Paging and Packing with Possibly Precise Predictions

Lene Favrholdt
University of Southern Denmark

MAPSP 22

Paging and Packing with Possibly Precise Predictions

Online algorithms with advice (oracle)
Online algorithms with predictions (e.g. ML)

Upper & lower bound techniques

Running example: Paging
Vertex Cover, Knapsack, Bin Packing

Running example :

Paging

Cache with room for k pages

Input : Sequence of page requests

Requested page, p, not in cache : fault

→ Evict a page from the cache to make room for p

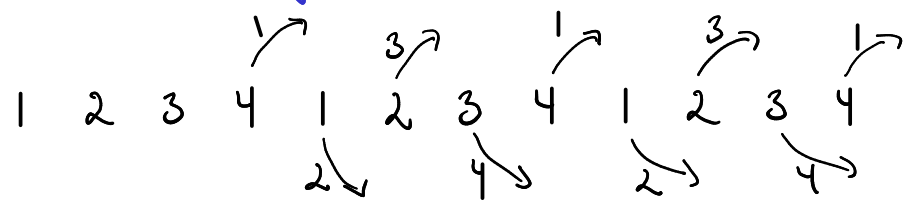
Objective : Minimize #faults

Online problem :

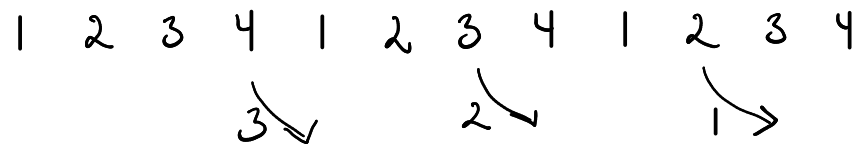
$I = r_1, r_2, r_3, \ldots, r_n$ → Online algorithm → $Y = y_1, y_2, y_3, \ldots, y_n$

Ex: $k = 3$

Least Recently Used (LRU)



Longest Forward Distance (LFD) (optimal offline alg.)



$$\frac{LRU}{LFD} = 3 = k$$

Competitive Ratio

If $\exists b : \forall I : \; ALG(I) \leq c \cdot OPT + b$
then ALG is c-competitive
The competitive ratio of ALG is
$CR(ALG) = \inf \{ c \mid ALG \text{ is } c\text{-competitive} \}$

Game between online player and malicious adversary

Adversary $\xrightarrow{I = r_1, r_2, r_3, \ldots, r_n}$ Online algorithm $\xrightarrow{Y = y_1, y_2, y_3, \ldots, y_n}$

Competitive Ratio

If $\exists b : \forall I : \; ALG(I) \leq c \cdot OPT + b$
then ALG is c-competitive

The competitive ratio of ALG is
$CR(ALG) = \inf \{ c \mid ALG \text{ is } c\text{-competitive} \}$

[ST85]:

For any deterministic paging alg. ALG, $CR(ALG) \geq k$

Proof:
Adversary Strategy:
  Use k+1 pages
  Always req. page ∉ ALG's cache

ALG faults on each req.

LFD faults on ≤ every k^{th} req.

Thus, $\frac{ALG}{LFD} \geq k$ □

[ST85]:
Sleator, Tarjan
Amortized Efficiency of List Update and Paging Rules
CACM 85

Competitive Ratio

If $\exists b : \forall I : E[ALG(I)] \leq c \cdot OPT + b$
then ALG is c-competitive

The competitive ratio of ALG is
$CR(ALG) = \inf \{ c \mid ALG \text{ is } c\text{-competitive} \}$

[FKLMSY 91]:

For any randomized paging alg. ALG, $CR(ALG) \geq H_k \approx \ln k$

[FKLMSY 91]:
Fiat, Karp, Luby, McGeoch, Sleator, Young
Competitive Paging Algorithms
J. Algorithms 91

[FKLMSY91]:

For any randomized paging alg. ALG, $CR(ALG) \geq H_k \approx \ln k$

Proof: (By Yao's Principle)

Adversary strategy:

Use $k+1$ pages

Choose page to be requested uniformly at random

ALG faults with prob. $\frac{1}{k+1}$

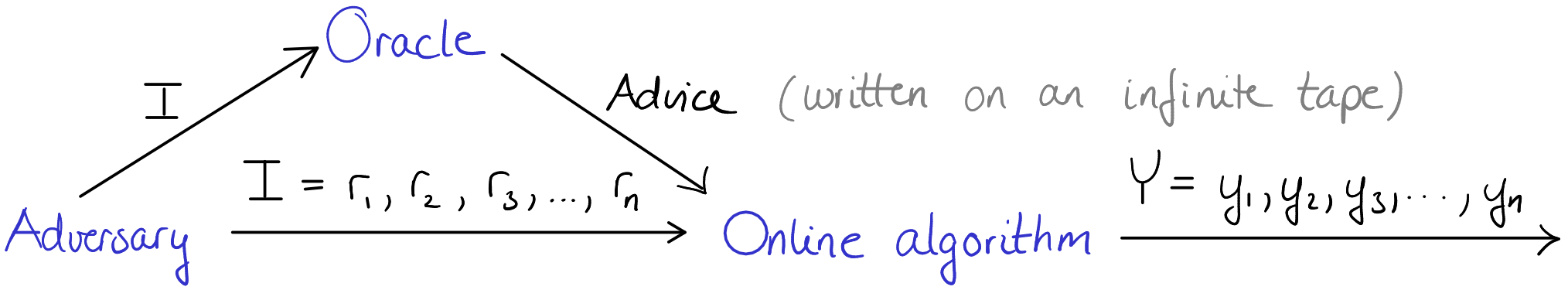
LFD faults once for each k distinct pages $\Rightarrow$

In expectation, LFD faults once for each $(k+1)H_{k+1} - 1 = (k+1)H_k$

Coupon Collector: $\underbrace{1\ 2\ 3\ 4 \cdots k}_{(k+1)H_{k+1}}\ \underbrace{k+1\ 1'\ 2'\ 3' \cdots k'}_{(k+1)H_{k+1}}$ □

$$\left(H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k} \approx \ln k\right)$$

Online Algorithms with Advice



The advice complexity, $b_A(n)$, of an algorithm, A, is the max #advice bits read by A on any I of length n

The advice complexity, $b_P(n,c)$, of a problem, P, is the #advice bits necessary and sufficient to be c-competitive (as a function of input length), i.e.,

$$b_P(n,c) = \inf_{\text{c-comp. } A} \left\{ \max_{|I|=n} \{ b_A(n) \} \right\}$$

Advice Complexity

How much extra information does it take to
- be (offline) optimal ?
- beat best online algorithm ?

Algorithms may be stepping stones to
- semi-online algorithms
- parallel solutions
- algorithms with predictions

Lower bounds give impossibility results for
- semi-online algorithms
- parallel solutions
- randomized online algorithms (sometimes)

CR = 1:

First (trivial) idea: Encode OPT

For each fault, state the page that OPT would evict.
$\Rightarrow$ $b(n,1) \leq n \log k$

Next idea: Now-or-later [DKP 09]

For each request, indicate whether the page requested would stay in OPT's cache until its next request.
$\Rightarrow$ $b(n,1) \leq n$

Ex: $k = 3$

I = 1 2 3 4 2 5 1 2 3 5

ϕ = 0 0 1 1 0 0 ...

1: safe to evict
0: stay in cache

Note: $OPT = |\phi|_1$

[DKP 09]: Dobrev, Královič, Pardubská
Measuring the problem relevant information in input
RAIRO – Theoretical Informatics and Applications 2009

CR > 1 :

First idea: Use advice only for „easy" sequences [BKKKM 17]

OBS: For any alg., A, and any input, I, $OPT(I) \geq \frac{n}{c} \Rightarrow \frac{A(I)}{OPT(I)} \leq c$

Hence, a c-competitive algorithm needs advice, only if $|\phi|_1 < n/c$.

Thus, $b(n,c) \leq \log(\#\text{advice strings with} < \frac{n}{c}\ 1\text{'s})$

$= O(\frac{\log c}{c} n)$, for $c > 1$.

[BKKKM 17]: Böckenhauer, Komm, Královič, Královič, Mömke
Online algorithms with advice: The tape model
Information and Computation 2017

Note: For inputs with $|\phi|_1 < \frac{n}{c}$, the algorithm is optimal.
In this case, we can afford $(c-1)|\phi|_1$ flips $0 \to 1$

Next idea: Covering designs

Construct new advice string ϕ' „covering" all 1's in ϕ:

$$\phi \xrightarrow{(c-1)|\phi|_1 \text{ flips } 0\to 1} \phi'$$

Ex: $c=2$

ϕ: 0 0 | | 0 0 | 0 | 0 0 0 | 0 0 0 0

ϕ': 0 | | | | | | | 0 0 | | 0 0 0 0

The resulting algorithm is

- well-defined: there is always a page that is safe to evict, since $\phi_i = 1 \Rightarrow \phi'_i = 1$
- c-competitive: $|\phi'|_1 \leq c \cdot |\phi|_1$ and each 1 gives ≤ 1 fault

Since it is not important which 0's are flipped, we can use covering designs...

