Learning-Augmented Online Algorithms

Bertrand Simon

IN2P3 Computing Center / CNRS, Villeurbanne

Journées CALAMAR – January 2024

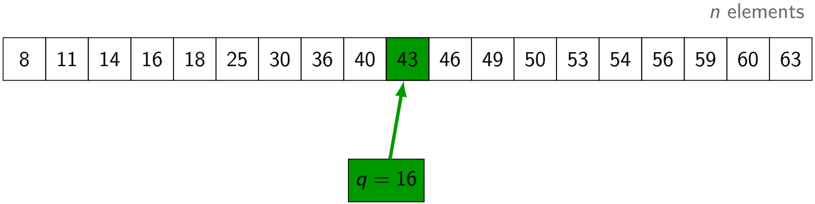
Motivating example: binary search

n elements

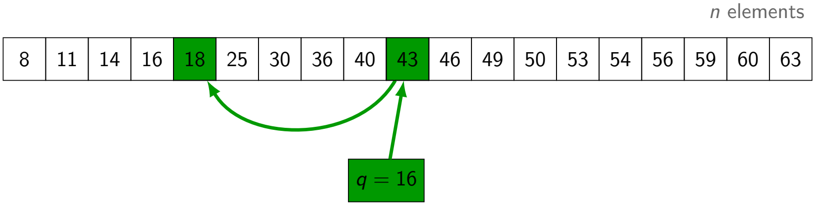
8	11	14	16	18	25	30	36	40	43	46	49	50	53	54	56	59	60	63

$q = 16$

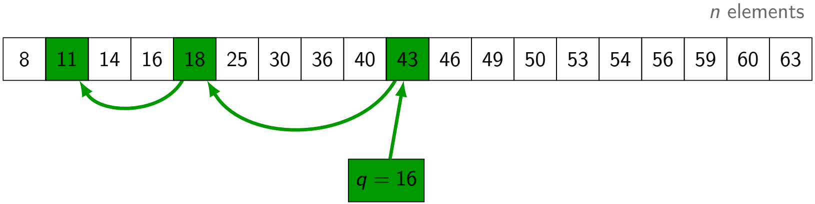
Motivating example: binary search



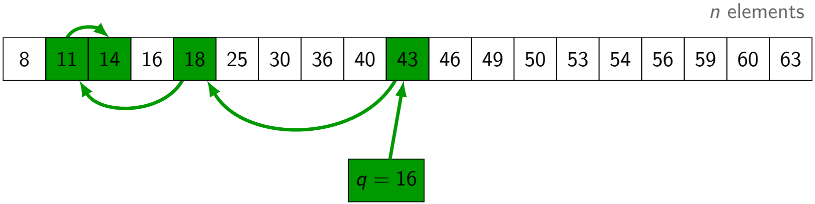
Motivating example: binary search



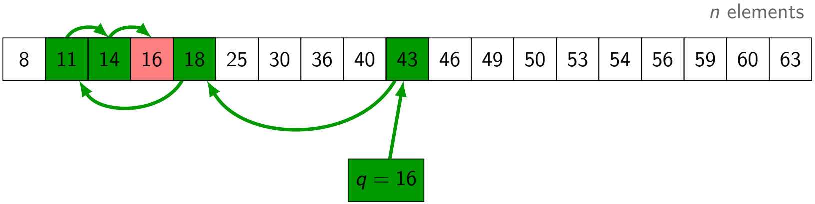
Motivating example: binary search



Motivating example: binary search



Motivating example: binary search



Motivating example: binary search

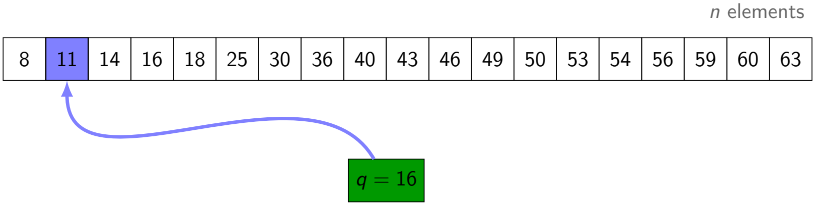
n elements

8	11	14	16	18	25	30	36	40	43	46	49	50	53	54	56	59	60	63

$q = 16$

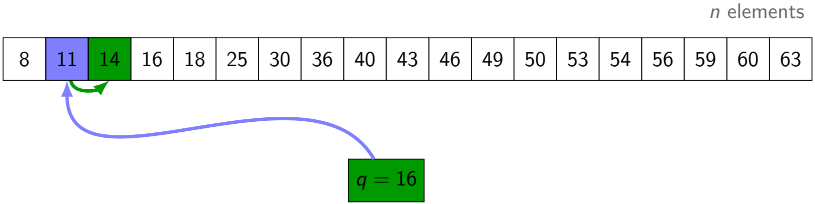
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



