# Paging with succinct predictions

Bertrand Simon – CNRS / CC-IN2P3

ROADEF, March 2024

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

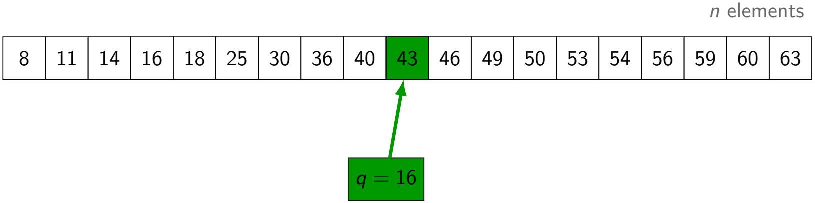
# Motivating example: binary search

*n* elements

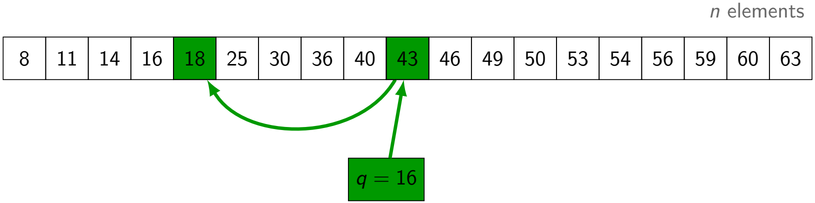
| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 11 | 14 | 16 | 18 | 25 | 30 | 36 | 40 | 43 | 46 | 49 | 50 | 53 | 54 | 56 | 59 | 60 | 63 |

$q = 16$

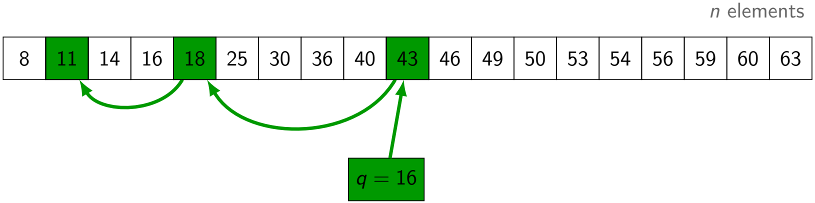
# Motivating example: binary search

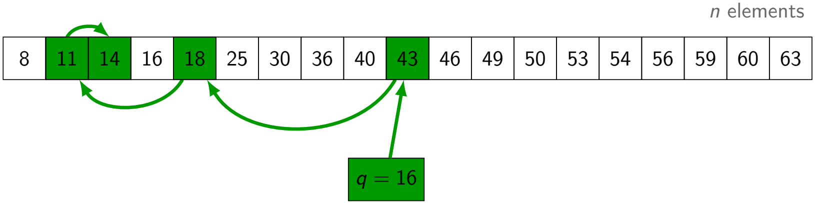


# Motivating example: binary search



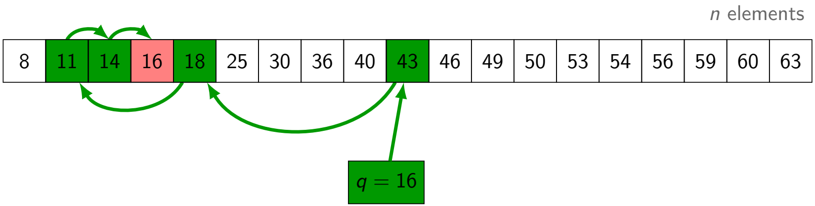
# Motivating example: binary search





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

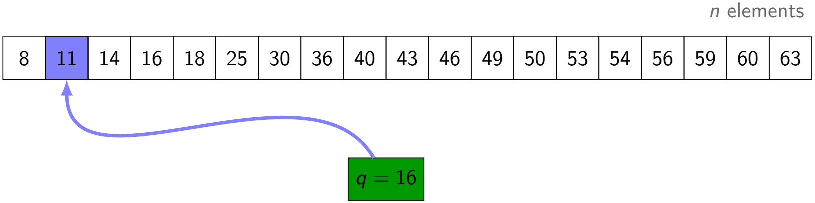
*n* elements

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

$q = 16$

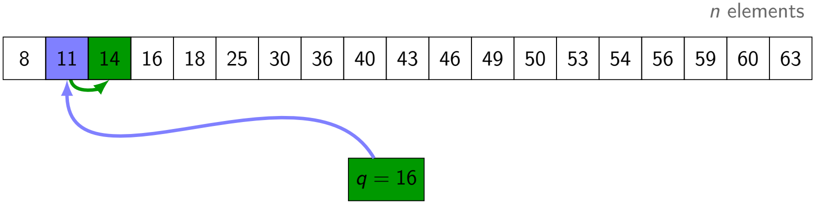
Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



