Learning-Augmented Online Algorithms & Paging

Bertrand Simon – CNRS / CC-IN2P3

CoA Workshop, September 2023
Based on work with Antonios Antoniadis, Joan Boyar, Marek Eliáš, Lene M. Favrholdt, Ruben Hoeksma, Kim S. Larsen, Adam Polak.

several slides inspired from J. Boyar

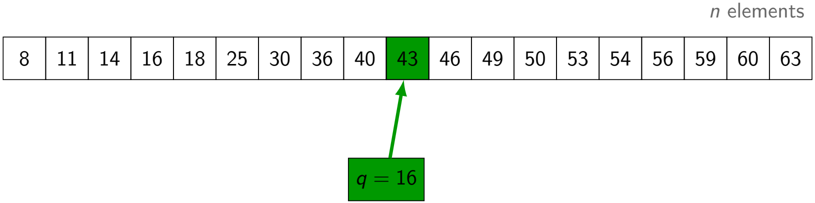
Motivating example: binary search

n elements

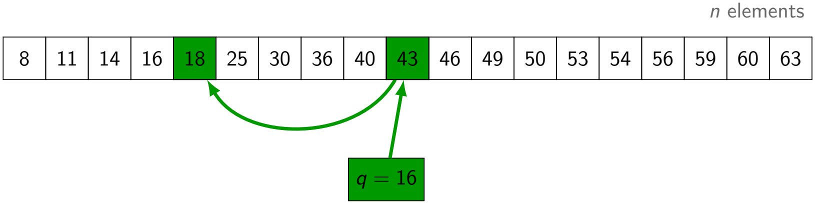
8	11	14	16	18	25	30	36	40	43	46	49	50	53	54	56	59	60	63