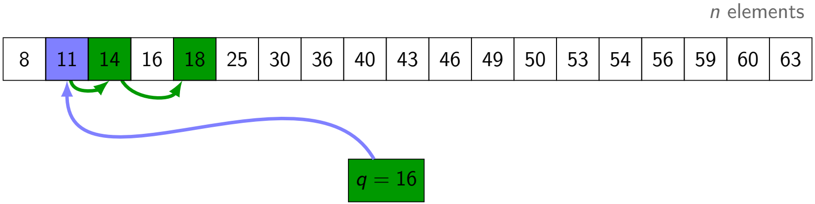
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



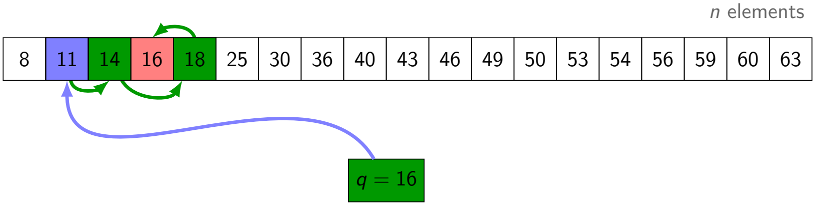
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



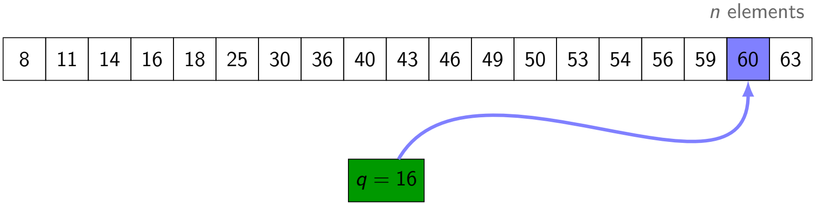
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



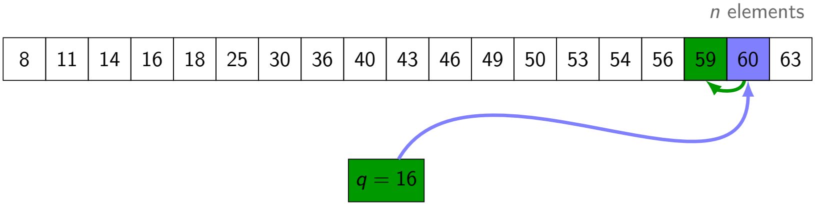
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



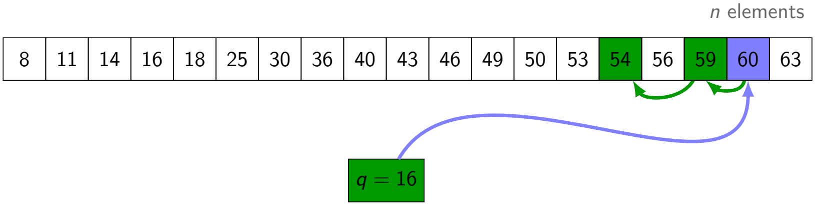
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



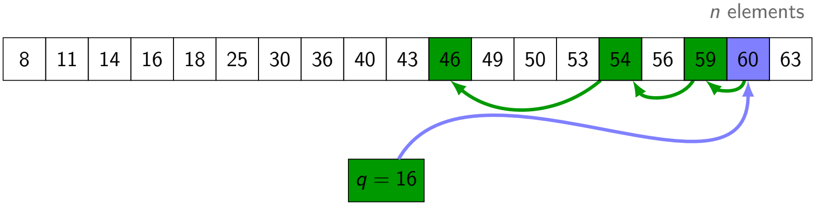
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



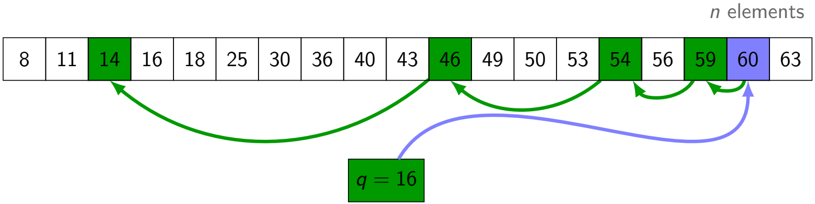
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



