Learning-Augmented Online Algorithms

Bertrand Simon

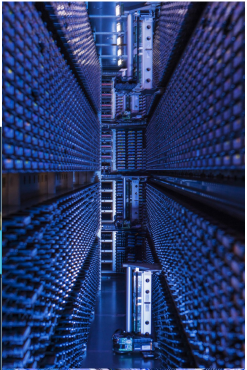
IN2P3 Computing Center / CNRS, Villeurbanne

ROADEF – February 2022

Research Infrastructure

Data processing of physics experiments

85 people (70 IT engineers)
80 international experiments
Annual budget : 7M €



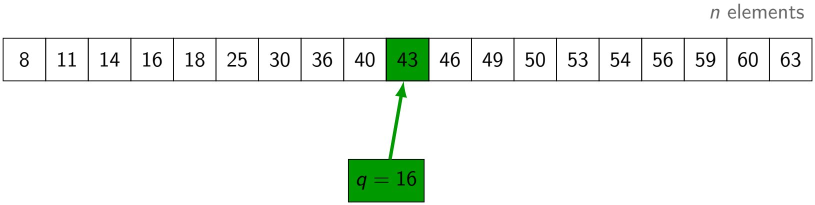
Motivating example: binary search

n elements

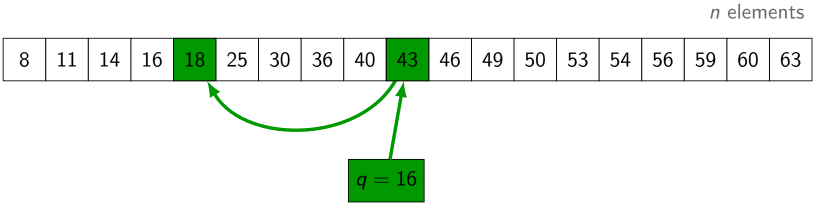
8	11	14	16	18	25	30	36	40	43	46	49	50	53	54	56	59	60	63

$q = 16$

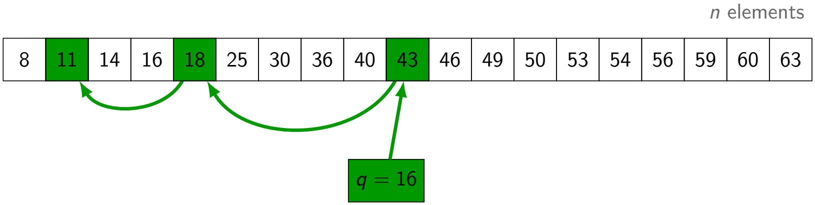
Motivating example: binary search



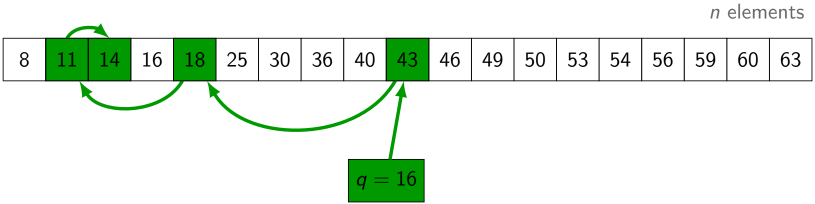
Motivating example: binary search



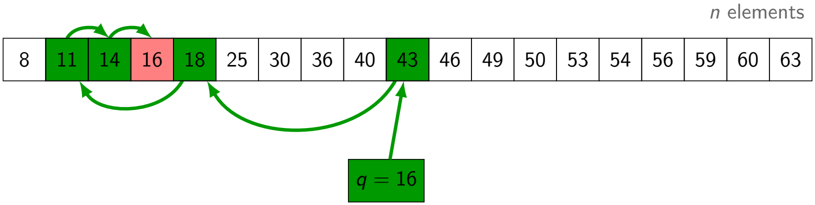
Motivating example: binary search



Motivating example: binary search



Motivating example: binary search



Motivating example: binary search

n elements

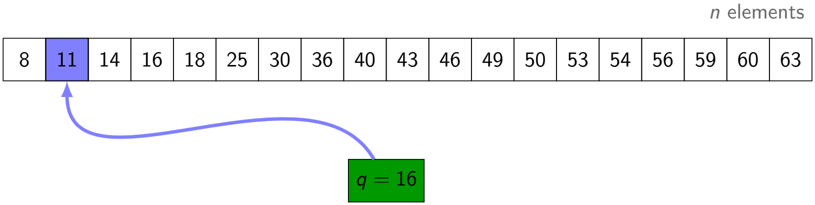
8	11	14	16	18	25	30	36	40	43	46	49	50	53	54	56	59	60	63

