Karlsruhe Institute of Technology

Decision-Making in Data Streams under Limited Feedback

Marco Heyden

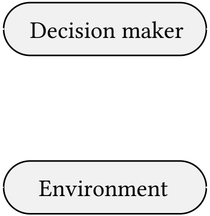
Doctoral Defense | 13. 02. 2025

The Decision-Making Process

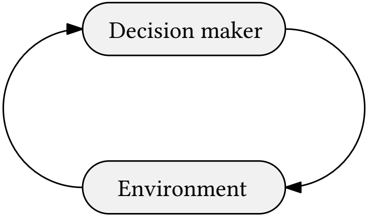
The Decision-Making Process

Decision maker

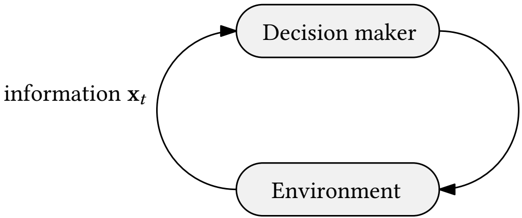
The Decision-Making Process



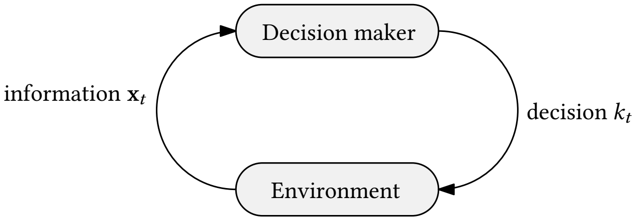
The Decision-Making Process



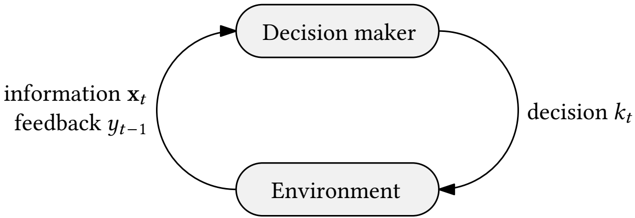
The Decision-Making Process



The Decision-Making Process

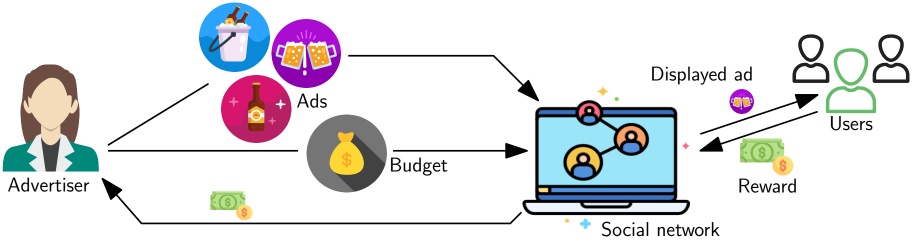


The Decision-Making Process



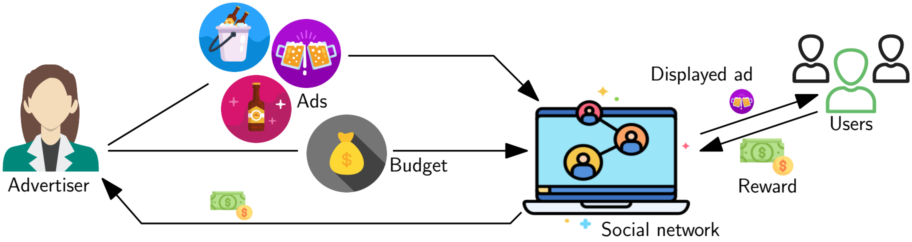
The Decision-Making Process
Example 1: Online advertising

- **Decision maker** is responsible for selecting ads
- **Decision:** which ad to show to users
- **Information:** none / user information (e.g., age, gender)
- **Feedback:** conversions, advertising costs



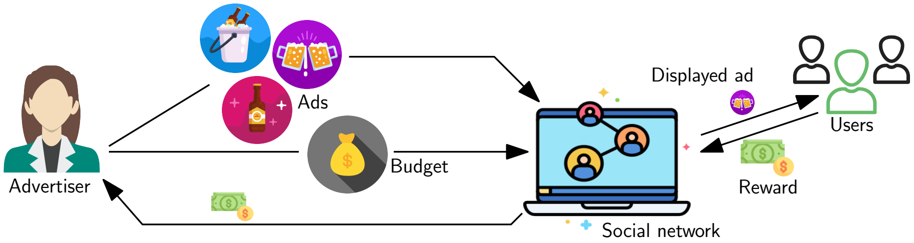
The Decision-Making Process
Example 1: Online advertising

- **Decision maker** is responsible for selecting ads
- **Decision:** which ad to show to users
- **Information:** none / user information (e.g., age, gender)
- **Feedback:** conversions, advertising costs



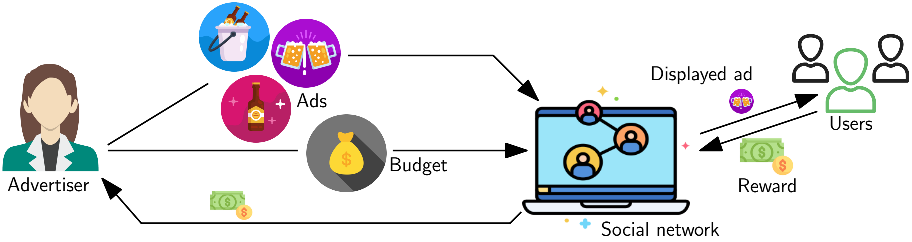
The Decision-Making Process
Example 1: Online advertising

- **Decision maker** is responsible for selecting ads
- **Decision:** which ad to show to users
- **Information:** none / user information (e.g., age, gender)
- **Feedback:** conversions, advertising costs



The Decision-Making Process
Example 1: Online advertising

- **Decision maker** is responsible for selecting ads
- **Decision:** which ad to show to users
- **Information:** none / user information (e.g., age, gender)
- **Feedback:** conversions, advertising costs



The Decision-Making Process
Example 2: Biofuel production

- **Decision maker** is responsible for smooth operation of biofuel production plant
- **Decision:** stop the plant or continue
- **Information:** sensor readings (vibration, heat, pressure, chemical compounds)
- **Feedback:** Produced biofuel is of expected quality

Example 2: Biofuel production

Karlsruhe Institute of Technology

- **Decision maker** is responsible for smooth operation of biofuel production plant
- **Decision:** stop the plant or continue
- **Information:** sensor readings (vibration, heat, pressure, chemical compounds)
- **Feedback:** Produced biofuel is of expected quality

The Decision-Making Process
Example 2: Biofuel production

- **Decision maker** is responsible for smooth operation of biofuel production plant
- **Decision:** stop the plant or continue
- **Information:** sensor readings (vibration, heat, pressure, chemical compounds)
- **Feedback:** Produced biofuel is of expected quality

Example 2: Biofuel production

Karlsruhe Institute of Technology

- **Decision maker** is responsible for smooth operation of biofuel production plant
- **Decision:** stop the plant or continue
- **Information:** sensor readings (vibration, heat, pressure, chemical compounds)
- **Feedback:** Produced biofuel is of expected quality

Decision-Making
Decision support systems

- Human decision-making is prone to errors and bias [TK74]

⇒ Use **decision support systems** (DSS) to guide the decision maker

Decision-Making
Decision support systems

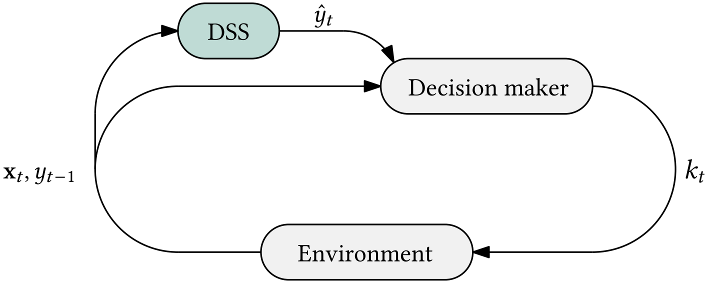
- Human decision-making is prone to errors and bias [TK74]

$\Rightarrow$ Use **decision support systems** (DSS) to guide the decision maker

Decision-Making
Decision support systems

- Human decision-making is prone to errors and bias [TK74]

$\Rightarrow$ Use **decision support systems** (DSS) to guide the decision maker



How to design them using machine learning?

Traditional process:

1. Collect data
2. Apply supervised learning

Challenges:

- **Sequential data:** New data only becomes available over time
- **Dynamic environments** change over time, e.g., due to wear and tear or shifting user preferences

$\Rightarrow$ **Main drivers of research on data streams [Bif+18]**

Decision Support Systems
How to design them using machine learning?

Traditional process:

1. Collect data
2. Apply supervised learning

Challenges:

- **Sequential data:** New data only becomes available over time
- **Dynamic environments** change over time, e.g., due to wear and tear or shifting user preferences

$\Rightarrow$ **Main drivers of research on data streams [Bif+18]**

Decision Support Systems
How to design them using machine learning?

Traditional process:

1. Collect data
2. Apply supervised learning

Challenges:

- **Sequential data:** New data only becomes available over time
- **Dynamic environments** change over time, e.g., due to wear and tear or shifting user preferences

$\Rightarrow$ **Main drivers of research on data streams [Bif+18]**

Decision Support Systems
How to design them using machine learning?

Traditional process:

1. Collect data
2. Apply supervised learning

Challenges:

- **Sequential data:** New data only becomes available over time
- **Dynamic environments** change over time, e.g., due to wear and tear or shifting user preferences

$\Rightarrow$ **Main drivers of research on data streams [Bif+18]**

Never-ending sequences of observations

Karlsruhe Institute of Technology

Data Stream

A *data stream* S is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Data Streams
Never-ending sequences of observations

Data Stream

A *data stream* S is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Data Streams
Never-ending sequences of observations

Data Stream

A *data stream* S is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Never-ending sequences of observations

Karlsruhe Institute of Technology

Data Stream

A *data stream S* is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Data Streams
Never-ending sequences of observations

Data Stream

A *data stream S* is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Data Streams
Never-ending sequences of observations

Data Stream

A *data stream S* is a possibly never-ending sequence of observations $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_t, y_t), \ldots\}$ drawn from an ordered set of data generating distributions $\{\mathcal{S}_{\tau_1,\tau_2}, \mathcal{S}_{\tau_2,\tau_3}, \mathcal{S}_{\tau_3,\tau_4}, \ldots\}$, called *concepts*, such that

$$\forall t \in [\tau_i, \tau_{i+1}) : (\mathbf{x}_t, y_t) \overset{iid}{\sim} \mathcal{S}_{\tau_i,\tau_{i+1}}.$$

ML algorithms for data streams should

- Inspect each observation only once
- Use limited amount of time and memory
- Adapt to concept drift (change from one concept to another)

Research Gap
Learning from limited feedback

- Most algorithms for data streams assume **plenty and cheap** feedback
- Many applications **violate** these assumptions

Research Gap
Learning from limited feedback

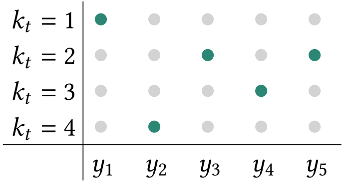
- Most algorithms for data streams assume **plenty and cheap** feedback
- Many applications **violate** these assumptions

Research Gap
Learning from limited feedback

- Most algorithms for data streams assume **plenty and cheap** feedback
- Many applications **violate** these assumptions

Decision-based feedback

- Feedback only available for the chosen decision

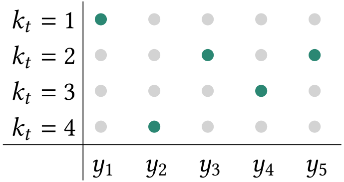


Research Gap
Learning from limited feedback

- Most algorithms for data streams assume **plenty and cheap** feedback
- Many applications **violate** these assumptions

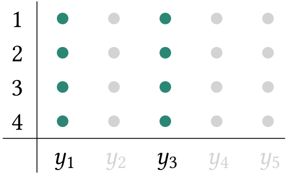
Decision-based feedback

- Feedback only available for the chosen decision



Observation-based feedback

- Feedback is only available for some observations
- Extreme case: unavailable feedback

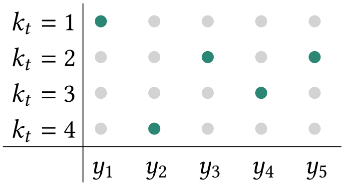


Research Gap
Learning from limited feedback

- Most algorithms for data streams assume **plenty and cheap** feedback
- Many applications **violate** these assumptions

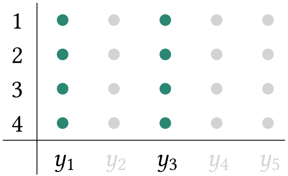
Decision-based feedback

- Feedback only available for the chosen decision



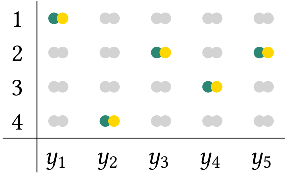
Observation-based feedback

- Feedback is only available for some observations
- Extreme case: unavailable feedback



Costly feedback

- Obtaining feedback comes at a cost



This Dissertation
Addresses limited feedback from three perspectives

Limited Feedback

This Dissertation
Addresses limited feedback from three perspectives

Part I
- decision-based
- costly

Budgeted multi-armed bandits with asymmetric confidence intervals

KDD '24

Limited Feedback

This Dissertation
Addresses limited feedback from three perspectives

Limited Feedback

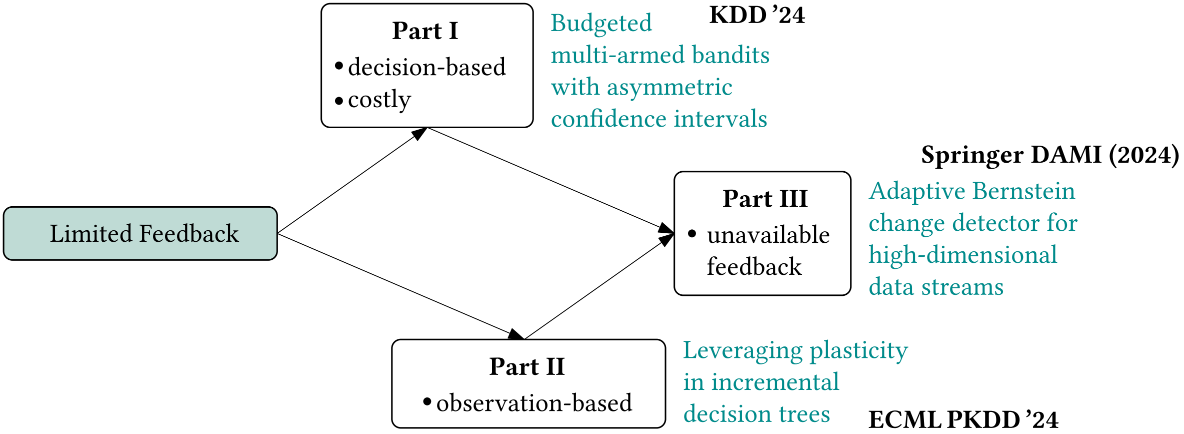
Part I
- decision-based
- costly

Budgeted multi-armed bandits with asymmetric confidence intervals **KDD '24**

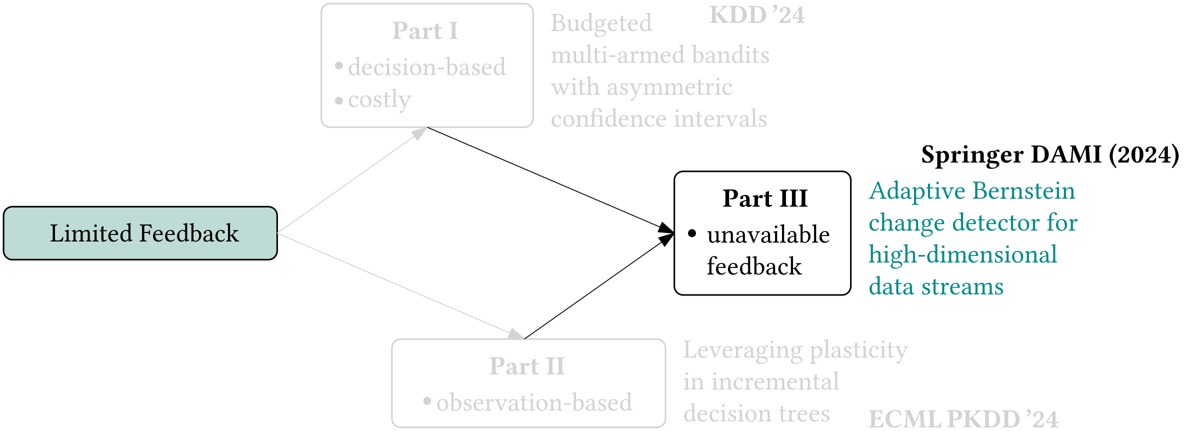
Part II
- observation-based

Leveraging plasticity in incremental decision trees **ECML PKDD '24**

This Dissertation
Addresses limited feedback from three perspectives



Next up
Part III — ABCD



Contribution
Feedback generation in complex systems

Research question:

How to guide decision-making when no direct feedback from the environment is available?