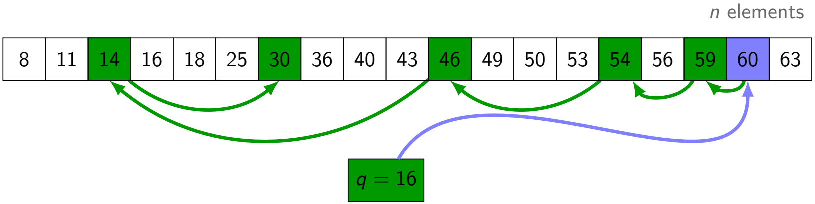
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



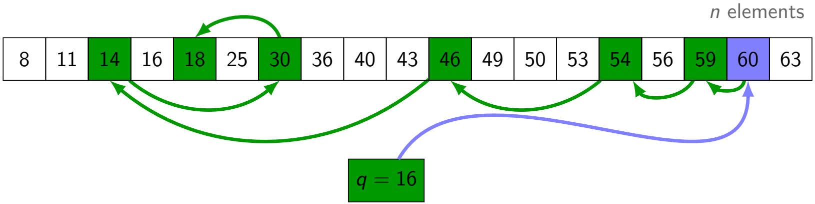
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



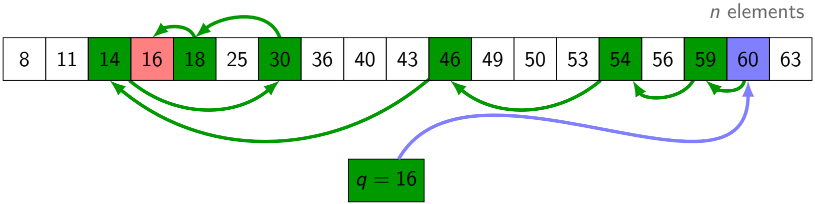
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



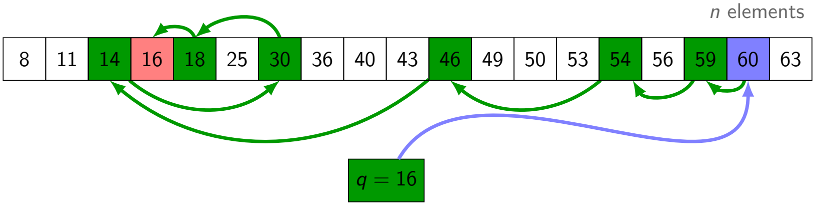
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search

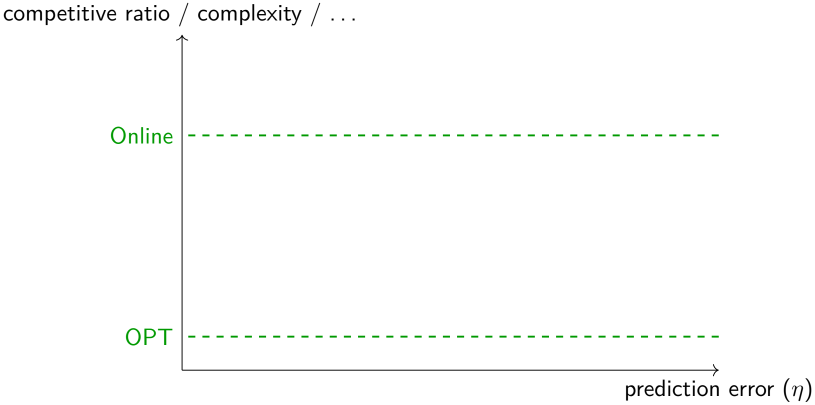


Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Classic: $\Theta(\log n)$ $\xrightarrow{\text{predictions}}$ $\Theta(\log \eta)$

Practical applications [KraskaBeutalChiDeanPolyzotis'18]

