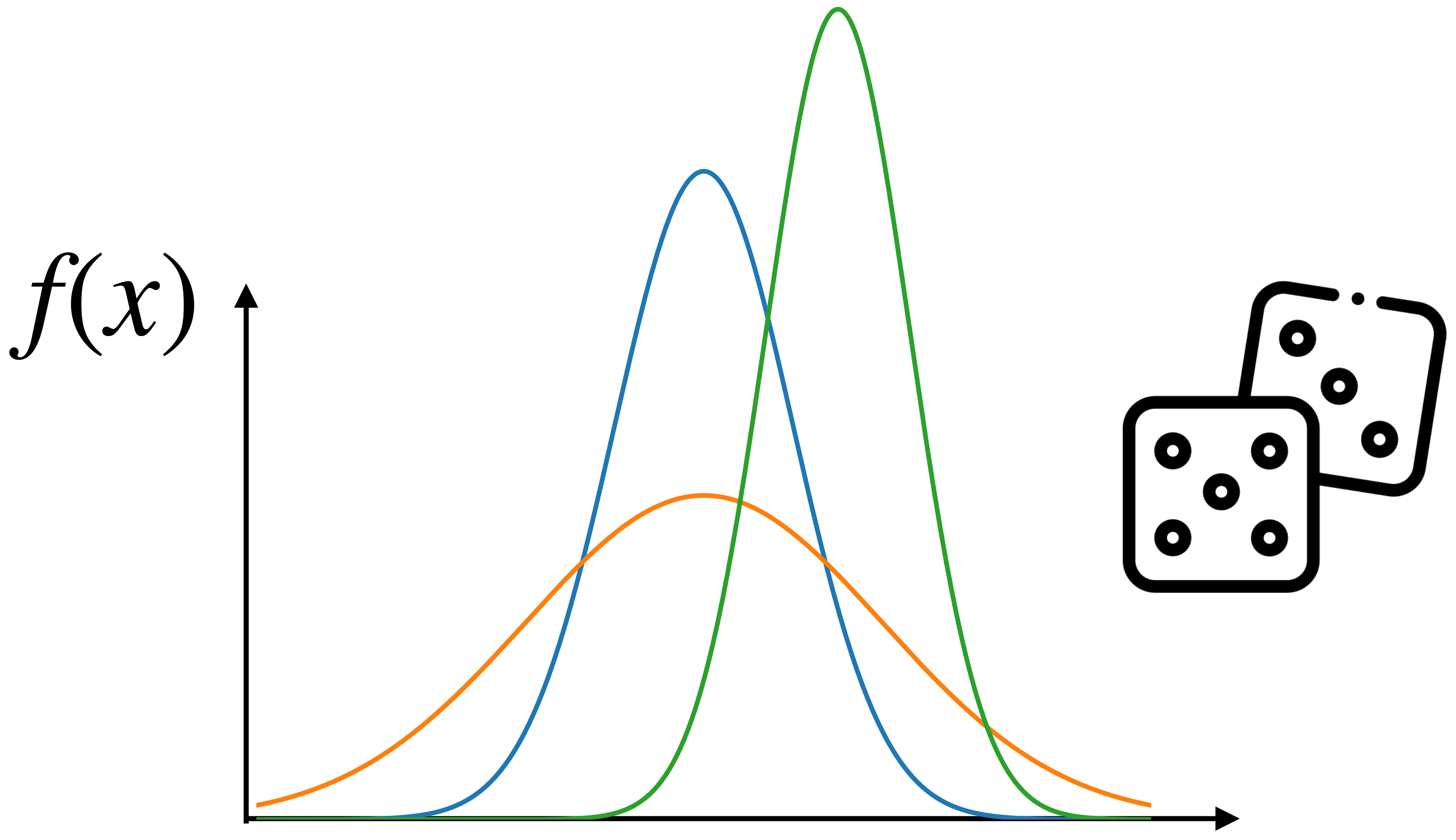
Optimization under uncertainty



online input



uncertainty set



stochastic information

+ predictions (e.g., ML-based)

How can algorithms stay reliable under uncertainty, yet benefit from accurate predictions?

Objectives of the thesis

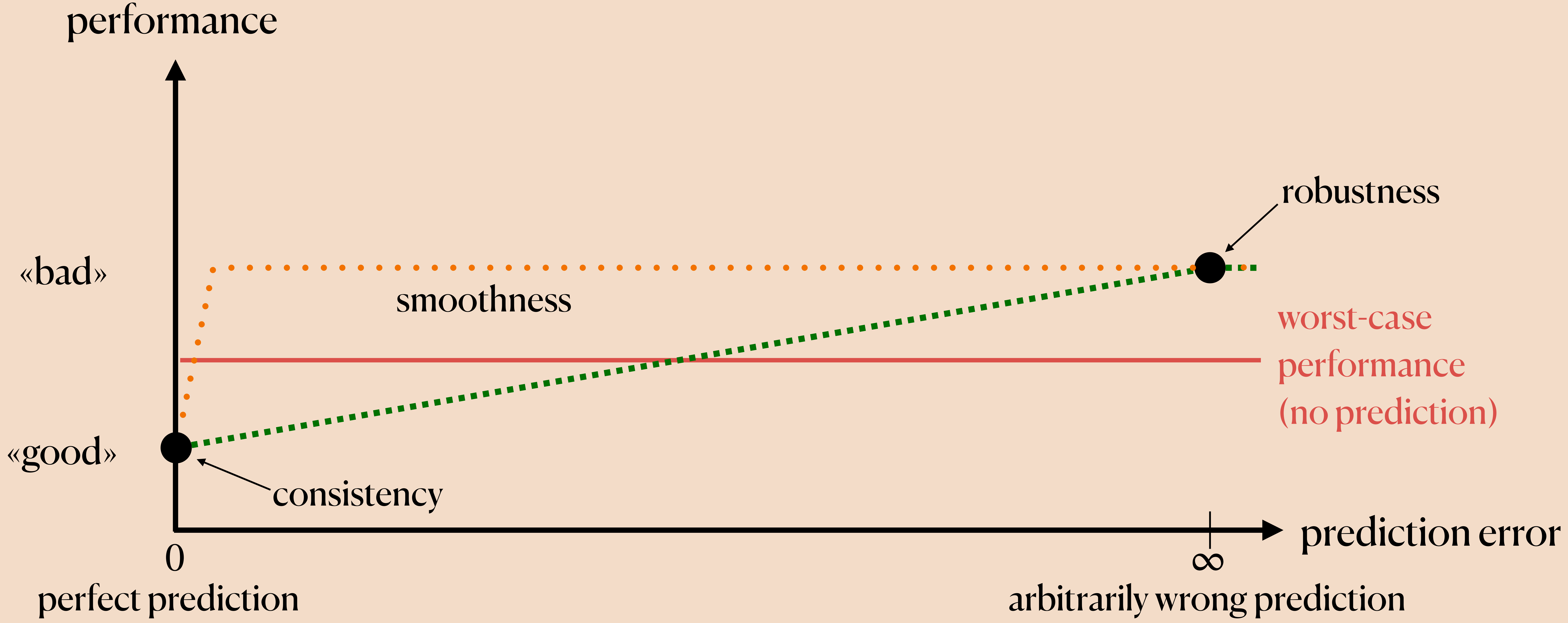
- New evaluation Metrics for Learning-Augmented Algorithms
- Scenario-Based Analysis

Three parts of the presentation:

1. **Decision-Theoretic Approaches in Learning-Augmented Algorithms**
2. **Prophet inequalities with scenarios**
3. **Scenario-Based Robust Optimization of Tree Structures**

1. Decision-Theoretic Approaches in Learning-Augmented Algorithms

Learning-augmented algorithms



Motivation

Pareto-optimal algorithms [1]

+ best possible trade-off between consistency and robustness

– might be brittle

– how to select the «best» one?

[1] **Bo Sun, et al.** Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. **2021**

Motivation

Pareto-optimal algorithms [1]

+ best possible trade-off between consistency and robustness

– might be brittle

– how to select the «best» one?

Smooth algorithms [2]

+ performance degrades gradually as the prediction error increases

+ avoid brittleness

– assumption about known upper bound on the prediction error (sometimes)

– might be suboptimal when prediction is accurate

[1] **Bo Sun, et al.** Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. **2021**

[2] **Spyros Angelopoulos, Shahin Kamali, and Dehou Zhang.** Online Search with Best-Price and Query-Based Predictions. **2022.**

Motivation

Pareto-optimal algorithms [1]

+ best possible trade-off between consistency and robustness

– might be brittle

– how to select the «best» one?

Smooth algorithms [2]

+ performance degrades gradually as the prediction error increases

+ avoid brittleness

– assumption about known upper bound on the prediction error (sometimes)

– might be suboptimal when prediction is accurate

Our approach:

We aim to evaluate and compare algorithms based on how they perform across the entire range of possible errors

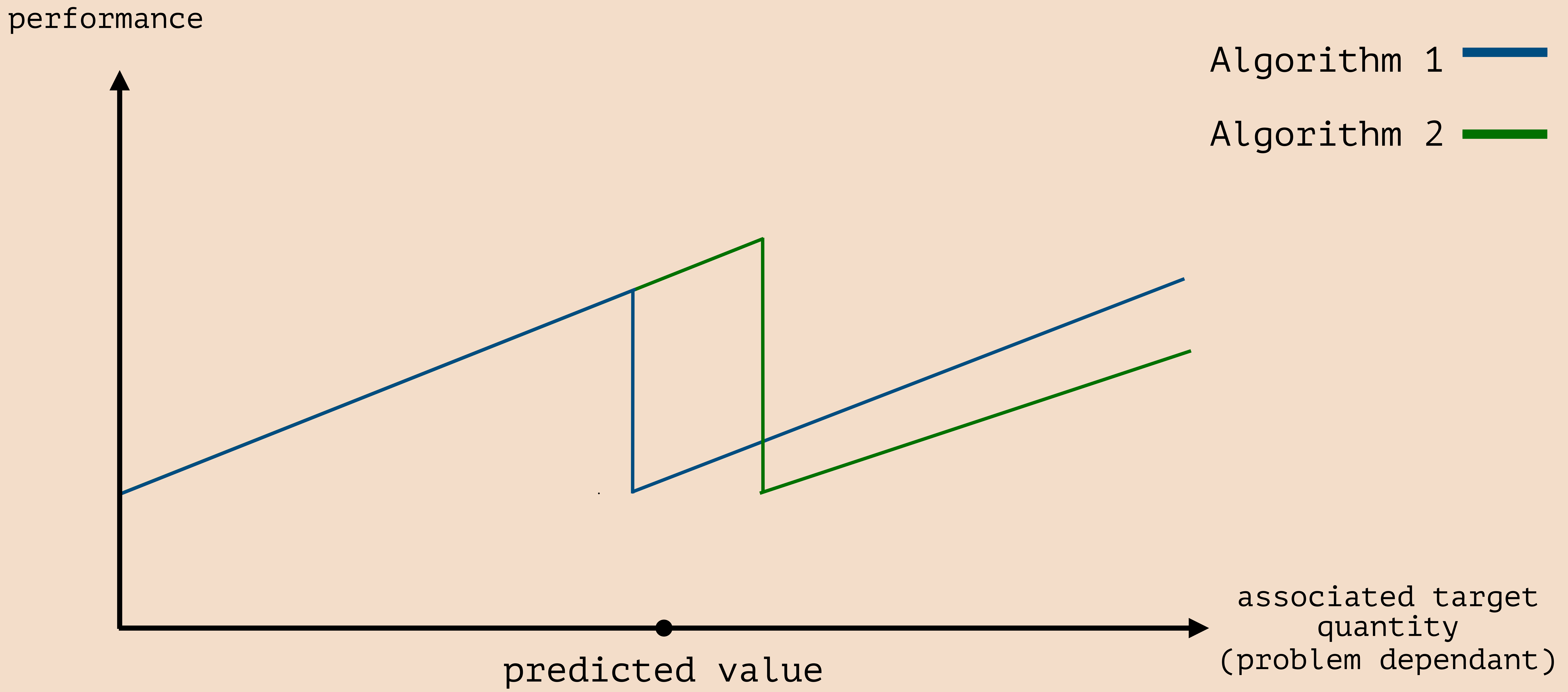
Measures:

- deterministic (distance-based)
- stochastic (risk-based)

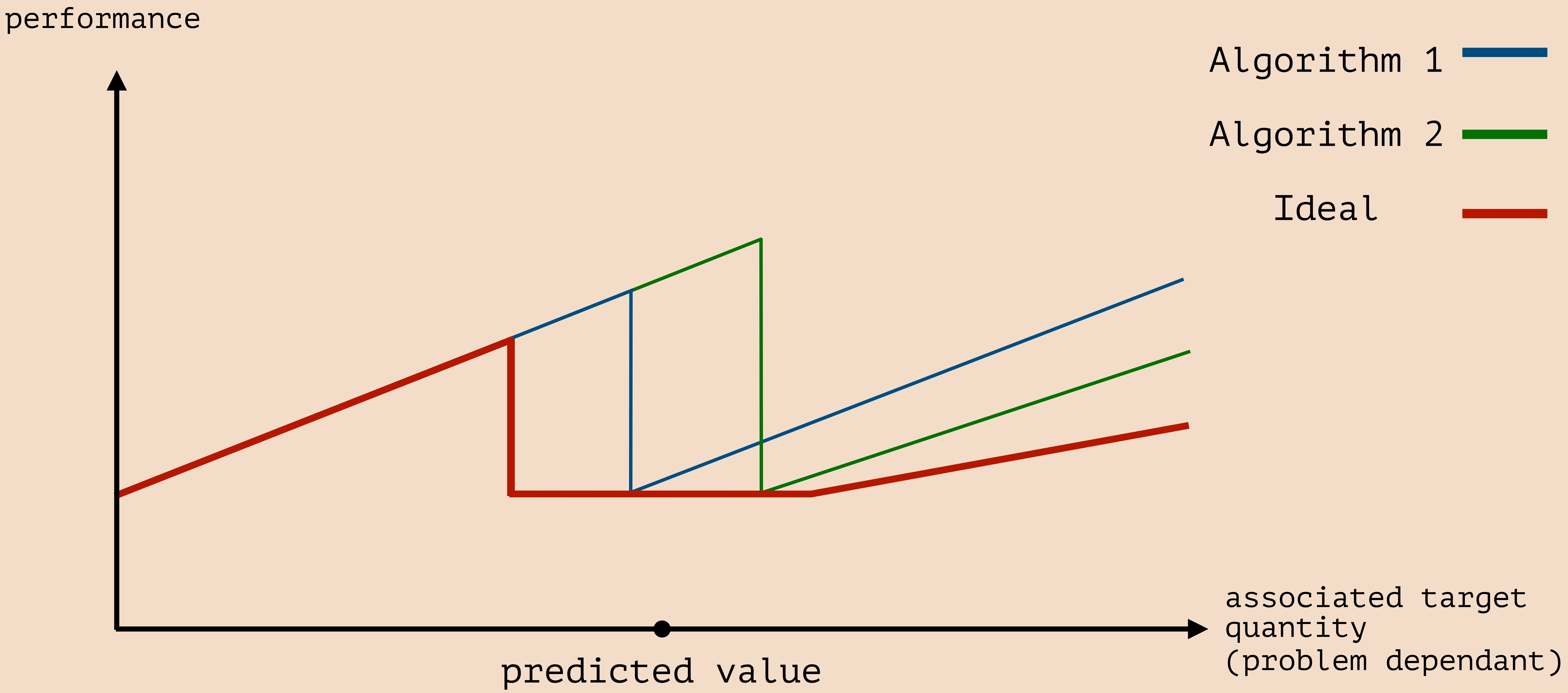
[1] **Bo Sun, et al.** Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. **2021**

[2] **Spyros Angelopoulos, Shahin Kamali, and Dehou Zhang.** Online Search with Best-Price and Query-Based Predictions. **2022.**