Solution: Let the algorithm generate feedback!

Technical contributions:

- ABCD, a change detection and characterization algorithm for high-dimensional data streams
  - “When”, “where”, and “how severely”
- Formalization of change, change subspace, and change severity
- Stream aggregates for adaptive windows

Contribution
Feedback generation in complex systems

Research question:

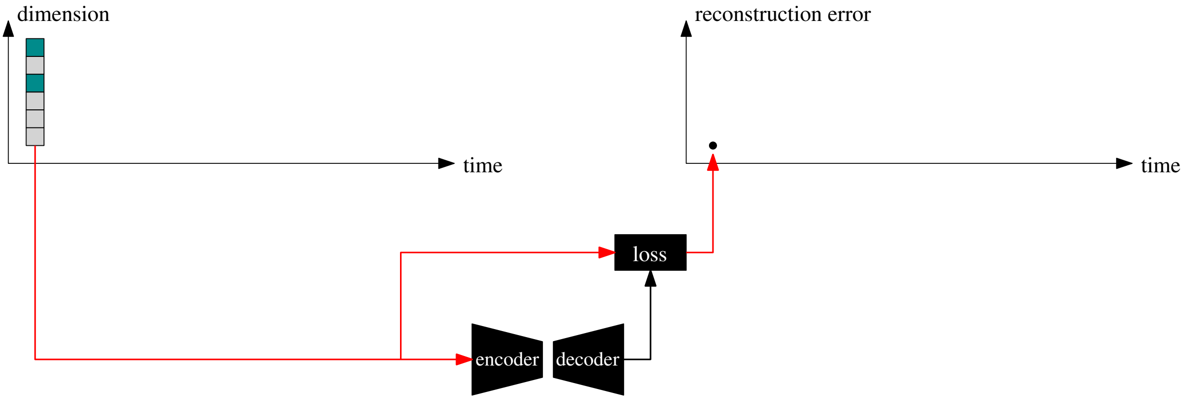
How to guide decision-making when no direct feedback from the environment is available?

Solution: **Let the algorithm generate feedback!**

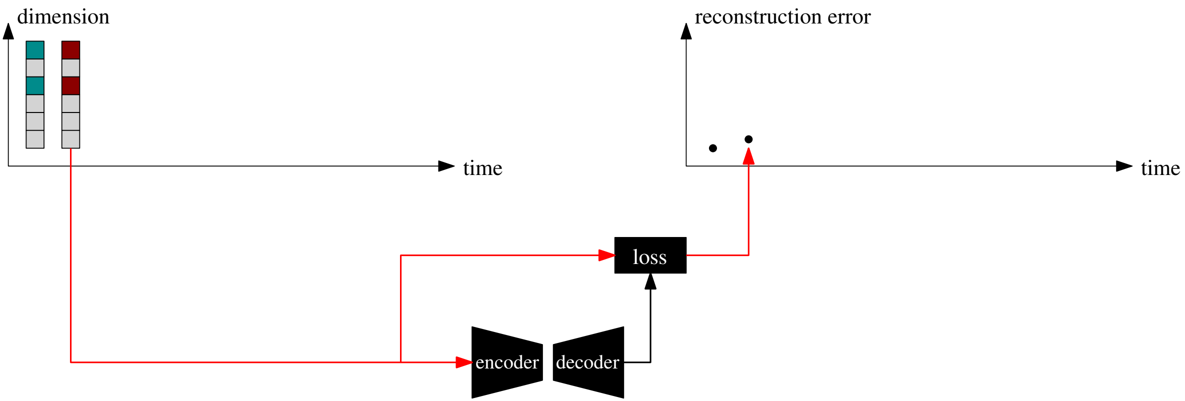
Technical contributions:

- ABCD, a change detection and characterization algorithm for high-dimensional data streams
  - “When”, “where”, and “how severely”
- Formalization of change, change subspace, and change severity
- Stream aggregates for adaptive windows

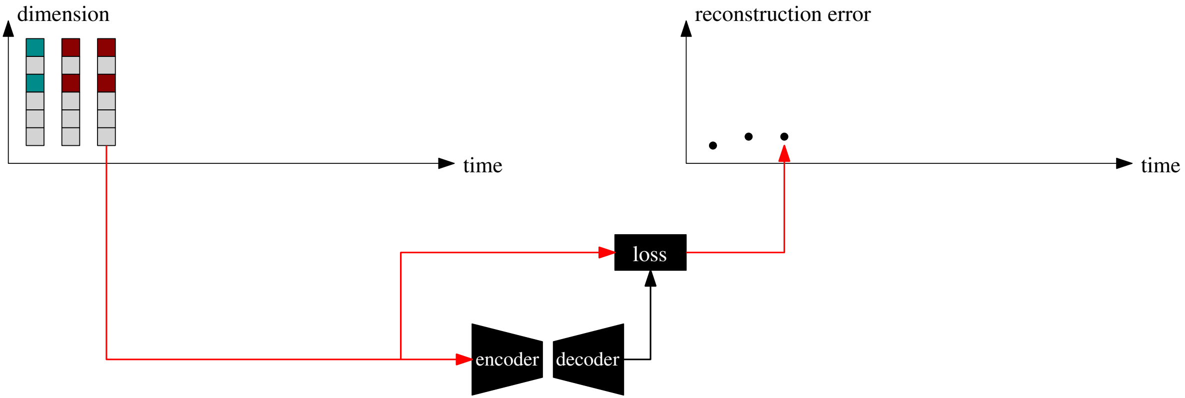
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



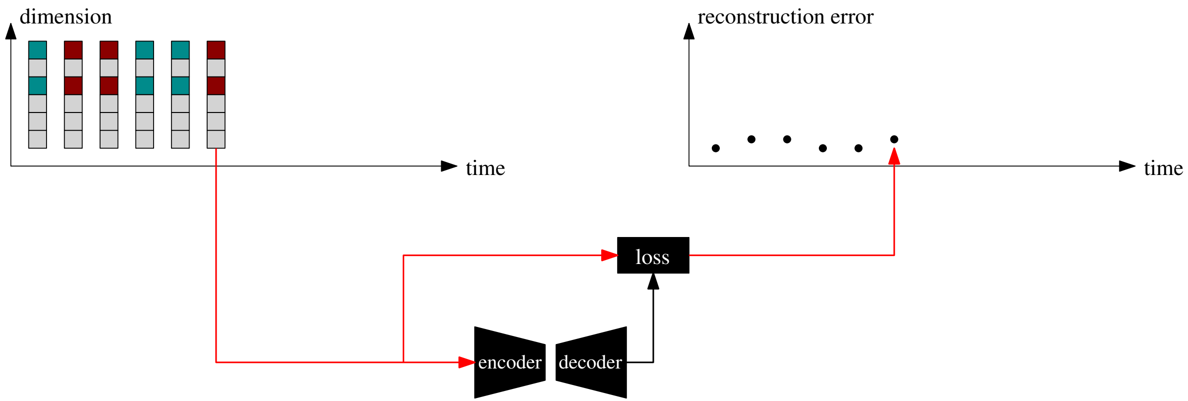
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



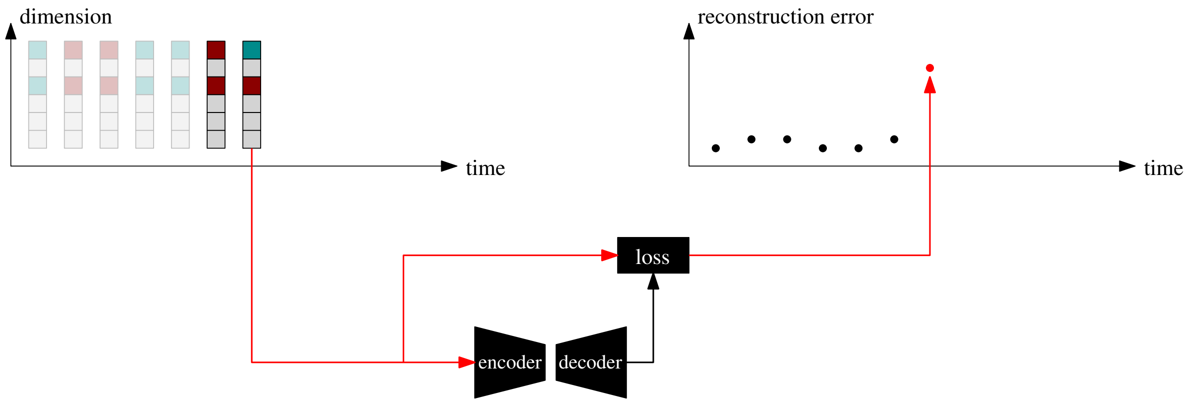
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



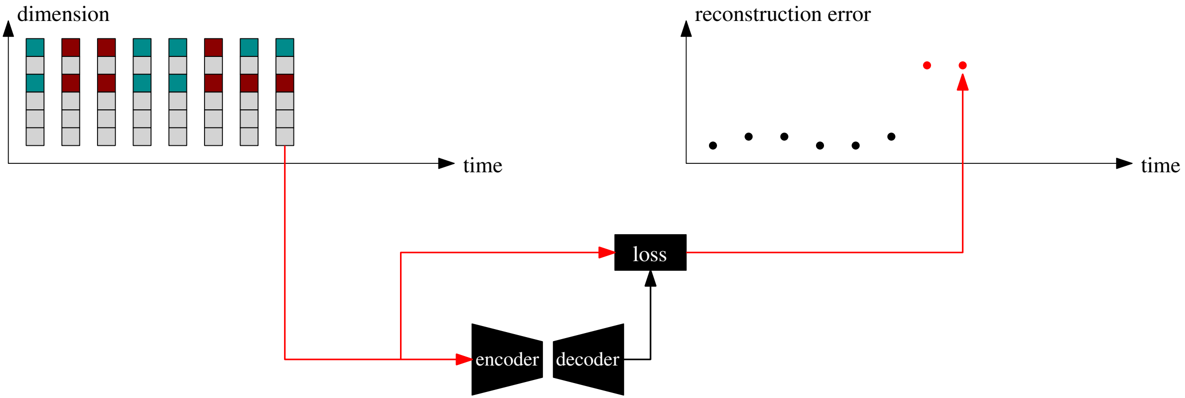
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



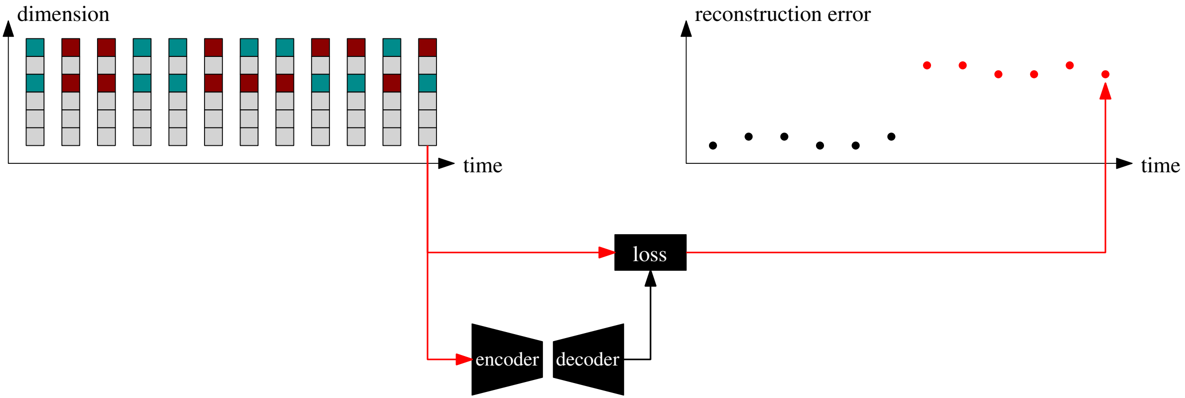
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



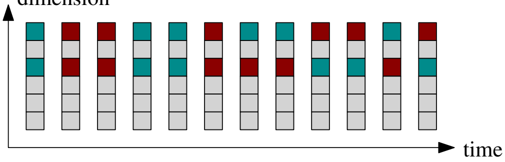
High-level Algorithm
Monitor reconstruction loss of encoder-decoder model