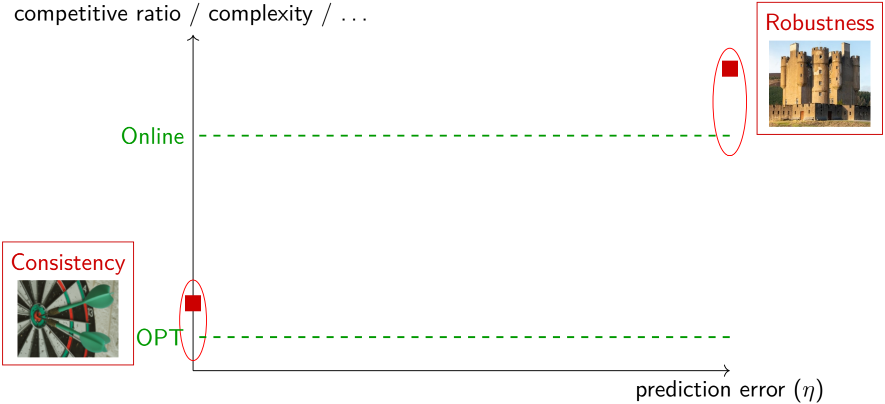
Properties we seek



Algorithms are oblivious to η

Prediction *h* should be *learnable*, e.g., compact

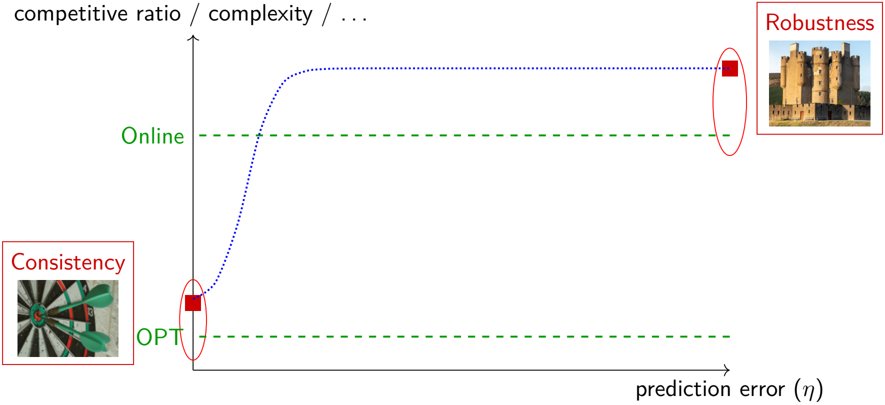
Properties we seek



Algorithms are oblivious to η

Prediction *h* should be *learnable*, e.g., compact

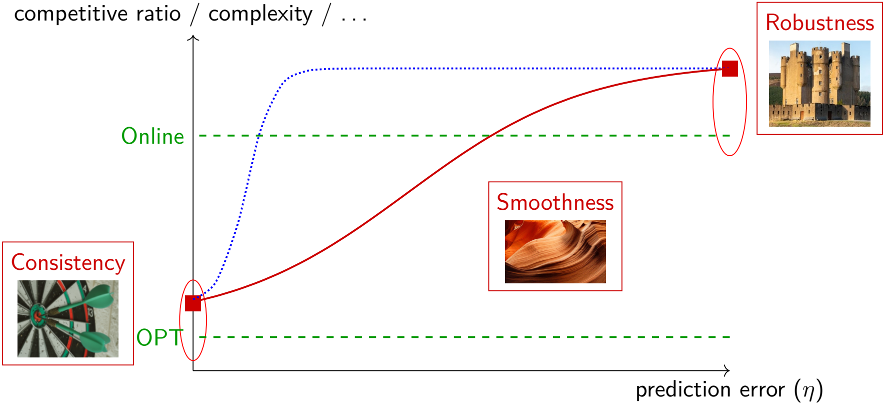
Properties we seek



Algorithms are oblivious to η

Prediction *h* should be *learnable*, e.g., compact

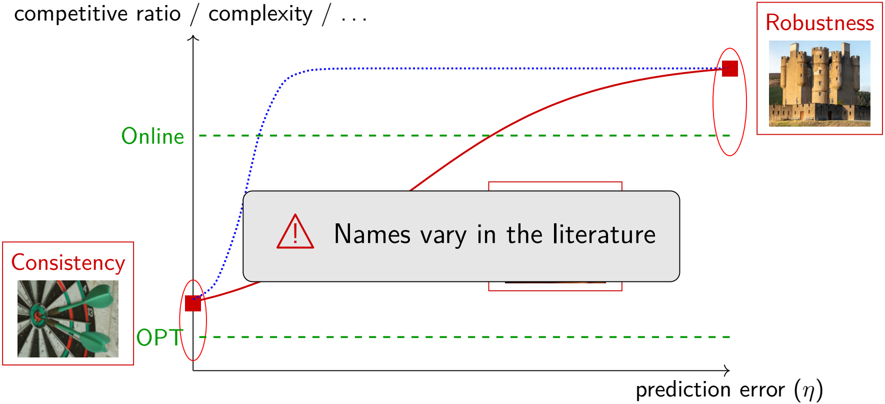
Properties we seek



Algorithms are oblivious to η

Prediction h should be *learnable*, e.g., compact

Properties we seek



Algorithms are oblivious to η

Prediction h should be *learnable*, e.g., compact

“Classic” Beyond worst-case analysis

Future instance: X_1 ; X_2 ; X_3 ; X_4 ; X_5 ; ...