$q = 16$

Prediction: position $h(q)$

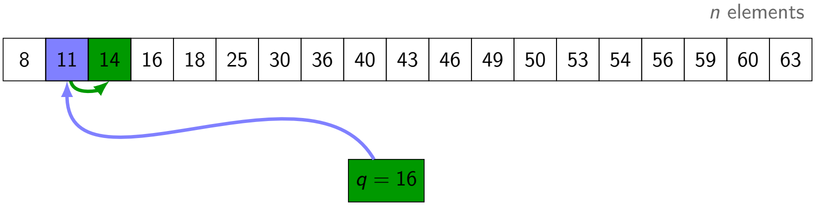
Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



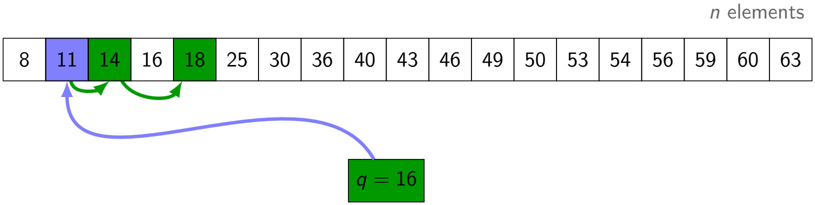
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



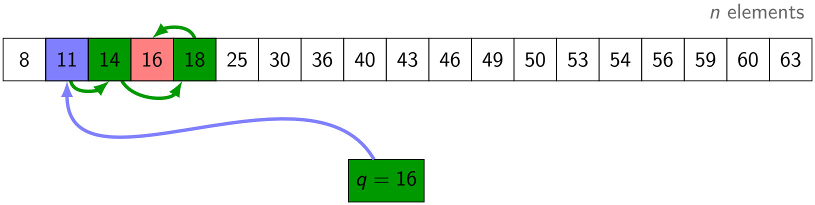
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



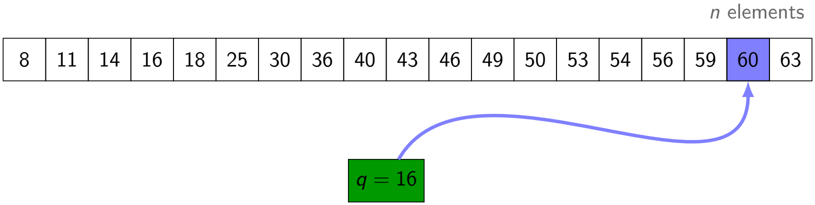
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



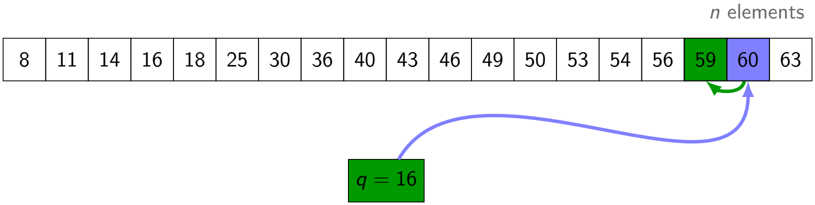
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



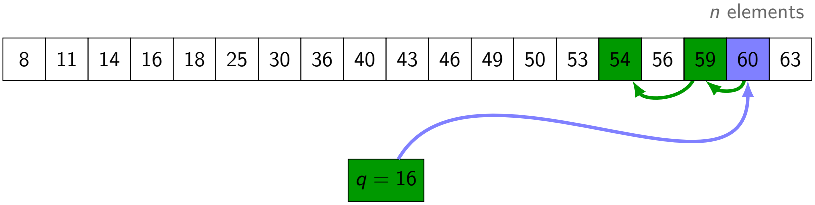
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



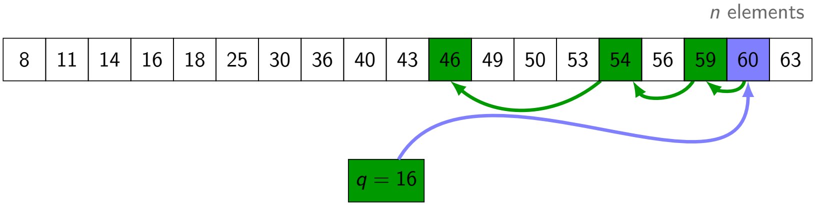
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



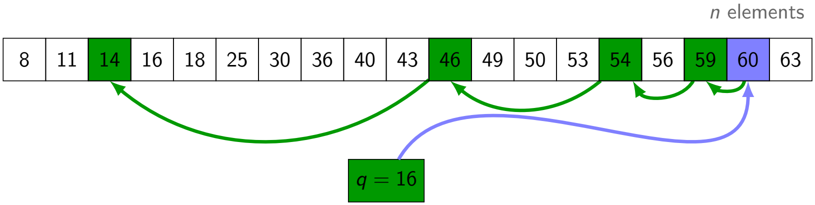
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