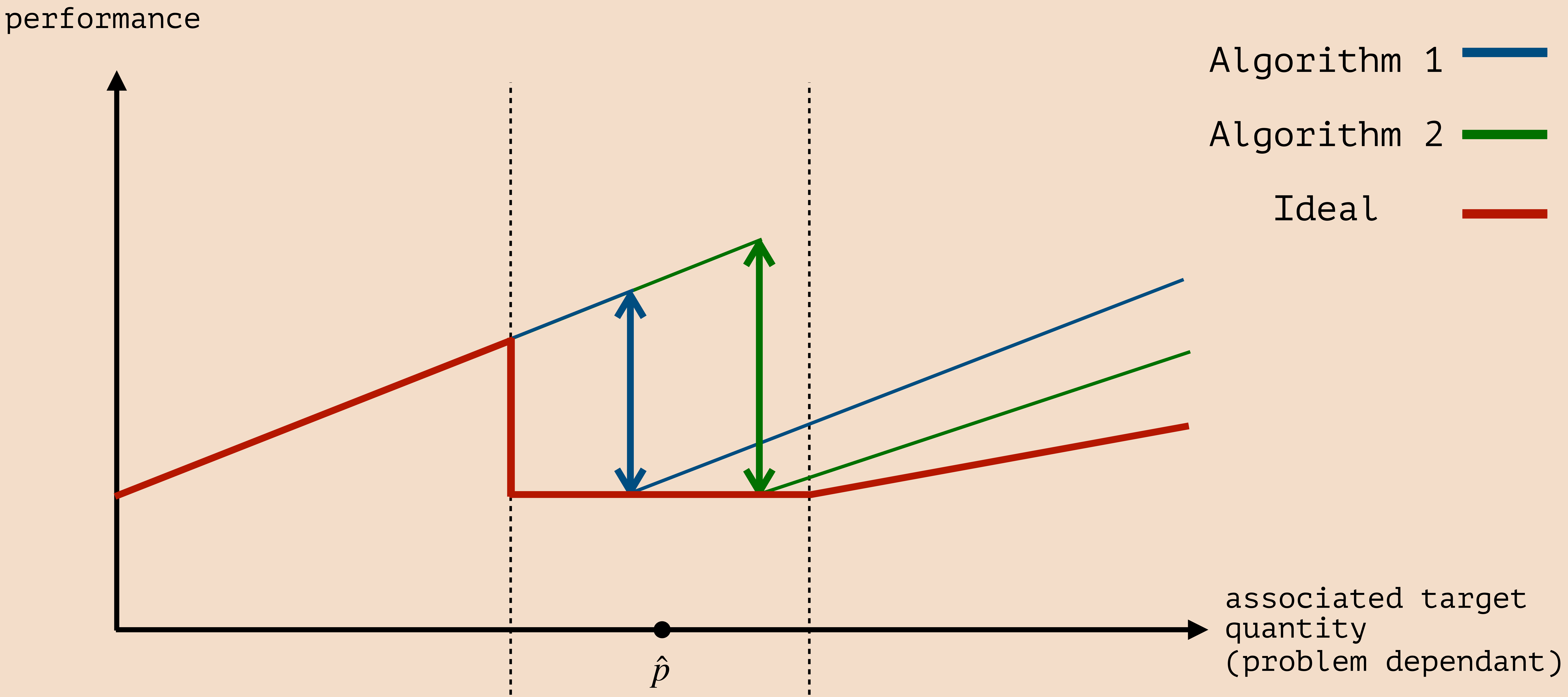
Distance-based analysis



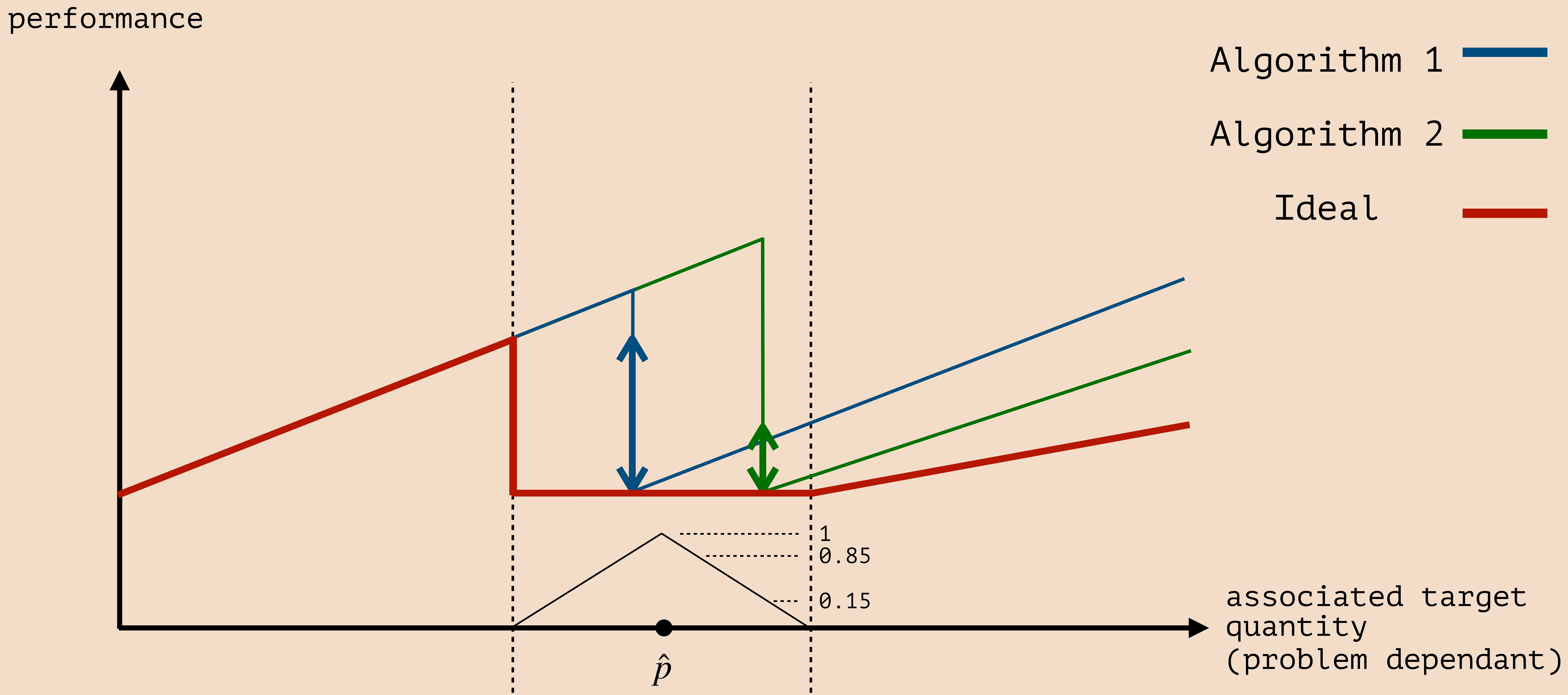
Distance-based analysis



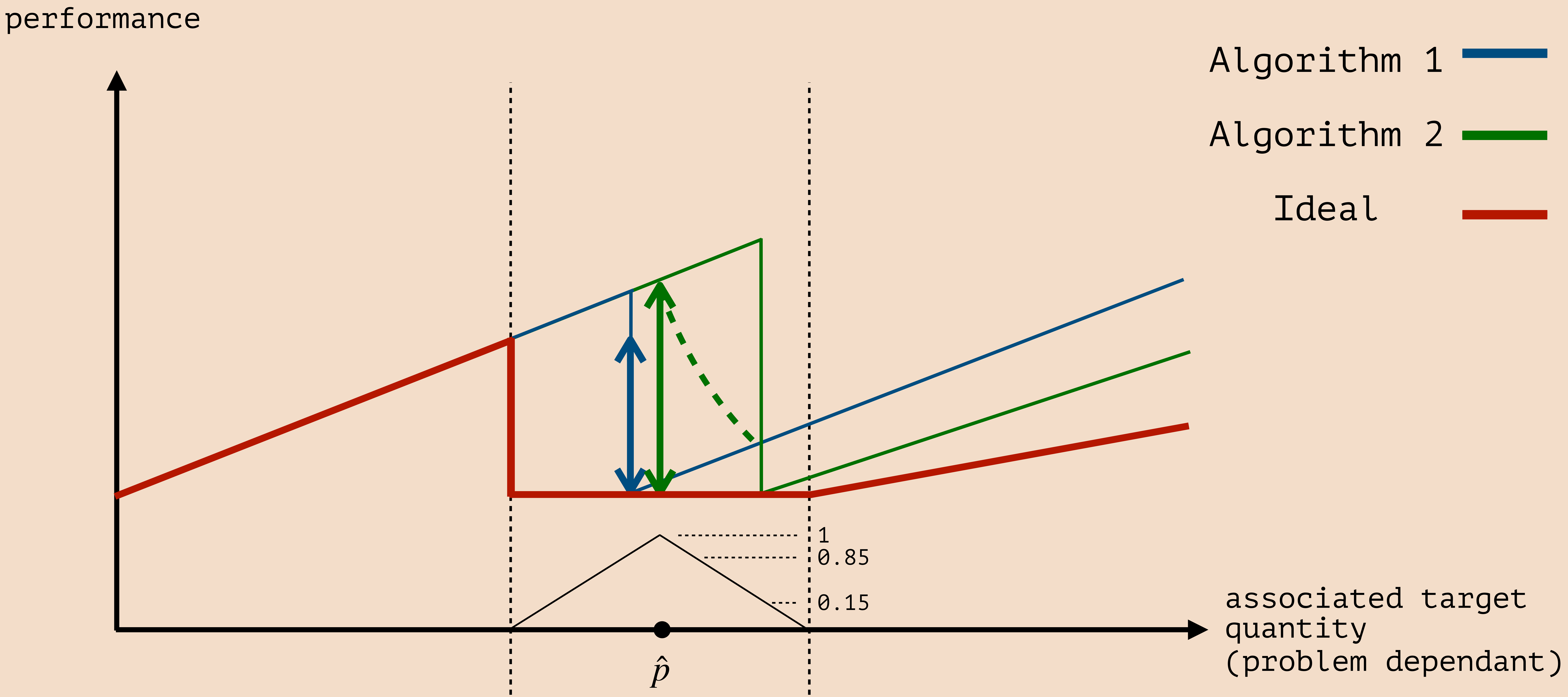
Distance-based analysis



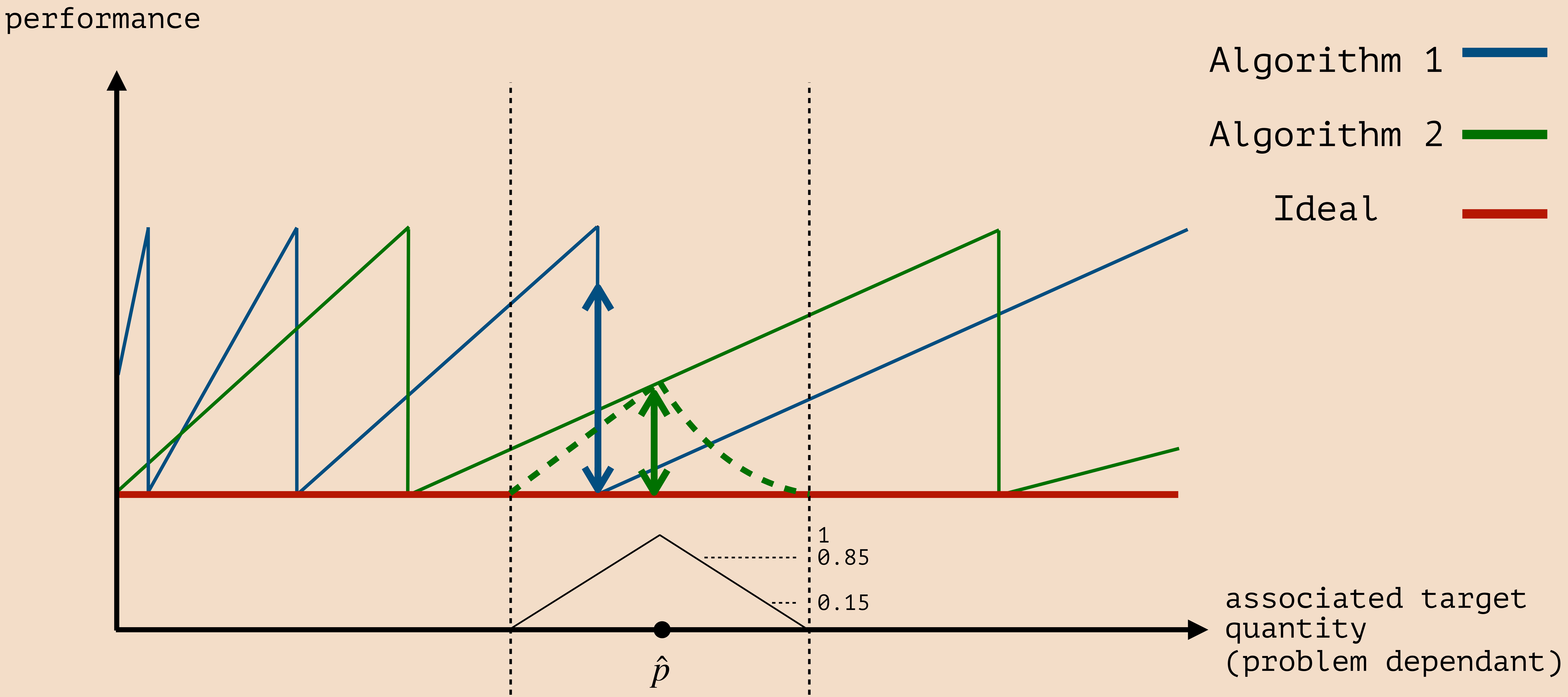
Distance-based analysis



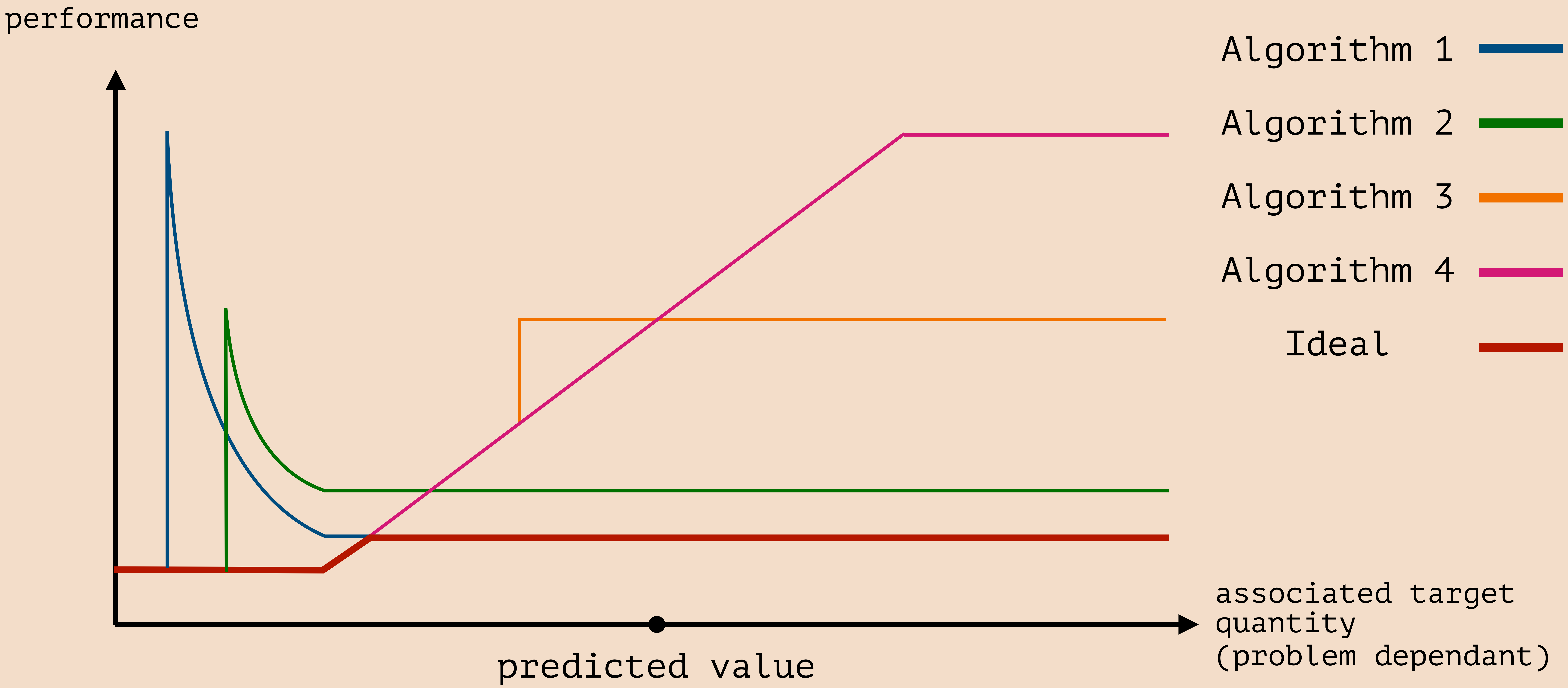
Distance-based analysis



Distance-based analysis



Distance-based analysis



Distance-based analysis

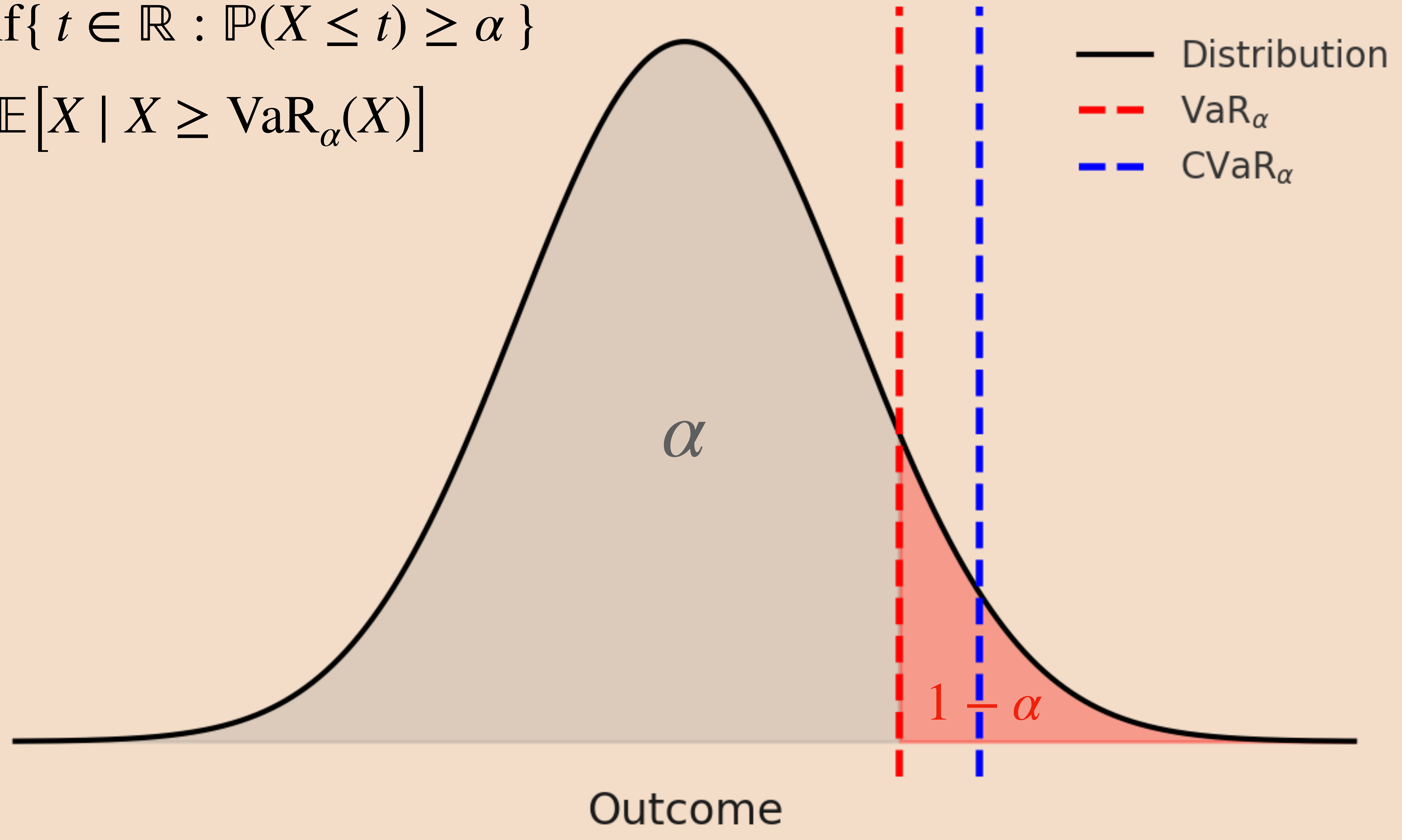
Let R be the class of the r-robust algorithms.

- Algorithm **Max**: the best algorithm from R, that minimizes the *maximum* weighted distance from the ideal performance.
- Algorithm **Avg**: the best algorithm from R, that minimizes the *average* weighted distance from the ideal performance.