Lookahead
$X_1 = 5$

Semi-online
$\sum_i X_i = 30$

Random arrival


Advice
1101110

Stochastic input
$X_i \sim \mathcal{N}(10, 5)$

Robust analysis
$X_1 = 5 \pm 2$, $X_2 = 7 \pm 3$, ...

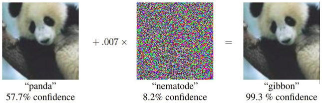
...

☹ Strong assumptions, needs some perfect information (oracle)

HERE: no assumption on the predictor

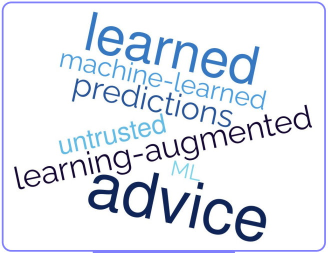
allows plug-and-play predictors

 $\Longrightarrow$ 



arxiv.org/abs/1412.6572

Landscape preview

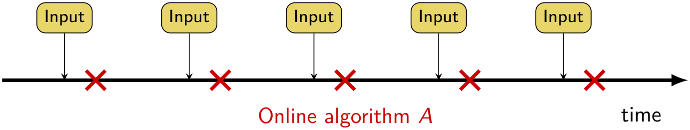


Denomination

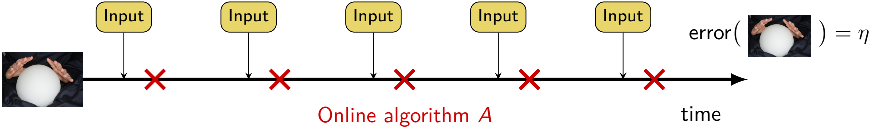
Themes

100s publications since 2018
`https://algorithms-with-predictions.github.io/`

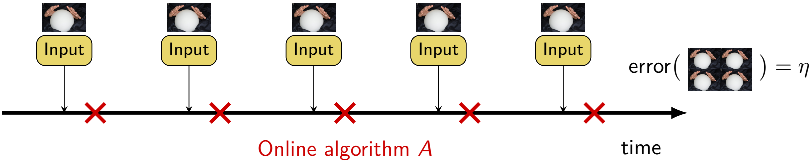
Most common framework used



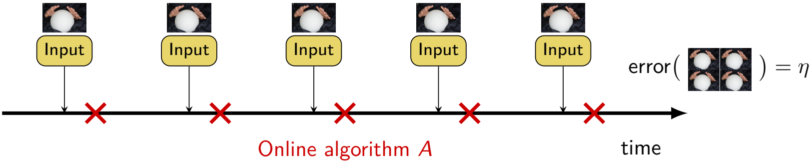
Most common framework used



Most common framework used



Most common framework used



Objective: “minimize” competitive ratio $c_A(\eta) = \max_I \frac{cost_A(I)}{OPT(I)}$

Consistency

$c_A(0)$

Robustness

$c_A(\infty)$

Smoothness

“slope” of $c_A(\eta)$

Outline

First example: Ski rental [PurohitSK'18]

b — cost to buy skis

1 — daily rent price

x ? — # ski days (unknown)

Online algorithm: minimize max $\frac{\text{cost}}{\textsc{Opt}}$

Best deterministic algo: buy at day $\approx b$

Worst-case = stop after day b:

- $\textsc{Opt} = b$
- cost $= 2b$
- $\implies$ competitive ratio $= 2$

First example: Ski rental [PurohitSK'18]

b — cost to buy skis

1 — daily rent price

x ? — # ski days (unknown)

What should h predict ?

- ☹ $h \longrightarrow$ 0/1: rent or buy ? cannot measure η
- ☺ $h \longrightarrow$ x with $\eta = |h - x|$

What should the algorithm do ?

First example: Ski rental [PurohitSK'18]

b — cost to buy skis

1 — daily rent price

x ? — # ski days (unknown)

What should h predict ?

- ☹ $h \longrightarrow$ 0/1: rent or buy ? cannot measure η
- ☺ $h \longrightarrow x$ with $\eta = |h - x|$

What should the algorithm do ?

NAIVE : if $h \geq b$ then buy on day 1 else rent forever

First example: Ski rental [PurohitSK'18]

b — cost to buy skis

1 — daily rent price

x ? — # ski days (unknown)

What should h predict ?

- ☹ $h \longrightarrow$ 0/1: rent or buy ? cannot measure η
- ☺ $h \longrightarrow$ x with $\eta = |h - x|$

What should the algorithm do ?