Covering design

For any set S, $|S| = n$, an (n,k,ℓ)-covering design, $\ell \le k \le n$, is a set $\mathcal{G}_{k,\ell}$ of k-subsets of S s.t. each ℓ-subset of S is contained in (covered by) one of the sets in $\mathcal{G}_{k,\ell}$.

Ex: A $(4,3,2)$-covering design

$S = \{1,2,3,4\}$

$\mathcal{G}_{3,2} = \{\{1,2,3\}, \{1,2,4\}, \{2,3,4\}\}$

$\{1,2\}$ $\{1,4\}$ $\{3,4\}$
$\{1,3\}$ $\{2,4\}$
$\{2,3\}$

Indicating one of the three 3-subsets in $\mathcal{G}_{3,2}$ takes $\lceil \log |\mathcal{G}_{3,2}| \rceil = 2$ bits.

For an $(n, c\cdot|\Phi|_1, |\Phi|_1)$-covering design $\mathcal{G}$ give n, $|\Phi|_1$, and the index of Φ' in $\mathcal{G}$ as advice.

Use an $(n, c|\phi|_1, |\phi|_1)$ - covering design $\mathcal{G}_{c|\phi|_1, |\phi|_1}$ to indicate a string ϕ' s.t.

- $\phi_i = 1 \Rightarrow \phi'_i = 1$
- $|\phi'|_1 \leq c \cdot |\phi|_1$

Give $\underbrace{n, |\phi|_1}_{O(\log n)}$, and the $\underbrace{\text{index of } \phi' \text{ in } \mathcal{G}_{c|\phi|_1, |\phi|_1}}_{\leq \frac{n}{c} + O(\log n), \text{ since}}$ as advice.

$$\max_{1 \leq |\phi|_1 \leq \frac{n}{c}} \log \left| \mathcal{G}_{c|\phi|_1, |\phi|_1} \right| = \underbrace{\log \left(1 + \frac{(c-1)^{c-1}}{c^c} \right) \cdot n}_{\frac{n}{2c} \leq B(n,c) \leq \frac{n}{c}} + O(\log n)$$

$\therefore b(n,c) \leq \frac{n}{c} + O(\log n)$

Covering designs can also be used for accept/reject problems, e.g.,

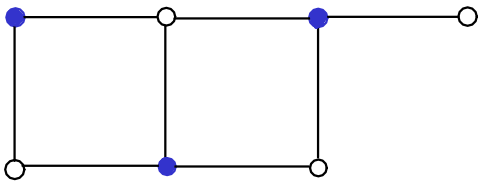
- Vertex Cover (VC): Any superset of a VC is a VC

VC

Input: $v_1 v_2 \cdots v_n$
With v_i, edges to $v_1, \ldots, v_{i-1}$ are revealed

Output: Set of vertices touching all edges

Ex:



- Independent Set (IS): Any subset of an IS is an IS

Use a $(n, n - \frac{|\phi|_1}{c}, |\phi|_0)$-covering design

$b(n,c) = B(n,c) + O(\log n) \leq \frac{n}{c} + O(\log n)$

VC and IS belong to

AOC (Asymmetric Online Covering):

Minimization problems:

For any feasible solution, X, cost(X) = |X|

Any superset of an optimal solution is a feasible solution

Maximization problems:

For any feasible solution, X, gain(X) = |X|

Any subset of an optimal solution is a feasible solution

Examples:

IS, VC, Dom. Set, Set Cover, Disj. Path Alloc., Cycle Finding

Knapsack (unit profit), Matching (edge arrival)

A problem $P \in AOC$ is AOC-complete, if

$b_p(n,c) \geq B(n,c) - O(\log n)$

A problem $P \in AOC$ is AOC-complete, if

$$b_P(n,c) \geq B(n,c) - O(\log n)$$

Completeness can be proven via advice preserving reductions.
from, e.g., Asymmetric String Guessing (ASG) [BFKM 16]:

ASG (with known history)

Input: $x_1 x_2 \cdots x_n$, $x_i \in \{0,1\}$
On request i, the alg. must guess x_i
(After the guess, x_i is revealed.)