Risk-based analysis

$$\mathrm{VaR}_\alpha(X) = \inf\{\, t \in \mathbb{R} : \mathbb{P}(X \le t) \ge \alpha \,\}$$

$$\mathrm{CVaR}_\alpha(X) = \mathbb{E}\left[X \mid X \ge \mathrm{VaR}_\alpha(X)\right]$$



Risk-based analysis

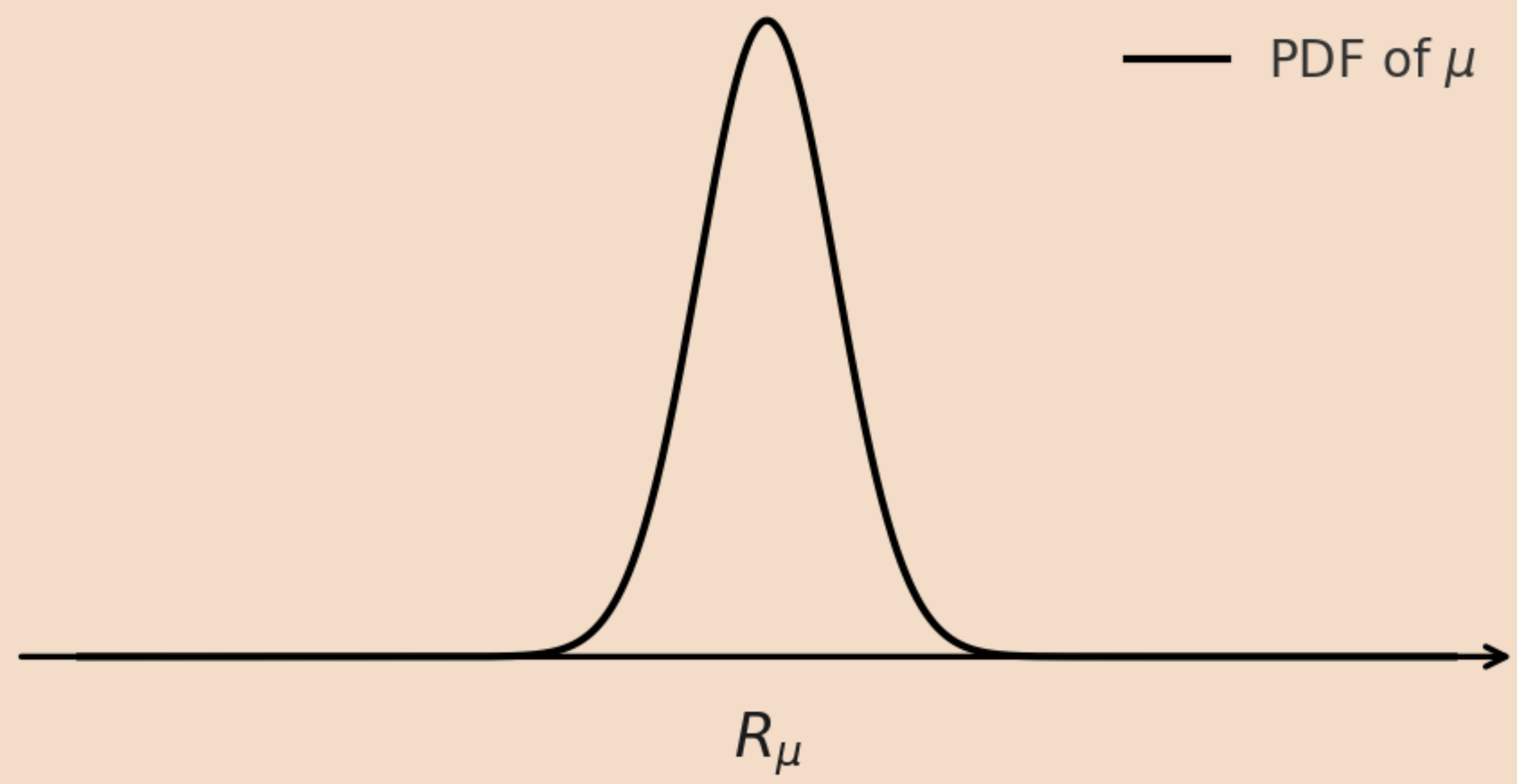
$$\mathrm{CVaR}_\alpha(\mathtt{ALG}) = \inf_t \left\{ t + \frac{1}{1-\alpha}\mathbb{E}[(\mathtt{ALG} - t)^+] \right\},$$

where $\alpha \in [0,1)$ [3].

prediction is in the form of a distribution μ, with support over an interval R_μ

$\alpha = 0$: risk-seeking regime

$\alpha \to 1$: risk averse regime



[3] **R Tyrrell Rockafellar, Stanislav Uryasev, et al.** Optimization of conditional value-at-risk. **2000.**
[4] **Nicolas Christianson, Bo Sun, Steven H. Low, and Adam Wierman.** Risk-Sensitive Online Algorithms. **2024.**

Risk-based analysis

For an algorithm A, and given $\alpha \in [0,1)$, we define the α-consistency:

$$\alpha - \text{cons}(A) = \sup_{F \in \mathcal{F}} \frac{\text{CVaR}_{\alpha,F}(A(\sigma))}{\mathbb{E}_{\sigma \sim F}[\text{OPT}(\sigma)]}$$

$\alpha = 0$: risk-seeking algorithm [5]

$$\text{CVaR}_{\alpha,F}(A) = \mathbb{E}_{\sigma \sim F}[A(\sigma)]$$

$\alpha \to 1$: risk averse algorithm

$$\text{CVaR}_{\alpha,F}(A) = \inf_{\sigma \in \text{supp}(F)} A(\sigma)$$

[5] **Ilias Diakonikolas, Vasilis Kontonis, Christos Tzamos, Ali Vakilian, and Nikos Zarifis.** Learning Online Algorithms with DistributionalAdvice. **2021**

Risk-based analysis

For an algorithm A, and given $\alpha \in [0,1)$, we define the α-consistency:

$$\alpha - \text{cons}(A) = \sup_{F \in \mathcal{F}} \frac{\text{CVaR}_{\alpha,F}(A(\sigma))}{\mathbb{E}_{\sigma \sim F}[\text{OPT}(\sigma)]}$$

$\alpha = 0$: risk-seeking algorithm [5]

$$\text{CVaR}_{\alpha,F}(A) = \mathbb{E}_{\sigma \sim F}[A(\sigma)]$$

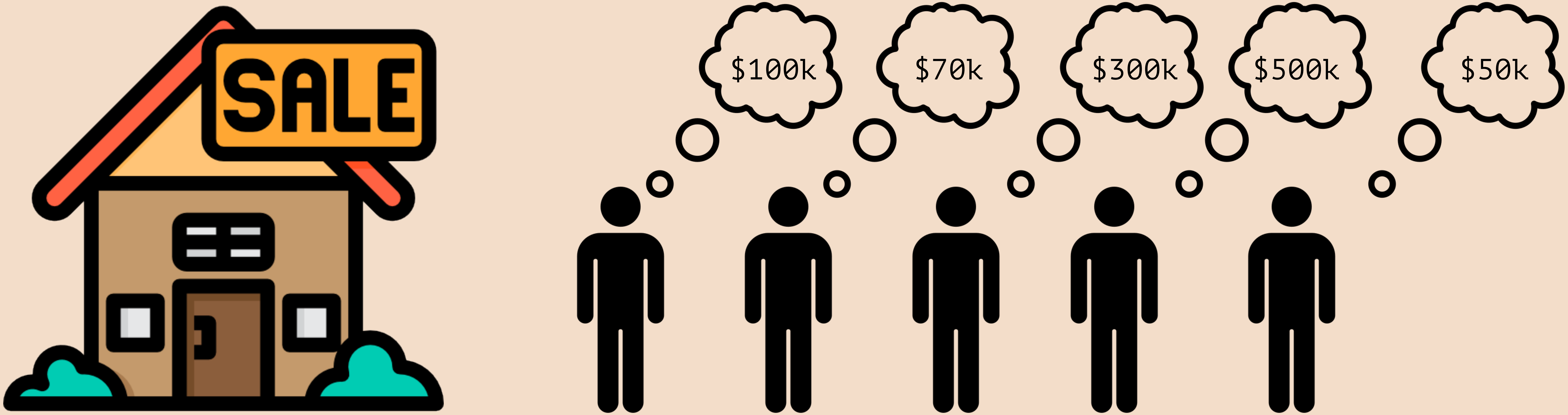
$\alpha \to 1$: risk averse algorithm

$$\text{CVaR}_{\alpha,F}(A) = \inf_{\sigma \in \text{supp}(F)} A(\sigma)$$

Algorithm $\mathbf{CVaR}_\alpha$: the best algorithm A from R, that minimizes the $\text{CVaR}_\alpha(A)$.

[5] **Ilias Diakonikolas, Vasilis Kontonis, Christos Tzamos, Ali Vakilian, and Nikos Zarifis.**
Learning Online Algorithms with DistributionalAdvice. **2021**

One-Max Search



One-Max Search

Input σ : sequence of prices in $[1, M]$

A_T : threshold algorithm with $T \in [1, M]$

Optimal competitive ratio without prediction is $\sqrt{M}$ [6] .

For any $r \geq \sqrt{M}$, algorithm A_T is r-robust if and only if $T \in [M/r, r]$ [1].

Objective

For a given robustness level r, find the threshold $T \in [M/r, r]$, that optimizes the chosen decision-theoretic measure.

[1] **Bo Sun, et al.** Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. **2021**

[6] **Ran El-Yaniv.** Competitive Solutions for On-line Financial Problems. **1996.**

best worst

Risk-based analysis

	PO	HC	**Max**	**Avg**	**CVaR**		
					$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 0.9$
Avg perf. ratio	4.630	10.009	4.394	4.447	9.771	8.144	6.022
Exp. profit	13.904	5.475	15.614	20.528	35.795	34.402	27.500

Pareto Optimal algorithm (PO) [1]

h-confidence algorithm (HC) [2]

[1] **Bo Sun, et al.** Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. **2021**

[2] **Spyros Angelopoulos, Shahin Kamali, and Dehou Zhang.** Online Search with Best-Price and Query-Based Predictions. **2022.**

One-Max Search | Real-World Experiments

best — worst

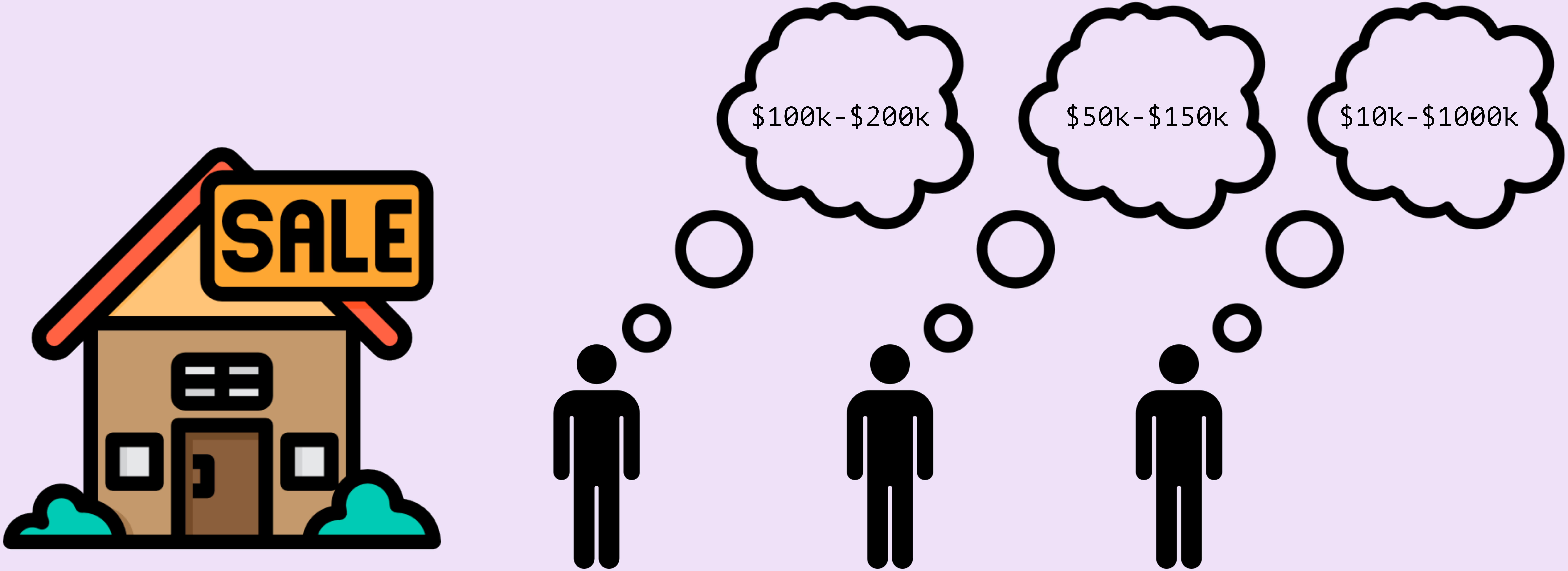
Currency	PO	HC	Max	Avg	CVaR
CHF	1.4853	1.5261	1.2611	1.2709	1.3328
GBP	1.4018	1.1480	1.7115	1.7134	1.4249
JPY	1.5115	1.0846	1.0869	1.0869	1.1650
USD	1.5229	1.3561	1.2339	1.2386	1.3012
BTC	14.9876	8.9721	8.7997	8.7643	7.0627

Results

- Introduced new evaluation metrics that enable the optimization of learning-augmented algorithms over the entire range of prediction error.
- Designed theoretically optimal algorithms for several well-studied problems (**1-max search, ski rental, contract scheduling**) and demonstrated that they outperform the known, extreme case approaches in practice.

2. Prophet inequalities with scenarios

Prophet inequalities



Prophet inequalities

1	0, w.p. 0.99 100, w.p. 0.01

Prophet inequalities

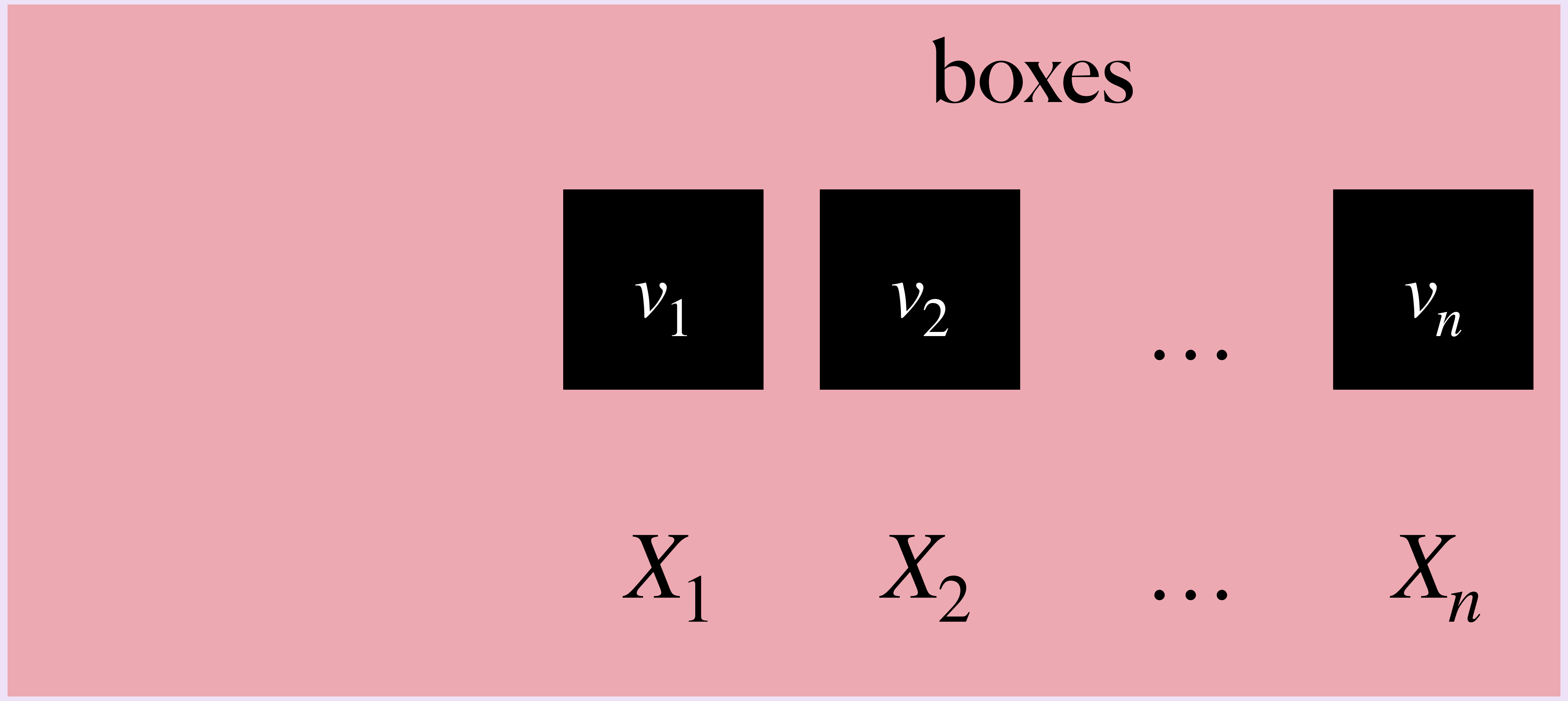
1

0, w.p. 0.99
100, w.p. 0.01

$$\mathbb{E}[\text{Prophet's reward}] = 1 \cdot 0.99 + 100 \cdot 0.01 = 0.99 + 1$$

Model

There are n random variables $X_1, X_2, \ldots, X_n$. Each has a known distribution. Realizations $v_1, \ldots, v_n$ are revealed sequentially, one at a time.



Algorithm A is a stopping rule.

- Ratio with prophet: $\mathbb{E}[A]/\mathbb{E}[\max(X_1, \ldots, X_n)]$

The best known ratio is 1/2 [7].

Threshold: set price τ, accept first $v_i \geq \tau$ with

- Median rule [8]

  τ s.t. $\mathbb{P}(\exists v_i \geq \tau) = 1/2$

- Mean rule [9]

  $\tau = 1/2\mathbb{E}[\max(v_1, \ldots, v_n)]$

- Backward induction [10]

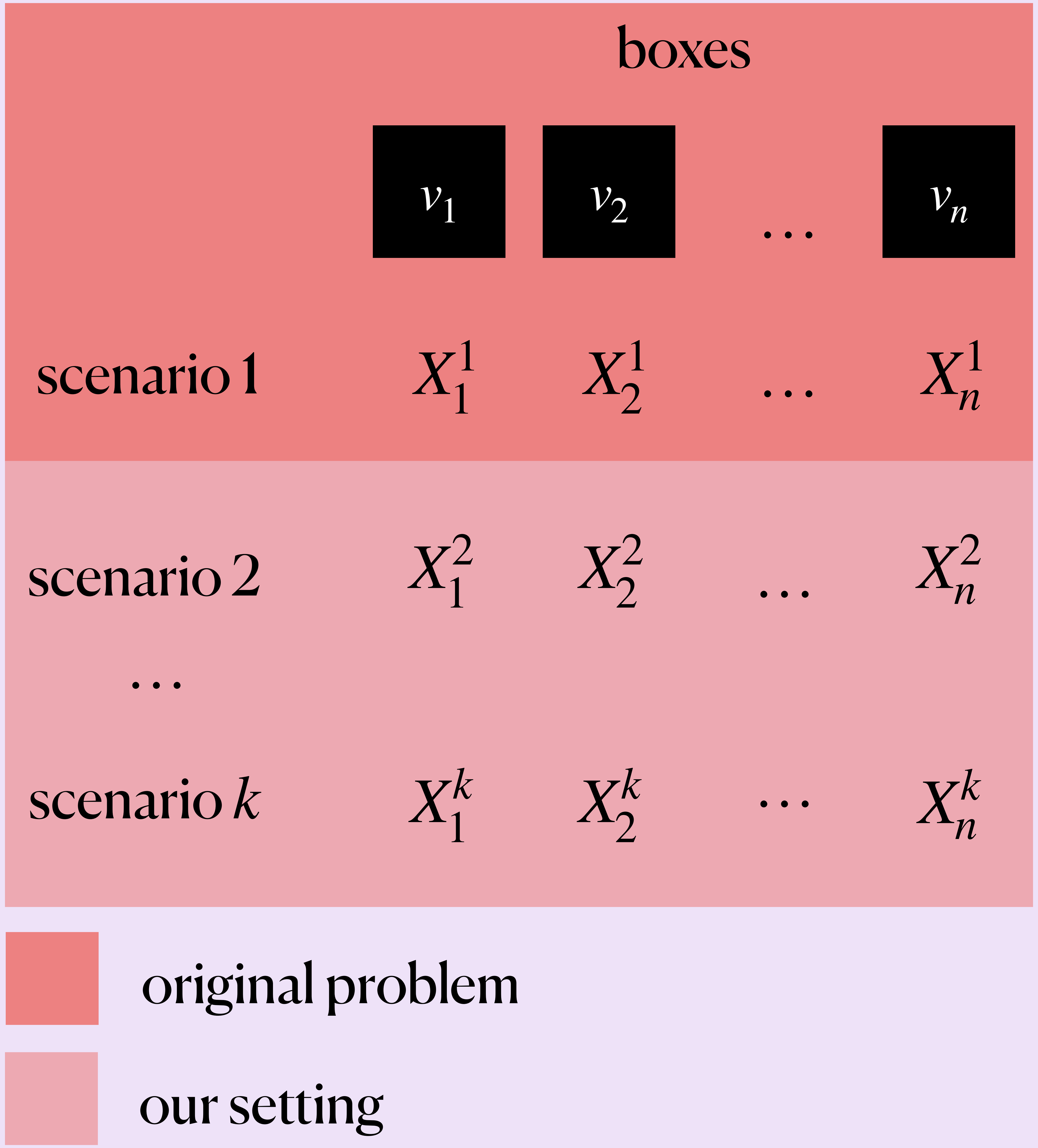
[7] **Ulrich Krengel and Louis Sucheston**. Semiamarts and finite values. **1977.**
[8] **Ester Samuel-Cahn.** Comparison of threshold stop rules and maximum for independent nonnegative random variables. **1984.**
[9] **Robert Kleinberg and S. Matthew Weinberg.** Matroid prophet inequalities. **2012**
[10] **Theodore P Hill and Robert P Kertz.** Ratio comparisons of supremumand stop rule expectations. **1981.**

Model

There are kn random variables $X_1^s, X_2^s, \ldots, X_n^s$, $s \in \{1, \ldots, k\}$.