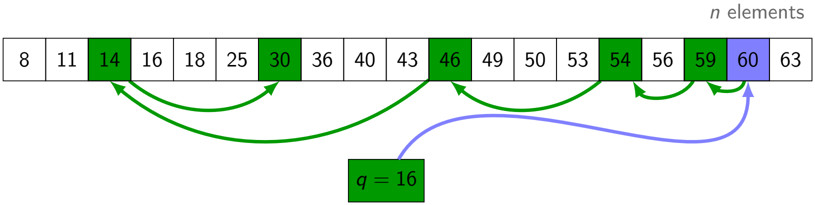
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



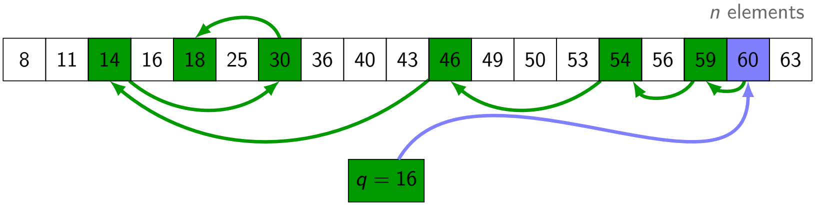
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



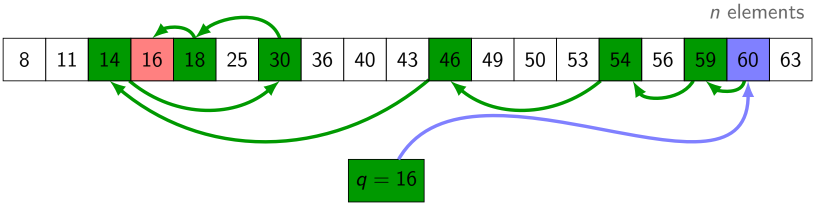
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



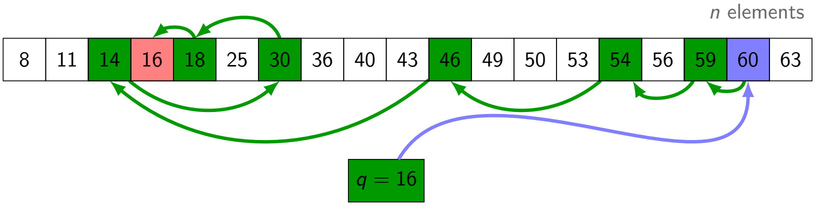
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



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

Motivating example: binary search

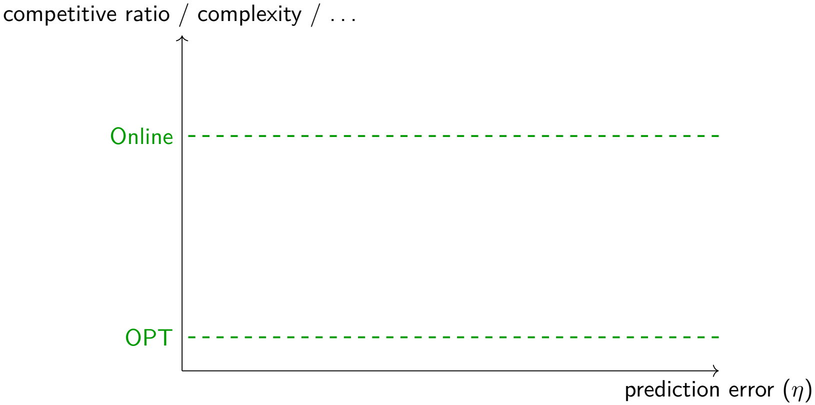


Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Classic: $\Theta(\log n)$ —predictions→ $\Theta(\log \eta)$

Practical applications [KraskaBCDP '18]