NAIVE : if $h \geq b$ then buy on day 1 else rent forever

Lemma

The competitive ratio of NAIVE *is* $1 + \eta / \textbf{OPT}$.

~~Robust~~

A robust algorithm for Ski Rental [Purohit Switkina Kumar'18]

Intuition: if h x , we should not buy at day 1

How long should we rent ? depends on the predictor's "trustworthiness"

Consistent


$\lambda = 0$ $\lambda = 1$



Robust


$\textsc{SkiPred}(\lambda)$: ▶ If $h \geq b$: rent $\lceil \lambda b \rceil$ days ▶ Else: rent $\lceil b/\lambda \rceil$ days

rent	b	h

→ time

Theorem

$\textsc{SkiPred}(\frac{1}{2})$ *is:* $\min\left(3 \ , \ 1.5 \ + \ 2 \ \cdot \frac{\eta}{\textsc{Opt}} \right)$ *- competitive.*



A robust algorithm for Ski Rental [Purohit Switkina Kumar'18]

Intuition: if h x , we should not buy at day 1

How long should we rent ? depends on the predictor's "trustworthiness"

Consistent


$\lambda = 0$ $\lambda = 1$






Robust


SkiPred(λ): ▶ If $h \geq b$: rent $\lceil \lambda b \rceil$ days ▶ Else: rent $\lceil b/\lambda \rceil$ days

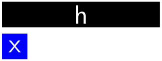
rent h b → time

Theorem

SkiPred($\frac{1}{2}$) *is:* min $\left(3 \quad , \quad 1.5 \quad + \quad 2 \quad \cdot \frac{\eta}{\textsc{Opt}} \right)$ *- competitive.*



A robust algorithm for Ski Rental [Purohit Switkina Kumar'18]

Intuition: if 

, we should not buy at day 1

How long should we rent ? depends on the predictor's "trustworthiness"

$\textsc{SkiPred}(\lambda)$:

- If $h \geq b$: rent $\lceil \lambda b \rceil$ days
- Else: rent $\lceil b/\lambda \rceil$ days

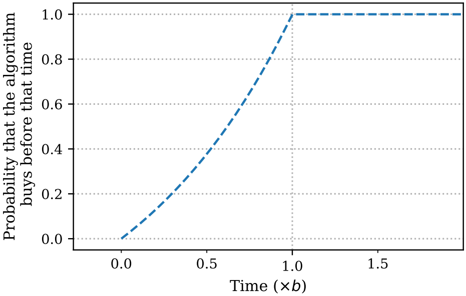
→ time

Theorem

$\textsc{SkiPred}(\lambda)$ *is:* $\min\left(\frac{1+\lambda}{\lambda} \;,\; (1+\lambda) + \frac{1}{1-\lambda} \cdot \frac{\eta}{\textsc{Opt}} \right)$ *- competitive.*

Randomized ski rental [Purohit Switkina Kumar'18]

Classic randomized ski rental → $\frac{e}{e-1} \approx 1.58$-competitive



Randomized ski rental [Purohit Switkina Kumar'18]

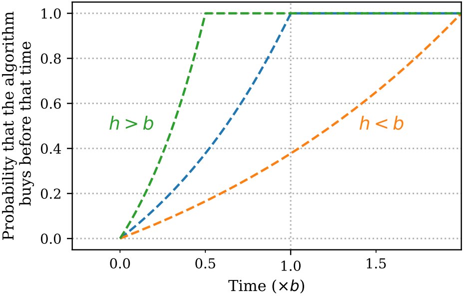
Classic randomized ski rental $\rightarrow$ $\frac{e}{e-1} \approx 1.58$-competitive

Theorem

There is a $O\left(\min\left(\frac{1}{1-e^{\lambda}}\ ,\ \frac{\lambda}{1-e^{-\lambda}}\left(1+\frac{\eta}{\textsc{Opt}}\right)\right)\right)$*-competitive algorithm.*

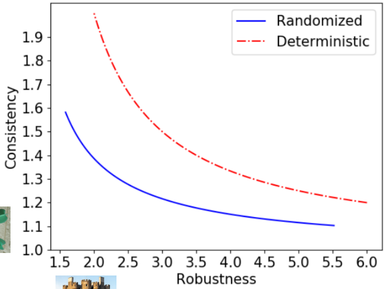


e.g., $\lambda = 1/2$



Consistency vs Robustness [Purohit Switkina Kumar'18]

ONLINE

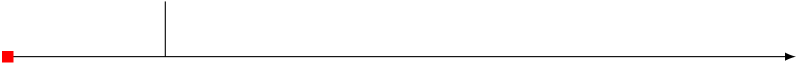


NAIVE

Lower bounds:

- Randomized: matches UB [WeiZhang'20]
- Deterministic: LB a bit lower but  [AngelopoulosDürrJinKamaliRenault'19]

Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]



state() $\in$ {    }

Dynamic Power Management: shift of focus

- focus first on 2 states
-  is "free" over many rounds (experts framework)

Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]

state() $\in$ {    }

Dynamic Power Management: shift of focus

- focus first on 2 states
-  is "free" over many rounds (experts framework)

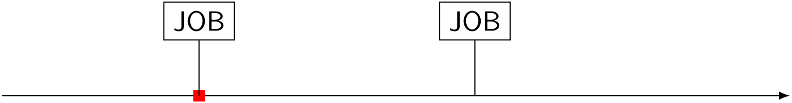
Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]

state() $\in$ {    }

Dynamic Power Management: shift of focus

- focus first on 2 states
-  is "free" over many rounds (experts framework)

Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]



state() $\in$ {    }

Dynamic Power Management: shift of focus

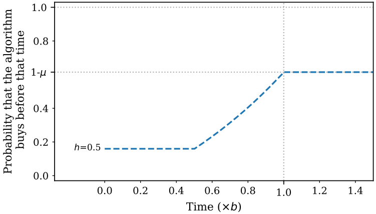
- focus first on 2 states
-  is "free" over many rounds (experts framework)

Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]

New tradeoff:  vs  in cost $=$  $\cdot \textsc{Opt} +$  $\cdot \eta$

Use the whole prediction

Example for a ≈ 1.16-consistent, 0.38-smooth solution:

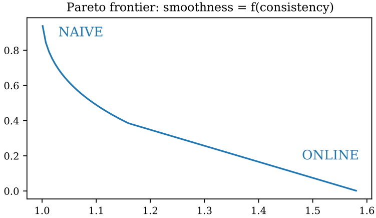


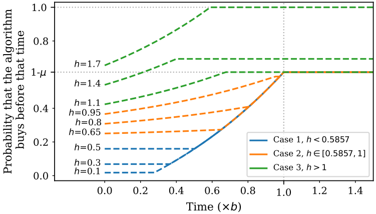
Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]

New tradeoff:  vs  in cost $=$  $\cdot\, \textsc{Opt} +$  $\cdot\, \eta$

Use the whole prediction

Example for a ≈ 1.16-consistent, 0.38-smooth solution:



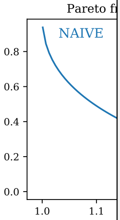


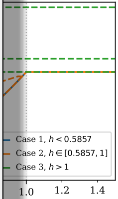
Multi-round ski rental [Antoniadis Coester Elias Polak Simon '21]

New tradeo  $\cdot \eta$

Use the wh

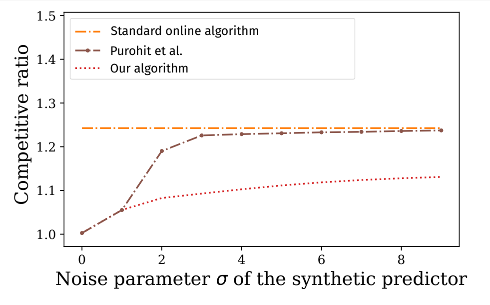
Example fo





Example of simulation results

4 actual idle states, random instances + predictions



Outline

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 1

pages $\in \{A, B, \ldots, F\}$

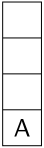
1

A

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 2

pages $\in \{A, B, \ldots, F\}$



1 2

A B

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 2

pages $\in \{A, B, \ldots, F\}$

B
A

1	2	3
A	B	A

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 3

pages $\in \{A, B, \ldots, F\}$

B
A

1	2	3	4
A	B	A	C

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 4

pages $\in \{A, B, \dots, F\}$

C
B
A

1	2	3	4	5
A	B	A	C	D

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 5

pages $\in \{A, B, \ldots, F\}$

D
C
B
A

1	2	3	4	5	6
A	B	A	C	D	E

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 6

pages $\in \{A, B, \ldots, F\}$

D
C
E
A

1	2	3	4	5	6	7
A	B	A	C	D	E	F

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 6

pages $\in \{A, B, \ldots, F\}$

D
F
E
A

1	2	3	4	5	6	7	8
A	B	A	C	D	E	F	A

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 7

pages $\in \{A, B, \ldots, F\}$

D
F
E
A

1	2	3	4	5	6	7	8	9
A	B	A	C	D	E	F	A	B

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 7

pages $\in \{A, B, \ldots, F\}$

D
B
E
A

1	2	3	4	5	6	7	8	9	10
A	B	A	C	D	E	F	A	B	E

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 8

pages $\in \{A, B, \ldots, F\}$

D
B
E
A

1	2	3	4	5	6	7	8	9	10	11
A	B	A	C	D	E	F	A	B	E	F

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 8

pages $\in \{A, B, \ldots, F\}$

F
B
E
A

1	2	3	4	5	6	7	8	9	10	11
A	B	A	C	D	E	F	A	B	E	F

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 8

pages $\in \{A, B, \ldots, F\}$

F
B
E
A

1	2	3	4	5	6	7	8	9	10	11
A	B	A	C	D	E	F	A	B	E	F

Q: What to predict?