Cost:

x_i \ y_i	0	1
0	0	1
1	∞	1

$$b(n,c) = B(n,c) \pm O(\log n)$$

Recall:

$$\frac{n}{2c} \leq B(n,c) \leq \frac{n}{c}$$

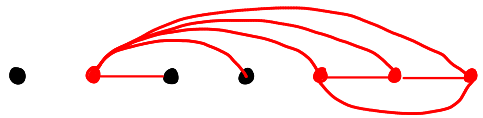
[BFKM 16]: Boyar, Favrholdt, Kudahl, Mikkelsen
A class of hard online problems
Theory of Computing Systems 2016

Ex: Reduction ASG → VC

A_{VC}: c-comp. VC-alg. with advice string ϕ_{VC}
A_{ASG}: c-comp. ASG-alg using A_{VC} and ϕ_{VC}

For any input X, A_{ASG} builds a graph G_X:

$X = 0\ 1\ 0\ 0\ 1\ 1\ 1$



●: V_0
●: V_1

V_1 induces a clique $\Rightarrow$ A_{VC} rejects ≤ 1 vertex from V_1

$OPT \leq |V_1| = |X|_1 \Rightarrow A_{VC}$ accepts $\leq c \cdot |X|_1$ vertices in total

Hence, $|\phi_{VC}| + O(\log n)$ bits suffice to
- guess all 1's correctly
- return $\leq c|X|_1$ 1's

Hence, $b_{VC}(n,c) \geq b_{ASG}(n,c) - O(\log n) \geq \frac{n}{2c} - O(\log n)$,

i.e., for VC, covering designs are optimal, up to an additive term of $O(\log n)$.

Derandomization [BKKK 17], [M16]

For any problem, P, with $I(n)$ possible input sequences of length n

$\exists$ rand. c-comp. alg. for P $\Rightarrow$ $b_P(n,c) \in O(\log\log I(n) + \log n)$

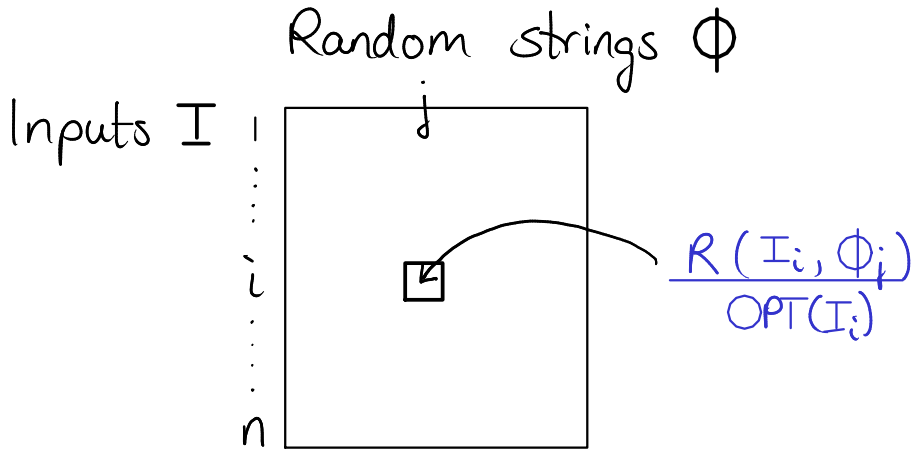
$\therefore$ For Paging, $b(n, H_k) \in O(\log n)$

[BKKK 17]:
Böckenhauer, Komm, Královič, Královič
On the advice complexity of the k-Server problem
Journal of Computer and System Sciences 2017

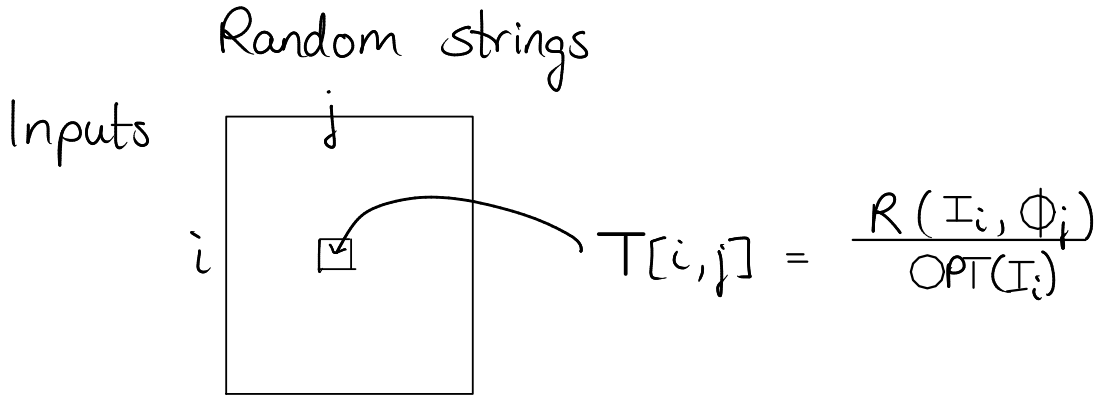
[M16]:
Jesper W Mikkelsen
Randomization can be as helpful as a glimpse of the future in online computation
ICALP 2016