Each has a known distribution. Realizations $v_1, \ldots, v_n$ are revealed sequentially, one at a time.

- Ratio with prophet:
  $$\min_{s \in k} \mathbb{E}[A(X^s)] / \mathbb{E}[\max(X_1^s, \ldots, X_n^s)]$$



Upper bound $1/k$

Construction

Inspired by the instance for the One-Max Search Problem

	■	■	■	■
Scenario 1	10	0	0	0
Scenario 2	10	100	0	0
Scenario 3	10	100	1000	0
Scenario 4	10	100	1000	10000
	p_1	p_2	p_3	p_4

$\mathbb{E}[A(X^s)] \leq p_s 10^s + 10^{s-1}$ $\Rightarrow$ Ratio $\leq 1/k$

$\max(X^s) = 10^s$ $p_i \geq 1/k$

Upper bound $1/k$

Construction

Inspired by the instance for the One-Max Search Problem

	1	2	3	4
Scenario 1	M	0	0	0
Scenario 2	M	M^2	0	0
Scenario 3	M	M^2	M^3	0
Scenario 4	M	M^2	M^3	M^4
	p_1	p_2	p_3	p_4

$\mathbb{E}[A(X^s)] \leq p_s M^s + M^{s-1}$ $\Rightarrow$ Ratio $\leq 1/k$

$\max(X^s) = M^s$

$p_i \geq 1/k$

Upper bound $1/2k$

Construction

Inspired by the instance for the One-Max Search Problem

	1	2	3	4	5	6	7	8
Scenario 1	M	M^2 w. p. $1/M$	0	0	0	0	0	0
Scenario 2	M	M^2 w. p. $1/M$	M^2	M^3 w. p. $1/M$	0	0	0	0
Scenario 3	M	M^2 w. p. $1/M$	M^2	M^3 w. p. $1/M$	M^3	M^4 w. p. $1/M$	0	0
Scenario 4	M	M^2 w. p. $1/M$	M^2	M^3 w. p. $1/M$	M^3	M^4 w. p. $1/M$	M^4	M^5 w. p. $1/M$
	p_1		p_2		p_3		p_4	

$\mathbb{E}[A(X^s)] \leq p_s M^s + M^{s-1}$ $\Rightarrow$ Ratio $\leq 1/2k$

$\mathbb{E}[\max(X^s)] = 2M^s - M^{s-1}$ $p_i \geq 1/k$

Baseline strategy

Choose a scenario $s \in \{1, \ldots, k\}$ u.a.r. and apply the optimal strategy for that scenario.

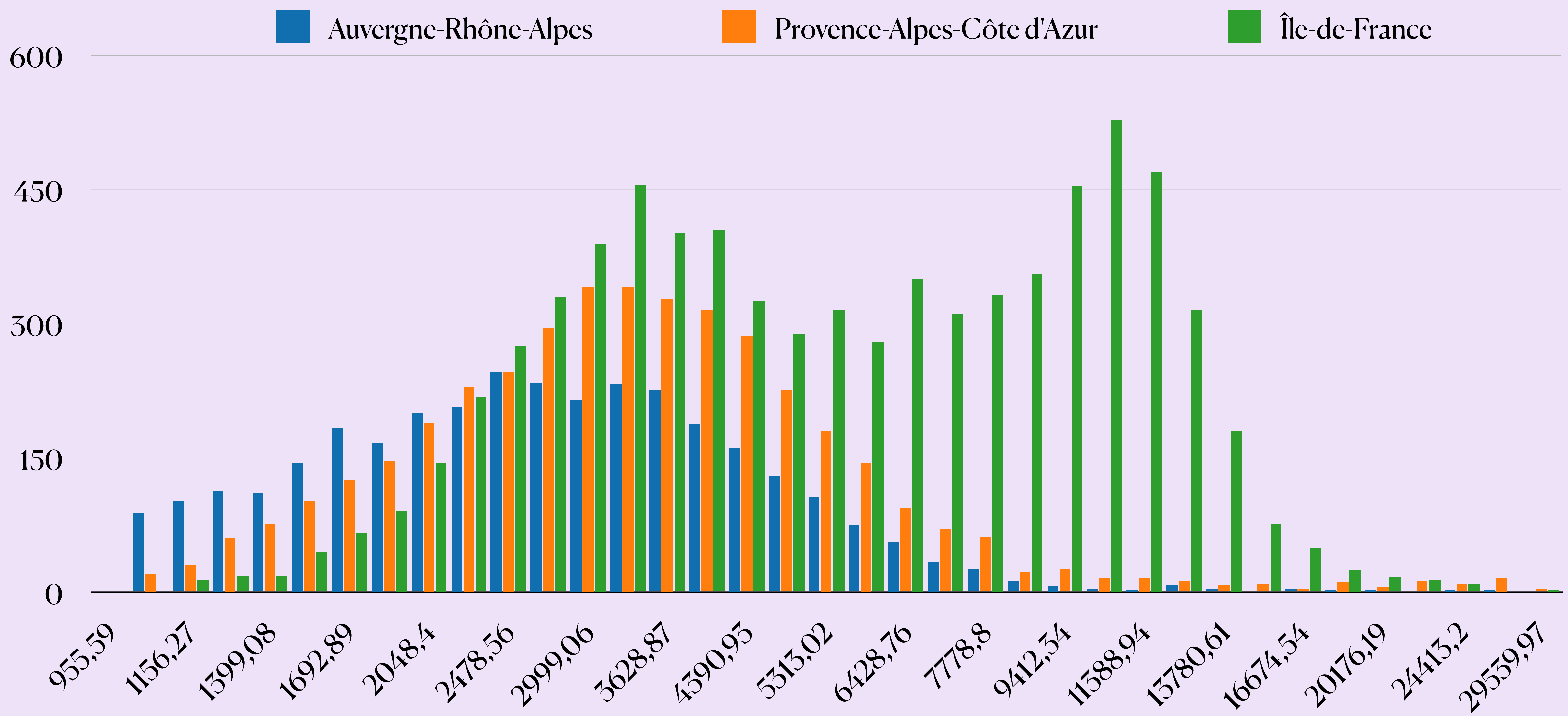
Theorem.

Algorithm Uniform achieves a competitive ratio of at least $1/2k$.

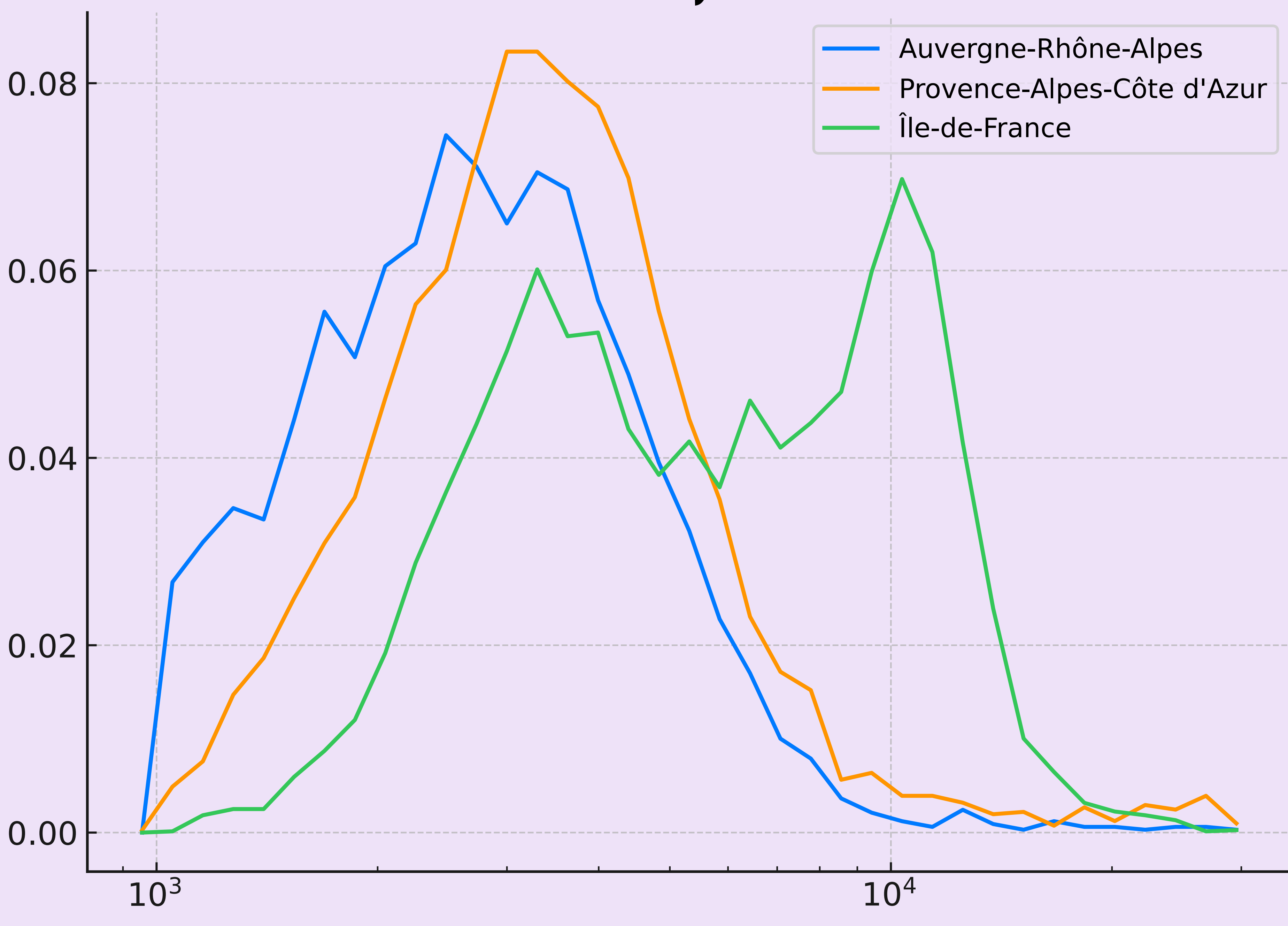
Dataset

- Housing market prices in France from gouv.fr in 2020 — 2023
- Box = month, scenario = region

In total, we have 48 boxes and 13 scenarios



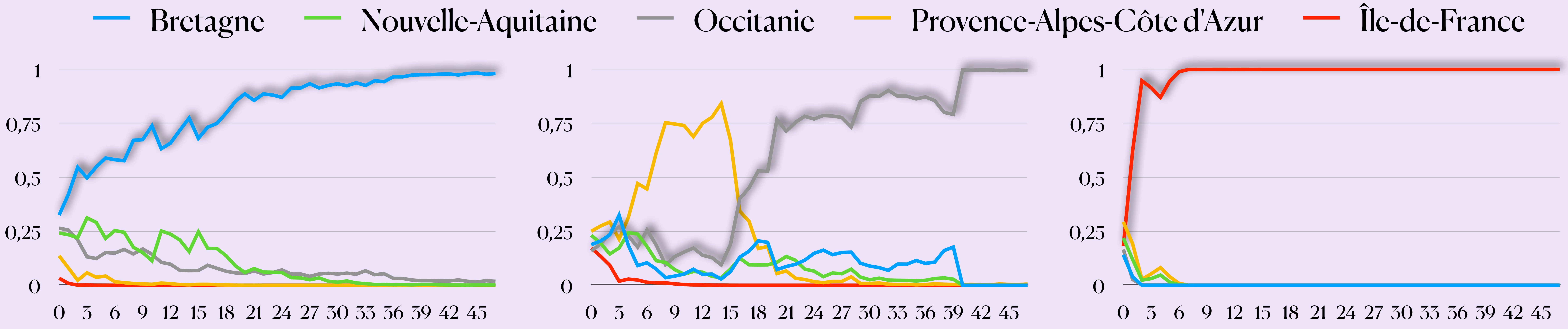
January 2020



Learning: Bayes Rule

1. Maintain distribution b over scenarios. When observing $X_i = v$ in box i, update b_s proportionally to probability that box i has value v in scenario s.

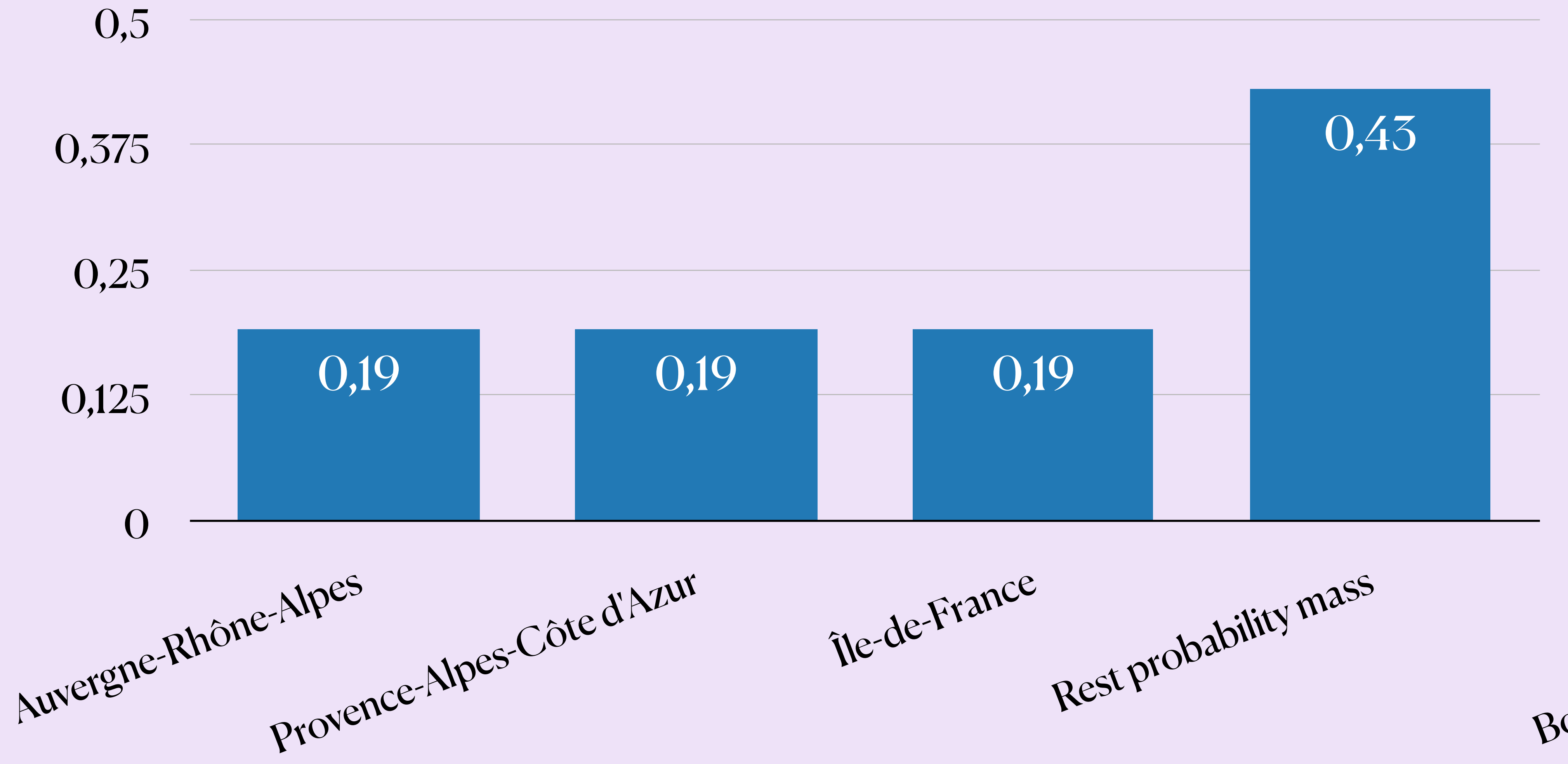
2. Initially $b_s = 1/k$ for all $s \in \{1,\ldots,k\}$. After the first box, $b_s = \dfrac{\mathbb{P}(X_1 \mid s)}{\sum_{i=1}^{k} \mathbb{P}(X_1 \mid i)}$ is the belief that the true scenario is s.

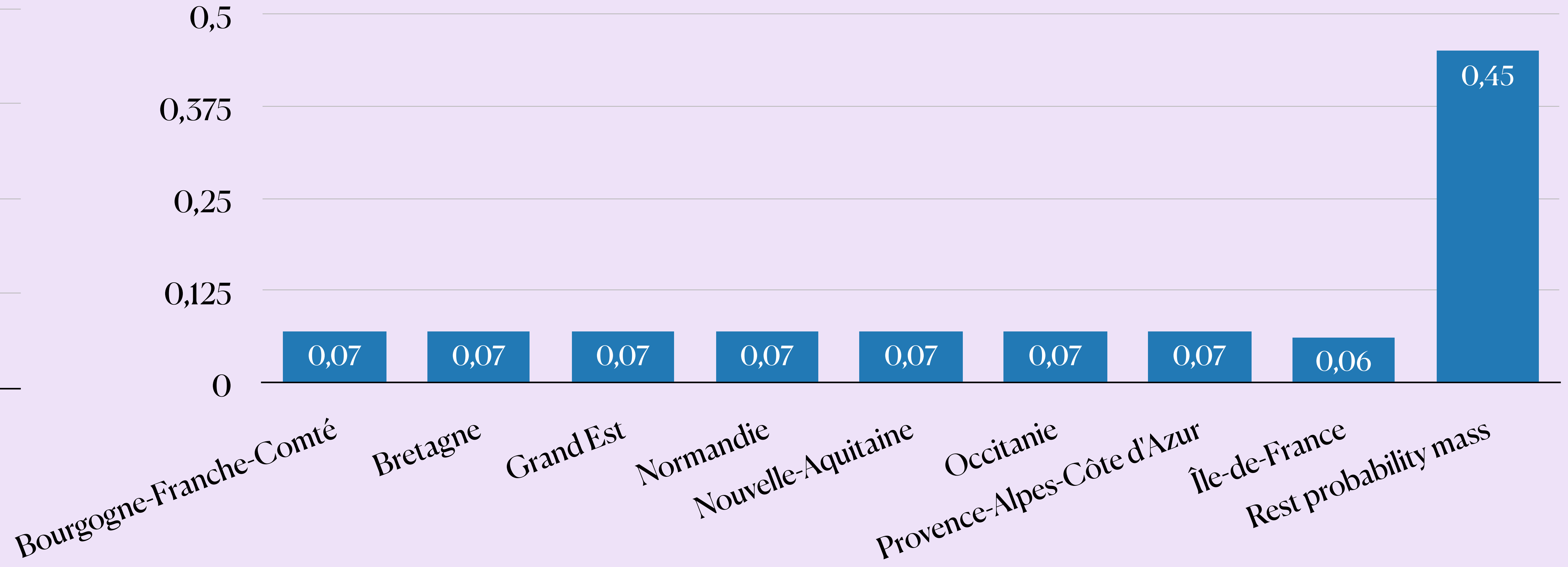


Commitment

1. Define the minimum probability mass for the first box in scenario s to ensure a guaranteed ratio of $1/2k$ by $p_s = \dfrac{\mathbb{E}[\max(X^s)]}{2k\,\mathbb{E}[A(X^s)]}$.