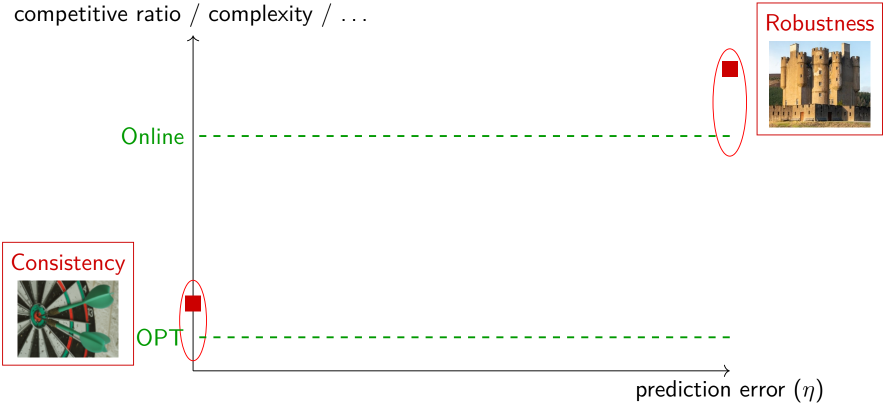
Properties we seek



Algorithms are oblivious to η

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

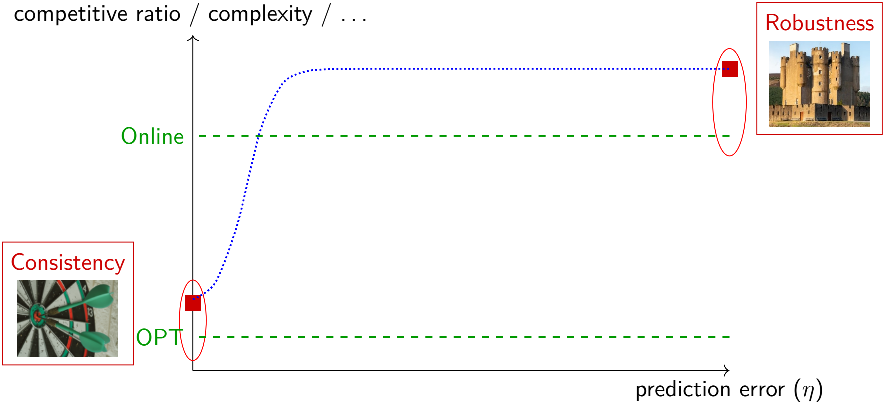
Properties we seek



Algorithms are oblivious to η

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

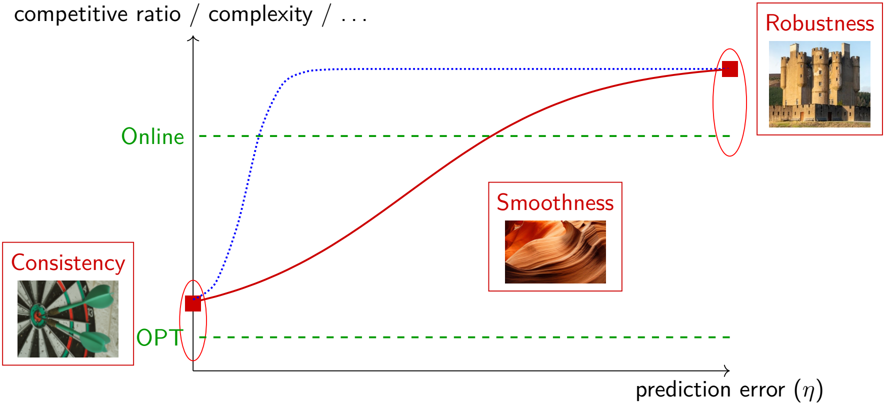
Properties we seek



Algorithms are oblivious to η

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

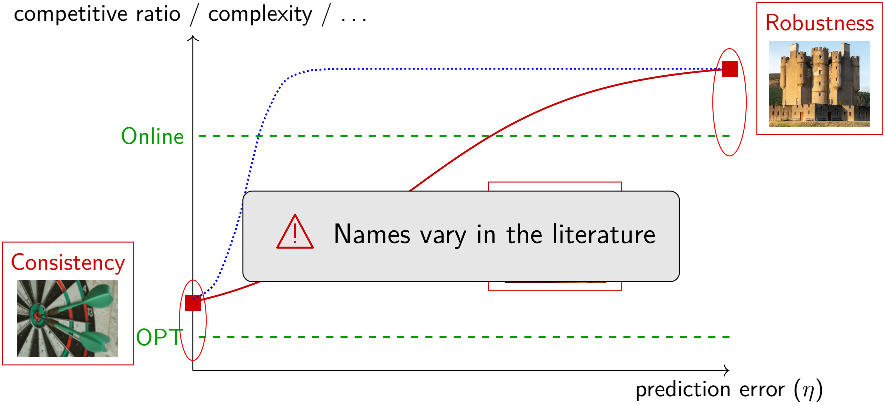
Properties we seek



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

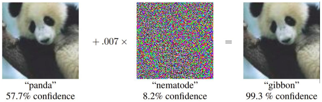
...