$q = 16$

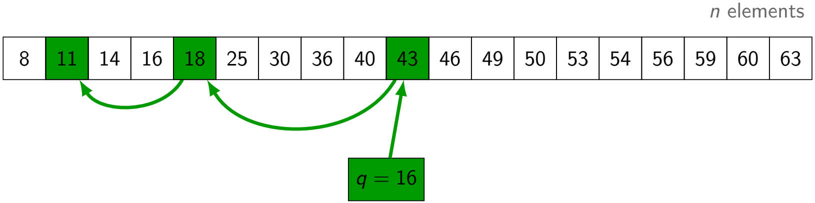
Motivating example: binary search



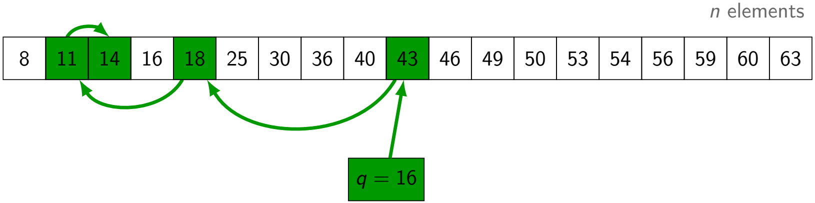
Motivating example: binary search



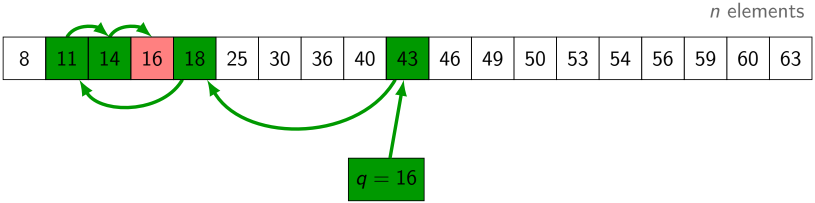
Motivating example: binary search



Motivating example: binary search



Motivating example: binary search



Motivating example: binary search

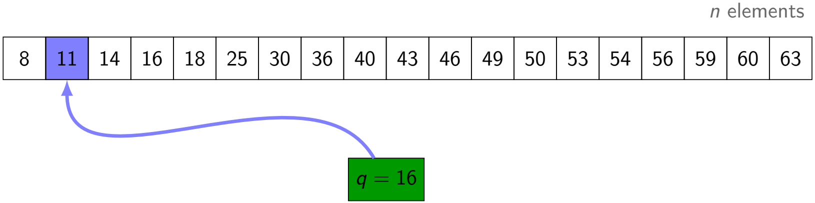
n elements

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

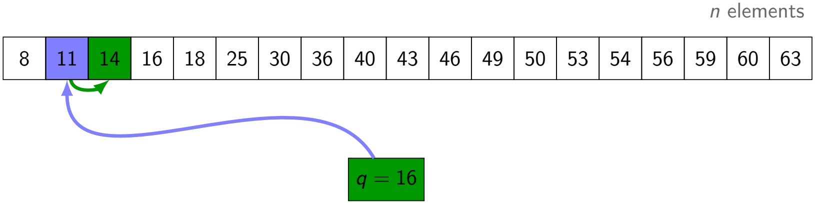
Motivating example: binary search



Prediction: position $h(q)$

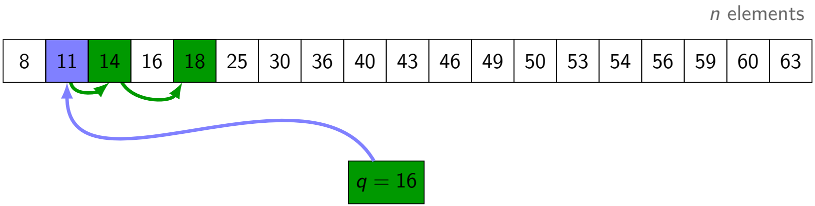
Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



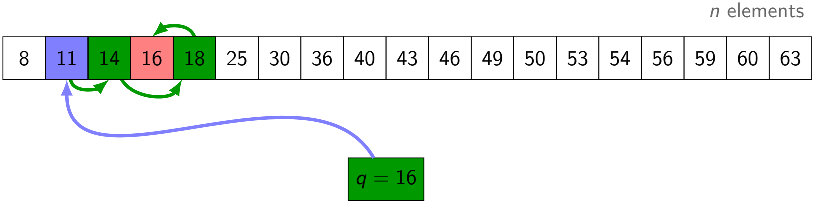
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



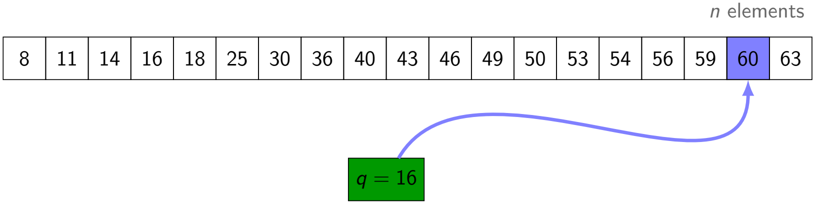
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



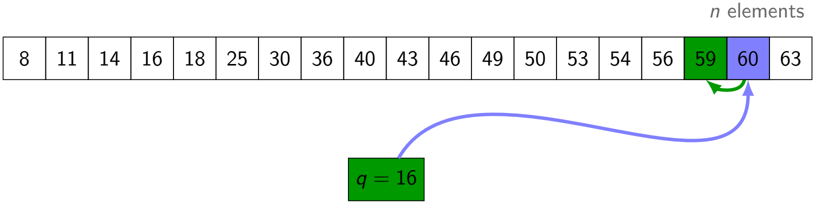
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



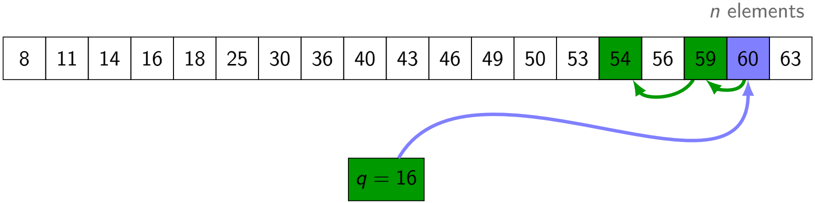
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



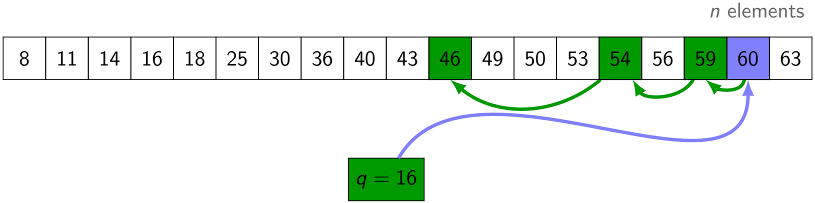
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