Derandomization of c-competitive randomized alg. R: [BKKK 17]

Using a table:



Obtaining a $c\cdot(1+\varepsilon)$-competitive deterministic algorithm using $O\left(\log \frac{\log I(n)}{\log(1+\varepsilon)} + \log n\right)$ advice bits.

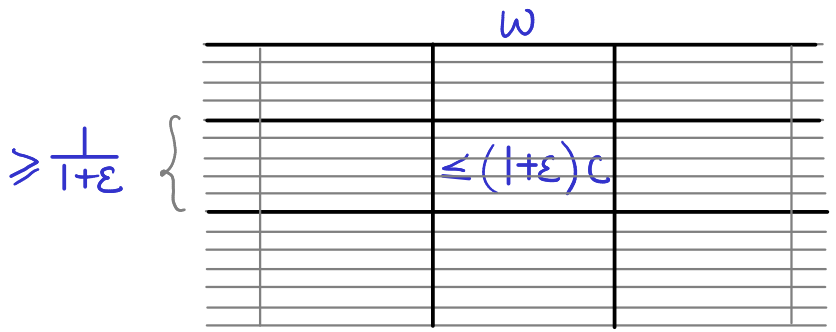


Invariant: $\forall$ row i : $E_j[A[i,j]] \le c \quad \Rightarrow$

$\exists$ column w : $E_i[A[i,w]] \le c$

In column w, $\ge \frac{1}{1+\varepsilon}$ of the entries are $\le (1+\varepsilon)c$ (Markov's Inequality)

The corresponding inputs are assigned w as advice string.



$\le \ell$ strings, where $(1+\varepsilon)^\ell \ge I(n) \quad \Leftrightarrow \quad \ell \le \frac{\log I(n)}{\log(1+\varepsilon)}$

[BKKKM 17]:

Even a few advice bits may improve the competitive ratio:

b advice bits $\Rightarrow$ $CR \leq \frac{2k+2}{2^b} + 3b$ (average of 2^b algorithms)

$\log k$ advice bits $\Rightarrow$ $C.R. \in O(\log k)$

On the other hand:

b advice bits $\Rightarrow$ $CR \geq \frac{k}{2^b}$

Follows from:

- $\forall I$ with $k+1$ different pages : $OPT(I) \leq \frac{n}{k}$
- $\forall A : \exists I$ with $k+1$ different pages : $A(I) \geq \frac{n}{2^b}$

  By induction on i:

  $\leq i$ different advice strings $\Rightarrow$

  $\exists I$ with $k+1$ diff. pages s.t. $|I| = \frac{n}{2^b} \cdot i$ and $ALG(I) \geq \frac{n}{2^b}$

[M16]:

Without advice, best randomized c.r. $= H_k$

$\Rightarrow$ $b(n, H_k - \varepsilon) \in \Omega(n)$ (even for randomized algorithms)

since Paging is Σ-repeatable and compact

$$A(\sigma_1 \sigma_2 \cdots \sigma_r) = \sum_{i=1}^{r} A(\sigma_i) + O(r)$$

Lower bound can be proven via input distribution with finite support

More generally, MTS is Σ-repeatable and compact.

Hence, for any problem that can be modeled as a Metrical Task System,

any lower bound for randomized algorithms without advice is a lower bound for randomized algorithms with $O(n)$ advice bits.

Similarly, deterministic lower bounds carry over to (randomized) algorithms with $o(n)$ advice bits, if
- the bound holds even if OPT and n are known to the alg. and
- the problem is V-repeatable.

$$A(\sigma_1 \sigma_2 \cdots \sigma_r) = \max_{1 \le i \le r} A(\sigma_i)$$

Paging with Predictions [LV18]

For each request,
predict when the same page will be requested again.

Combine advantages of ML with worst-case guarantees:

- If predictions are good,
  the algorithm should be close to optimal (consistency)
- If predictions are bad,
  the algorithm should not be too much worse than
  the best online algorithm (robustness)

[LV18]:
Lykouris, Vassilvitski
Competitive Caching with Machine Learned Advice
ICML 18 and JACM 21

For each request,
predict when the same page will be requested again.