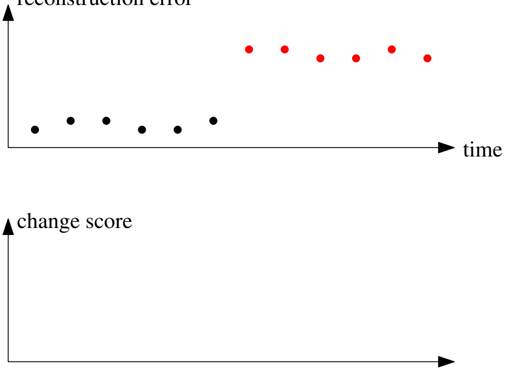


High-level Algorithm
Monitor reconstruction loss of encoder-decoder model



High-level Algorithm
Compute change score for possible change points

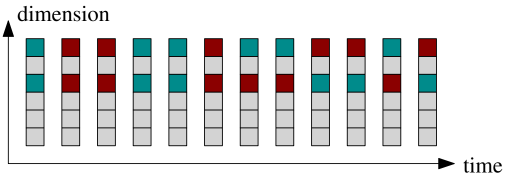


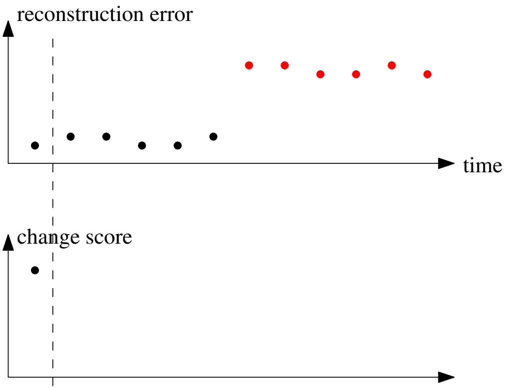


$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Compute change score for possible change points

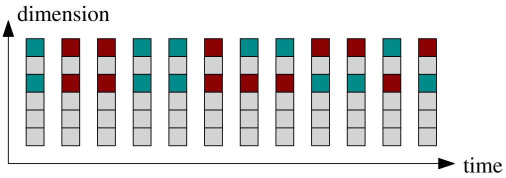


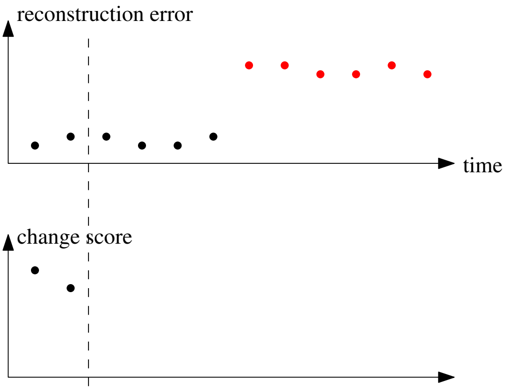


$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Compute change score for possible change points

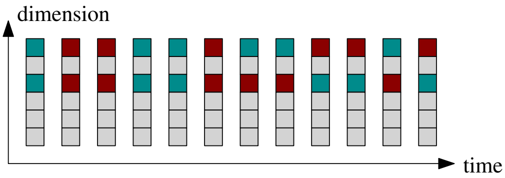


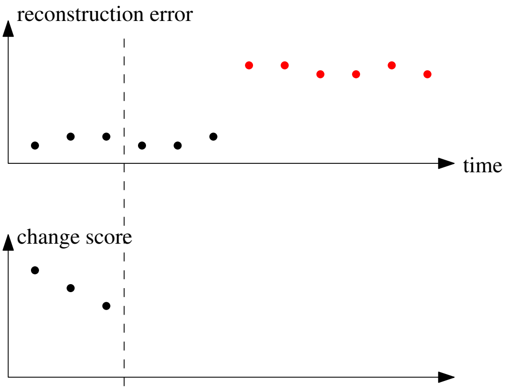


$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Compute change score for possible change points

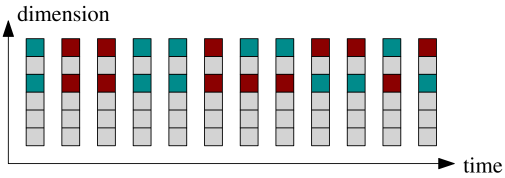


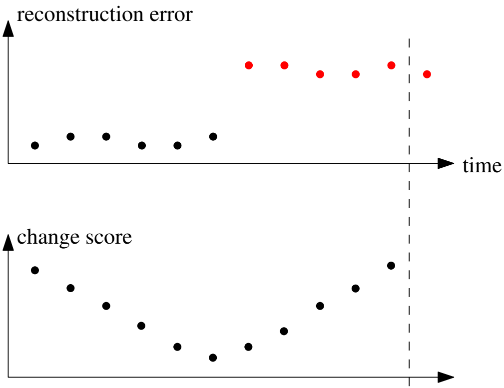


$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Compute change score for possible change points

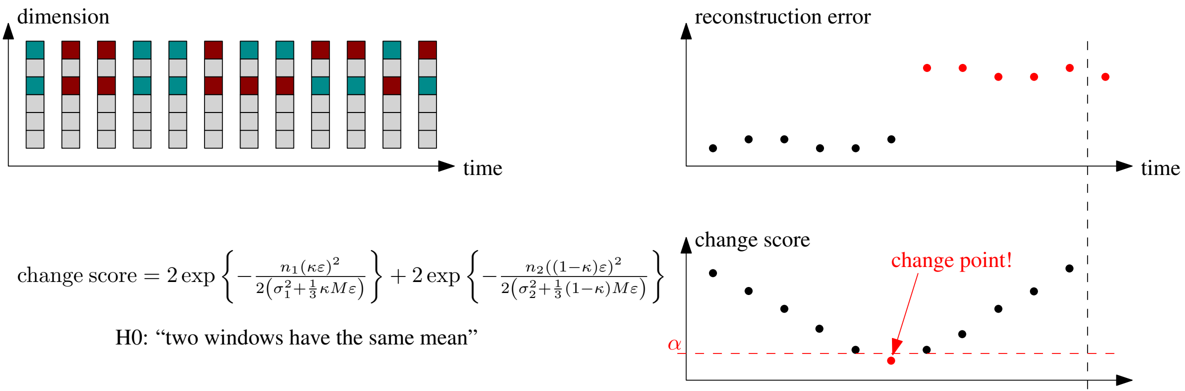




$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Is change significant?



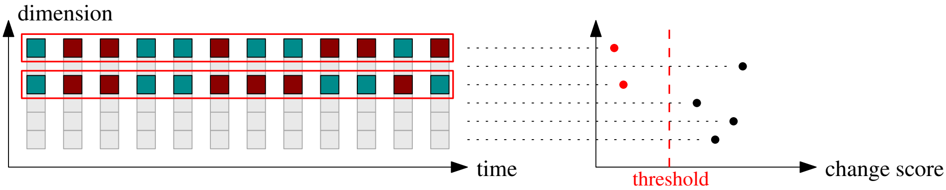
$$\text{change score} = 2\exp\left\{-\frac{n_1(\kappa\varepsilon)^2}{2\left(\sigma_1^2+\frac{1}{3}\kappa M\varepsilon\right)}\right\} + 2\exp\left\{-\frac{n_2((1-\kappa)\varepsilon)^2}{2\left(\sigma_2^2+\frac{1}{3}(1-\kappa)M\varepsilon\right)}\right\}$$

H0: “two windows have the same mean”

High-level Algorithm
Change subspace and severity

After detecting a change:

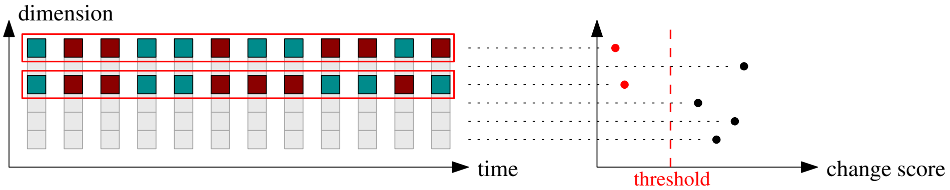
1. Identify dimensions that changed the most
   - Apply change score to each dimension
2. Quantify change severity
   - Normalize loss **in the change subspace**



High-level Algorithm
Change subspace and severity

After detecting a change:

1. Identify dimensions that changed the most
   - Apply change score to each dimension
2. Quantify change severity
   - Normalize loss **in the change subspace**

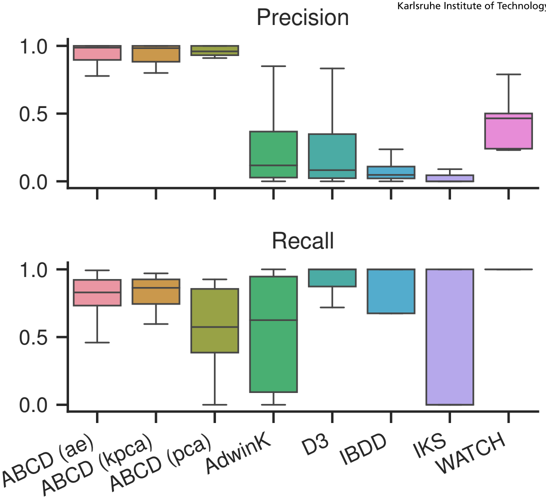


Experiments
Change detection results

Insights:

1. Precision is very high
2. Lower sensitivity than competitors

⇒ Which method to choose depends on the cost of FP and FN

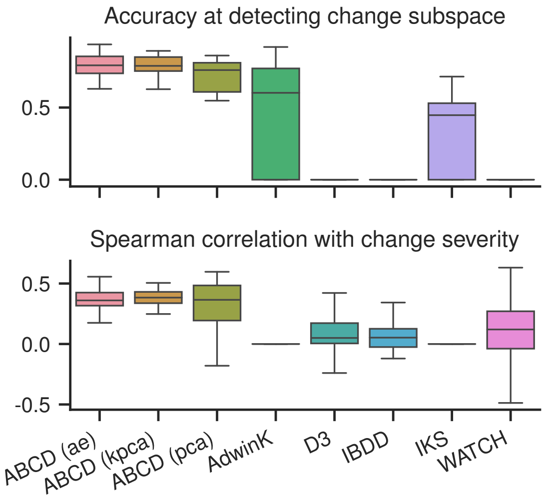


Experiments
Change subspace and severity

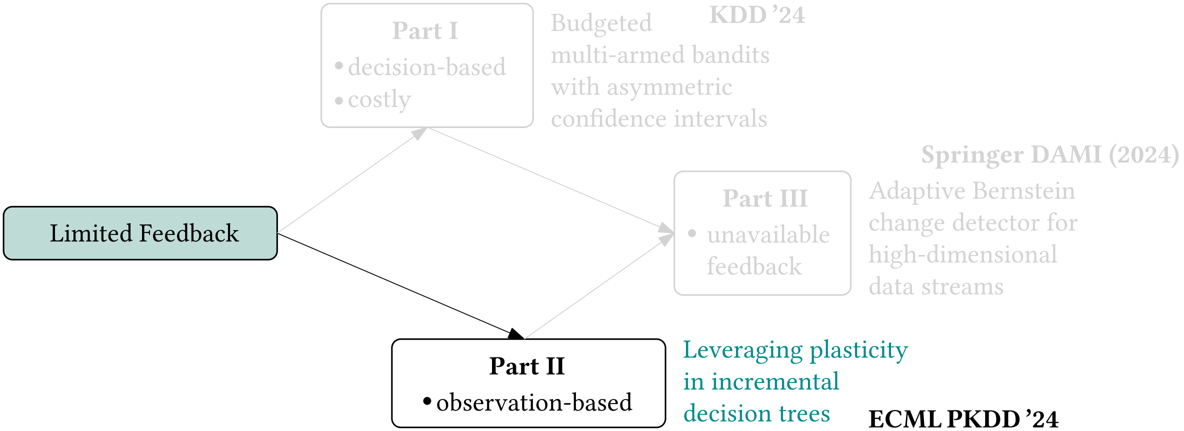
Insights:

1. Both metrics are higher than for competitors
2. However, there is still room for improvement

⇒ First strides towards drift characterization



Next up
Part II — PLASTIC



Contribution
Feedback-efficient incremental decision tree mining

Research question:

How to improve feedback efficiency of widely used algorithms for decision support systems?

Solution: Improve incremental decision trees!

Technical contributions:

- PLASTIC, a feedback-efficient incremental decision tree algorithm
- Decision tree restructuring based on the concept of plasticity
- PLASTIC-A, a change-adaptive version of PLASTIC

Contribution
Feedback-efficient incremental decision tree mining

Research question:

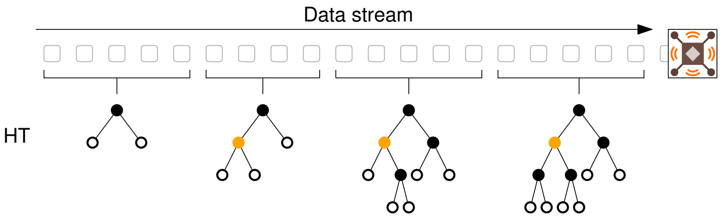
How to improve feedback efficiency of widely used algorithms for decision support systems?

Solution: Improve incremental decision trees!

Technical contributions:

- PLASTIC, a feedback-efficient incremental decision tree algorithm
- Decision tree restructuring based on the concept of plasticity
- PLASTIC-A, a change-adaptive version of PLASTIC

Incremental Decision Trees
Foundation



Hoeffding Trees [DH00]

- Feedback-inefficient but accurate

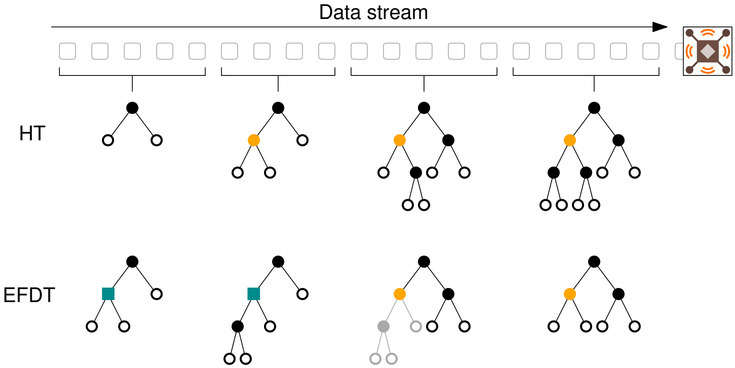
Karlsruhe Institute of Technology

Hoeffding Trees [DH00]

- Feedback-inefficient but accurate

Extremely Fast Decision Trees [MWS18]

- More feedback-efficient
- but unreliable

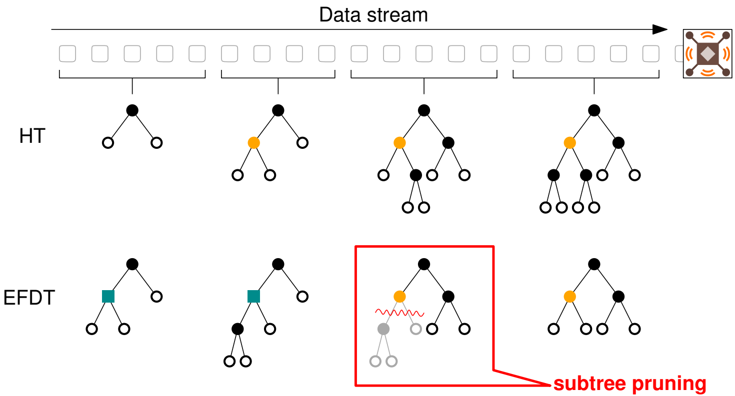


Hoeffding Trees [DH00]

- Feedback-inefficient but accurate

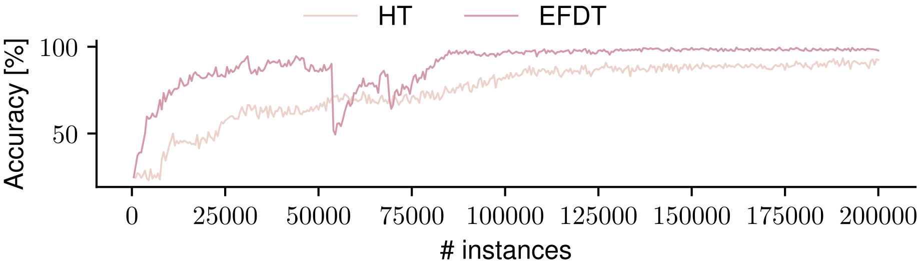
Extremely Fast Decision Trees [MWS18]

- More feedback-efficient
- but unreliable



HT vs. EFDT
EFDT learns faster than HT but suffers from accuracy drops

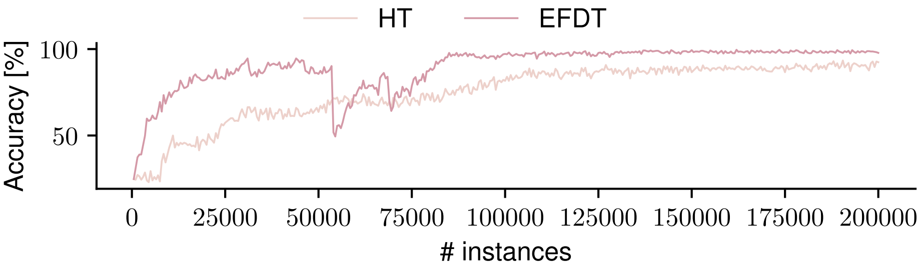
- Illustrative example on synthetic data



Can we maintain EFDT's fast learning but avoid the accuracy drops?