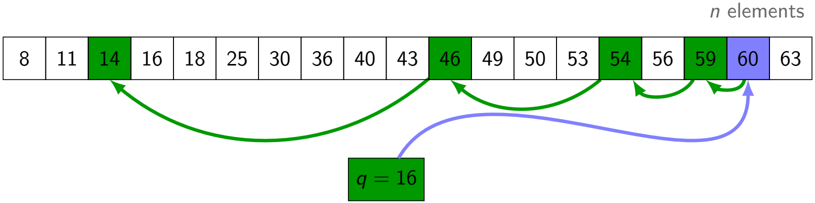
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



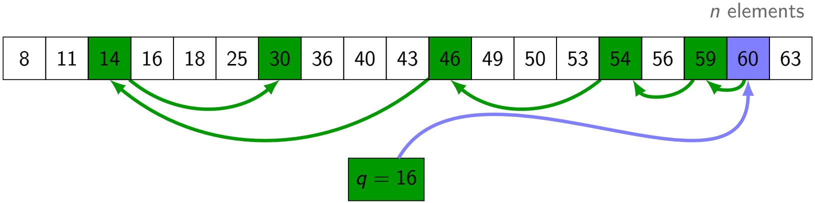
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



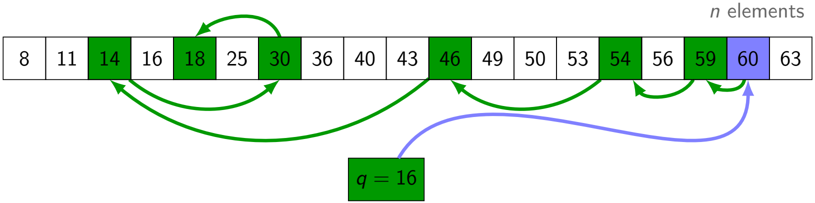
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



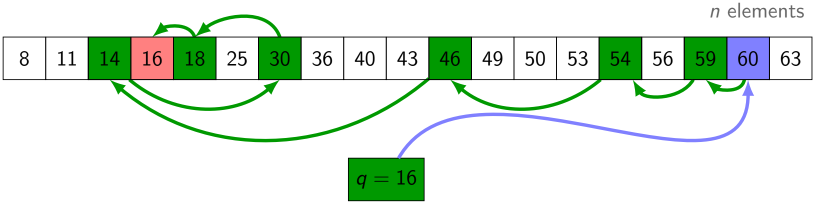
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



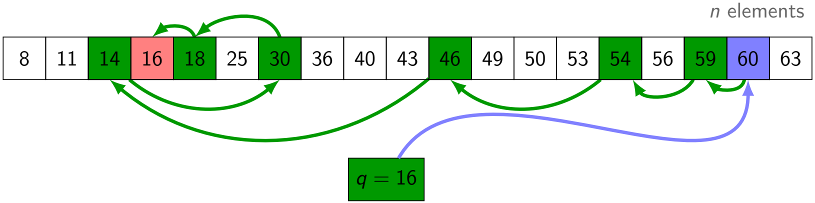
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

Motivating example: binary search



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

Motivating example: binary search

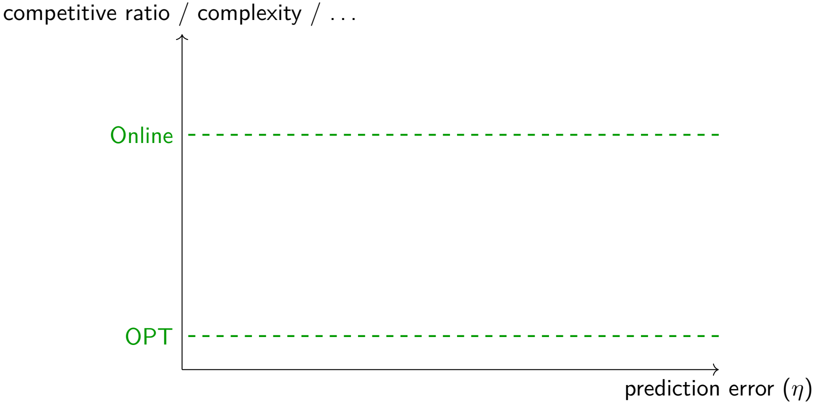


Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

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

Practical applications [KraskaBCDP '18]