2. With probability p_s we *commit*, meaning we fix one scenario until the end. With this probability $1 - \sum_{s=1}^{k} p_s$, we employ *learning strategies.*





Learning Strategies

1. Algorithm *Independent* applies the threshold of a scenario s, according to the distribution defined by b_s.

2. Algorithm *Max-belief* selects the threshold of the most believed scenario so far.

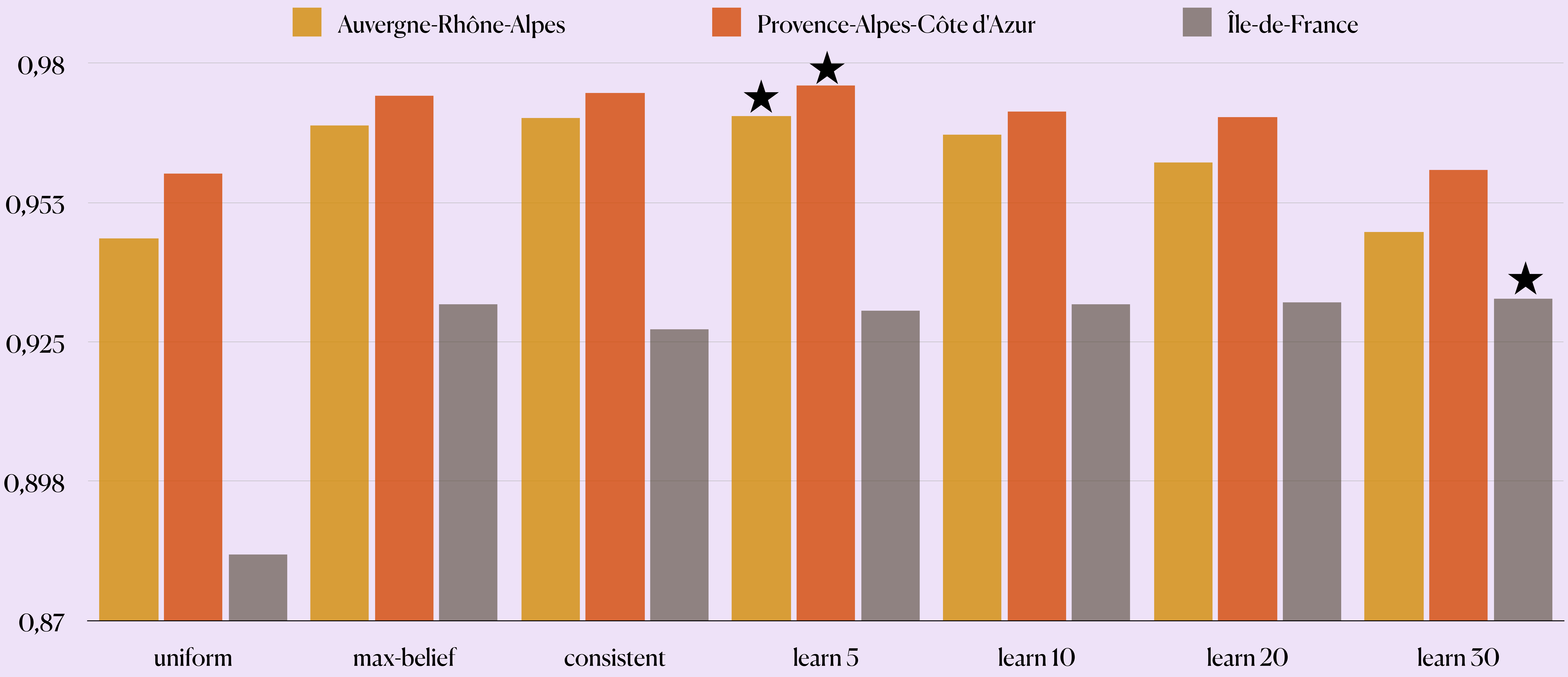
3. Algorithm *Consistent* minimizes the expected number of strategy changes over time.

4. Algorithm *learn x* waits* for x boxes and updates the belief. After x boxes it selects the threshold used by the *consistent* algorithm.

**waiting* here means using the maximum threshold from all scenarios, as we must accept the price in this case.

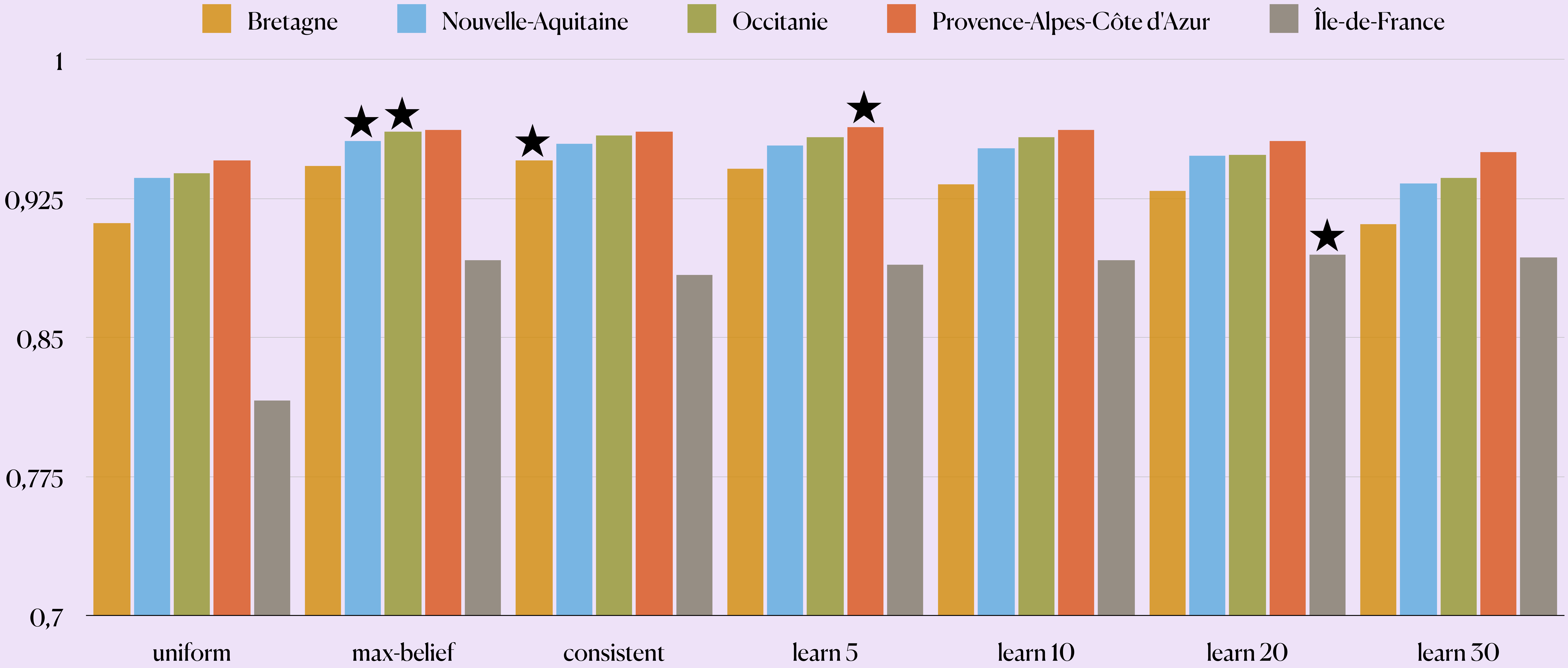
Evaluation

★ max ratio among all algorithms for that scenario



Evaluation

★ max ratio among all algorithms for that scenario



Results

- Extension of the prophet inequalities problem to a scenario-based setting.
- Tight upper bound on the competitive ratio.
- Algorithms that combine the optimal theoretical guarantee with adaptivity on the observed values.

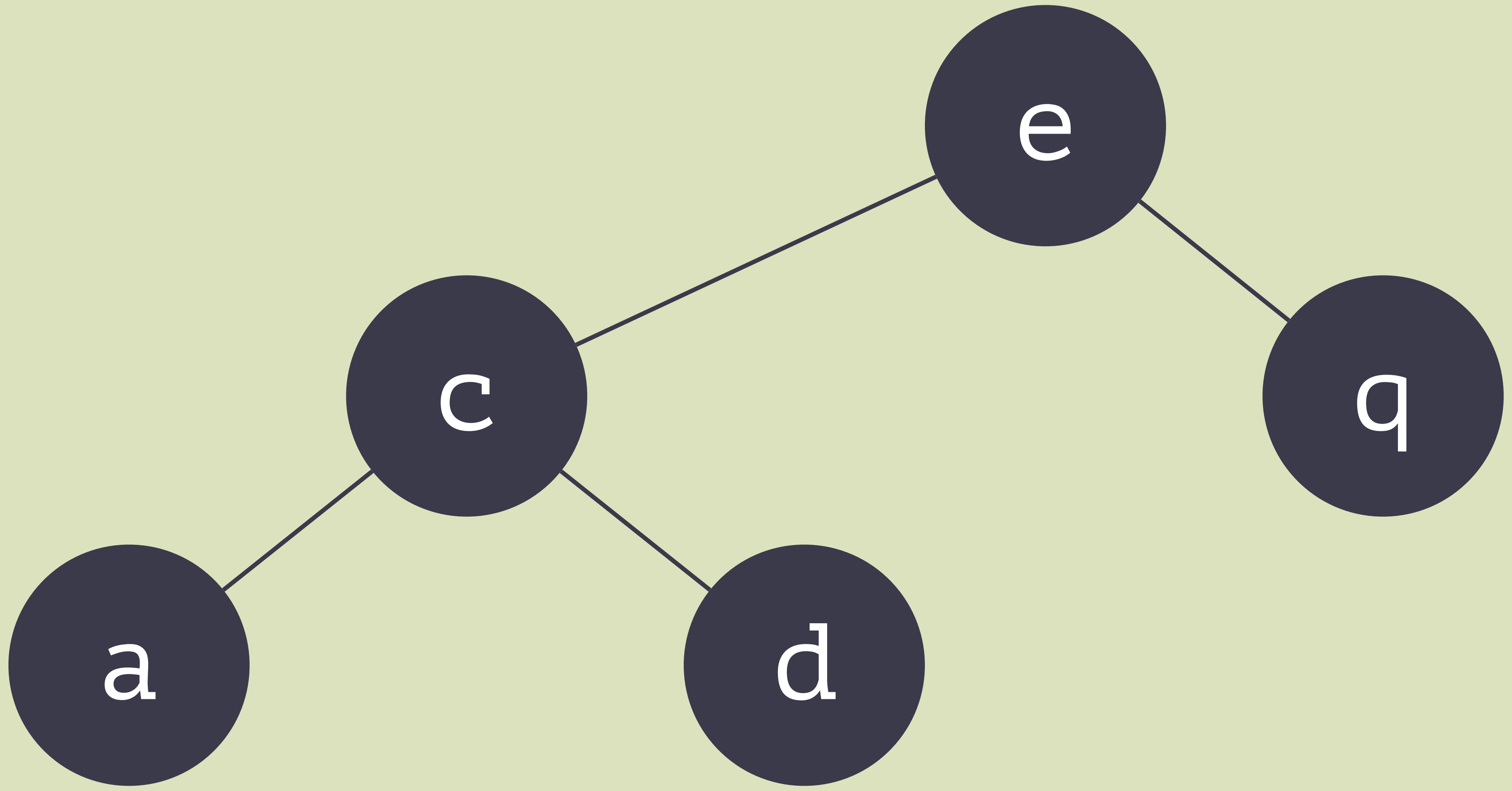
3. Scenario-Based Robust Optimization of Tree Structures

Binary search tree (BST) problem

BST stores n keys from a given ordered set $S = \{a, c, d, e, q\}$.

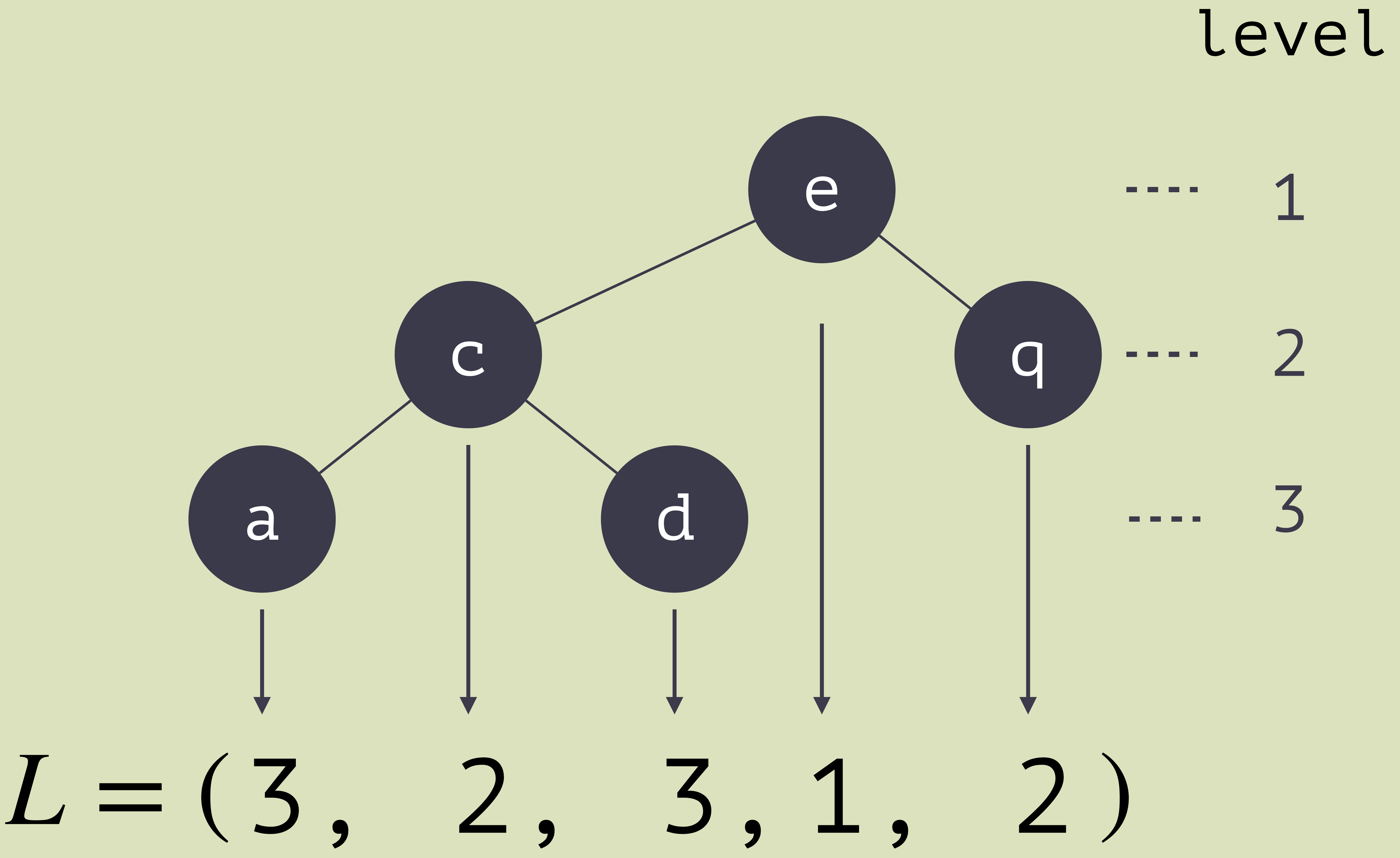
The keys are stored in nodes, and satisfy the ordering property:

the key of any node is larger than all keys in its left sub-tree, and smaller than all keys in its right sub-tree.



Binary search tree (BST) problem

A BST can be conveniently represented by its level vector L: every key i has level L_i in the tree.

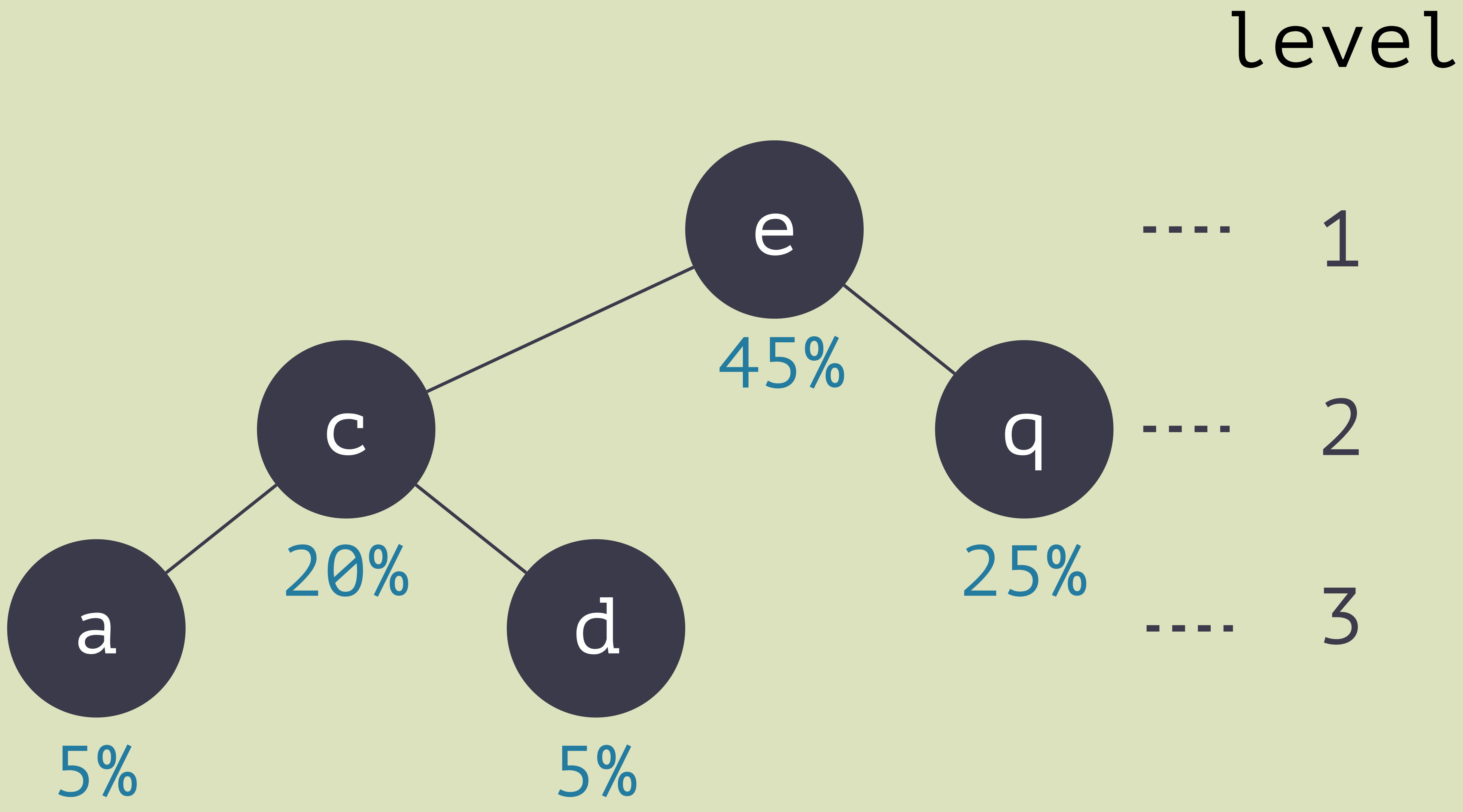


Binary search tree (BST) problem

Input: frequency vector F.