HT vs. EFDT
EFDT learns faster than HT but suffers from accuracy drops

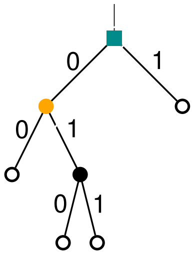
- Illustrative example on synthetic data



Can we maintain EFDT's fast learning but avoid the accuracy drops?

Idea behind PLASTIC
Decision tree plasticity

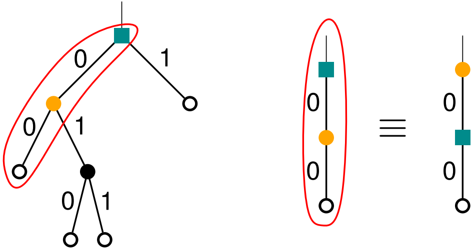
- In the left-most branch, any instance with attribute values ■ = 0 and ● = 0 will arrive at ○
- Hence, from the viewpoint of the leaf, ■—●—○ ≡ ●—■—○



- **PLASTIC revises splits by restructuring the affected subtree**

Idea behind PLASTIC
Decision tree plasticity

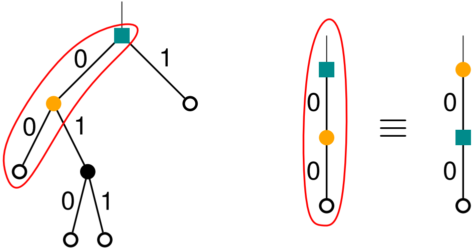
- In the left-most branch, any instance with attribute values ■ = 0 and ● = 0 will arrive at ○
- Hence, from the viewpoint of the leaf, ■–●–○ ≡ ●–■–○



- **PLASTIC revises splits by restructuring the affected subtree**

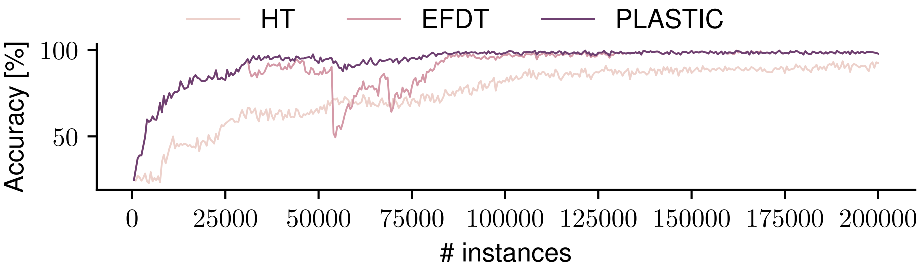
Idea behind PLASTIC
Decision tree plasticity

- In the left-most branch, any instance with attribute values ■ = 0 and ● = 0 will arrive at ○
- Hence, from the viewpoint of the leaf, ■—●—○ ≡ ●—■—○



- **PLASTIC revises splits by restructuring the affected subtree**

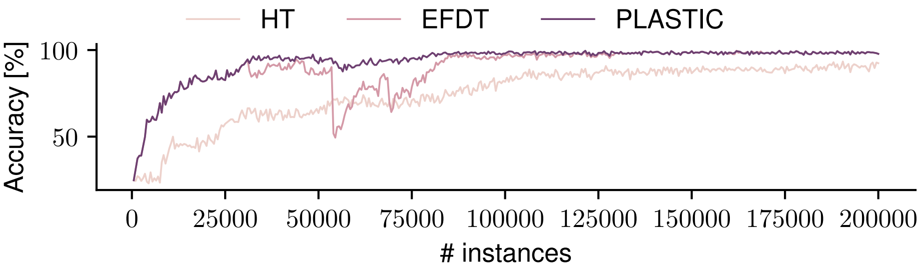
- Restructuring **avoids accuracy drops** caused by subtree pruning in EFDT
- Improvements in worst-case accuracy up to 50 % compared to EFDT



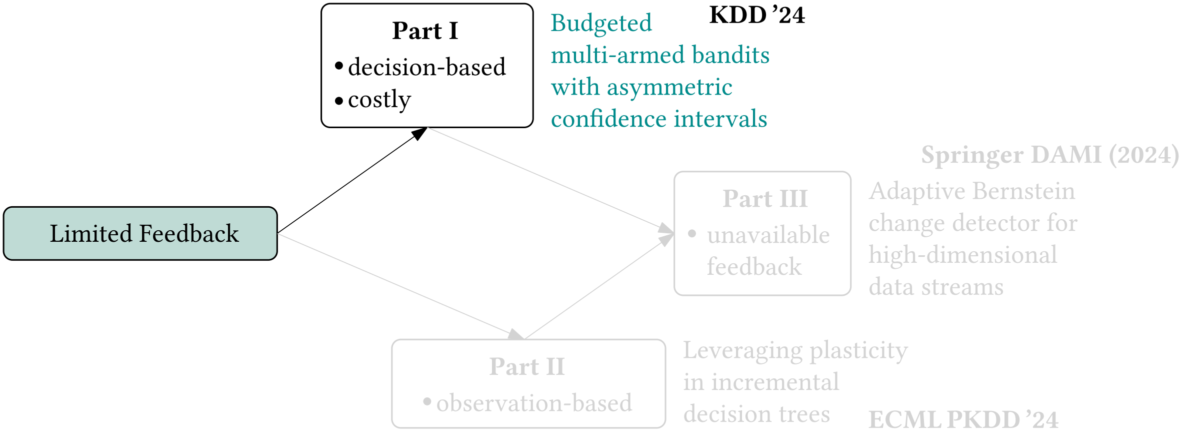
Results

Karlsruhe Institute of Technology

- Restructuring **avoids accuracy drops** caused by subtree pruning in EFDT
- Improvements in worst-case accuracy up to 50 % compared to EFDT



Next up
Part I — ω-UCB



Contribution
Sequential decision-making under budget constraints

Research question:

How to optimize sequential decisions under budget constraints when feedback is costly and resources are limited?

Solution: **Use budgeted multi-armed bandit algorithms!**

Technical contributions:

- ω-UCB, a budget-aware multi-armed bandit algorithm based on asymmetric confidence intervals
- Derivation of asymmetric confidence intervals
- Theoretical analysis and empirical evaluation

Contribution
Sequential decision-making under budget constraints

Research question:

How to optimize sequential decisions under budget constraints when feedback is costly and resources are limited?

Solution: **Use budgeted multi-armed bandit algorithms!**

Technical contributions:

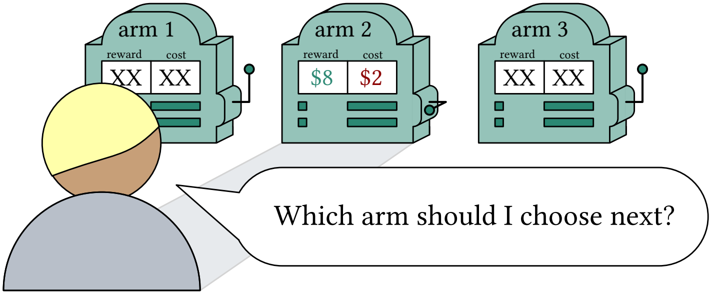
- ω-UCB, a budget-aware multi-armed bandit algorithm based on asymmetric confidence intervals
- Derivation of asymmetric confidence intervals
- Theoretical analysis and empirical evaluation

Budgeted Multi-Armed Bandits
Generic algorithm

Goal: Maximize the total reward until the available budget runs out

While budget B not empty:

1. play one of K arms
2. observe reward and cost
3. adjust arm selection strategy

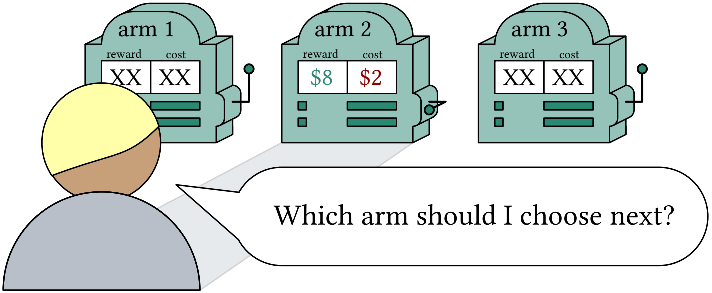


Budgeted Multi-Armed Bandits
Generic algorithm

Goal: Maximize the total reward until the available budget runs out

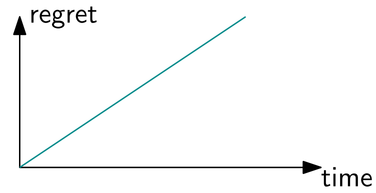
While budget B not empty:

1. play one of K arms
2. observe reward and cost
3. adjust arm selection strategy

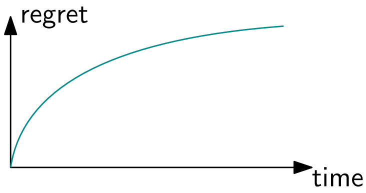


Foundation
Regret

- **Regret**: Difference in reward compared to the optimal strategy



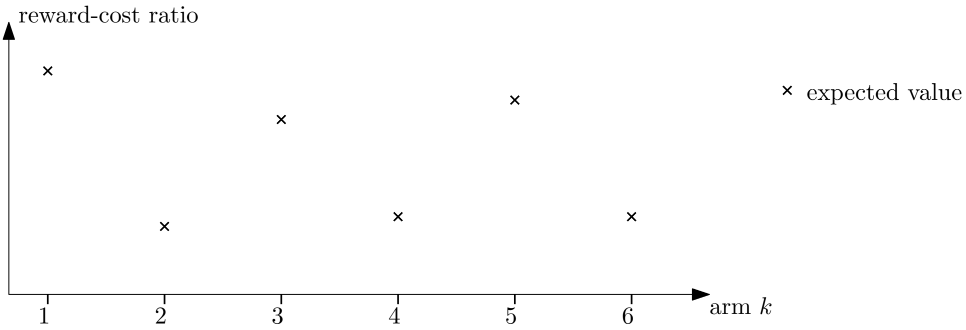
linear regret
→ algorithm does not learn



sublinear regret
→ algorithm learns

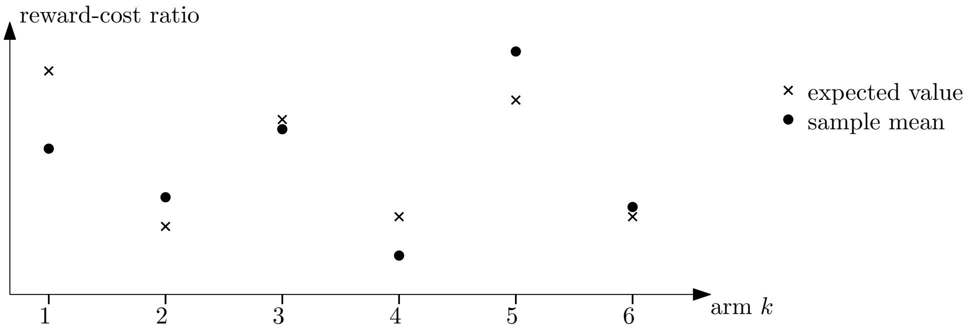
Foundation
Upper confidence bound (UCB) sampling

- Best arm = arm with **highest ratio** between expected rewards and costs



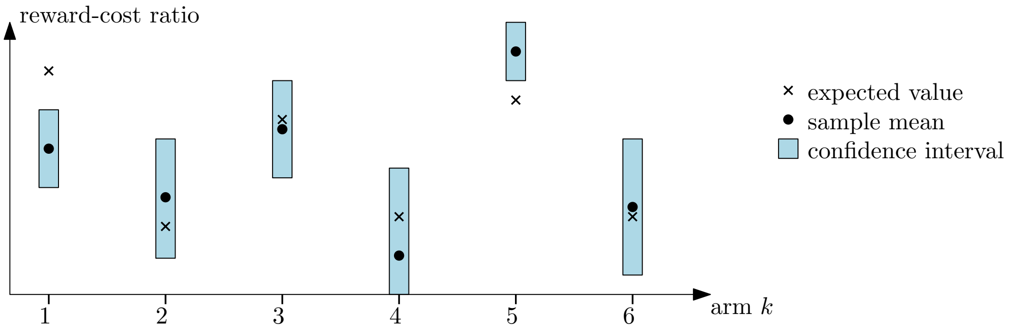
Foundation
Upper confidence bound (UCB) sampling

- Best arm = arm with **highest ratio** between expected rewards and costs



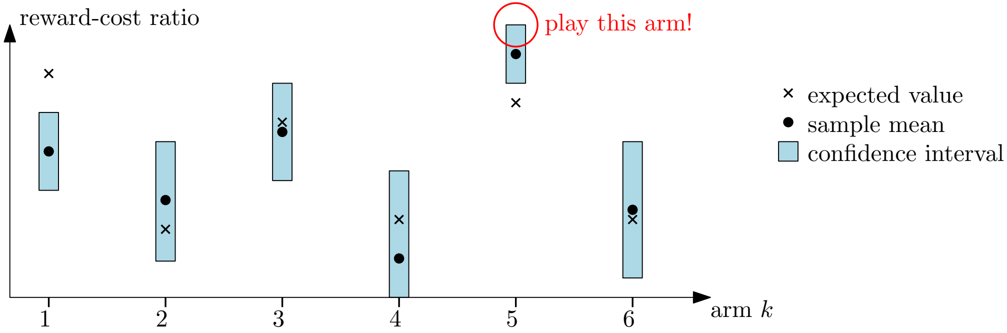
Foundation
Upper confidence bound (UCB) sampling