Error measure : ℓ_1

Ex :

	1	2	3	4	5	6	7	8	9	10	11	12	13
I :	1	2	1	3	4	1	3	4	1	2	1	3	2
Prediction :	4	8	6	7	8	10	9	15	11	13	14	14	16
Actual :	3	10	6	7	8	9	12	14	11	13	14	14	14
Error :	1	2	0	0	0	1	3	1	0	0	0	0	2

Total error : $\eta = 1 + 2 + 1 + 3 + 1 + 2 = 10$

$$\varepsilon = \frac{\eta}{OPT} = \frac{10}{2} = 5$$

↑ if $k = 3$

Marking alg (Class of alg.)

For each request
  Mark requested page
  If fault
    If all pages in cache are marked
      Remove all marks (Current req. starts a new k-phase)
    Evict an unmarked page

Ex: LRU: Least Recently Used

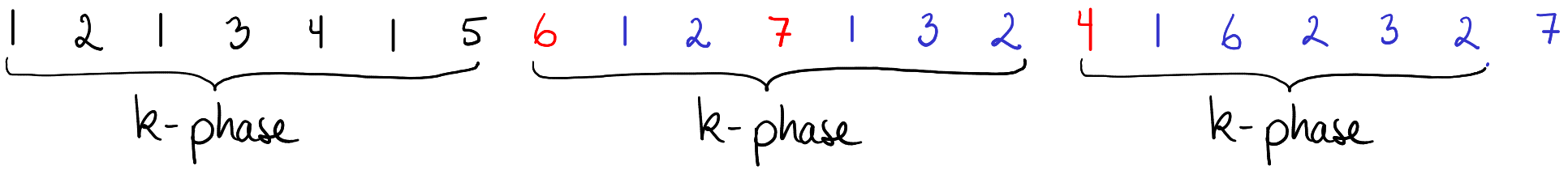
$CR = k$

Mark: Evict random unmarked page

$CR = 2H_k - 1$, where

$$H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k} \approx \ln k$$

Ex: $k=5$

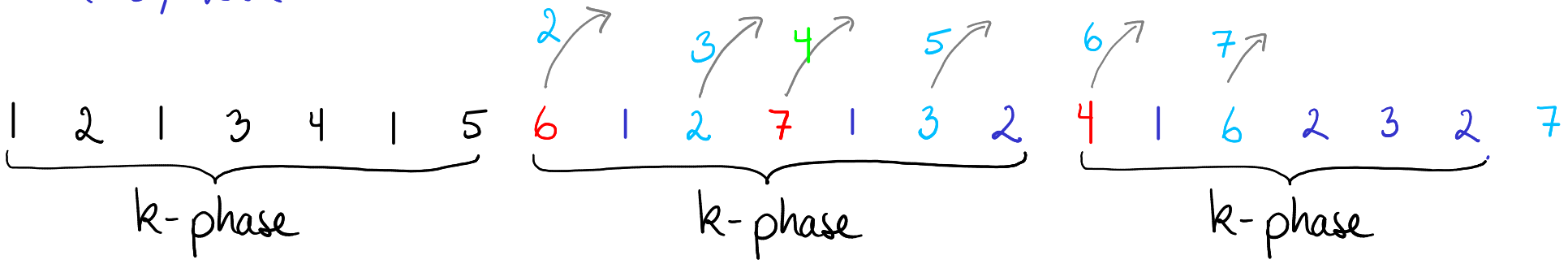


k-phase: Maximal subsequence with k distinct pages

New page: Not req. in previous phase
Old page: Requested in previous phase

Note:
$OPT \geq \frac{N}{2}$, where N = total # new pages

Ex: $k=5$, LRU



Two chains:
$6 \to 2 \to 3 \to 5$
$7 \to 4$

One chain:
$4 \to 6 \to 7$

Note: A chain always starts with a new page.
($\Rightarrow$ #chains $\leq 2 \cdot OPT$)

Note: total #faults = total chain length

For Mark (evicting random unmarked pages),
the expected length of a chain is $\leq H_k \approx \ln k$

Predictive Marker [LV18]

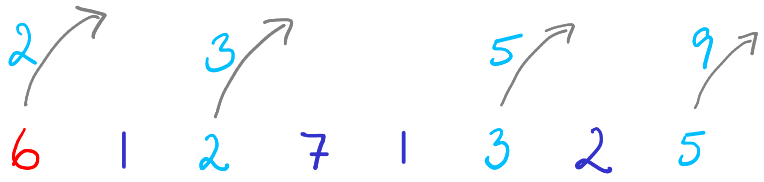
On a fault

...