☹ Strong assumptions, needs some perfect information (oracle)

HERE: no assumption on the predictor

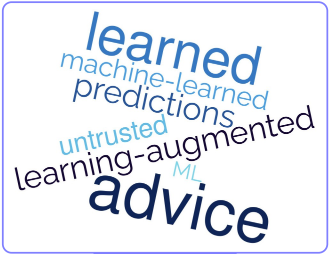
allows plug-and-play predictors

 $\Longrightarrow$ 



arxiv.org/abs/1412.6572

Landscape

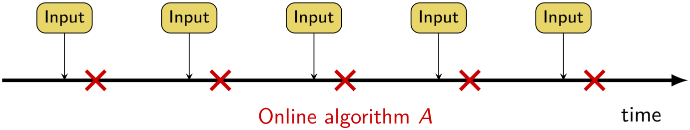


Denomination

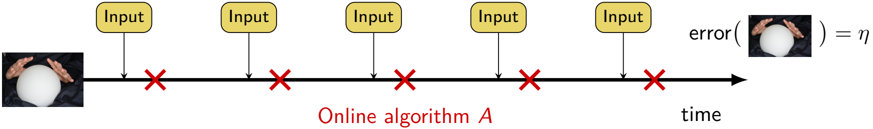
Themes

≈ 100 publications
since 2018

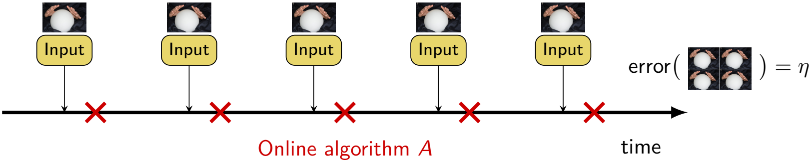
Most common framework used



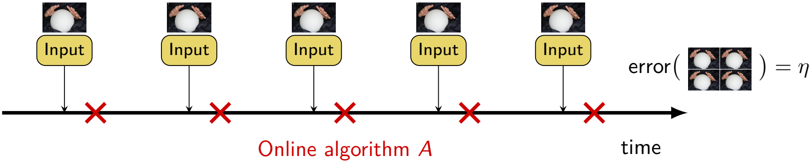
Most common framework used



Most common framework used



Most common framework used



Objective: "minimize" competitive ratio $c_A(\eta)$ (may need $\textsc{Opt}$ to scale)

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)

What should h predict ?

- ☹ $h \longrightarrow 0/1$: rent or buy ? cannot measure η

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* $1 + \eta/\textsc{Opt}$.

A robust algorithm for Ski Rental [PurohitSK'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

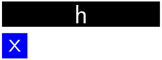
⟶ time

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 [PurohitSK'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

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 [PurohitSK'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

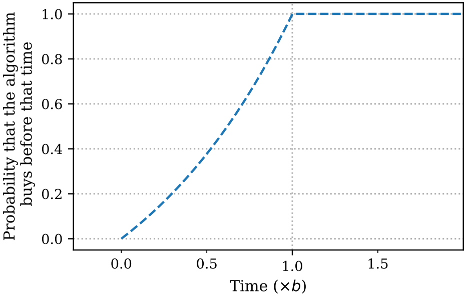
Theorem

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



Randomized ski rental [PurohitSK'18]

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



Randomized ski rental [PurohitSK'18]

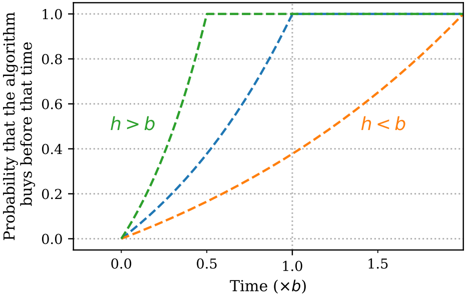
Classic randomized ski rental → $\frac{e}{e-1} \approx 1.58$-competitive

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.*

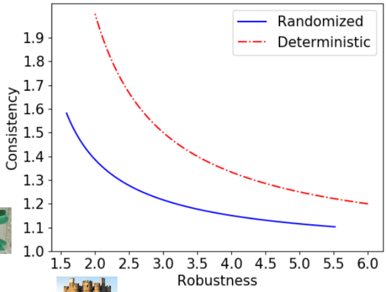


e.g., $\lambda = 1/2$



Consistency vs Robustness [PurohitSK'18]

ONLINE



NAIVE

Lower bounds:

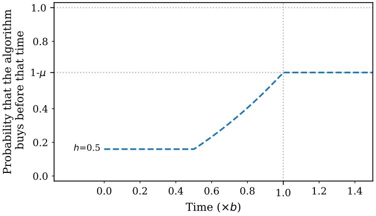
- Randomized: matches UB [WeiZhang'20]
- Deterministic: LB a bit lower but  [AngelopoulosDJKR'19]

Multi-round ski rental [AntoniadisCEPS'21]

Shift of focus (1 prediction per round)

-  is "free" over many rounds (experts framework)
- new tradeoff:  vs  in cost =  $\cdot \textsc{Opt} +$  $\cdot \eta$
- motivation: dynamic power management

New: use the whole prediction

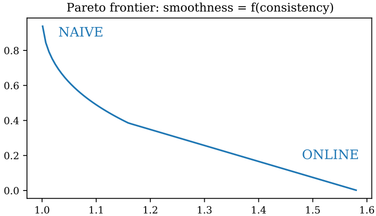


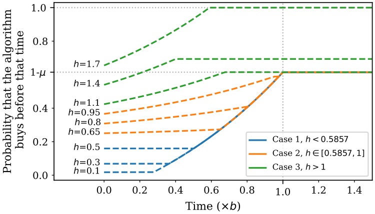
Multi-round ski rental [AntoniadisCEPS'21]