- Best arm = arm with **highest ratio** between expected rewards and costs



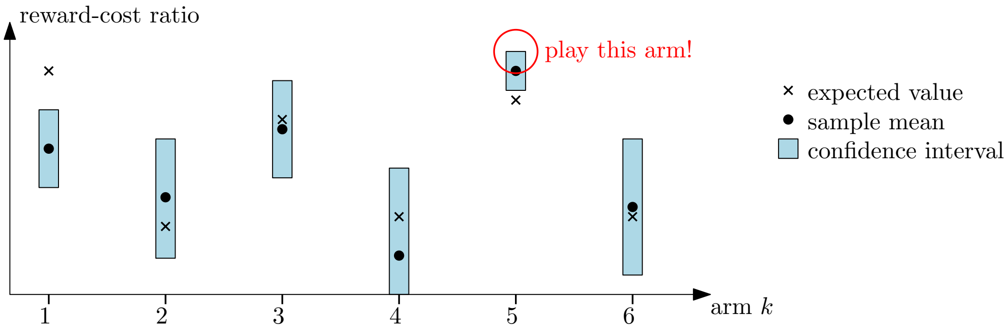
Foundation
Upper confidence bound (UCB) sampling

- Best arm = arm with **highest ratio** between expected rewards and costs



Upper confidence bound (UCB) sampling

- Best arm = arm with **highest ratio** between expected rewards and costs



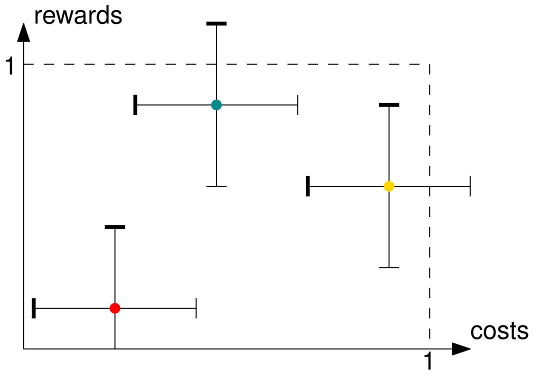
Problem
Symmetric CIs lead to increased UCB for reward-cost ratio

The UCB for the reward-cost ratio should be

- as **accurate** as possible (UCB > expected value)
- as **tight** as possible

→ but this is not the case in existing algorithms.

$$UCB = \frac{\text{average reward} + \text{uncertainty}}{\text{average cost} - \text{uncertainty}}$$



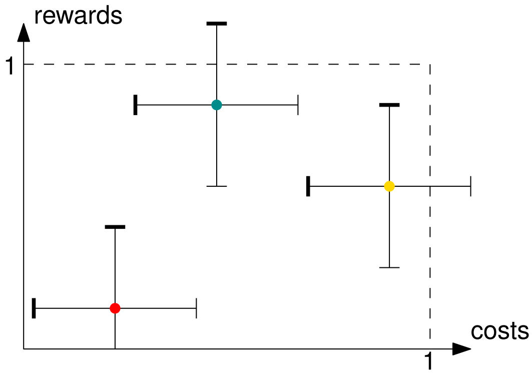
Symmetric CIs lead to increased UCB for reward-cost ratio

The UCB for the reward-cost ratio should be

- as **accurate** as possible (UCB > expected value)
- as **tight** as possible

$\rightarrow$ but this is not the case in existing algorithms.

$$UCB = \frac{\text{average reward} + \text{uncertainty}}{\text{average cost} - \text{uncertainty}}$$



Idea
Use asymmetric confidence intervals instead

The UCB for the reward-cost ratio should be

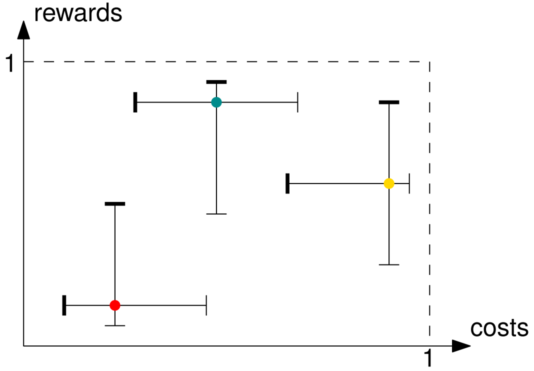
- as **accurate** as possible (UCB > expected value)
- as **tight** as possible

$\rightarrow$ but this is not the case in existing algorithms.

$$UCB = \frac{\textit{average reward} + \textit{uncertainty}}{\textit{average cost} - \textit{uncertainty}}$$

Our core idea:

- Use **asymmetric confidence intervals**
- Tighten confidence intervals when variance is low (our η-parameter, $\eta = 1 \rightarrow$ Bernoulli)



Idea
Use asymmetric confidence intervals instead

The UCB for the reward-cost ratio should be

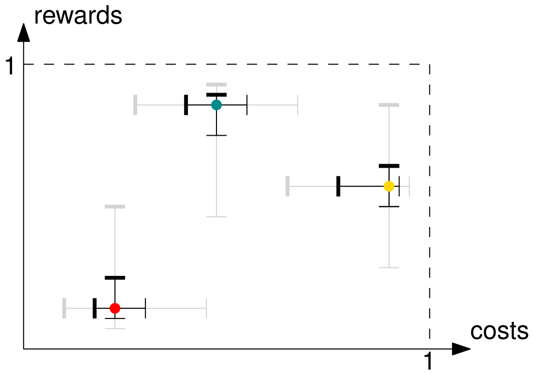
- as **accurate** as possible (UCB > expected value)
- as **tight** as possible

$\rightarrow$ but this is not the case in existing algorithms.

$$UCB = \frac{\textit{average reward} + \textit{uncertainty}}{\textit{average cost} - \textit{uncertainty}}$$

Our core idea:

- Use **asymmetric confidence intervals**
- Tighten confidence intervals when variance is low (our η-parameter, $\eta = 1 \rightarrow$ Bernoulli)

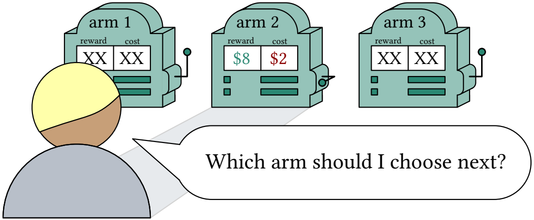


Our Approach
ω-UCB algorithm

Goal: Maximize the total reward until the available budget runs out

While budget B not empty:

1. play one of K arms
   - UCB sampling with **asymmetric confidence intervals**
2. observe reward and cost
   - Track mean and **variance** $\Rightarrow \omega^*$-UCB
3. adjust arm selection strategy
   - **Increase confidence intervals over time** according to $\alpha(t) = 1 - \sqrt{1 - t^{-\rho}}$

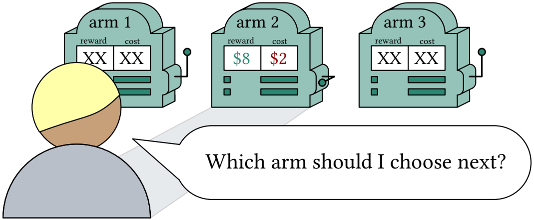


Our Approach
ω-UCB algorithm

Goal: Maximize the total reward until the available budget runs out

While budget B not empty:

1. play one of K arms
   - UCB sampling with **asymmetric confidence intervals**
2. observe reward and cost
   - Track mean and **variance** $\Rightarrow \omega^*$-UCB
3. adjust arm selection strategy
   - **Increase confidence intervals over time** according to $\alpha(t) = 1 - \sqrt{1 - t^{-\rho}}$

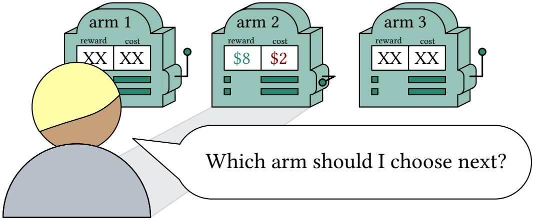


Our Approach
ω-UCB algorithm

Goal: Maximize the total reward until the available budget runs out

While budget B not empty:

1. play one of K arms
   - UCB sampling with **asymmetric confidence intervals**
2. observe reward and cost
   - Track mean and **variance** $\Rightarrow \omega^*$-UCB
3. adjust arm selection strategy
   - **Increase confidence intervals over time** according to $\alpha(t) = 1 - \sqrt{1 - t^{-\rho}}$

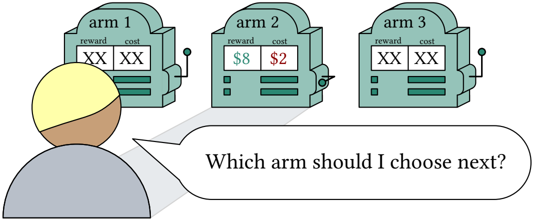


Karlsruhe Institute of Technology

Goal: Maximize the total reward until the available budget runs out

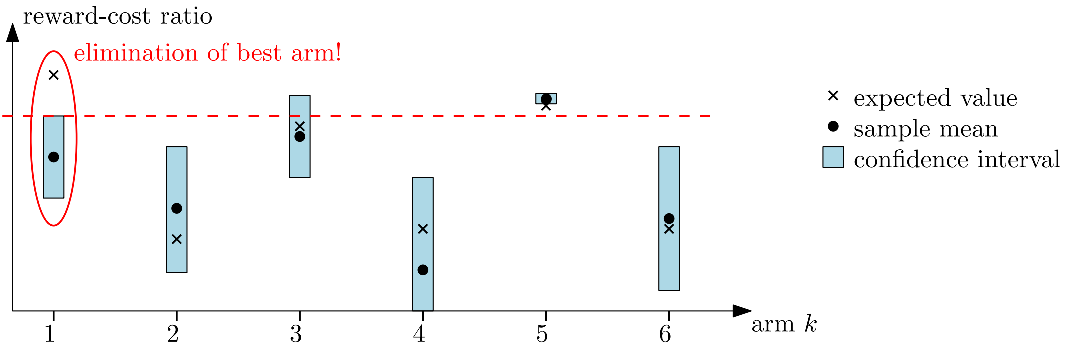
While budget B not empty:

1. play one of K arms
   - UCB sampling with **asymmetric confidence intervals**
2. observe reward and cost
   - Track mean and **variance** $\Rightarrow \omega^*$-UCB
3. adjust arm selection strategy
   - **Increase confidence intervals over time** according to $\alpha(t) = 1 - \sqrt{1 - t^{-\rho}}$



ω-UCB
Time-adaptive confidence interval

- Time-adaptive confidence intervals prevent elimination of best arm

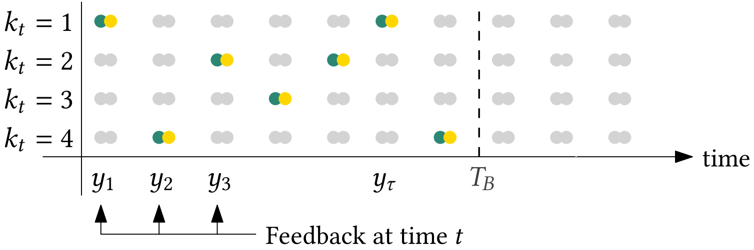


Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)

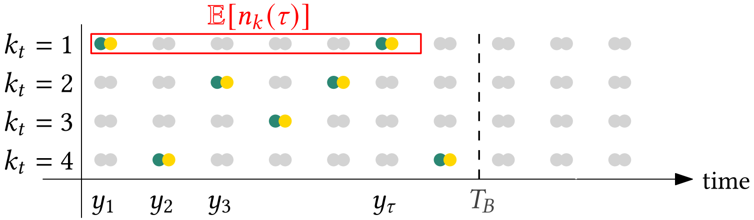
Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



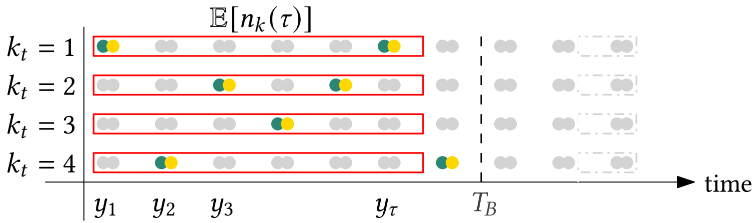
Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



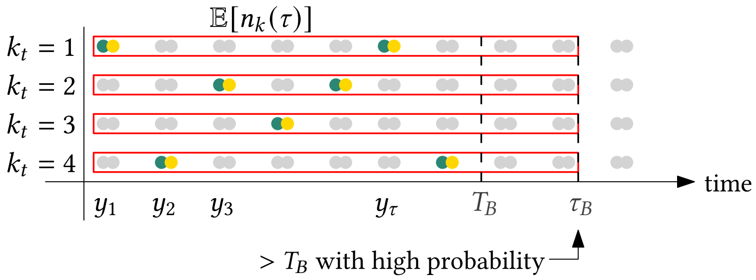
Proof of sub-linear regret
Proof structure

- Regret $=\sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



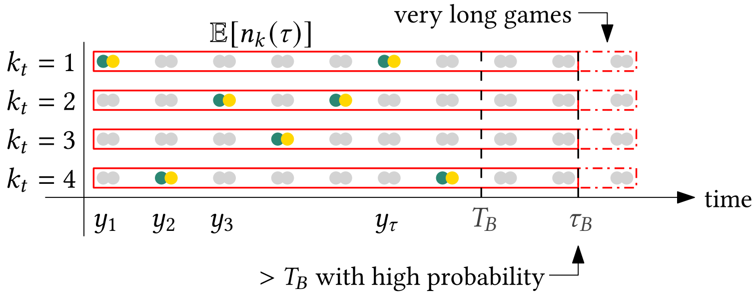
Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



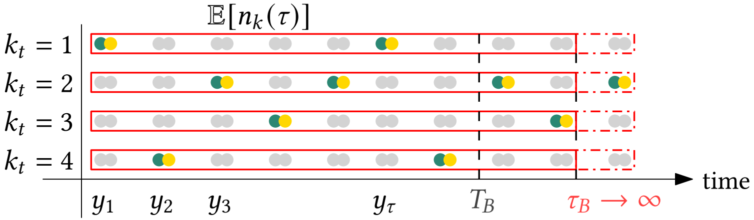
Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



Proof of sub-linear regret
Proof structure

- Regret $= \sum_{\text{arms } k}$ regret increment $\cdot$ number of plays until T_B
- T_B: number of plays until budget B is empty ($\leftarrow$ a random variable!)