LFD_p on unmarked pages

...

Ex:



Chain: $6 \to 2 \to 3 \to 5 \to 9$

The pages of the chain are predicted to arrive in the relative order 9, 5, 3, 2 (by the def. of LFD_p)

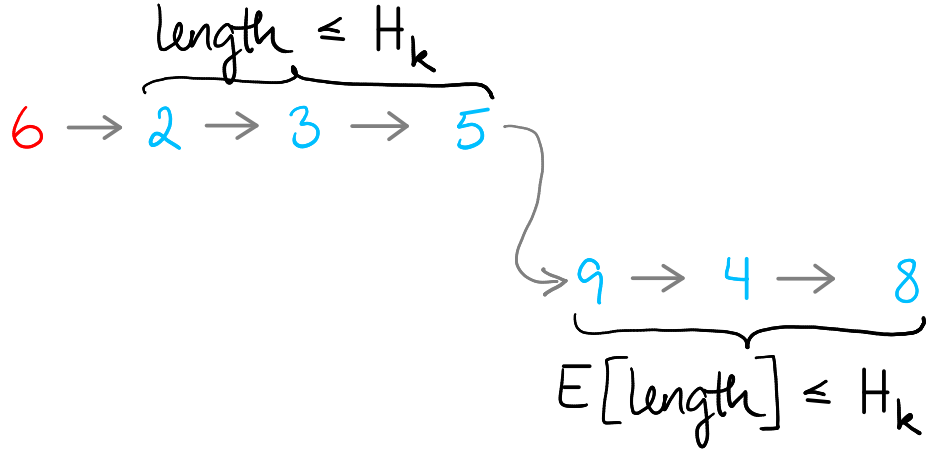
$\Rightarrow$ #inversions $\geq 3+2+1 = \frac{4 \cdot 3}{2}$ $\Rightarrow$ $\eta \geq \frac{4 \cdot 3}{2}$

A chain of length ℓ contributes $\Omega(\ell^2)$ to η.

Predictive Marker is $(2 + O(\sqrt{\varepsilon}))$ - consistent

Predictive Marker [LV18]

On a fault
- If chain shorter than H_k
  - LFD_P on unmarked pages
- Else
  - Evict random unmarked page



Predictive Marker is $4H_k$-robust

$$C.R.(P.M.) \leq \min\{\underbrace{2+4\sqrt{5\epsilon}}_{\text{worst-case}}, \underbrace{4H_k}_{\text{expected}}\}$$

Predictive Marker [LV18]

On a fault
  If chain shorter than H_k
    LFD_p on unmarked pages
  Else
    Evict random unmarked page

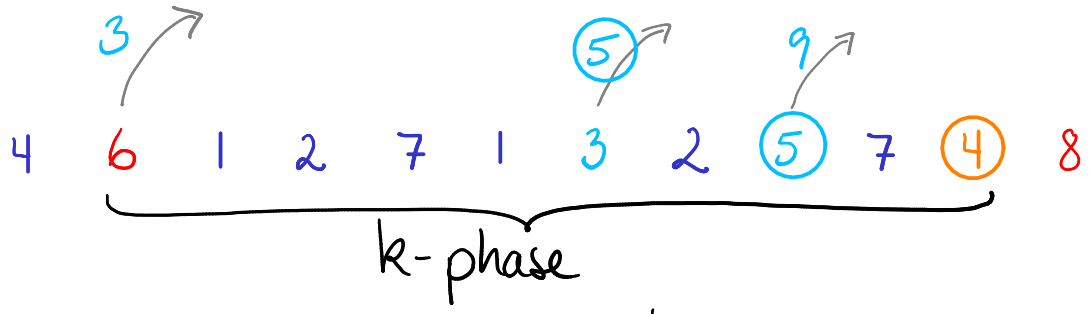
$$C.R.(P.M.) \leq 2 \min\{1 + 2\sqrt{5\varepsilon}, 2H_k\}$$

„Modified Predictive Marker"

On a fault
  If the requested page is new
    LFD_p on unmarked pages
  Else
    Evict random unmarked page

$$C.R.(MPM) \leq 2 \cdot (2 + \min\{H_{2\varepsilon}, H_k\})$$

[R20]:
Rohatgi
Near-Optimal Bounds for
Online Caching with
Machine Learned Advice
SODA 20



4 is predicted to arrive before 5

m unmarked pages ⇒ m inversions

„Modified Predictive Marker" [R20]

On a fault
  If the requested page is new
    LFD_p on unmarked pages
  Else
    Evict random unmarked page