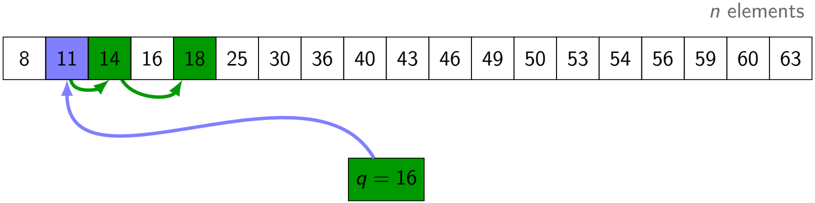

Prediction: position $h(q)$    Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



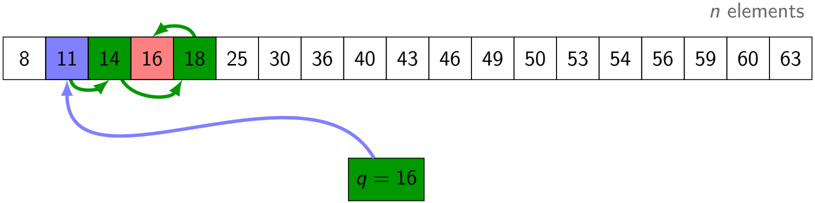

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



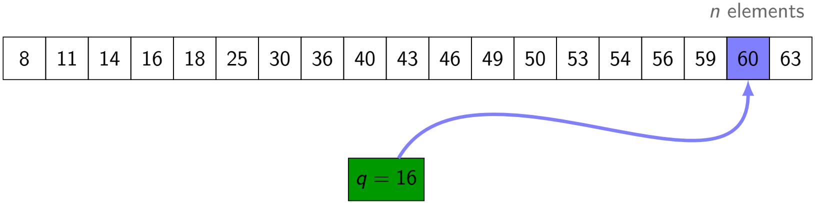

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



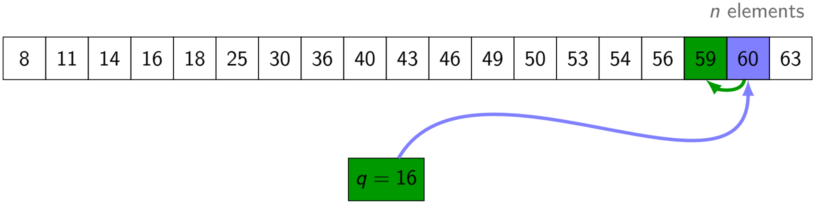

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



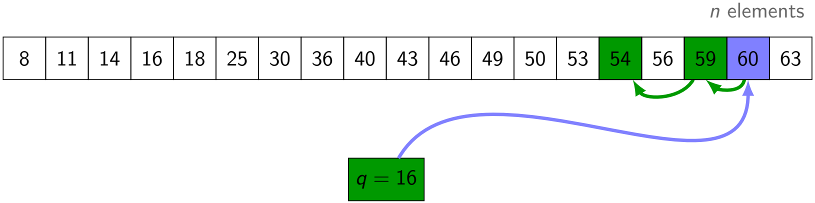

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



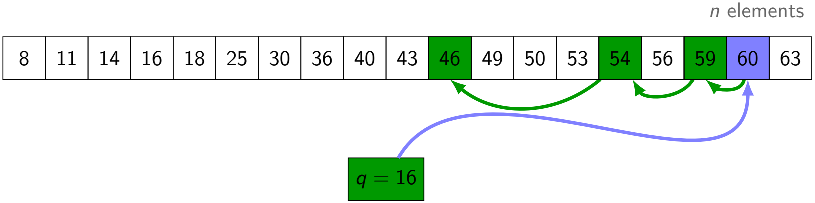

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



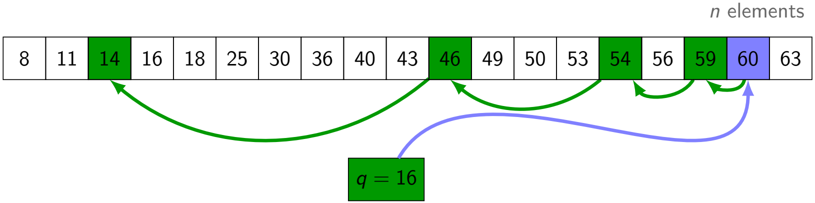

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