Goal: find a BST (level vector L) that minimises the average cost.

$$\text{cost}(L, F) = \sum_{i=1}^{n} L_i \cdot F_i$$



$L = (\ 3,\ 2,\ 3,\ 1,\ 2\ )$

$F = (\ 5\%,\ 20\%,\ 5\%,\ 45\%,\ 25\%\ )$

Donald E. Knuth. Optimum binary search trees. **1971.**

Robust Binary Search Tree

Input: k frequency vectors $F^1, \ldots, F^k$, each of dimension n, called *scenarios*.

Goal: produce a single binary search tree.

Robust Binary Search Tree

Input: k frequency vectors $F^1, \ldots, F^k$, each of dimension n, called *scenarios*.

Goal: produce a single binary search tree.

Three possible minimization problems on the BST level vector L:

Robust Binary Search Tree

Input: k frequency vectors $F^1, \ldots, F^k$, each of dimension n, called *scenarios*.

Goal: produce a single binary search tree.

Three possible minimization problems on the BST level vector L:

1. Worst-case cost: minimize $\max_{s \in k} \text{cost}(L, F^s)$

Robust Binary Search Tree

Input: k frequency vectors $F^1, \ldots, F^k$, each of dimension n, called *scenarios*.

Goal: produce a single binary search tree.

Three possible minimization problems on the BST level vector L:

1. Worst-case cost: minimize $\max_{s \in k} \text{cost}(L, F^s)$

2. Competitive ratio: minimize $\max_{s \in k} \dfrac{\text{cost}(L, F^s)}{\min_{L^*} \text{cost}(L^*, F^s)}$

Opt bst for F^s

Robust Binary Search Tree

Input: k frequency vectors $F^1, \ldots, F^k$, each of dimension n, called *scenarios*.

Goal: produce a single binary search tree.

Three possible minimization problems on the BST level vector L:

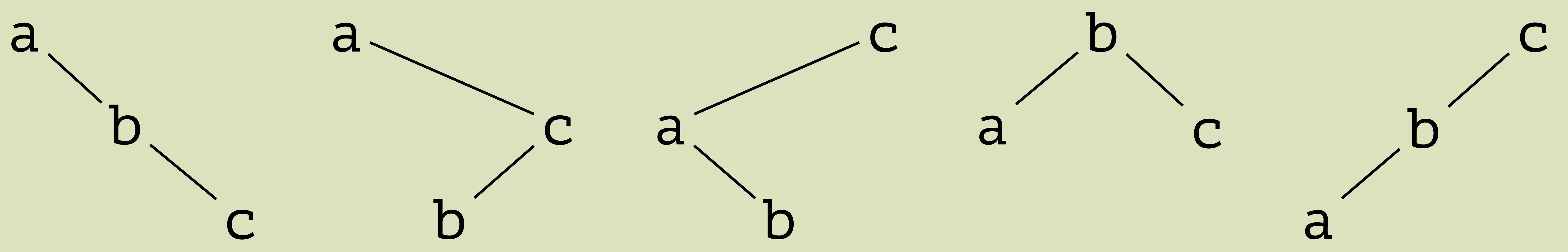
1. Worst-case cost: minimize $\max_{s \in k} \text{cost}(L, F^s)$

2. Competitive ratio: minimize $\max_{s \in k} \dfrac{\text{cost}(L, F^s)}{\min_{L^*} \text{cost}(L^*, F^s)}$

3. Regret: minimize $\max_{s \in k} \left\{ \text{cost}(L, F^s) - \min_{L^*} \text{cost}(L^*, F^s) \right\}$

Example

Let $F^1 = \left(0, \frac{1}{4}, \frac{3}{4}\right)$ and $F^2 = \left(\frac{4}{9}, \frac{2}{9}, \frac{1}{3}\right)$ denote two scenarios for three keys $\{a, b, c\}$.

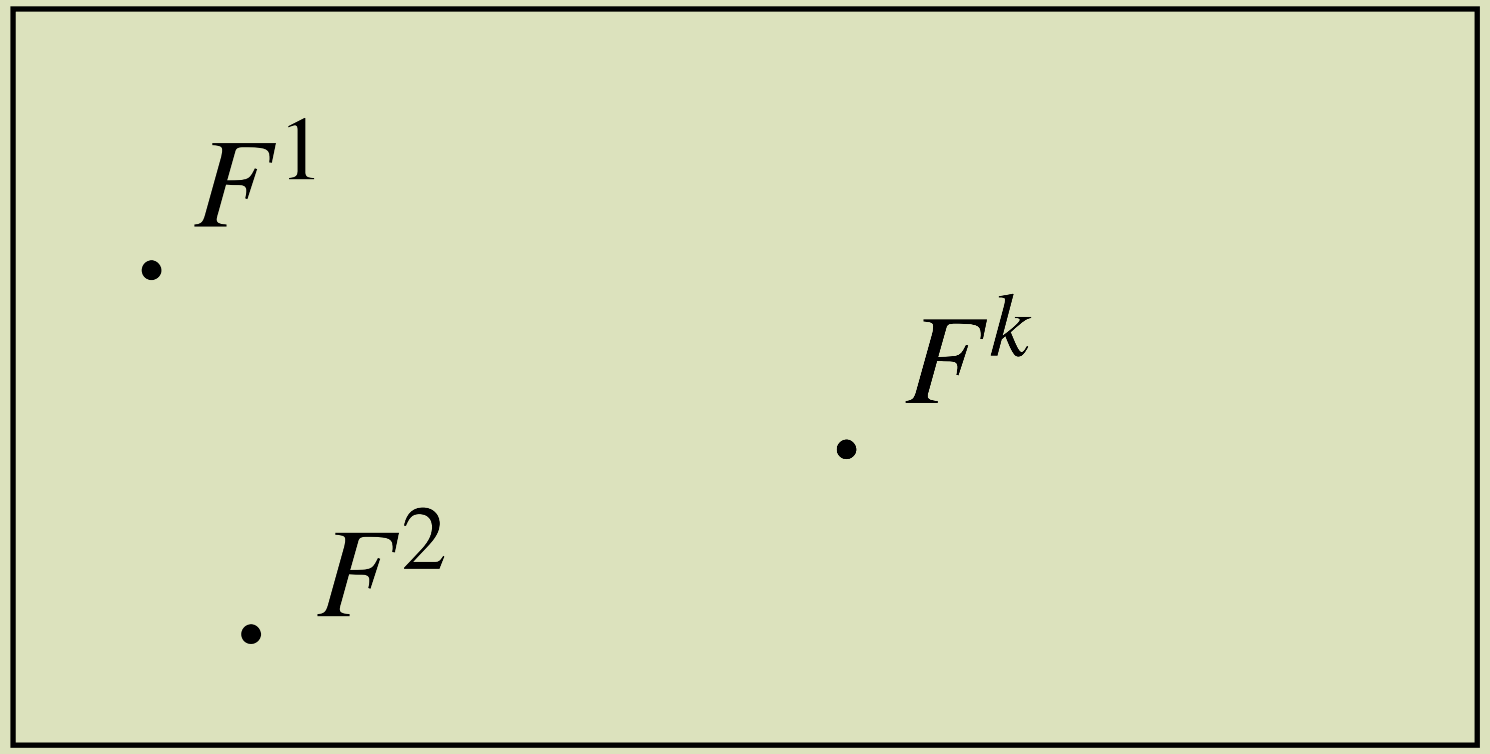


Worst cost	11/4	9/4	16/9	17/9	19/9
Competitive ratio	11/5	9/5	7/5	6/5	19/16
Regret	3/2	1	1/2	1/4	1/3

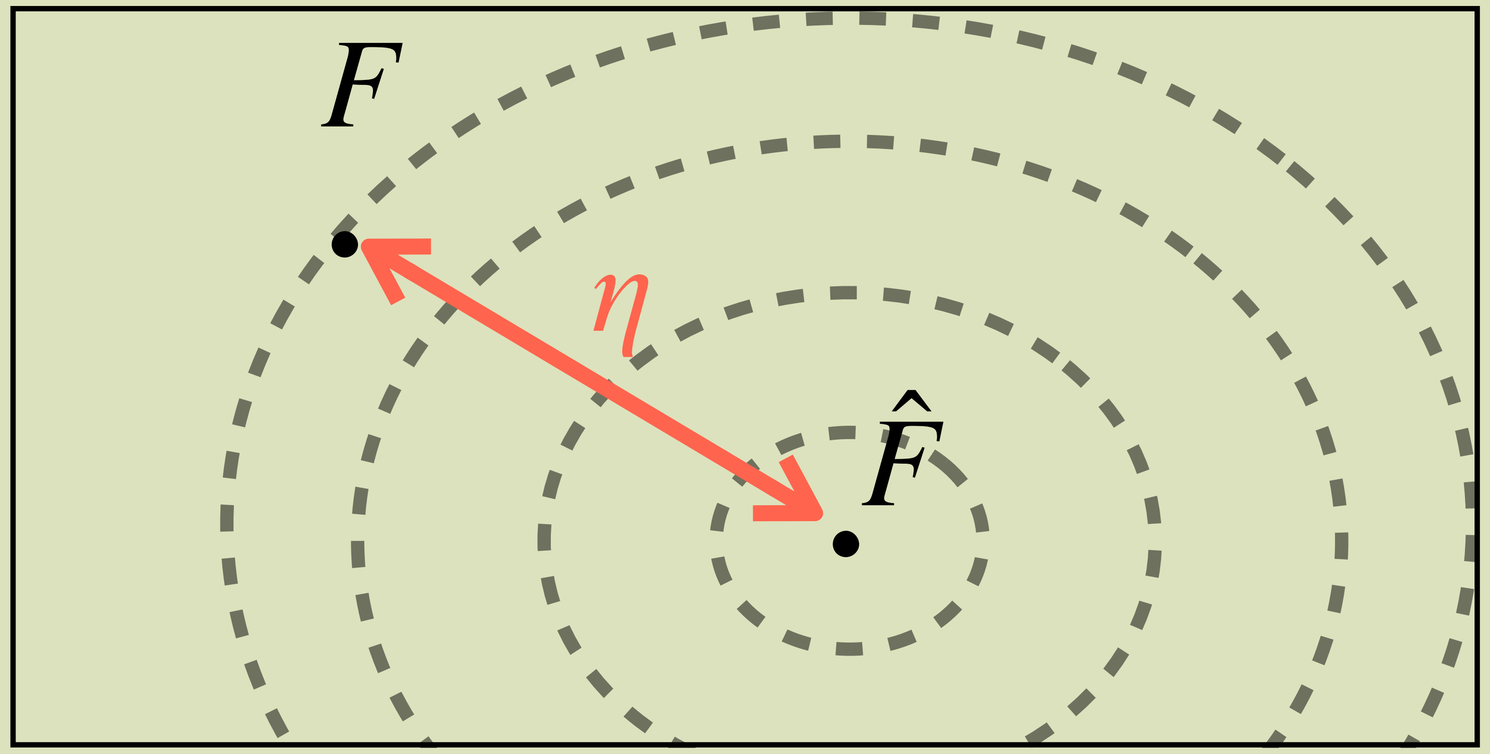
Related work

Binary Search with Distributional Predictions [11]

- Prediction: distribution $\hat{F}$
- True distribution F with distance η from $\hat{F}$
- Result: optimal binary search tree with cost $O(H(F) + \eta)$



Our model



Their model

[11] **Dinitz, Im, Lavastida, Moseley, Niaparast, Vasilvitskii. 2024**

Overview of our results

	Robust BST		
Worst-case cost	NP-hard	MILP formulation for the OPT	
Competitive ratio			Optimal $\lceil \log_2(k+1) \rceil$ algorithm
Regret			Investigate fairness*

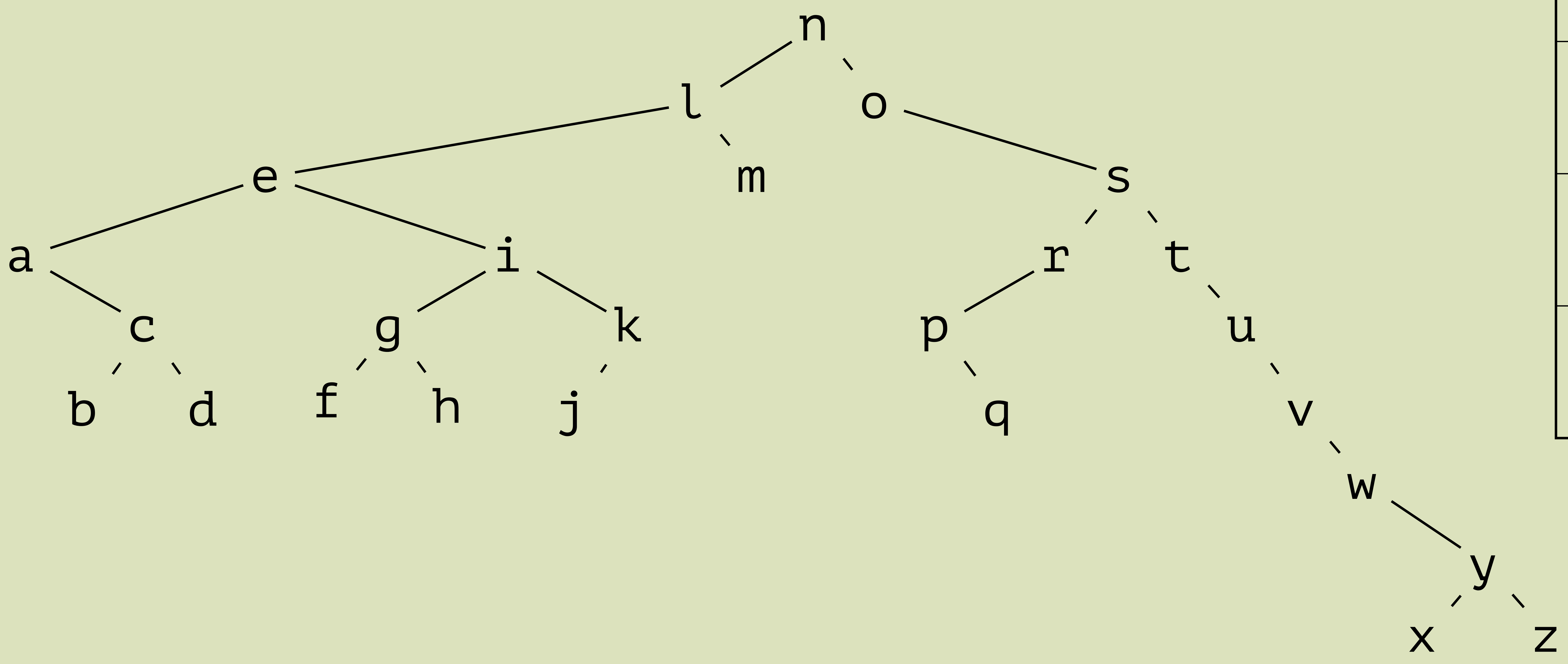
Computational experiments

10 scenarios are different European languages, based on the frequency of each letter in the corresponding language.

Letter	English	French	German	Spanish	Portuguese	
a	8.167	7.636	6.516	11.525	14.634	
b	1.492	0.901	1.886	2.215	1.043	
c	2.782	3.260	2.732	4.019	3.882	
d	4.253	3.669	5.076	5.010	4.992	
e	12.702	14.715	16.396	12.181	12.570	

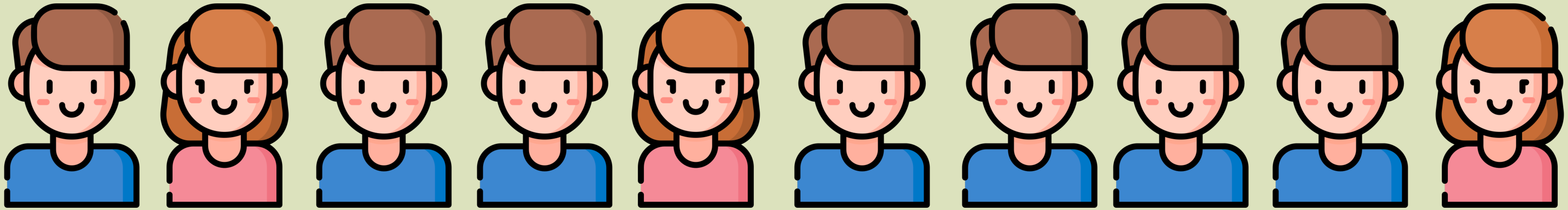
Computational experiments

Binary search tree for all languages:



	Optimal	R-bst
Cost	3.389	3.940
Ratio	1.047	1.215
Regret	0.151	0.680

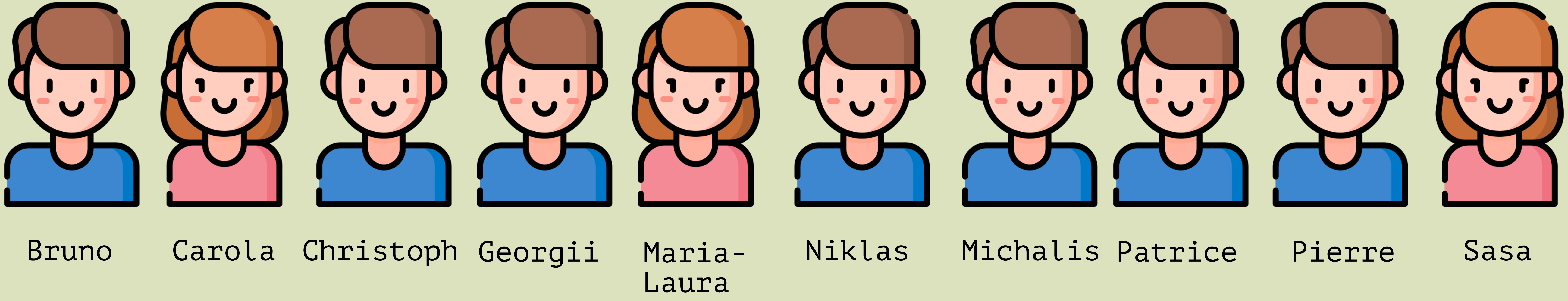
Regret and fairness in BST



$$F^1 = \left(0, \frac{1}{3}, 0, 0, \frac{1}{3}, 0, 0, 0, 0, \frac{1}{3}\right)$$

$$F^2 = \left(\frac{1}{7}, 0, \frac{1}{7}, \frac{1}{7}, 0, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, 0\right)$$