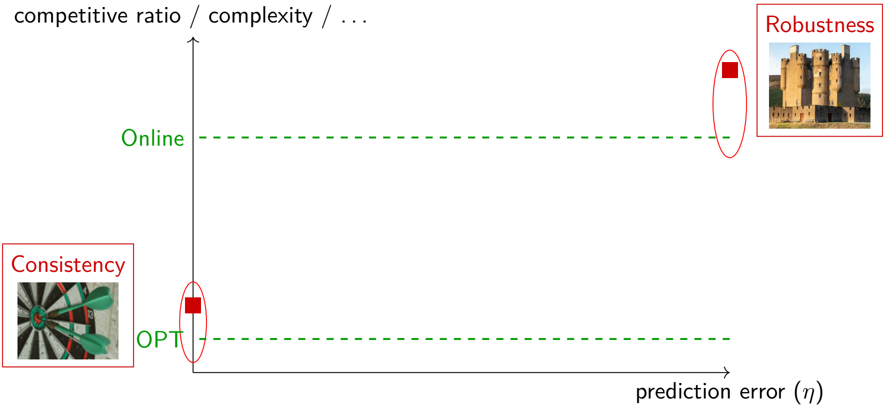
Properties we seek



Algorithms are oblivious to η

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

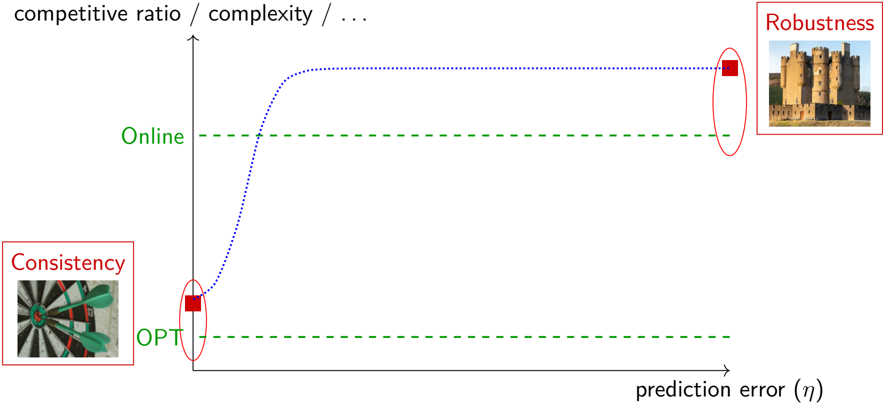
Properties we seek



Algorithms are oblivious to η

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

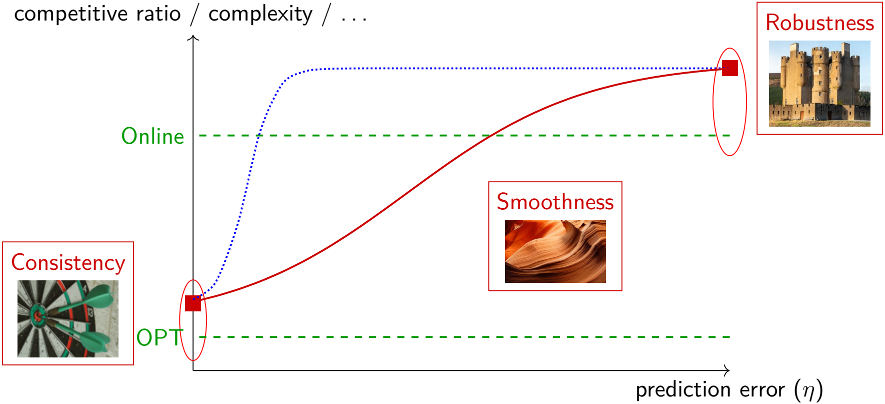
Properties we seek



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

“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**

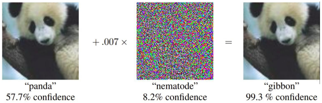
- **Advice**: 1101110
- **Stochastic input**: $X_i \sim \mathcal{N}(10, 5)$
- **Robust analysis**: $X_1 = 5 \pm 2$, $X_2 = 7 \pm 3$, . . .
- . . .