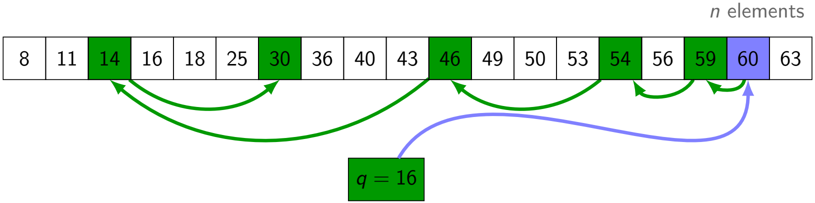

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



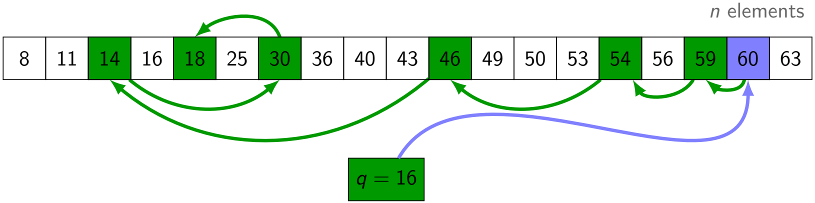

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



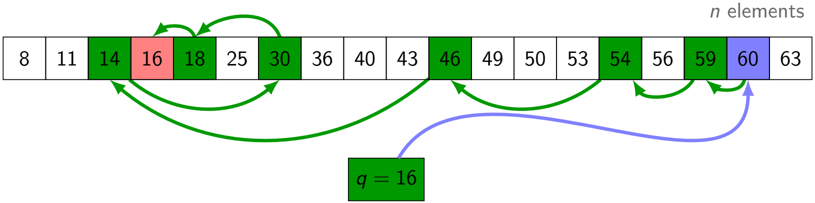

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search



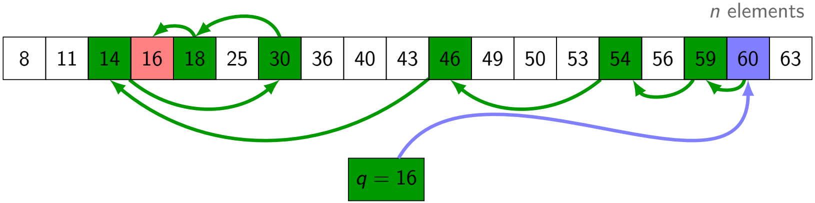

Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search

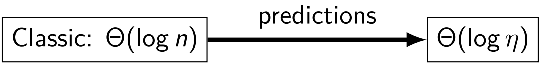



Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$

# Motivating example: binary search

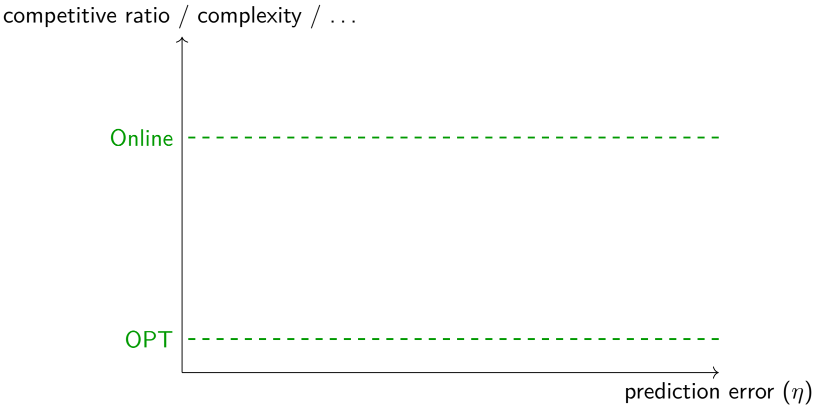



Prediction: position $h(q)$ Error: $\eta = |h(q) - index(q)|$




Practical applications [KraskaBCDP '18]