Proof of sub-linear regret
Results (I)

Theorem (Number of suboptimal plays)

With ω-UCB, the expected number of plays of a suboptimal arm $k > 1$ before time step τ, $\mathbb{E}[n_k(\tau)]$, is upper-bounded by:

$$\mathbb{E}[n_k(\tau)] \leq 1 + n_k^*(\tau) + \xi(\tau, \rho),$$

where

$$\xi(\tau, \rho) = (\tau - K)\left(2 - \sqrt{1 - \tau^{-\rho}}\right) - \sum_{t=K+1}^{\tau} \sqrt{1 - t^{-\rho}},$$

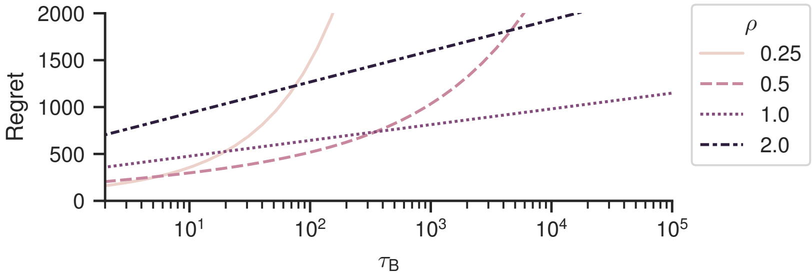
$$n_k^*(\tau) = \frac{8\rho \log \tau}{\delta_k^2} \max\left\{\frac{\eta_k^r \mu_k^r}{1 - \mu_k^r}, \frac{\eta_k^c (1 - \mu_k^c)}{\mu_k^c}\right\}, \quad \delta_k = \frac{\Delta_k}{\Delta_k + \frac{1}{\mu_k^c}},$$

and K and Δ_k are defined as before.

Regret illustration for 2-armed bandit

Hyperparameter ρ controlls amount of exploration

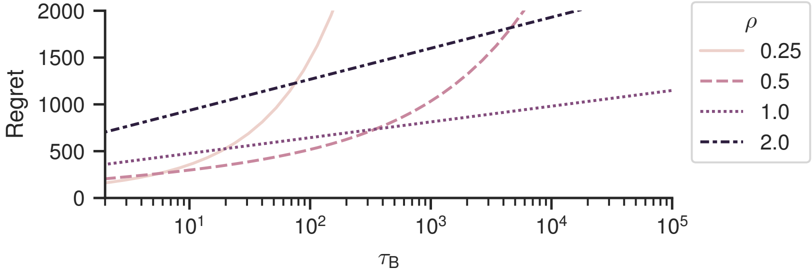
- $\rho > 1$ leads to logarithmic growth
- $\rho \leq 1$ leads to super-logarithmic growth



Proof of sub-linear regret
Regret illustration for 2-armed bandit

Hyperparameter ρ controlls amount of exploration

- $\rho > 1$ leads to logarithmic growth
- $\rho \leq 1$ leads to super-logarithmic growth



Experiments

Karlsruhe Institute of Technology

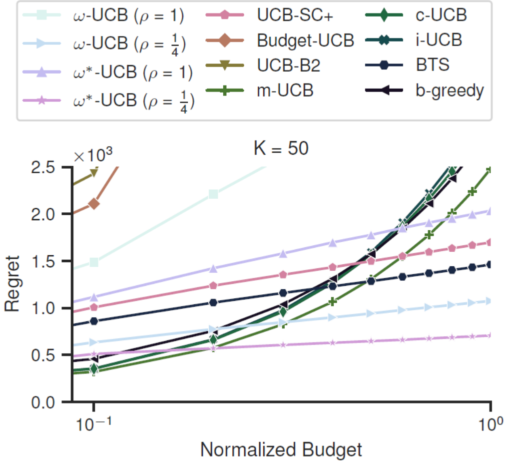
On the right:

- Regret over time

Insights:

1. $\rho = 1$ is too conservative in practice
2. Estimating η as in ω^*-UCB reduces regret

$\Rightarrow$ **"Use ω^*-UCB with $\rho = 1/4$!"**



Experiments

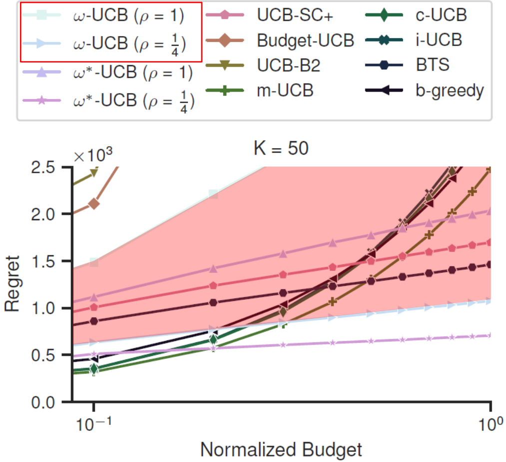
On the right:

- Regret over time

Insights:

1. $\rho = 1$ is too conservative in practice
2. Estimating η as in ω^*-UCB reduces regret

$\Rightarrow$ **"Use ω^*-UCB with $\rho = 1/4$!"**



Experiments

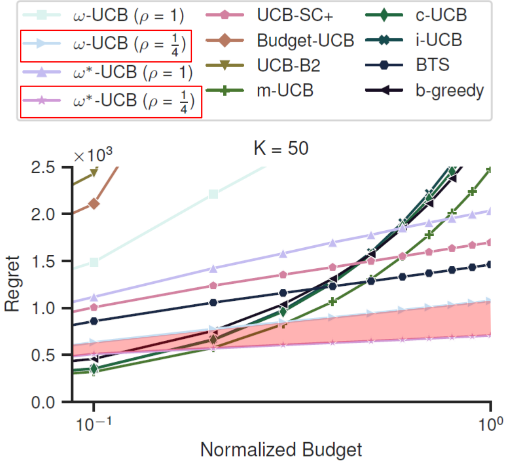
On the right:

- Regret over time

Insights:

1. $\rho = 1$ is too conservative in practice
2. Estimating η as in ω^*-UCB reduces regret

$\Rightarrow$ **“Use ω^*-UCB with $\rho = 1/4$!”**



Experiments

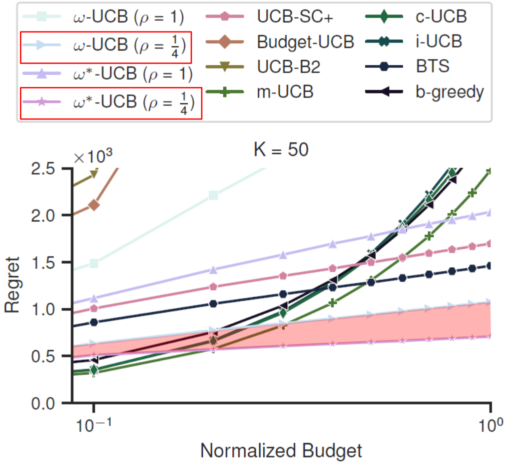
On the right:

- Regret over time

Insights:

1. $\rho = 1$ is too conservative in practice
2. Estimating η as in ω^*-UCB reduces regret

$\Rightarrow$ **"Use ω^*-UCB with $\rho = 1/4$!"**



Conclusions

Karlsruhe Institute of Technology

Most existing algorithms for decision making in data streams assume **plenty and cheap feedback**. My dissertation addresses limited feedback from **three perspectives**:

- Costly decision-based feedback ⇒ ω-UCB (Sequential decision-making under budget constraints) [Hey+24b]
- Observation-based feedback ⇒ PLASTIC (feedback-efficient incremental decision tree mining) [Hey+24c]
- Unavailable feedback ⇒ ABCD (characterizing change in high-dimensional data streams) [Hey+24a]

Additional materials

- Complete source code available on GitHub (`https://github.com/heymarco`)
- Released PLASTIC and ABCD as part of open-source projects (`https://capymoa.org/`)
- `Advertisement video` and `blog post` (by Vadim Arzamasov and me) showcasing ω-UCB

Conclusions

Most existing algorithms for decision making in data streams assume **plenty and cheap feedback**. My dissertation addresses limited feedback from **three perspectives**:

- Costly decision-based feedback ⇒ ω-UCB (Sequential decision-making under budget constraints) [Hey+24b]
- Observation-based feedback ⇒ PLASTIC (feedback-efficient incremental decision tree mining) [Hey+24c]
- Unavailable feedback ⇒ ABCD (characterizing change in high-dimensional data streams) [Hey+24a]

Additional materials

- Complete source code available on GitHub (`https://github.com/heymarco`)
- Released PLASTIC and ABCD as part of open-source projects (`https://capymoa.org/`)
- `Advertisement video` and `blog post` (by Vadim Arzamasov and me) showcasing ω-UCB

Conclusions

Most existing algorithms for decision making in data streams assume **plenty and cheap feedback**. My dissertation addresses limited feedback from **three perspectives**:

- Costly decision-based feedback $\Rightarrow$ ω-UCB (Sequential decision-making under budget constraints) [Hey+24b]
- Observation-based feedback $\Rightarrow$ PLASTIC (feedback-efficient incremental decision tree mining) [Hey+24c]
- Unavailable feedback $\Rightarrow$ ABCD (characterizing change in high-dimensional data streams) [Hey+24a]

Additional materials

- Complete source code available on GitHub (`https://github.com/heymarco`)
- Released PLASTIC and ABCD as part of open-source projects (`https://capymoa.org/`)
- `Advertisement video` and `blog post` (by Vadim Arzamasov and me) showcasing ω-UCB

References I

[1] Albert Bifet and Ricard Gavaldà. “Adaptive learning from evolving data streams”. In: *IDA 2009*. Vol. 5772. Springer, 2009, pp. 249–260. DOI: 10.1007/978-3-642-03915-7_22.

[2] Albert Bifet and Ricard Gavaldà. “Learning from time-changing data with adaptive windowing”. In: *SDM*. Minneapolis, USA: SIAM, 2007, pp. 443–448. DOI: 10.1137/1.9781611972771.42.

[3] Albert Bifet et al. “Big data stream mining”. In: *Machine learning for data streams: with practical examples in MOA*. The MIT Press, 2018. ISBN: 978-0-262-34604-7. URL: https://doi.org/10.7551/mitpress/10654.003.0006.

[4] Michelangelo Ceci et al. “ECHAD: Embedding-based change detection from multivariate time series in smart grids”. In: *IEEE Access* 8 (2020), pp. 156053–156066. DOI: 10.1109/ACCESS.2020.3019095.

[5] T. F. Chan, G. H. Golub, and R. J. LeVeque. “Updating formulae and a pairwise algorithm for computing sample variances”. In: *COMPSTAT 1982*. Toulouse, France: Physica, 1982, pp. 30–41. ISBN: 978-3-642-51461-6. DOI: 10.1007/978-3-642-51461-6_3.

References II

[6] Tamraparni Dasu et al. "An information-theoretic approach to detecting changes in multi-dimensional data streams". In: *Interface 2006*. Pasadena, USA: Curran Associates, Inc., 2006. ISBN: 978-1-62276-715-1.

[7] Pedro M. Domingos and Geoff Hulten. "Mining high-speed data streams". In: *KDD '00*. ACM, 2000, pp. 71–80. ISBN: 1-58113-233-6. DOI: 10.1145/347090.347107.

[8] Pietro Ducange, Francesco Marcelloni, and Riccardo Pecori. "Fuzzy Hoeffding decision tree for data stream classification". In: 14.1 (2021), pp. 946–964. DOI: 10.2991/ijcis.d.210212.001.

[9] Kamil Faber et al. "WATCH: Wasserstein change point detection for high-dimensional time series data". In: *BigData 2021*. IEEE, 2021, pp. 4450–4459. DOI: 10.1109/BIGDATA52589.2021.9671962.

[10] William J. Faithfull, Juan José Rodríguez Diez, and Ludmila I. Kuncheva. "Combining univariate approaches for ensemble change detection in multivariate data". In: *Information Fusion* 45 (2019), pp. 202–214. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2018.02.003.

References III

[11] João Gama, Ricardo Fernandes, and Ricardo Rocha. “Decision trees for mining data streams”. In: *Intelligent Data Analysis* 10.1 (2006), pp. 23–45. DOI: 10.3233/IDA-2006-10103.

[12] Heitor Murilo Gomes et al. “A survey on semi-supervised learning for delayed partially labelled data streams”. In: *ACM Computing Surveys* 55.4 (2023), 75:1–75:42. DOI: 10.1145/3523055.

[13] Ömer Gözüaçik et al. “Unsupervised concept drift detection with a discriminative classifier”. In: *CIKM ’19*. Beijing, China: ACM, 2019, pp. 2365–2368. ISBN: 978-1-4503-6976-3. DOI: 10.1145/3357384.3358144.

[14] Sattar Hashemi and Ying Yang. “Flexible decision tree for data stream classification in the presence of concept change, noise and missing values”. In: *Data Mining and Knowledge Discovery* 19.1 (2009), pp. 95–131. DOI: 10.1007/s10618-009-0130-9.

[15] Marco Heyden et al. “Adaptive Bernstein change detector for high-dimensional data streams”. In: *Data Mining and Knowledge Discovery* 38.3 (2024), pp. 1334–1363. DOI: 10.1007/S10618-023-00999-5.

References IV

[16] Marco Heyden et al. “Budgeted multi-armed bandits with asymmetric confidence intervals”. In: *KDD ’24: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Barcelona, Spain: ACM, 2024, pp. 1073–1084. ISBN: 9798400704901. DOI: 10.1145/3637528.3671833.

[17] Marco Heyden et al. “Leveraging plasticity in incremental decision trees”. In: *Machine Learning and Knowledge Discovery in Databases. Research Track. ECML PKDD 2024*. Vilnius, Lithuania: Springer Nature, 2024, pp. 38–54. ISBN: 978-3-031-70362-1. DOI: 10.1007/978-3-031-70362-1_3.

[18] Geoff Hulten, Laurie Spencer, and Pedro M. Domingos. “Mining time-changing data streams”. In: *KDD ’01*. San Francisco, California, USA: ACM, 2001, pp. 97–106. ISBN: 1-58113-391-X. DOI: 10.1145/502512.502529.

[19] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge: Cambridge University Press, 2020. ISBN: 978-1-108-57140-1. URL: https://doi.org/10.1017/9781108571401.

[20] Anjin Liu et al. “Regional concept drift detection and density synchronized drift adaptation”. In: *IJCAI-17*. Melbourne, Australia: ijcai.org, 2017, pp. 2280–2286. DOI: 10.24963/ijcai.2017/317.

References V

Karlsruhe Institute of Technology

[21] Chaitanya Manapragada, Mahsa Salehi, and Geoffrey I. Webb. “Extremely fast Hoeffding adaptive tree”. In: *ICDM 2022*. Orlando, FL, USA: IEEE, 2022, pp. 319–328. DOI: 10.1109/ICDM54844.2022.00042.

[22] Chaitanya Manapragada, Geoffrey I. Webb, and Mahsa Salehi. “Extremely fast decision tree”. In: *KDD ’18*. ACM, 2018, pp. 1953–1962. ISBN: 978-1-4503-5552-0. DOI: 10.1145/3219819.3220005.