Regret and fairness in BST

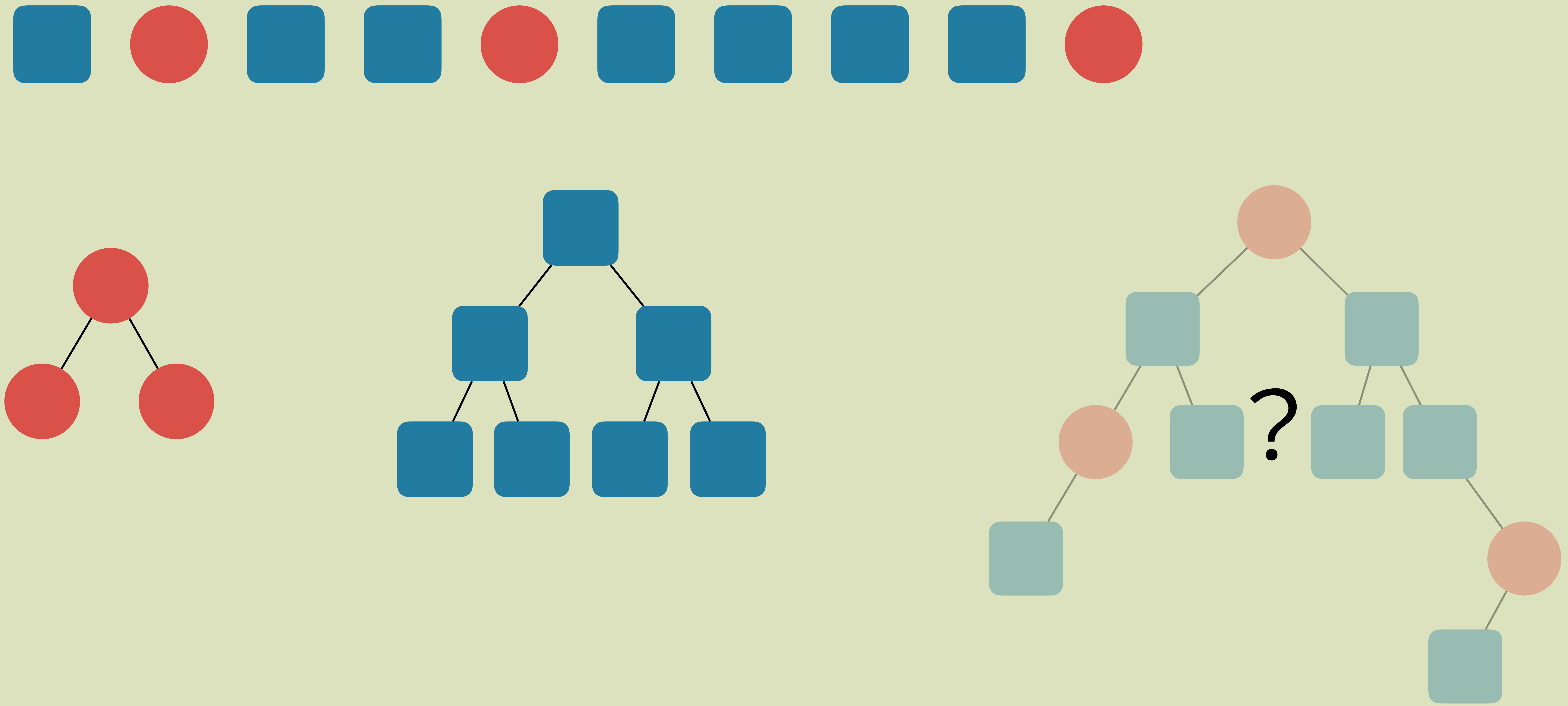


$$F^1 = \left(0, \frac{1}{3}, 0, 0, \frac{1}{3}, 0, 0, 0, 0, \frac{1}{3}\right)$$

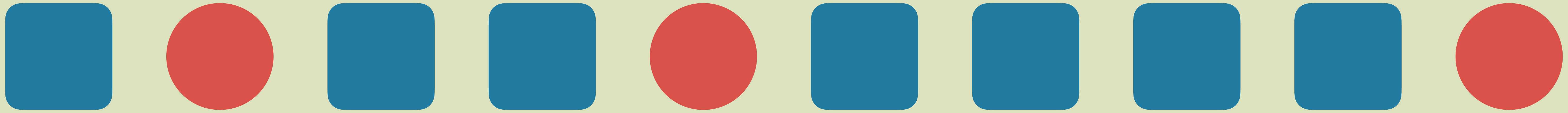
$$F^2 = \left(\frac{1}{7}, 0, \frac{1}{7}, \frac{1}{7}, 0, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, 0\right)$$