Shift of focus (1 prediction per round)

-  is "free" over many rounds (experts framework)
- new tradeoff:  vs  in cost =  $\cdot$ OPT +  $\cdot\, \eta$
- motivation: dynamic power management

New: use the whole prediction



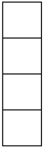


Outline

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 1

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



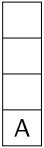
1

A

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 2

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

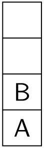


1	2
A	B

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 2

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



1	2	3
A	B	A

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 3

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

B
A

1	2	3	4
A	B	A	C

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 4

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

C
B
A

1	2	3	4	5
A	B	A	C	D

Caching 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

Caching 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

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 6

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

D
F
E
A

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

Caching 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

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 7

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

Cache
D
B
E
A

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

Caching 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

Caching 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

Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 8

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

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 OPT, arguably learnable
- error η_i at round i : distance between predicted time and actual time combined error $\eta = \sum \eta_i$.

Caching 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 OPT, arguably learnable
- error η_i at round i : distance between predicted time and actual time combined error $\eta = \sum \eta_i$.

What if we "Follow The Predictions"?

FTP: evict the latest predicted page

- ☺ If $\eta = 0 \rightarrow$ OPT 
- ☺ get  $(\log k)$ by combination
- Is it a good candidate? What about  ?

[L&V'18] : for $k = 2$, take the sequence

$$A\ BCBCBCBC\ A\ BCBCBCBC\ A\ \ldots$$

Predict B, C correctly and A asap: $\eta =$ total length ; $\text{OPT} = \#A$

FTP's competitive ratio is at least $\Omega(\eta/\text{OPT})$ for $k = 2$.
No trivial fix known.

☹ We need better smoothness 

Classic online solution: Marker

Divide input in phases: maximum subsequences of $\leq k$ distinct pages

Example for $k = 3$: $A, B, D, A, \mid C, E, C, B, E, C, C, \mid A, B, E, \mid D, \ldots$

Definition (marking algorithms)

Marked pages: previously requested in the current phase.
A *Marking algorithm* **never** evicts *marked pages*.

Marker algorithm: evict an unmarked page uniformly at random

Classic results:
- Marker is $2H_k$-competitive ($O(\log k)$)
- Opt $\geq$ #phases
- marking algorithms $\in [2, k]$-competitive

Classic online solution: Marker

Divide input in phases: maximum subsequences of $\leq k$ distinct pages

Example for $k = 3$: $A, B, D, A,$ | $C, E, C, B, E, C, C,$ | $A, B, E,$ | $D, \ldots$

clean / new

Definition (marking algorithms)

Marked pages: previously requested in the current phase.
A *Marking algorithm* **never** evicts *marked pages*.

Marker algorithm: evict an unmarked page uniformly at random

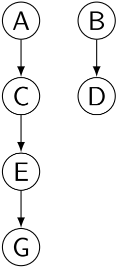
Classic results:
- Marker is $2H_k$-competitive ($O(\log k)$)
- Opt $\geq$ #phases
- marking algorithms $\in [2, k]$-competitive

Predictive Marker [LykourisVassilvitskii'18]

Main idea: use a marking framework to bring more structure

Version 1: MARKER but evict the predicted *unmarked* page

  is only k



Define *eviction chains*: build a graph between the pages:

- when a *stale* (not new) page q evicts a page p, add an edge from p to q

Note: big $\eta \implies$ long chains

Predictive Marker: revert to random unmarked eviction for chains $> H_k$.

Theorem

Predictive marker is $2 + O(\min(\log k\ ,\ \sqrt{\eta/\textsc{Opt}}))$*-competitive.*

Key: ℓ-long chain means ℓ pages predicted in reverse order $\Rightarrow \eta = \Omega(\ell^2)$

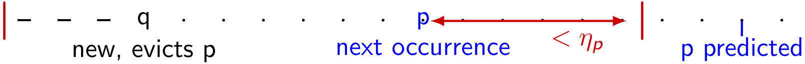
Improvements from [Rohatgi'20]

LMARKER: revert to random unmarked evictions for chains > 1

Theorem

LMARKER is $O(1 + \min(\log k\ ,\ \log \frac{\eta}{\text{OPT}}))$*-competitive.*

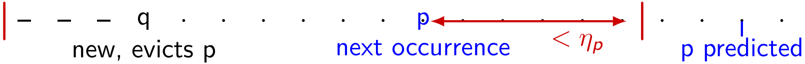
Key: the furthest predicted element is "close" to the end of the phase, so an analysis similar to MARKER with a shorter phase length works