☹ Strong assumptions, needs some perfect information (oracle)

HERE: no assumption on the predictor

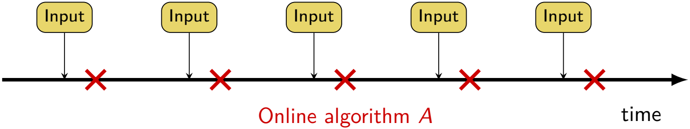
allows plug-and-play predictors

$\Longrightarrow$

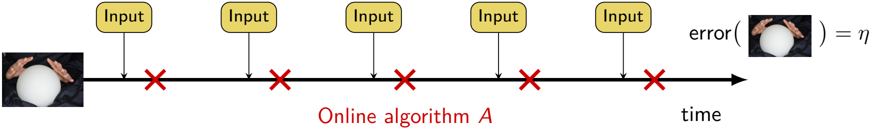


arxiv.org/abs/1412.6572

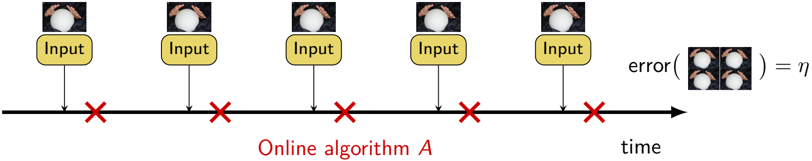
Most common framework used



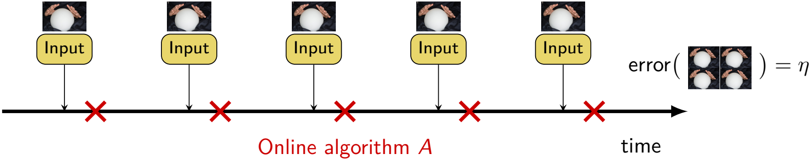
Most common framework used



Most common framework used



Most common framework used



Objective: "minimize" competitive ratio $c_A(\eta)$ (may need OPT to scale)

Consistency

$c_A(0)$

Robustness

$c_A(\infty)$

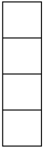
Smoothness

"slope" of $c_A(\eta)$

`https://algorithms-with-predictions.github.io`

$k = 4$ misses: 1

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



1

A

$k = 4$ misses: 2

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

A

1	2
A	B

$k = 4$ misses: 2

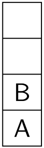
pages $\in \{A, B, \ldots, F\}$

B	
A	

1	2	3
A	B	A

$k = 4$ misses: 3

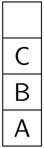
pages $\in \{A, B, \ldots, F\}$



1	2	3	4
A	B	A	C

$k = 4$ misses: 4

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



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

$k = 4$ 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

$k = 4$ 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

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

$k = 4$ 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

$k = 4$ 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

$k = 4$ 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 $\in \{A, B, \ldots, F\}$

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

$k = 4$ 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 $\eta = 0 \to$ 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 ; $\textsc{Opt} = \#A$

FTP’s competitive ratio is at least $\Omega(\eta / \textsc{Opt})$ 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 $k = 3$: $A, B, D, A,$ | $C, E, C, B, E, C, C,$ | $A, B, E,$ | $D, \ldots$

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, OPT $\geq \frac{1}{2}$#clean pages
- marking algorithms $\in [2, k]$-competitive

Classic online solution: MARKER

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

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

clean / new

Definition (marking algorithms)

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

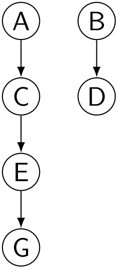
MARKER algorithm: evict an unmarked page uniformly at random

Classic results:
- MARKER is $2H_k$-competitive ($O(\log k)$)
- OPT $\geq$ #phases, OPT $\geq \frac{1}{2}$#clean pages
- marking algorithms $\in [2, k]$-competitive

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 $\Rightarrow \eta = \Omega(\ell^2)$

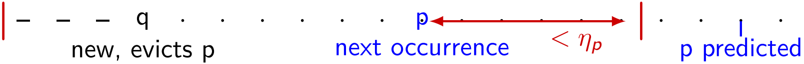
Improvements from [Rohatgi'20]

LMARKER: revert to random unmarked evictions for chains > 1

Theorem

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

Key: the furthest predicted element is "close" to the end of the phase, so an analysis similar to $\textsc{Marker}$ 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

| – – – q . . . . . . p ←——— $< \eta_p$ ———→ | . . . .