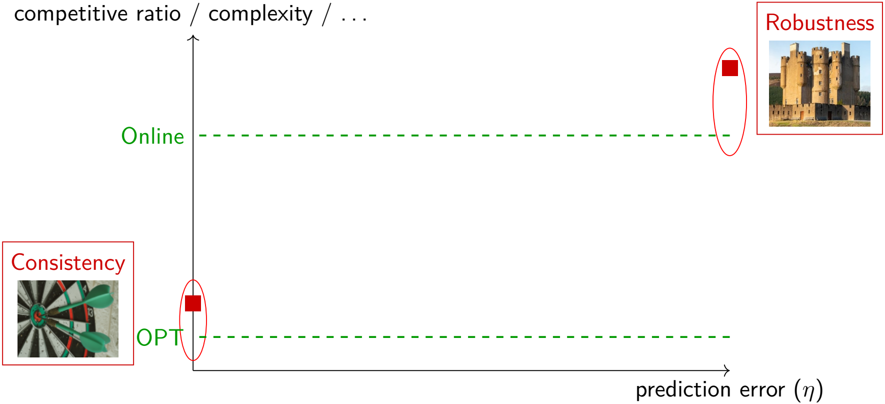
# Properties we seek




Algorithms are oblivious to $\eta$

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

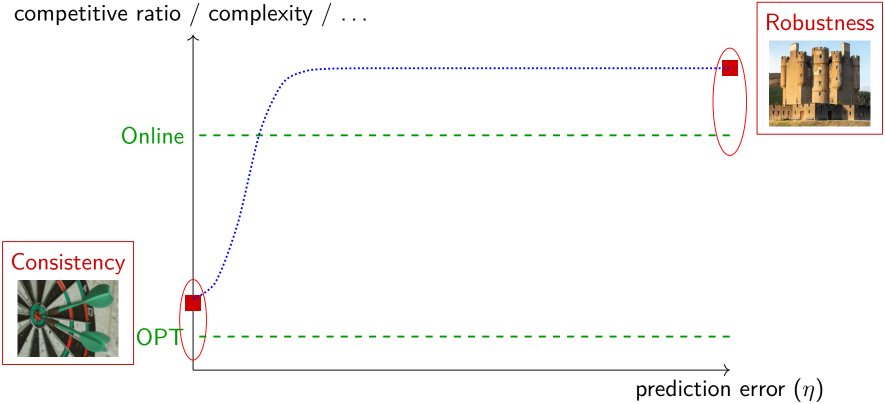
# Properties we seek




Algorithms are oblivious to $\eta$

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

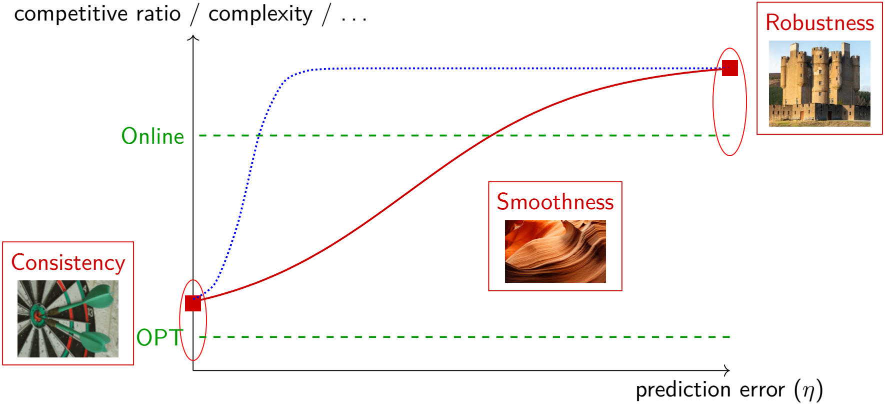
# Properties we seek




Algorithms are oblivious to $\eta$

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

# Properties we seek

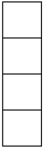



Algorithms are oblivious to $\eta$

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

$k = 4$ misses: 1

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



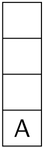
1

A

# Caching with predictions [LykourisVassilvitskii'18]

$k = 4$ misses: 2

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




| 1 | 2 |
|---|---|
| A | B |

$k = 4$ misses: 2

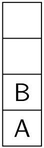
pages $\in \{A, B, \ldots, F\}$

Cache: 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 $\in \{A, B, \ldots, F\}$

| |
|---|
| |
| 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 |

$k = 4$ 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 |

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

| 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 $\eta_i$ at round $i$ : distance between predicted time and actual time combined error $\eta = \sum \eta_i$.

$k = 4$ misses: 8

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

| | | | |
|---|---|---|---|
| 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 |
| 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 $\eta_i$ at round $i$ : distance between predicted time and actual time
  combined error $\eta = \sum \eta_i$.

# 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 ($O(\log k)$)
- marking algorithms $\in [2, k]$-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 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 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

Question: Can we do this with succinct predictions?

# 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 $\eta$.

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 $\eta$.

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 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

$\eta_0$: Number of incorrect 0-predictions.
$\eta_1$: Number of incorrect 1-predictions.

# Notation

$\eta_0$: Number of incorrect 0-predictions.
$\eta_1$: Number of incorrect 1-predictions.

$\mathbb{A}$ is $(\alpha, \beta, \gamma)$-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**

*For any $\alpha \geq 1$, **Flush-When-All-0s** is $(\alpha, k - \alpha, 1)$-competitive, this is "best possible".*
*(Lower bound: for any $(\alpha, \beta, \gamma)$-competitive algorithm ALG, $\alpha + \beta \geq k$ and $\alpha + (k-1)\gamma \geq k$)*

**Corollary**

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

# 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

**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