Further improvement from [Rohatgi'20]

LNONMARKER: - use predictions only when new pages are requested
- evict a random page if chain length $= 1$
- otherwise evict a random unmarked page

Motivation (hand wavy) for good predictors :
- 2nd element of a chain is "close" to the end of the phase
- totally random eviction $\rightarrow$ only prob. $< \eta_p / k$ to be wrong in this phase



Theorem

LNONMARKER combined is $O(1 + \min(\log k\ ,\ \frac{\eta}{k \cdot \text{OPT}} \log k))$*-competitive.*

Further improvement from [Rohatgi'20]

LNONMARKER: - use predictions only when new pages are requested
- evict a random page if chain length $= 1$
- otherwise evict a random unmarked page

Motivation (hand wavy) for good predictors :
- 2nd element of a chain is "close" to the end of the phase
- totally random eviction $\rightarrow$ only prob. $< \eta_p / k$ to be wrong in this phase

| – – – q . . . . . . p ←——→ | . . . . |

new, evicts p — next occurrence — $< \eta_p$ — p predicted

Theorem

LNONMARKER combined is $O(1 + \min(\log k \, , \frac{\eta}{k \cdot \textsc{Opt}} \log k))$*-competitive.*

Theorem (Wei'20)

FTP combined is $O(1 + \min(\log k \, , \frac{\eta}{k \cdot \textsc{Opt}}))$*-competitive.*

Weighted caching

Model: each page p fits in 1 slot but has a cost w_p

☹ previous predictions "useless" :  $= \Omega(\log k)$

Strong Per Request Predictions [JiangPS'20]

- prediction = all requests until the current request is repeated
- Resource augmentation (1 cache slot) $\rightarrow O(\textsc{Opt} + \ell_{ed})$

ℓ_{ed} = edit distance variant

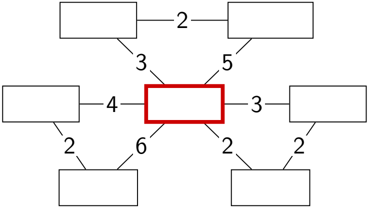
Weight classes parameterization [BansalCKPV'22]

- next arrival time prediction, $\eta <$ weighted norm
- assume pages weight belong to $\{w_1, w_2, \dots, w_\ell\}$
- $O\left(\min\left\{\log \ell + \frac{\ell\eta}{\textsc{Opt}}\ ,\ \log k\right\}\right)$ - competitive

☹ $\rightarrow$ predictions not easily generalizable

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

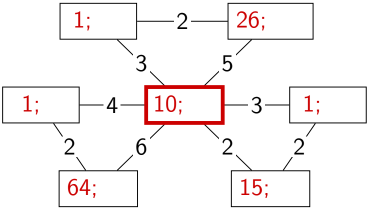
Definition generalizes cachings, *k*-server, convex body chasing. . .



Round 0
Cost incurred: 0

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

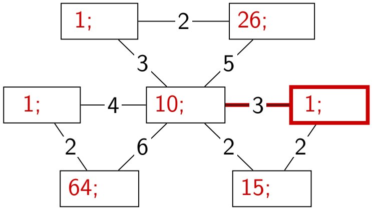
Definition generalizes cachings, k-server, convex body chasing...



Round 1 before serving
Cost incurred: 0

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

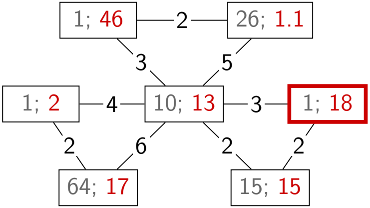
Definition generalizes cachings, *k*-server, convex body chasing. . .



Round 1 after serving
Cost incurred: 3+1

Metrical Task System (MTS) [AntoniadisCEPS'20]

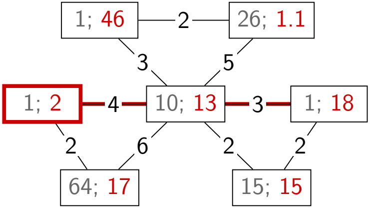
Definition generalizes cachings, k-server, convex body chasing...



Round 2 before serving
Cost incurred: 3+1

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

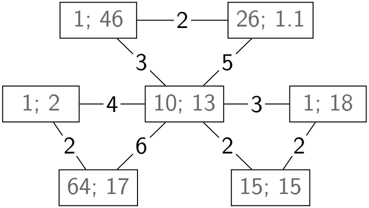
Definition generalizes cachings, *k*-server, convex body chasing...



Round 2 after serving
Cost incurred: 3+1+7+2

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

Definition generalizes cachings, k-server, convex body chasing...



Round 2 after serving
Cost incurred: 3+1+7+2

What should we predict?