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

Theorem

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

Theorem (Wei'20)

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

Paging with predictions – Overview

Predictions = time of next occurrence of current page

- Lykouris,Vassilvitskii, 2018 (2021 JACM)
- Rohatgi, SODA 2020
- Wei, APPROX/RANDOM 2020

Paging with predictions – Overview

Predictions = time of next occurrence of current page

- Lykouris,Vassilvitskii, 2018 (2021 JACM)
- Rohatgi, SODA 2020
- Wei, APPROX/RANDOM 2020

Predictions = all pages before next occurrence of current page

- Jiang Panigrahi Su, ICALP 2020

Paging with predictions – Overview

Predictions = time of next occurrence of current page

- Lykouris,Vassilvitskii, 2018 (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 $\textsc{Opt}$ (which pages in cache)

- Antoniadis Coester Elias Polak Simon, (ICML 2020)

Paging with predictions – Overview

Predictions = time of next occurrence of current page

- Lykouris,Vassilvitskii, 2018 (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)

Paging with predictions – Overview

Predictions = time of next occurrence of current page

- Lykouris,Vassilvitskii, 2018 (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 $\textsc{Opt}$ (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)
  CR $= O(\min\{log_{b+1} n + E[\eta]/\textsc{Opt}, \log k\})$, $b =$ number of queries

Paging with succinct predictions [ABEFHLP**S**, ICML'23]

Question: Can we do this with succinct predictions?

Question: Can we do this with succinct predictions?

Next request to a page is a lot of information.

- Is it too hard to obtain?
- Does it make it too easy to get a good competitive ratio, based on η.

Paging with succinct predictions [ABEFHLP**S**, ICML'23]

Question: Can we do this with succinct predictions?

Next request to a page is a lot of information.

- Is it too hard to obtain?
- Does it make it too easy to get a good competitive ratio, based on η.

Advice complexity says:

Theorem (Mikkelsen, 2016)

Even with correct advice, a linear number of bits are necessary to be better than H_k-competitive

Succinct predictions

Predictions: 1 bit per request

Discard predictions — same as for advice complexity

$$b_i = \begin{cases} 0 & \text{if } \textsc{Opt} \text{ would have } r_i \text{ in cache next time it is requested} \\ 1 & \text{otherwise} \end{cases}$$

Phase predictions — based on max. sequences with $\leq k$ distinct pages

$$b_i = \begin{cases} 0 & \text{if } r_i \text{ is in the next phase} \\ 1 & \text{otherwise} \end{cases}$$

Both cases: 0-predictions = should stay in cache.

Discard predictions — deterministic

Obvious deterministic algorithm (OBVIOUS)

- On a fault, evict a page with a 1-prediction, if there is one. (OPT should not have it in cache next time.)
- Otherwise, evict any page.

All predictions correct $\Longrightarrow$ **OBVIOUS** keeps same pages as OPT

Observation: OBVIOUS is 1-consistent 

Discard predictions — deterministic algorithms

Obvious deterministic algorithm (OBVIOUS)

- On a fault, evict a page with a 1-prediction, if there is one. (OPT should not have it in cache next time.)
- Otherwise, evict any page.

Suppose for r_i, the prediction for p is 1, but the correct prediction is 0
OPT would keep it in cache.
When p is requested again, there is one fault.

Discard predictions — deterministic algorithms

Obvious deterministic algorithm (OBVIOUS)

- On a fault, evict a page with a 1-prediction, if there is one. (OPT should not have it in cache next time.)
- Otherwise, evict any page.

Suppose for r_i, the prediction for p is 1, but the correct prediction is 0
OPT would keep it in cache.
When p is requested again, there is one fault.

Suppose for r_i, the prediction is 0, but the correct prediction is 1
Problem: Cache may have no 1-predictions.
Could evict sequence in the opposite of the correct order (like **LRU**), so OPT faults once and **OBVIOUS** faults k times.

Discard predictions — deterministic algorithms

Obvious deterministic algorithm (OBVIOUS)

- On a fault, evict a page with a 1-prediction, if there is one. (OPT should not have it in cache next time.)
- Otherwise, evict any page.

Suppose for r_i, the prediction for p is 1, but the correct prediction is 0
OPT would keep it in cache.
When p is requested again, there is one fault.

Suppose for r_i, the prediction is 0, but the correct prediction is 1
Problem: Cache may have no 1-predictions.
Could evict sequence in the opposite of the correct order (like **LRU**), so OPT faults once and **OBVIOUS** faults k times.

Observation: False 0-predictions are much worse than false 1-predictions.