Lookahead *(next q requests)*

- ☹ useless in the worst case

Strong Lookahead *(next requests until q distinct)*

- ☹ huge, hard to predict

Next arrival time of the current request

- ☺ compact, enough to compute $\textsc{Opt}$, arguably learnable
- error η_i at round i : distance between predicted time and actual time combined error $\eta = \sum \eta_i$.
- $\Longrightarrow$ get a $\approx \min(\log k, \log \frac{\eta}{\textsc{Opt}})$-competitive algorithm

Paging with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 8

pages $\in \{A, B, \ldots, F\}$

F
B
E
A

	1	2	3	4	5	6	7	8	9	10	11
	A	B	A	C	D	E	F	A	B	E	F
next:	3	9	8	-	-	10	11	-	-	-	-

Q: What to predict?

Lookahead *(next q requests)*

- ☹ useless in the worst case

Strong Lookahead *(next requests until q distinct)*

- ☹ huge, hard to predict

Next arrival time of the current request

- ☺ compact, enough to compute $\textsc{Opt}$, arguably learnable
- error η_i at round i : distance between predicted time and actual time combined error $\eta = \sum \eta_i$.
- $\Longrightarrow$ get a $\approx \min(\log k, \log \frac{\eta}{\textsc{Opt}})$-competitive algorithm

Paging with predictions – Overview of models

Predictions = time of next occurrence of current page

- Lykouris Vassilvitskii (2021 JACM); Rohatgi (SODA 2020); Wei (APPROX/RANDOM 2020)

Predictions = all pages before next occurrence of current page

- Jiang Panigrahi Su (ICALP 2020)

Predictions = state of OPT (which pages in cache)

- Antoniadis Coester Elias Polak Simon (ICML 2020)

Multiple predictors — time of next occurrence of current page

- Emek Kutten Shi (ITCS 2020)

Prediction queries — obtain next occurrence of any page in cache

- Im Kumar Petety Purohit (ICML 2022)

Succinct predictions = 1 bit of information ($\approx$ to evicted or not)

- Antoniadis Boyar Eliáš Favrholdt Hoeksma Larsen Polak Simon (ICML 2023)

A general error measure for graph algorithms

[Azar Panigrahi Touitou, Online Graph Algorithms with Predictions, SODA 2022]

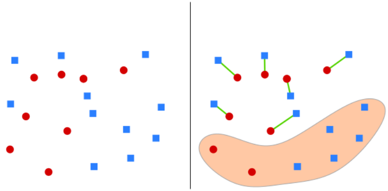


Figure 1: An illustration of metric error with outliers. The figur

Error measure for predicting a set of points in a metric space $\eta = (D, \Delta)$:
D = transportation distance ; Δ = #outliers

Theorem

For the STEINER TREE *and* FACILITY LOCATION *problems, if the error of the predicted input (resp. set of terminal and set of clients) is* (D, Δ)*, there is an algorithm of cost at most* $O(\log \Delta)\,\textbf{OPT} + O(D)$.

Faster matching via learned duals

[Dinitz Im Lavastida Moseley Vassilvitskii NeurIPS 2021]

[Chen Silwal Vakilian Zhang ICML 2022]

Theorem

Given a weighted bipartite graph and predicted dual $\hat{y}$ *, there exists an algorithm that finds a minimum weight perfect matching in time* $O(m\sqrt{n} + (m + n \log n)||y^* - \hat{y}||_0)$*, where* y^* *is an optimal dual solution.*

Main ideas:

- use predicted dual as a warm start for the Hungarian algorithm
- if this dual is not feasible, adapt it
- actually design the algorithm that get the solution faster if the prediction error is small
- show that the duals can be learned from samples of a probabilistic input

Outline

Conclusion

Take-back messages

-   
- fresh algorithm concepts
- relevant link ML – algorithms

Newer questions

- improve running time
- ensure learnability (*e.g.*, PAC-learnability) / $\eta \approx$ loss function
- extensive experiments including ML predictors
- multiple predictors
-  wrt renowned heuristics ?