- ☹ Next costs ? Useless or too much info
- ☺ Single state per round: where we should be
  - Distance VS Opt? There can be several good options...

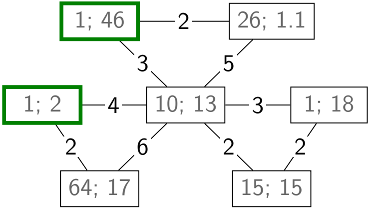
$\forall$ Off, $\eta := \sum_t d(\text{Off}_t, p_t) \longrightarrow \approx$best cost is $\text{Off} \cdot (1 + 4\eta/\text{Off})$

Combination $\rightarrow$ 

Metrical Task System (MTS) [AntoniadisCEP**S**'20]

Definition generalizes cachings, k-server, convex body chasing...



Round 2 after serving
Cost incurred: 3+1+7+2

What should we predict?

- ☹ Next costs ? Useless or too much info
- ☺ Single state per round: where we should be
  - Distance VS OPT? There can be several good options...

$\forall$ OFF, $\eta := \sum_t d(\text{OFF}_t, p_t) \longrightarrow \approx$best cost is OFF $\cdot\, (1 + 4\eta/\text{OFF})$

Combination $\rightarrow$ 

Logarithmic smoothness for caching [AntoniadisCEP**S**'20]

Prediction = cache P of some offline algorithm OFF

New: poor advice for a short time → low η → need other strategies

Algorithm TRUST&DOUBT: sketch

- Phases as MARKER (= k different requests)
- *Clean* page q arrives : "trust" for q – evict some $p_q \notin P$
- A p_q is requested : "doubt" for q – pick another $p_q \notin P$
- Regularly (depending on trustworthiness) : evict p_q, "trust" for q

Theorem (TRUST&DOUBT competitive ratio)

TRUST&DOUBT *costs* $O(\min\{\text{OFF} \cdot (1 + \log \frac{\eta}{\text{OFF}})\ ,\ \text{OPT} \cdot \log k\})$.

Hard to compare guarantees but can convert old prediction into this one

Outline

Conclusion

Take-back messages

-   
- fresh algorithm concepts
- relevant link ML – algorithms

Criticism: ANY OBSCURE PROBLEM $\longrightarrow$ *Learning-Augmented*

Newer questions

- improve running time [DinitzIMLV'21]
- parsimonious predictions [ImKPP'22]
- ensure learnability (*e.g.*, PAC-learnability) / $\eta \approx$ loss function
- extensive experiments including ML predictors [ChłedowskiPSŻ'21]
-  wrt renowned heuristics ?

References I

Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon, Online metric algorithms with untrusted predictions, International Conference on Machine Learning, PMLR, 2020, pp. 345–355.

______, Learning-augmented dynamic power management with multiple states via new ski rental bounds, Advances in Neural Information Processing Systems **34** (2021).

Spyros Angelopoulos, Christoph Dürr, Shendan Jin, Shahin Kamali, and Marc Renault, Online Computation with Untrusted Advice, Proceedings of ITCS'20, vol. 151, 2020, pp. 52:1–52:15.

Nikhil Bansal, Christian Coester, Ravi Kumar, Manish Purohit, and Erik Vee, Learning-augmented weighted paging, Proceedings of the Thirty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'22, 2022.

Jakub Chłędowski, Adam Polak, Bartosz Szabucki, and Konrad Tomasz Żołna, Robust learning-augmented caching: An experimental study, International Conference on Machine Learning, PMLR, 2021, pp. 1920–1930.

Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii, Faster matchings via learned duals, Advances in Neural Information Processing Systems **34** (2021).

References II

Sungjin Im, Ravi Kumar, Aditya Petety, and Manish Purohit, Parsimonious learning-augmented caching, arXiv preprint arXiv:2202.04262 (2022).

Zhihao Jiang, Debmalya Panigrahi, and Kevin Su, Online algorithms for weighted paging with predictions, 47th International Colloquium on Automata, Languages, and Programming (ICALP 2020), 2020.

Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis, The case for learned index structures, Proceedings of SIGMOD'18, 2018, pp. 489–504.

Thodoris Lykouris and Sergei Vassilvitskii, Competitive caching with machine learned advice, Proceedings of ICML'18, 2018, pp. 3302–3311.

Manish Purohit, Zoya Svitkina, and Ravi Kumar, Improving online algorithms via ML predictions, Proceedings of NeurIPS'18, 2018, pp. 9684–9693.

Dhruv Rohatgi, Near-optimal bounds for online caching with machine learned advice, Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'20, 2020, pp. 1834–1845.

Alexander Wei, Better and simpler learning-augmented online caching, Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020), 2020.

References III

Alexander Wei and Fred Zhang, Optimal robustness-consistency trade-offs for learning-augmented online algorithms, Advances in Neural Information Processing Systems **33** (2020), 8042–8053.