Notation

η_0: Number of incorrect 0-predictions.
η_1: Number of incorrect 1-predictions.

Notation

η_0: Number of incorrect 0-predictions.
η_1: Number of incorrect 1-predictions.

$\mathbb{A}$ is (α, β, γ)-competitive if for any input seq. I, $\exists b$

$$ALG(I) \leq \alpha \cdot \textsc{Opt}(I) + \beta \cdot \eta_0 + \gamma \cdot \eta_1 + b.$$

↖ constant

Discard predictions — deterministic

Modify **OBVIOUS — Flush-When-All-0s**

- On a fault, evict a page with a 1-prediction, if there is one. (OPT will not have it in cache next time.)
- Otherwise, flush the cache.

Theorem

Flush-When-All-0s *is* $(1, k-1, 1)$*-competitive.*

Corollary

Flush-When-All-0s *is* 1*-consistent* 

Discard predictions — Flush-When-All-0s

Theorem

Flush-When-All-0s *is* $(1, k-1, 1)$*-competitive.*

Between 2 flushes:

- $\textsc{Opt}$ evicts $\geq$one 0-predicted page
- **Flush-When-All-0s** evicts k 0-predicted pages

So:

- On 0-pages: $\text{Flush-When-All-0s}_0 \leq \textsc{Opt}_0 + (k-1)\eta_0$
- On 1-pages: $\text{Flush-When-All-0s}_1 \leq \textsc{Opt}_1 + \eta_1$

Flush-When-All-0s $\leq \textsc{Opt} + (k-1)\eta_0 + \eta_1$

Discard predictions — Flush-When-All-0s

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive.*

Between 2 flushes:

- $\textsc{Opt}$ evicts $\geq$one 0-predicted page
- **Flush-When-All-0s** evicts k 0-predicted pages

So:

- On 0-pages: $\text{Flush-When-All-0s}_0 \leq \alpha \, \textsc{Opt}_0 + (k - \alpha)\eta_0$
- On 1-pages: $\text{Flush-When-All-0s}_1 \leq \textsc{Opt}_1 + \eta_1$

Flush-When-All-0s $\leq \alpha \, \textsc{Opt} + (k - \alpha)\eta_0 + \eta_1$

Discard predictions — Deterministic lower bound

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive.*

Theorem

For discard-predictions, for a deterministic (α, β, γ)-competitive algorithm $\textsc{Alg}$, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$.

Discard predictions — Deterministic lower bound

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive.*

Theorem

For discard-predictions, for a deterministic (α, β, γ)-competitive algorithm $\textbf{Alg}$, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$.

Proof Use $k + 1$ pages and the cruel adversary against $\textsc{Alg}$.
(adversary always gives the page not in $\textsc{Alg}$'s cache)

Discard predictions — Deterministic lower bound

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive.*

Theorem

For discard-predictions, for a deterministic (α, β, γ)-competitive algorithm $\textbf{ALG}$, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$.

Proof Use $k + 1$ pages and the cruel adversary against $\textsc{Alg}$.
(adversary always gives the page not in $\textsc{Alg}$'s cache)

$\textsc{Alg} = n$ $\qquad\qquad$ $\textsc{Opt} \leq \frac{n}{k}$
Write $\textsc{Alg} \leq \alpha\,\textsc{Opt} + \beta\eta_0 + \gamma\eta_1$.

Discard predictions — Deterministic lower bound

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive.*

Theorem

For discard-predictions, for a deterministic (α, β, γ)-competitive algorithm $\textsc{Alg}$, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$.

Proof Use $k+1$ pages and the cruel adversary against $\textsc{Alg}$.
(adversary always gives the page not in $\textsc{Alg}$'s cache)

$\textsc{Alg} = n$ $\textsc{Opt} \leq \frac{n}{k}$
Write $\textsc{Alg} \leq \alpha\,\textsc{Opt} + \beta\eta_0 + \gamma\eta_1$.

Case predictions all zeros: $\eta_0 \leq \textsc{Opt}$

$$n = \textsc{Alg} \leq \alpha \cdot \left(\frac{n}{k}\right) + \beta \cdot \left(\frac{n}{k}\right)$$

So: $\alpha + \beta \geq k$.

Discard predictions — Deterministic lower bound

Theorem

*For $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k-\alpha, 1)$-competitive.*

Theorem

For discard-predictions, for a deterministic (α, β, γ)-competitive algorithm $\textsc{Alg}$, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$.