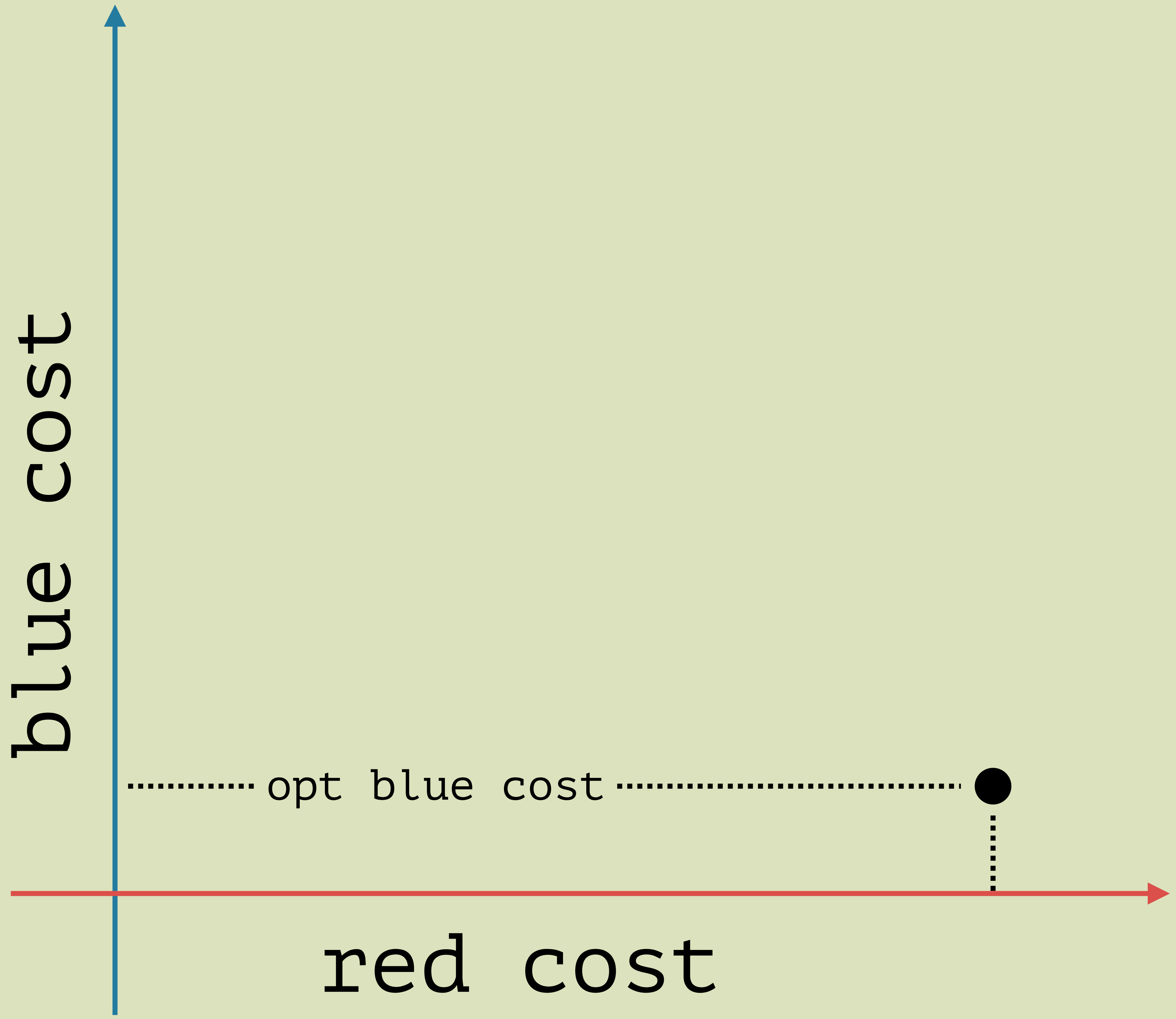
We want to store them in one database

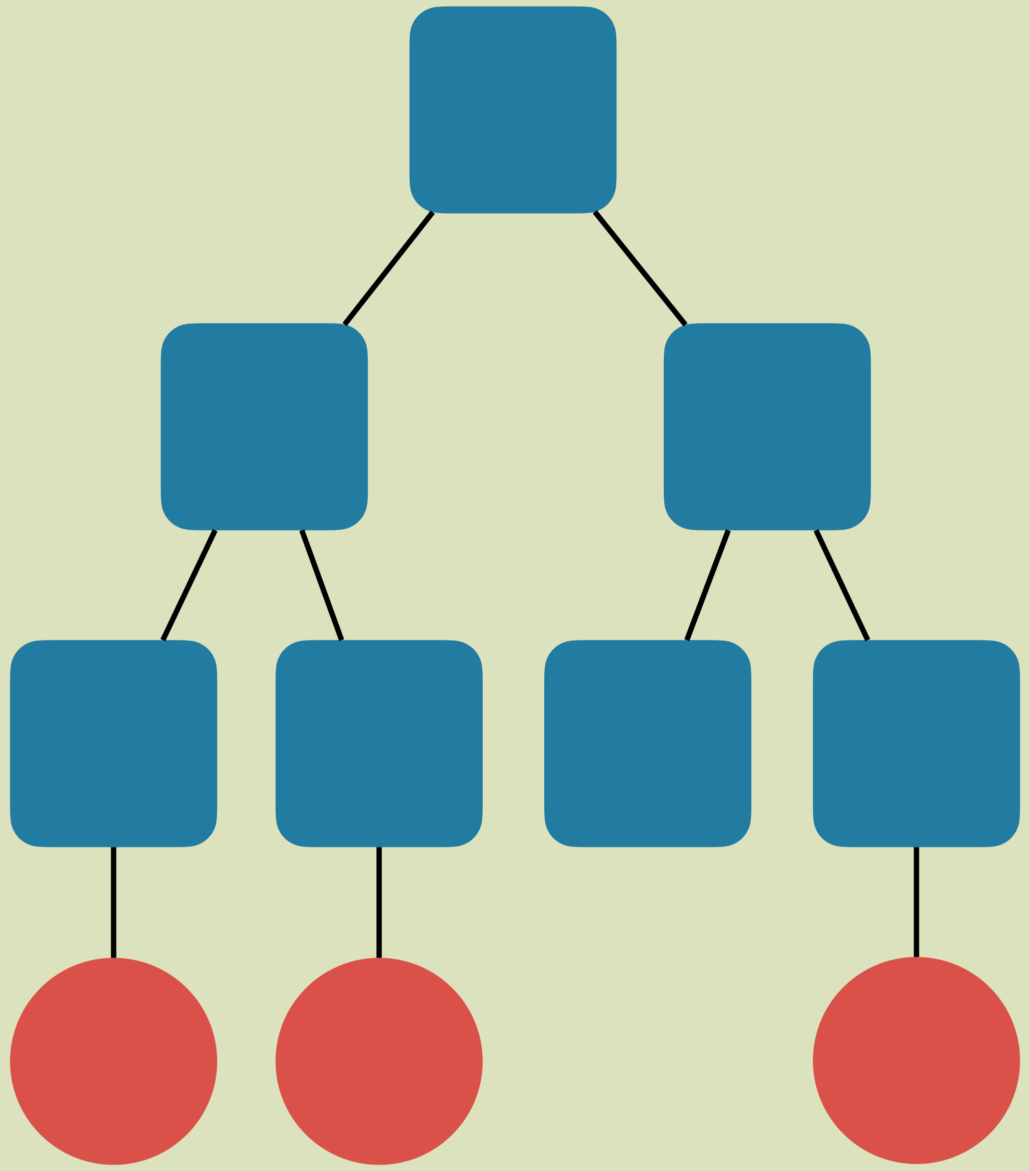
Regret and fairness in BST



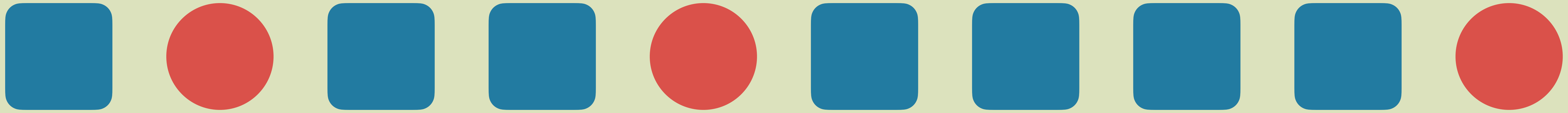
Regret and fairness in BST

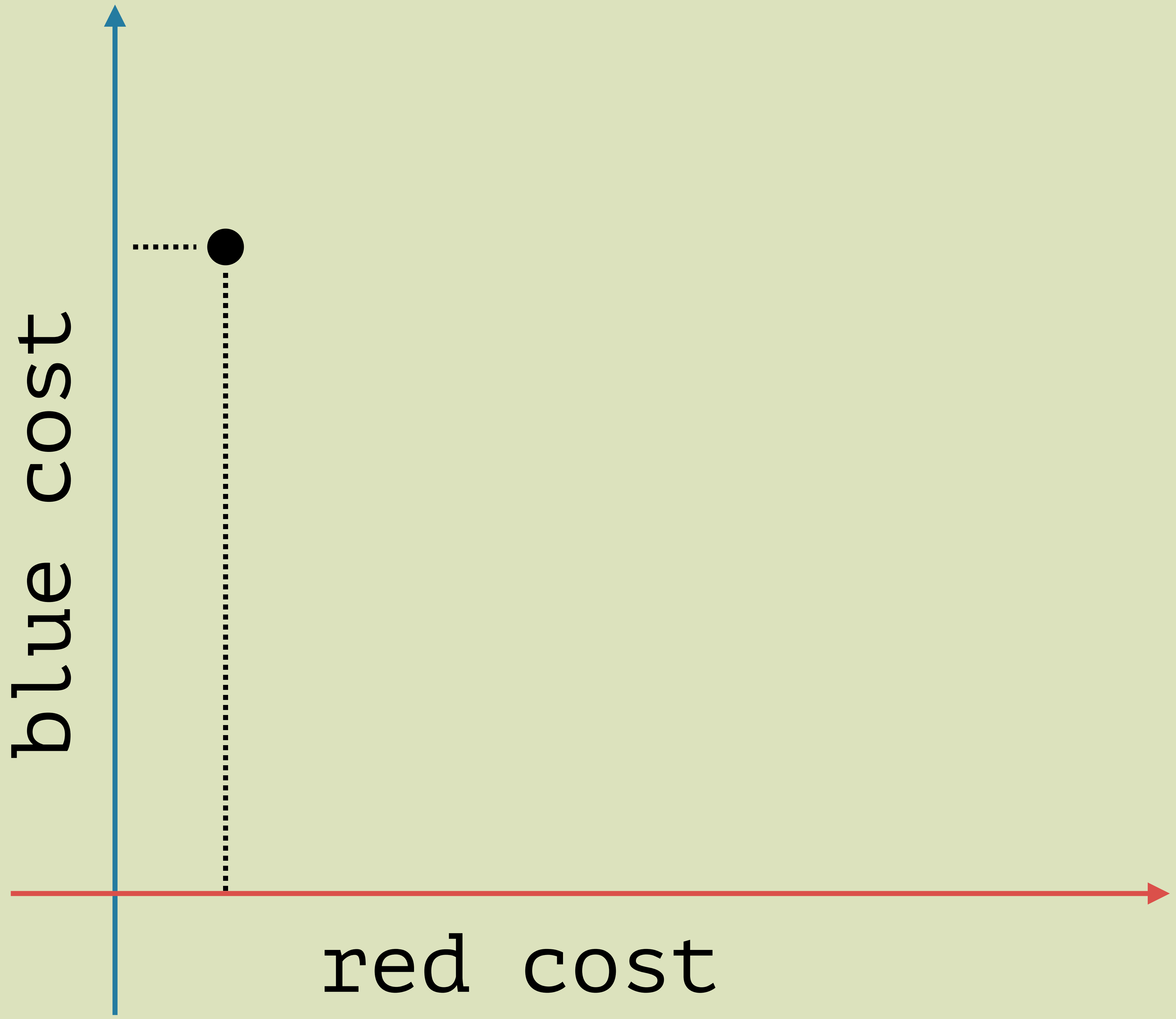


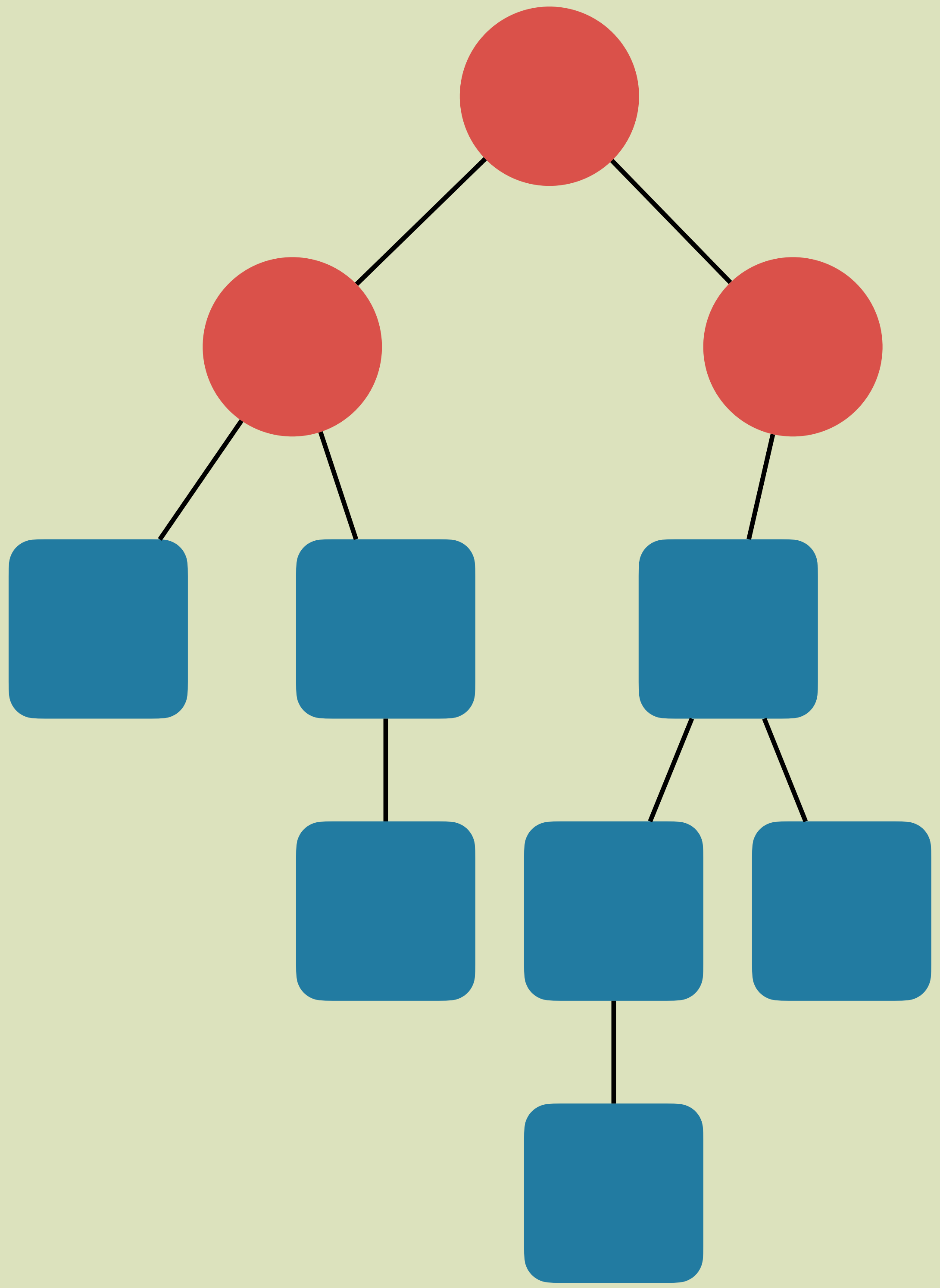




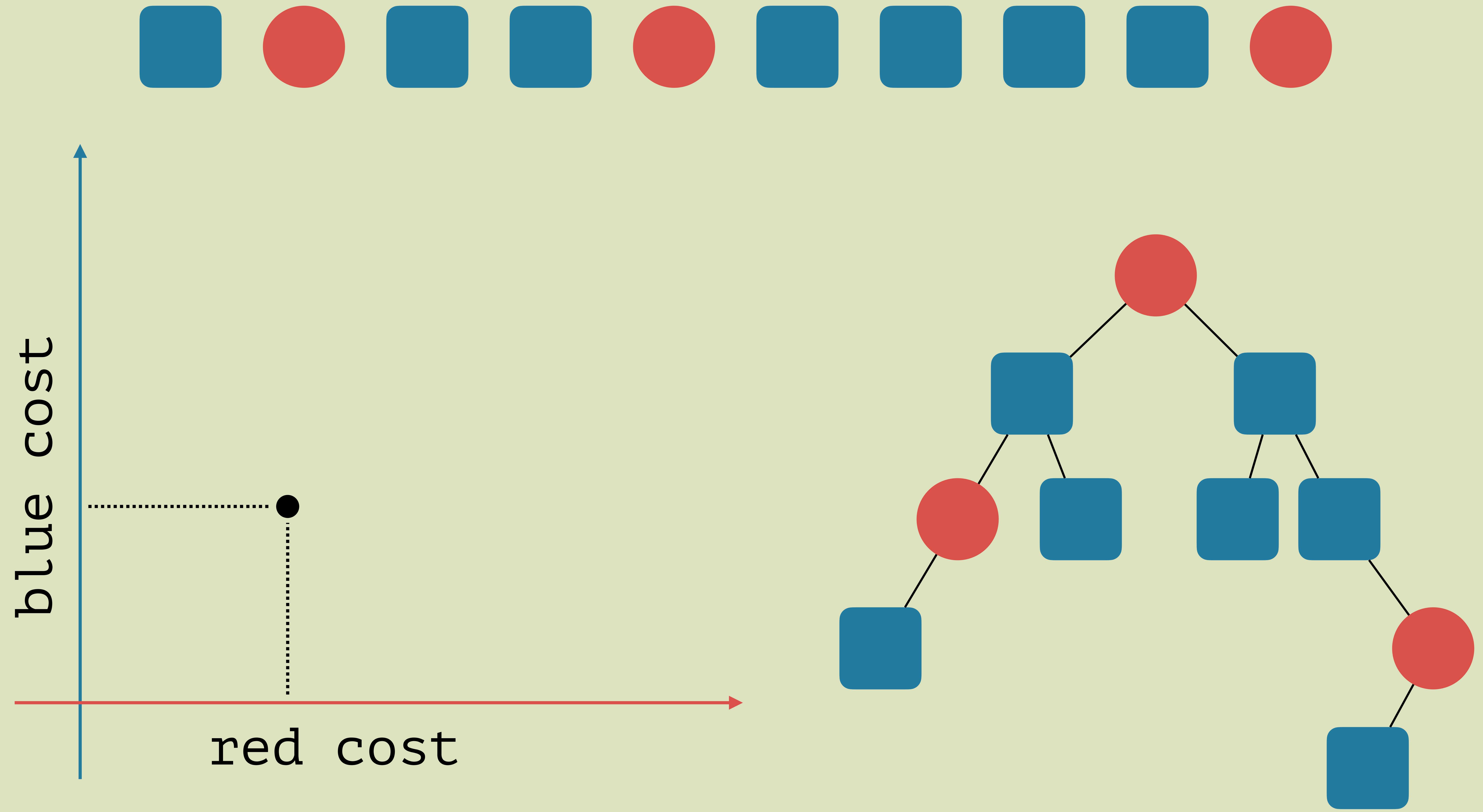
Regret and fairness in BST



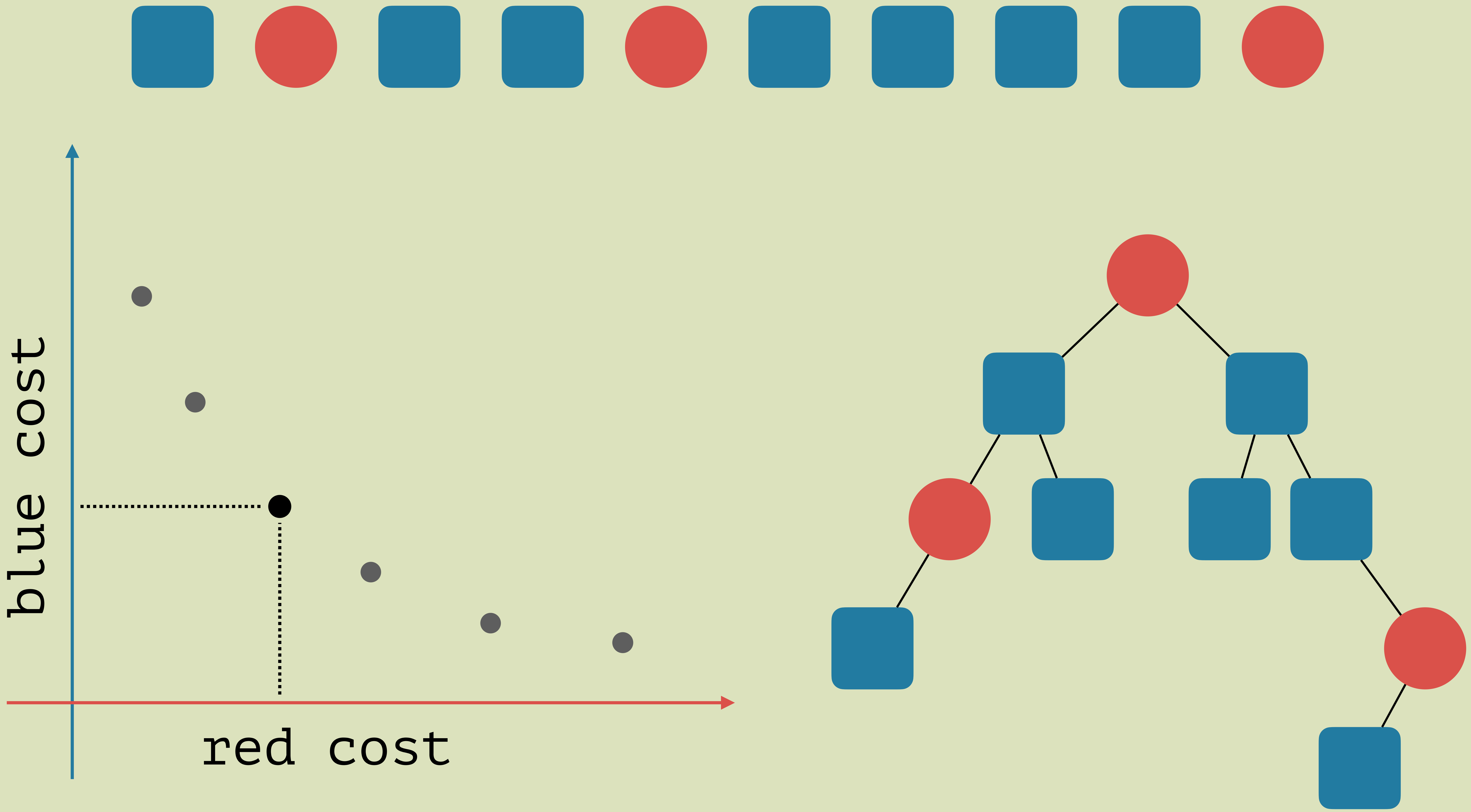




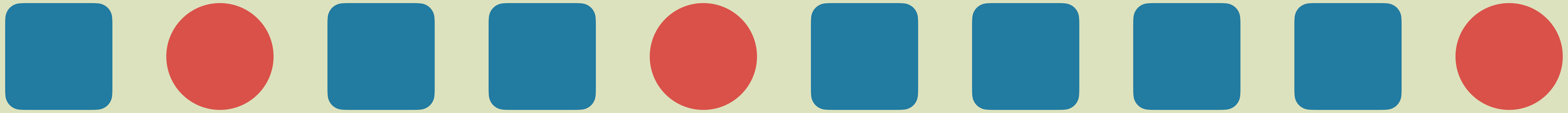
Regret and fairness in BST



Regret and fairness in BST



Regret and fairness in BST

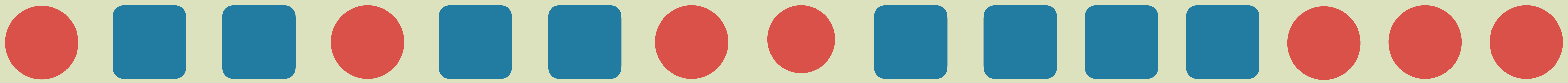


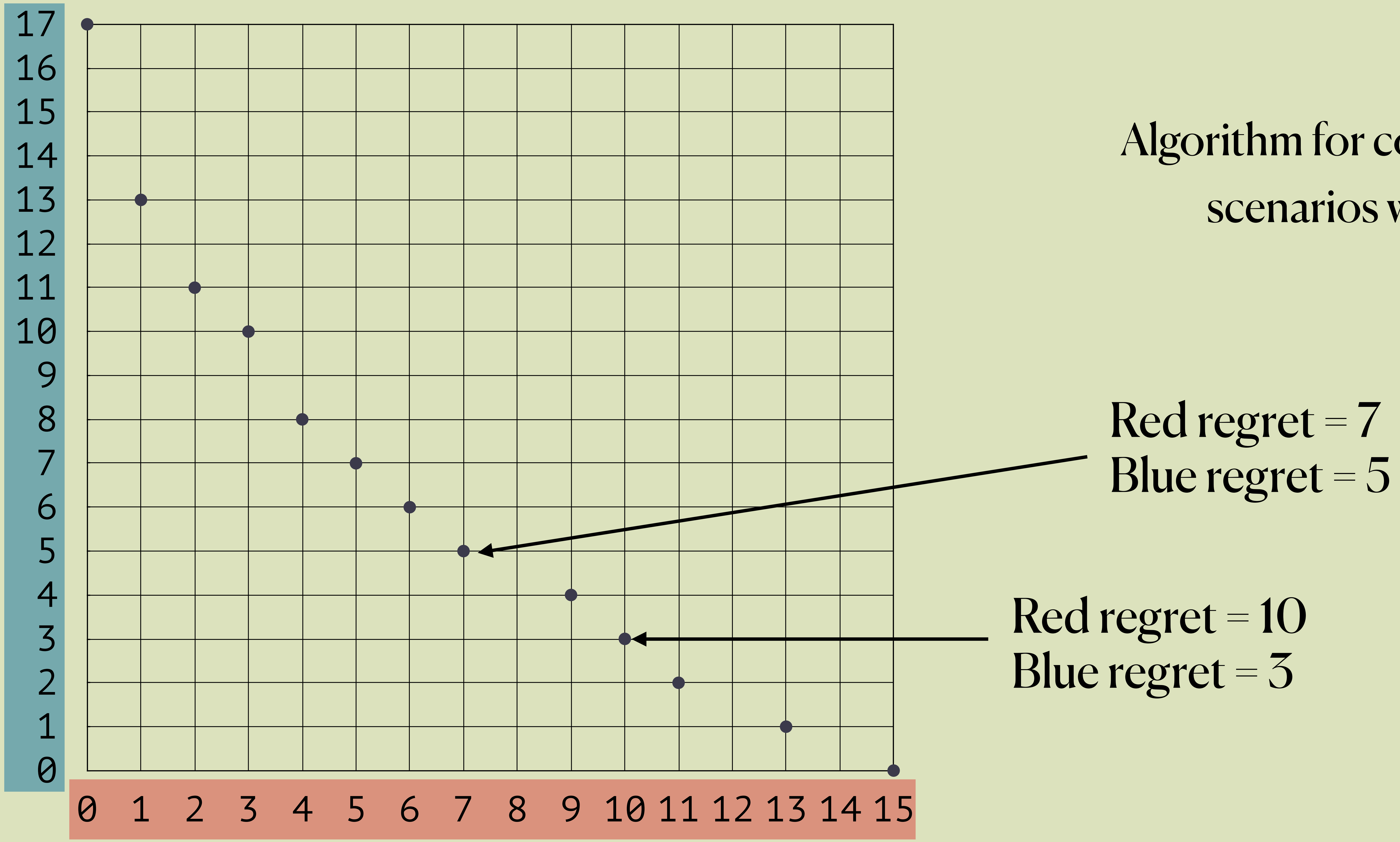
Red regret = red cost - opt red

Blue regret = blue cost - opt blue

«cost of being in a shared database»

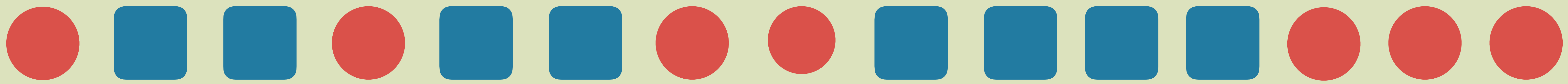
Computing the Pareto Frontier

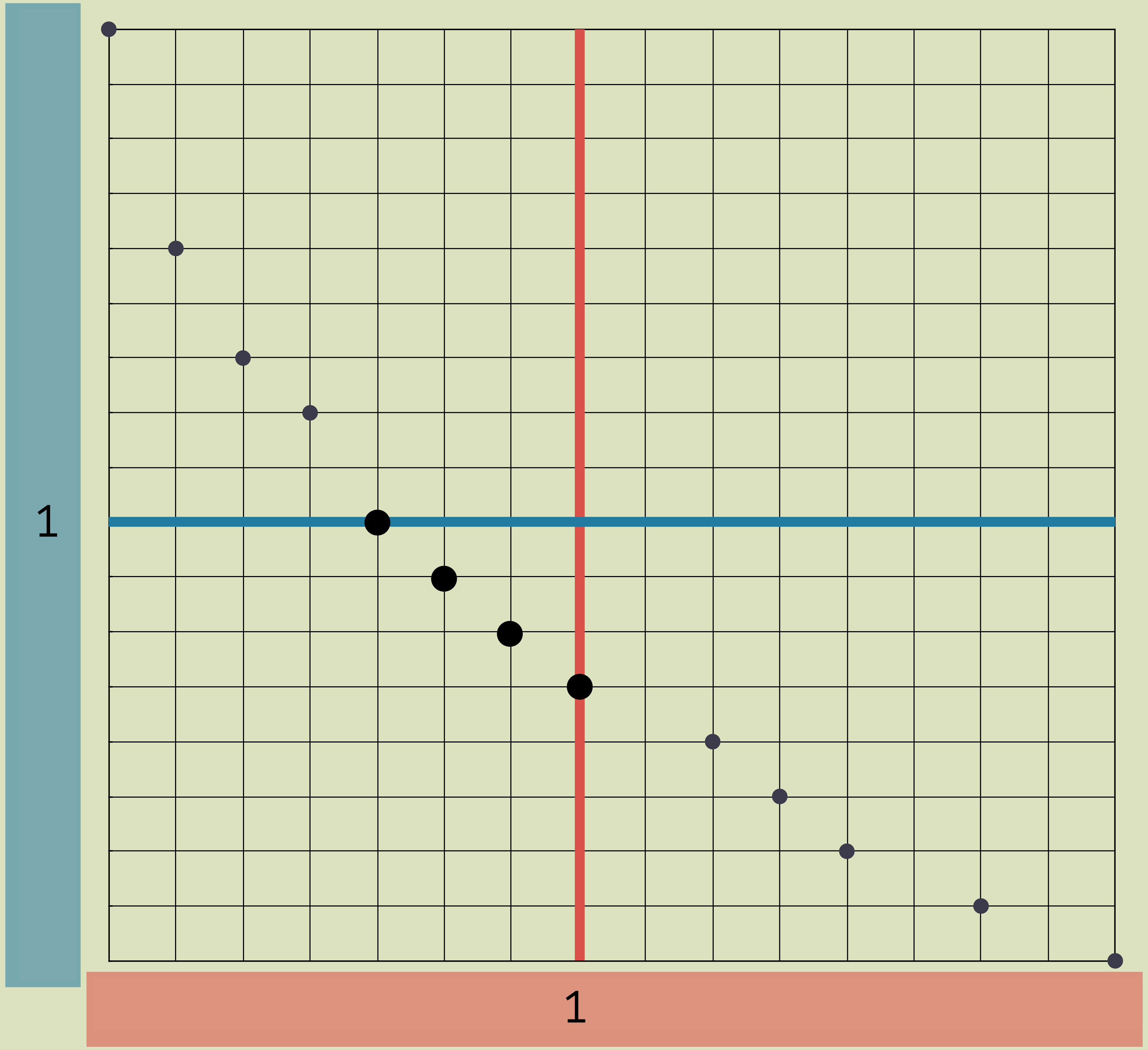




Algorithm for computing Pareto frontier for *k* scenarios with uniform distribution.

Regret and fairness in BST





Conjecture.

Red cost $\leq$ OPT(#red) + 1

Blue cost $\leq$ OPT(#blue) + 1

For all $a, b \in \{0,\ldots,11\}$, we generated all strings of size $a + b$ and checked the conjecture.

Regret and fairness in BST

Choose the root so that the total weight (frequency) of the left and right subtrees is as balanced as possible, and apply the rule recursively [12].

Theorem [13].
Let $\bar{F} = (F^1 + F^2)/2$ and $\bar{L}$ be the level vector produced by the Bisection rule construction on level vector $\bar{F}$, then

$$\text{cost}(\bar{L}, F^1) \leq \text{OPT}(F^1) + O(\log \log n),$$

and similarly for F^2.

[12] **Kurt Mehlhorn.** Nearly Optimal Binary Search Trees. **1975.**

[13] **Spyros Angelopoulos, Christoph Dürr, Alex Elenter and Georgii Melidi.** Scenario-Based Robust Optimization of Tree Structures. **2025.**

Conclusion

Two approaches:

- Scenario-based robust optimization
- Learning-augmented analysis

Main contributions:

- Formal frameworks + complexity results + fairness
- Positive and negative results
- Robust and decision-theoretic algorithms with provable guarantees
- New evaluation metrics (distance-based, risk-aware)
- Experimental validation on data structures, prophet inequalities, ski rental, contract scheduling and 1-max search

Future work

- **General data structures**: extend scenario-based analysis to B-trees, quad-trees, etc.
- **Combining predictions with scenarios**: algorithms that use predictions but stay robust across scenarios
- **Adaptive scenarios**: evolving/learned scenario sets over time (dynamic environments)
- **Real-world experiments**: finance, caching, sports → benchmark algorithms in practice

The big picture

- Modeling is important and is not obvious
- Our work contributes to theory → algorithms → experiments
- Goal: make algorithms reliable in uncertain environments

The big picture

- Modeling is important and is not obvious
- Our work contributes to theory → algorithms → experiments
- Goal: make algorithms reliable in uncertain environments

Thank you!