[23] Russel Pears, Sripirakas Sakthithasan, and Yun Sing Koh. “Detecting concept change in dynamic data streams – a sequential approach based on reservoir sampling”. In: *Machine Learning* 97.3 (2014), pp. 259–293. ISSN: 1573-0565. DOI: 10.1007/s10994-013-5433-9.

[24] Bernhard Pfahringer, Geoffrey Holmes, and Richard Kirkby. “New options for Hoeffding trees”. In: *Advances in Artificial Intelligence. AI 2007*. Springer, 2007, pp. 90–99. ISBN: 978-3-540-76928-6. DOI: 10.1007/978-3-540-76928-6_11.

References VI

[25] Abdulhakim A. Qahtan et al. "A PCA-based change detection framework for multidimensional data streams: change detection in multidimensional data streams". In: *KDD '15*. Sydney, Australia: ACM, Aug. 2015, pp. 935–944. ISBN: 978-1-4503-3664-2. DOI: 10.1145/2783258.2783359.

[26] Denis Moreira dos Reis et al. "Fast unsupervised online drift detection using incremental Kolmogorov-Smirnov test". In: *KDD '16*. San Francisco, California, USA: ACM, 2016, pp. 1545–1554. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939836.

[27] Leszek Rutkowski et al. "A new method for data stream mining based on the misclassification error". In: *IEEE Transactions on Neural Networks and Learning Systems* 26.5 (2015), pp. 1048–1059. DOI: 10.1109/TNNLS.2014.2333557.

[28] Leszek Rutkowski et al. "Decision trees for mining data streams based on the Gaussian approximation". In: *IEEE Transactions on Knowledge and Data Engineering* 26.1 (2014), pp. 108–119. DOI: 10.1109/TKDE.2013.34.

References VII

[29] Leszek Rutkowski et al. "Decision trees for mining data streams based on the McDiarmid's bound". In: *IEEE Transactions on Knowledge and Data Engineering* 25.6 (2013), pp. 1272–1279. DOI: 10.1109/TKDE.2012.66.

[30] Leszek Rutkowski et al. "The CART decision tree for mining data streams". In: *Information Sciences* 266 (2014), pp. 1–15. ISSN: 0020-0255. DOI: 10.1016/j.ins.2013.12.060.

[31] Vinícius M. A. de Souza, Farhan Asif Chowdhury, and Abdullah Mueen. "Unsupervised drift detection on high-speed data streams". In: *BigData 2020*. Atlanta, GA, USA: IEEE, 2020, pp. 102–111. DOI: 10.1109/BIGDATA50022.2020.9377880.

[32] Vinícius M. A. de Souza et al. "Efficient unsupervised drift detector for fast and high-dimensional data streams". In: *Knowl. Inf. Syst.* 63.6 (2021), pp. 1497–1527.

[33] Amos Tversky and Daniel Kahneman. "Judgment under uncertainty: Heuristics and biases". In: *Science* 185.4157 (1974), pp. 1124–1131. DOI: 10.1126/science.185.4157.1124.

References VIII

[34] B. P. Welford. “Note on a Method for Calculating Corrected Sums of Squares and Products”. In: *Technometrics* (1962), pp. 419–420. DOI: 10.1080/00401706.1962.10490022.

[35] Edwin B. Wilson. “Probable inference, the law of succession, and statistical inference”. In: *Journal of the American Statistical Association* 22.158 (1927), pp. 209–212. DOI: doi.org/10.2307/2276774.

[36] Xindong Wu, Pei-Pei Li, and Xuegang Hu. “Learning from concept drifting data streams with unlabeled data”. In: *Neurocomputing* 92 (2012), pp. 145–155. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2011.08.041.

[37] Yingce Xia et al. “Finite budget analysis of multi-armed bandit problems”. In: *Neurocomputing* 258 (2017), pp. 13–29. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2016.12.079.

Future Work
ω-UCB

1. **Contextual** budgeted multi-armed bandits
   - So far, ω-UCB does not use context information
   - Context information can improve regret drastically
   - Example:
     - Use Gaussian process (GP) to model context-reward and context-cost relationship
     - Estimate parameters of confidence interval based on GP
2. Adapt ω-UCB to **non-stationary** environments
   - Monitor statistics for each arm
     - **Use ABCD's adaptive windows!**
   - Adjust exploration strategy
   - Analyze regret theoretically

Future Work
PLASTIC

Karlsruhe Institute of Technology

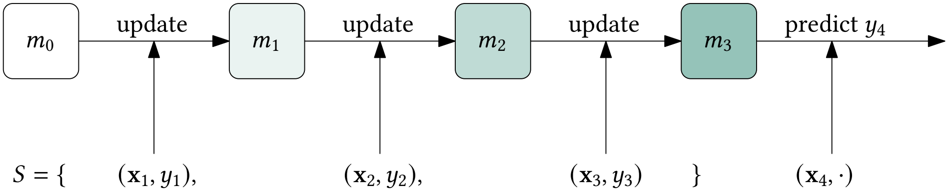
1. Extend PLASTIC to the **delayed-feedback** setting
   - Feedback usually arrives with a delay
   - Use self-training to bridge the delay period
   - Update tree with true feedback once available
   - Restructuring is beneficial for this!

2. Improve **change adaptability** of PLASTIC-A
   - Current change-adaptation procedure is rather simple
   - More sophisticated change adaptation mechanisms exist in the literature [BG09; MSW22]
   - Different types of change might require different adaptation strategies

Future Work
ABCD

1. Investigate change **severity**
   - Our results are better than for competitors but not perfect
   - This can have various reasons
     - definition of severity, subspace detection accuracy, experimental design, choice of encoder-decoder model
   - Possible research: theoretical investigation of change severity, its influencing factors and ways to establish a ground truth
2. Detect **gradual** changes
   - So far, ABCD does not distinguish between gradual and abrupt changes
   - Detecting gradual changes $\rightarrow$ more detailed change characterization
   - Split ABCD's adaptive windows into smaller sub-windows
   - Check whether multiple sub-windows contain change points

Data Streams
Deployed model = trainable model



Research Gap
Decision-Making in Data Streams under Limited Feedback

Related fields combine:

- ML for data streams
- Unsupervised learning, semi-supervised learning, active learning, change detection, multi-armed bandits

Shortcomings

1. Unable to deal with complexity (change detectors that only work with univariate data, e.g., [BG07])
2. Do not take the cost of decisions into account (e.g. most multi-armed bandit algorithms [LS20])
3. Have difficulty dealing with continuous arrival of new data or concept drift (e.g., active learning, SSL) [Gom+23])
4. Are hard to deploy in real data streams (e.g., active learning, SSL) [Gom+23])

Related Work
ω-UCB

UCB types:

- united (u)
- composite (c)
- hybrid (h)

$$UCB_u = \frac{average\ reward}{average\ cost} + uncertainty$$

$$UCB_c = \frac{average\ reward + uncertainty}{average\ cost - uncertainty}$$

Policy	Type	Compared
ε-first	–	×
KUBE	–	×
UCB-BV1	h	×
PD-BwK	c	×
Budget-UCB	h	✓
BTS	–	✓
MRCB	c	–
m-UCB	c	✓
b-greedy	–	✓
c-UCB	h	✓
i-UCB	u	✓
UCB-SC+	u	✓
UCB-B2	u	✓

PLASTIC

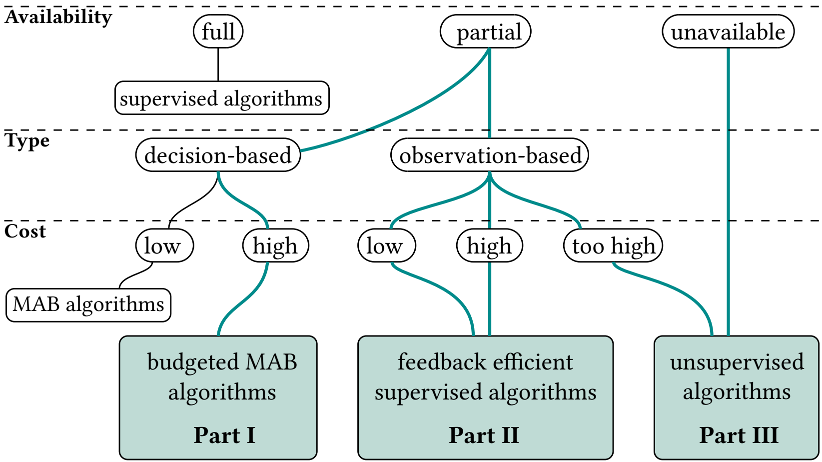
Existing decision trees focus on:

- feedback efficiency (EFDT) [MWS18; MSW22]
- adaptivity to concept drift [HSD01; BG09; GFR06; WLH12; MSW22]
- statistical foundation [Rut+13; Rut+14a; Rut+14b; Rut+15]
- semi-supervised learning [WLH12]
- fuzzy data [HY09; DMP21]
- stability [PHK07]

- **R1:** change detection
- **R2:** change subspace detection
- **R3:** Quantifying change severity
- **UV:** univariate data
- **MV:** multivariate data
- **HD:** high-dimensional data

Approach	Reference	Type	R1	R2	R3
ADWIN	[BG07]	UV	✓	–	–
SeqDrift2	[PSK14]	UV	✓	–	–
kdq-Tree	[Das+06]	MV	✓	–	✓
PCA-CD	[Qah+15]	MV	✓	–	✓
IKS	[Rei+16]	MV	✓	✓	–
LDD-DSDA	[Liu+17]	MV	✓	–	–
AdwinK	[FDK19]	MV	✓	✓	–
D3	[Göz+19]	MV	✓	–	✓
ECHAD	[Cec+20]	MV	✓	–	✓
IBDD	[SCM20]	HD	✓	–	✓
WATCH	[Fab+21]	HD	✓	–	✓
ABCD	ours	HD	✓	✓	✓

Contributions
Zooming out



Change

- Data stream = sequence of observations $S = x_1, x_2, \ldots, x_t$
- Each x_i comes from a distribution F_i and is d-dimensional
- **A change has occurred after t^* if $F_{t^*} \neq F_{t^*+1}$**

Change Subspace

- Set of all dimensions $D = \{1, 2, \ldots, d\}$
- **Union of all $D' \subseteq D$ in which the joint distribution $F^{D'}$ changed**
- **and which do not contain a subspace D'' for which $F_{t^*}^{D''} \neq F_{t^*+1}^{D''}$**

Change Severity

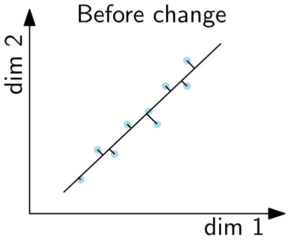
- Positive function Δ that quantifies the discrepancy between F_{t^*} and F_{t^*+1}

ABCD – Idea (1)
Track information loss of dimensionality reduction

- Finding changes in high-dimensional data is hard!
  - Exponential number of subspaces, correlation changes, etc.
- Dimensionality reduction = encode data in fewer dimensions while minimizing information loss
- Concept changes $\Rightarrow$ information loss increases

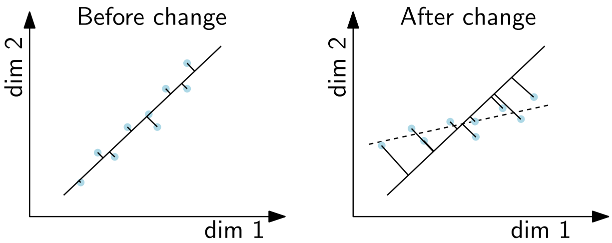
ABCD – Idea (1)
Track information loss of dimensionality reduction

- Finding changes in high-dimensional data is hard!
  - Exponential number of subspaces, correlation changes, etc.
- Dimensionality reduction = encode data in fewer dimensions while minimizing information loss
- Concept changes ⇒ information loss increases



ABCD – Idea (1)
Track information loss of dimensionality reduction

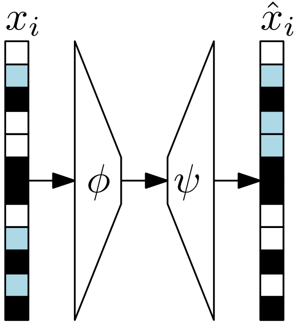
- Finding changes in high-dimensional data is hard!
  - Exponential number of subspaces, correlation changes, etc.
- Dimensionality reduction = encode data in fewer dimensions while minimizing information loss
- Concept changes $\Rightarrow$ information loss increases



Learn a model of the data and detect if it becomes obsolete

Identification of changes

- Learn lower-dimensional model of the data
- Model encodes data in $d' < d$ dimensions
  - Encoder: $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
  - Decoder: $\psi : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$
  - Reconstruction: $\bar{x}_i = \psi(\phi(x_i))$
- Loss L_i comes from some distribution L
  - $L_i = MSE(x_i, \bar{x}_i)$
- Detect changes in $(L_1, L_2, \ldots, L_t)$



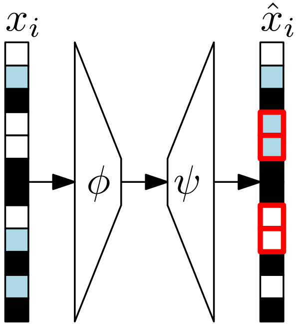
Change subspace and severity

Subspace

- Reconstruction is inaccurate in subspace D^*
- Identify change subspace by examining which dimensions are poorly reconstructed

Severity

- *After* approximation of change subspace
- How severe was the change in the affected subspace?
- Does the reconstruction error correlate with the severity of the change?



Detecting changes in reconstruction error
Adaptive windows

Karlsruhe Institute of Technology

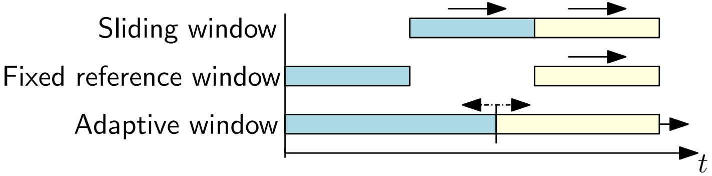
Advantages

- No need to specify window size
- Detection of changes at various time scales
- Larger amount of ‘consistent’ data available

Challenges

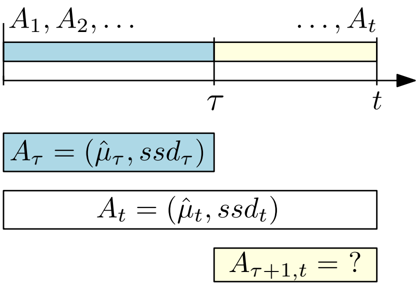
- Runtime and memory

How can we maintain and evaluate an adaptive window efficiently?