Proof Use $k+1$ pages and the cruel adversary against $\textsc{Alg}$.
(adversary always gives the page not in $\textsc{Alg}$'s cache)

$\textsc{Alg} = n$ $\qquad\qquad$ $\textsc{Opt} \leq \frac{n}{k}$
Write $\textsc{Alg} \leq \alpha \textsc{Opt} + \beta\eta_0 + \gamma\eta_1$.

Case predictions all ones: $\eta_1 \leq n - \textsc{Opt}$

$$n = \textsc{Alg} \leq \alpha \cdot \left(\frac{n}{k}\right) + \gamma\left(n - \frac{n}{k}\right)$$

So: $\alpha + (k-1)\gamma \geq k$.

Discard predictions — Randomized

Algorithm **Randomized Eagerly Evict**:
Uses ideas from marking algorithms.

Discard predictions — Randomized

Algorithm **Randomized Eagerly Evict**:
Uses ideas from marking algorithms.

- runs in phases, marking requested pages
- evicts all pages with prediction 1 immediately
- among pages with prediction 0, randomly evicts unmarked pages

Discard predictions — Randomized

Algorithm **Randomized Eagerly Evict**:
Uses ideas from marking algorithms.

- runs in phases, marking requested pages
- evicts all pages with prediction 1 immediately
- among pages with prediction 0, randomly evicts unmarked pages

Theorem

Algorithm ***Randomized Eagerly Evict*** *is* $(1, 2H_i, 1)$*-competitive.*

Corollary

Algorithm ***Randomized Eagerly Evict*** *is 1-consistent* 

$\approx$ corresponding lower bounds $\Longrightarrow$ results are quite tight

Phase predictions — Randomized

Algorithm **MARK & PREDICT**:
Follows MARKER closely.
Major difference: It prefers to evict pages with prediction 1.

Algorithm **MARK & PREDICT**:
Follows MARKER closely.
Major difference: It prefers to evict pages with prediction 1.

Algorithm MARKER

For $i = 1$ **to** n
If r_i is not in cache
    If all pages in cache are marked   { end phase }
        unmark all pages
    evict a random unmarked page
    bring r_i into cache
mark r_i

Phase predictions — Randomized

Algorithm **MARK & PREDICT**:
Follows MARKER closely.
Major difference: It prefers to evict pages with prediction 1.

Algorithm MARK & PREDICT

For $i = 1$ **to** n
If r_i is not in cache
  If all pages in cache are marked { end phase }
    unmark all pages
  If there is an unmarked 1-page
    evict a random unmarked 1-page
  Else
    evict a random unmarked 0-page
  bring r_i into cache
mark r_i

Phase predictions — Randomized

Theorem

*Algorithm **MARK & PREDICT** is $(2, H_k, 1)$-competitive. (Also holds if 1-pages are evicted deterministically.)*

Phase predictions — Randomized

Theorem

Algorithm ***MARK & PREDICT*** *is* $(2, H_k, 1)$*-competitive. (Also holds if* 1*-pages are evicted deterministically.)*

Theorem

Algorithm ***MARK & PREDICT*** *is* $(2, H_k, \frac{2\,\textsc{Opt}}{\eta_1}(\ln(2\frac{\eta_1}{\textsc{Opt}} + 1) + 1))$*-competitive.*

Phase predictions — Randomized

Theorem

*Algorithm **MARK & PREDICT** is* $(2, H_k, 1)$*-competitive. (Also holds if 1-pages are evicted deterministically.)*

Theorem

*Algorithm **MARK & PREDICT** is* $(2, H_k, \frac{2\,\textsc{Opt}}{\eta_1}(\ln(2\frac{\eta_1}{\textsc{Opt}} + 1) + 1))$*-competitive.*

Corollary

*Algorithm **MARK & PREDICT** is 2-consistent* 

$\approx$ corresponding lower bounds $\Longrightarrow$ results are quite tight

Conclusions

Learning-augmented algorithms

Paging with succinct predictions

- succinct predictions may be easier to obtain
- succinct predictions $\Longrightarrow$ similar guarantees

Future of Learning-Augmented algorithms

- "pick a new online problem and add predictions" done 100s of time
- new paradigms: multiple predictors, prediction scarcity, stochastic predictions, practical benchmark, new objective functions...
- ad: topic of the newly funded ANR project PREDICTIONS