Detecting changes in reconstruction error
Stream aggregates

- We keep stream aggregates A_i
  - Derived from Welford's and Chan's algorithm [Wel62; CGL82] for online variance updating
- Update and derive mean loss and variance for any time interval in constant time
- Drop old aggregates without information loss



$$\bar{\mu}_{\tau+1,t} = \frac{1}{t-\tau}(t\bar{\mu}_{1,t} - \tau\bar{\mu}_{1,\tau}) \qquad ssd_{\tau+1,t} = ssd_{1,t} - ssd_{1,\tau} - \frac{\tau(t-\tau)}{t}(\bar{\mu}_{1,\tau} - \bar{\mu}_{\tau+1,t})^2$$

Change Subspace and Severity

Change Subspace

1. Given change point t^*
3. Apply change score in j-th dimension
4. Thresholding of resulting value p_j to find D^*

Change Severity Δ is the normalized average loss $\bar{\mu}^{D^*}_{>t^*}$ observed in D^* after the change:

$$\Delta = \frac{\left|\bar{\mu}^{D^*}_{>t^*} - \bar{\mu}^{D^*}_{\leq t^*}\right|}{\sigma^{D^*}_{\leq t^*}}$$

ABCD Experiment Setup

Karlsruhe Institute of Technology

Data streams

- 7 data streams (simulated using real world and synthetic data)
- 3 additional synthetic data streams to evaluate change subspace detection and severity estimation
- $d \in [24, 1024]$

Baselines

- We use Autoencoders, PCA, and Kernel-PCA as encoder-decoder models
- We compare against IBDD [Sou+21], ADWIN-K [FDK19], D3 [Göz+19], WATCH [Fab+21], and IKS [Rei+16]
- For each approach we evaluate a grid of hyperparameters

Precision and recall are based on:

- **TP:** Change detected before the next one.
- **FN:** Change not detected before the next one.
- **FP:** Change detected although no change occurred.

Metrics for subspace and severity:

- Subspace accuracy: treat membership of change subspace as binary classification
- Spearman ρ: correlation coefficient between severity in subspace and ground truth

Incremental Decision Trees
Motivation

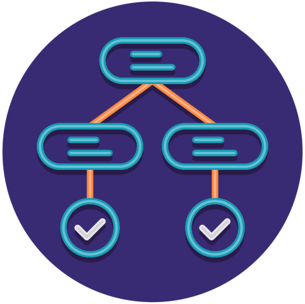
Decision Trees come with many benefits:

- They can handle mixed data
- They are interpretable
- They are popular choices in ensembles (e.g., Random Forest)
- They are robust to outliers

  $\Rightarrow$ Well suited for decision support systems

But how to build and maintain them in data streams?

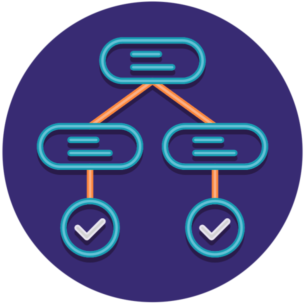
$\Rightarrow$ Incremental decision trees



Decision Trees come with many benefits:

- They can handle mixed data
- They are interpretable
- They are popular choices in ensembles (e.g., Random Forest)
- They are robust to outliers

  $\Rightarrow$ Well suited for decision support systems



But how to build and maintain them in data streams?

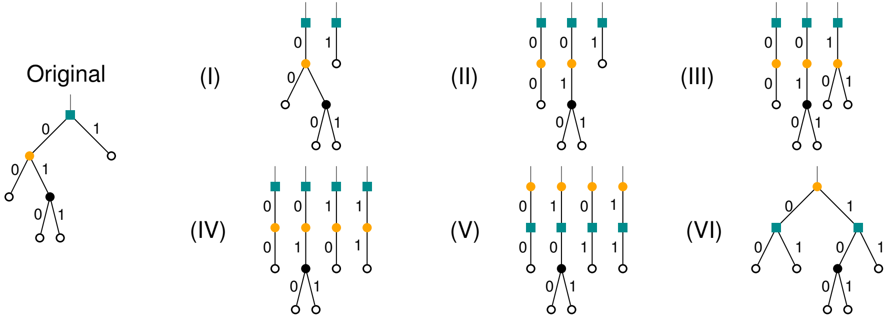
$\Rightarrow$ Incremental decision trees

PLASTIC
Algorithm – Decision tree restructuring

I–IV. Decouple the branches of the tree

V. Reorder each branch

VI. Re-build tree

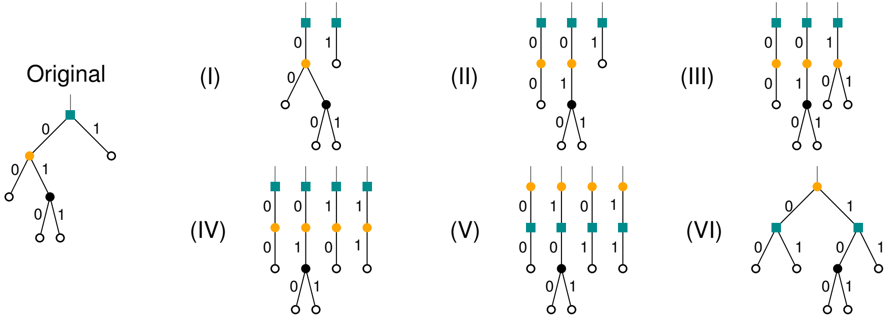


PLASTIC
Algorithm – Decision tree restructuring

I–IV. Decouple the branches of the tree

V. Reorder each branch

VI. Re-build tree

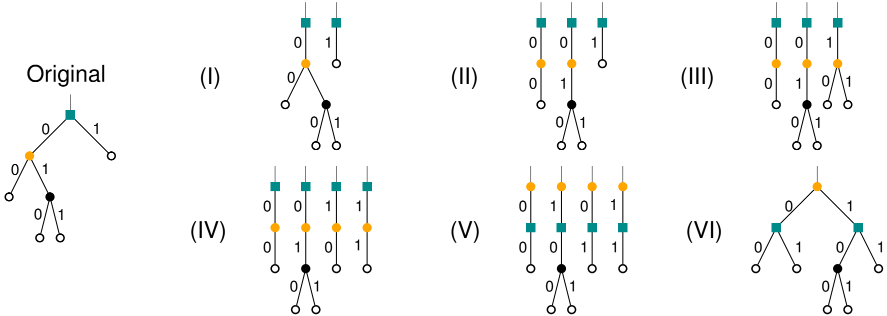


PLASTIC
Algorithm – Decision tree restructuring

I–IV. Decouple the branches of the tree

V. Reorder each branch

VI. Re-build tree



PLASTIC
Numerical and binary categorical splits

Numerical splits

- For numerical splits, split threshold ν^* typically changes

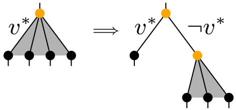
$\Rightarrow$ Adjust split threshold prior to restructuring

$\Rightarrow$ Remove unreachable subtree

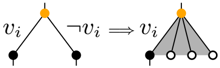
Binary categorical splits (e.g., “color=green $\Rightarrow$ go left”)

- We propose a set of transformations illustrated below

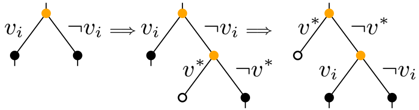
multiway – binary



binary – multiway



binary – binary



Setup and competitors

Karlsruhe Institute of Technology

Experiments

1. Comparison with EFDT

- Evaluates the effect of **decision tree restructuring**
- Comparison of PLASTIC and EFDT
- We use our own implementation of EFDT (based on the same code as PLASTIC)

2. Comparison with HT, EFDT and EFHAT

- Evaluation against **state of the art** decision trees
- We add a simple adaptive version of PLASTIC called PLASTIC-A
  - Trains a background tree when accuracy drops
  - Replaces current tree once it is more accurate

Data streams

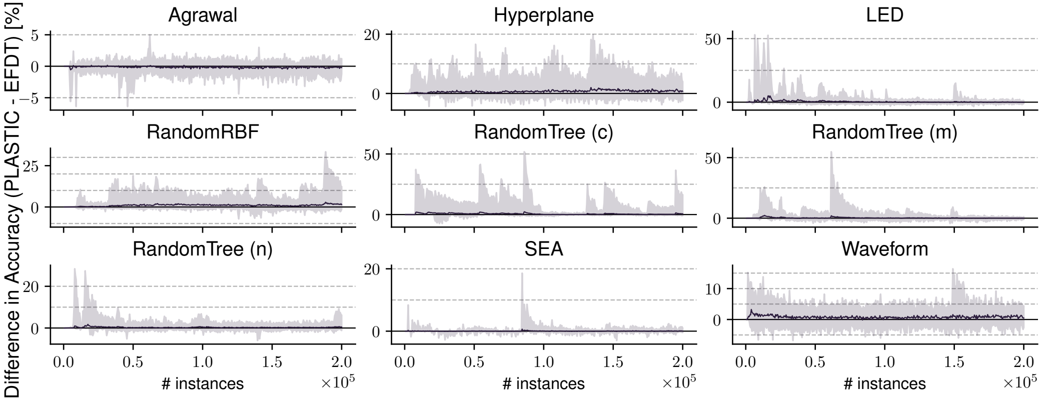
- 9 synthetic, 15 real-world data streams
- 200,000 instances on synthetic data
- Up to 15 million instances on real world data

Evaluation methodology

- Test-then-train evaluation
- Accuracy in sliding window of size 500 (synthetic data) and 1000 (real world data)

Experiments
Comparison to EFDT (synthetic data)

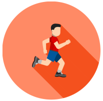
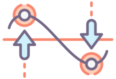
- Graphs show difference in accuracy between PLASTIC and EFDT
- Shaded area shows maximum difference across experiment repetitions



Backup

References

Experiments
Results on real-world data streams

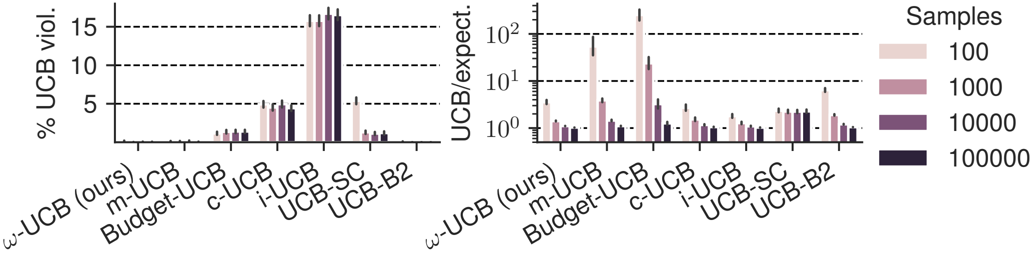
Approach	HT	EFDT	EFHAT	PLASTIC	PLASTIC-A	NoChange
RIALTO	24.2	37.8	42.3	**49.2**	47.4	0.0
SENSORS	15.8	38.2	42.7	**48.1**	47.1	0.1
COVTYPE	68.3	77.4	79.6	**82.1**	81.3	95.1
HARTH	79.5	86.5	89.2	88.3	**90.9**	99.9
PAMAP2	58.4	94.5	98.3	96.6	**98.6**	99.9
WISDM	65.6	80.6	89.0	82.6	**93.1**	99.9
...						
Accuracy	64.8	74.2	76.4	76.7	**77.8**	61.4
Rank	4.21	3.71	2.50	**2.29**	**2.29**	–
Runtime	**61.8**	110.6	198.6	141.6	175.0	27.1

Related work
Existing UCB approaches have issues

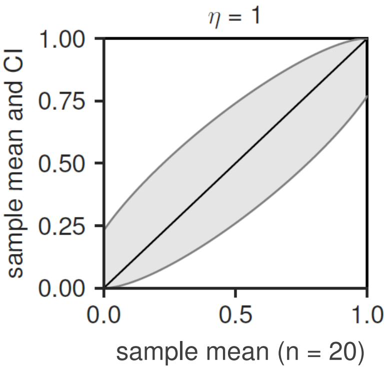
The UCB for the reward-cost ratio should be

- as **accurate** as possible (UCB > expected value)
- as **tight** as possible

$\rightarrow$ but this is not the case.



Our Approach
Asymmetric confidence interval (illustration)

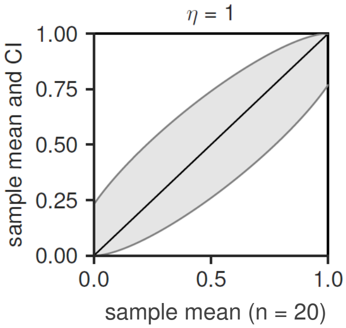


$$\omega_{\pm}(\alpha) = \frac{B}{2A} \pm \sqrt{\frac{B^2}{4A^2} - \frac{C}{A}},$$

$$A = n + z^2\eta, \quad B = 2n\bar{\mu} + z^2\eta(M + m), \quad C = n\bar{\mu}^2 + z^2\eta Mm,$$

Our Approach
Asymmetric confidence interval (illustration)

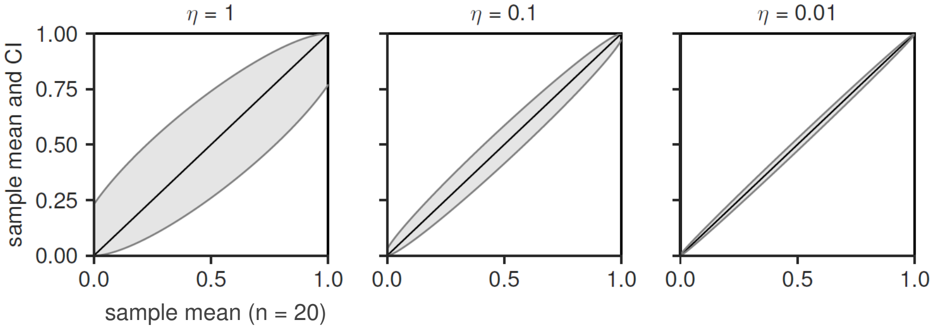
- Asymmetric CI (generalization of Wilson Score Interval [Wil27])
- η: variance parameter ($\eta = 1 \rightarrow$ Bernoulli random variable)



Our Approach
Asymmetric confidence interval (illustration)

- Asymmetric CI (generalization of Wilson Score Interval [Wil27])
- η: variance parameter ($\eta = 1 \rightarrow$ Bernoulli random variable)

$$\eta = \frac{\sigma^2}{(1-\mu)\mu}$$



Results (II)

Karlsruhe Institute of Technology

Theorem (Instance-dependent regret bound (based on [Xia+17]))

Define $\tau_B = \lfloor 2B/\min_{k\in[K]} \mu_k^c \rfloor$ and Δ_k, $n_k^*(\tau_B)$, and $\xi(\tau_B, \rho)$ as before. For any $\rho > 0$, the regret of ω-UCB is upper-bounded by

$$\textit{Regret} \leq \sum_{k=2}^{K} \Delta_k \left(1 + n_k^*(\tau_B) + \xi(\tau_B, \rho)\right) + \mathcal{X}(B) \sum_{k=2}^{K} \Delta_k + \frac{2\mu_1^r}{\mu_1^c},$$

where $\mathcal{X}(B)$ is in $\mathcal{O}\left(\frac{B}{\mu_{min}^c} e^{-0.5B\mu_{min}^c}\right)$.

Theorem (Asymptotic regret)

The regret of ω-UCB is

$$\textit{Regret} \in \mathcal{O}\left(B^{1-\rho}\right), \ \textit{for}\ 0 < \rho < 1; \qquad \textit{Regret} \in \mathcal{O}(\log B), \ \textit{for}\ \rho \geq 1$$