6 → 3 → 5 → 9

E[length] ≤ H_m

$$C.R.(MPM) \leq 2 \cdot (2 + \min\{H_{2\varepsilon}, H_k\})$$

[W20]:

$$C.R.(LFD_p) \leq \min\left\{1+2\varepsilon,\ 4+\frac{4\varepsilon}{k-1}\right\}$$

Combine LFD_p with LRU via a deterministic „blackbox" [FKLMSY91]

$$Combine_D(LFD_p, LRU)(I) \leq 2 \cdot \min\{LFD_p(I), LRU(I)\} \quad \text{for any } I$$

Combine LFD_p with Equitable via a randomized „blackbox" [BB00]

$$Combine_R(LFD_p, Eq, \gamma)(I) \leq (1+\gamma)\cdot\min\{LFD_p(I), Eq(I)\} + O\left(\frac{1}{\gamma}\right)$$

for any I and $0 < \gamma < \frac{1}{4}$

[W20]: Wei
Better and Simpler Learning-Augmented Online Caching
APPROX 20

[FKLMSY91]: Fiat, Karp, Luby, McGeoch, Sleator, Young
Competitive Paging Algorithms
J. Alg. 91

[BB00]: Blum, Burch
On-Line Learning and the Metrical Task Systems Problem
Mach. Learn. 00

[W20]:

$$C.R.(LFD_p) \leq \min\left\{1+2\varepsilon,\ 4+\frac{4\varepsilon}{k-1}\right\}$$

COMBINE$_D$ (LFD_p, LRU)

Simulate LFD_p and LRU
On every second fault
  Evict page $\notin$ LFD_p's cache after serving the request
On every second fault
  Evict page $\notin$ LRU's cache after serving the request

Or

Follow-the-Leader (LFD_p, LRU)

$$C.R.(\text{Combine}(LFD_p, LRU)) \leq \min\left\{2+4\varepsilon,\ 8+\frac{8\varepsilon}{k-1},\ 2k\right\}$$

For any deterministic alg. A, $C.R.(A) \in \Omega\left(1+\frac{\varepsilon}{k}\right)$

For any deterministic alg. A, $C.R.(A) \in \Omega\left(1+\frac{\varepsilon}{k}\right)$

Sketch:

$2 \leq j \leq k-1$

Repeat:

$$\langle p_1^k, p_2^k, \ldots, p_k^k \rangle^k \langle p_{k+1} \rangle \langle ? \rangle^j \langle p_{k+1} \rangle^{k^2+1}$$

just evicted ↑

$$\left.\begin{array}{l} \eta \in O(jk) \\ OPT = 2 \end{array}\right\} \Rightarrow \varepsilon \in O(jk) \Rightarrow j \in \Omega\left(\frac{\varepsilon}{k}\right)$$

$$ALG \geq j+1 = OPT + j-1 = \left(1 + \frac{j-1}{2}\right) OPT = \left(1 + \Omega\left(\frac{\varepsilon}{k}\right)\right) OPT$$

$$> \left(1 + \frac{j}{4}\right) OPT, \quad \text{since } j \geq 2$$

$$= \left(1 + \Omega\left(\frac{\varepsilon}{k}\right)\right) OPT$$

[W20]:

$\underline{\text{Combine}_R}(\text{LFD}_p, \text{Equitable}, 0<\gamma<\frac{1}{4})$ (Randomized Weighted Majority)

$\beta \leftarrow 1-\frac{\gamma}{2}$

$w_{LFD} \leftarrow 1$, $w_{EQ} \leftarrow 1$

On each request

For $A \in \{\text{LFD}_p, \text{EQUITABLE}\}$

If A faults

$w_A \leftarrow \beta \cdot w_A$

On each fault

$p_{\text{LFD}_p} \leftarrow \frac{w_{\text{LFD}_p}}{w_{\text{LFD}_p}+w_{EQ}}$, $p_{EQ} \leftarrow \frac{w_{EQ}}{w_{\text{LFD}_p}+w_{EQ}}$

Choose algorithm using probabilities p_{LFD_p} and p_{EQ}

$C.R.(\text{Combine}_R(\text{LFD}_p, \text{Equitable}, \gamma)) \leq (1+\gamma)\min\{1+2\varepsilon, 4+\frac{4\varepsilon}{k-1}, H_k\}$, $\gamma>0$

Unit Profit Knapsack: Maximize #items packed

Threshold Algorithm [BFK22]

Advice: average item size, a, in OPT

$$T(i) = \frac{ae}{aei+1}$$

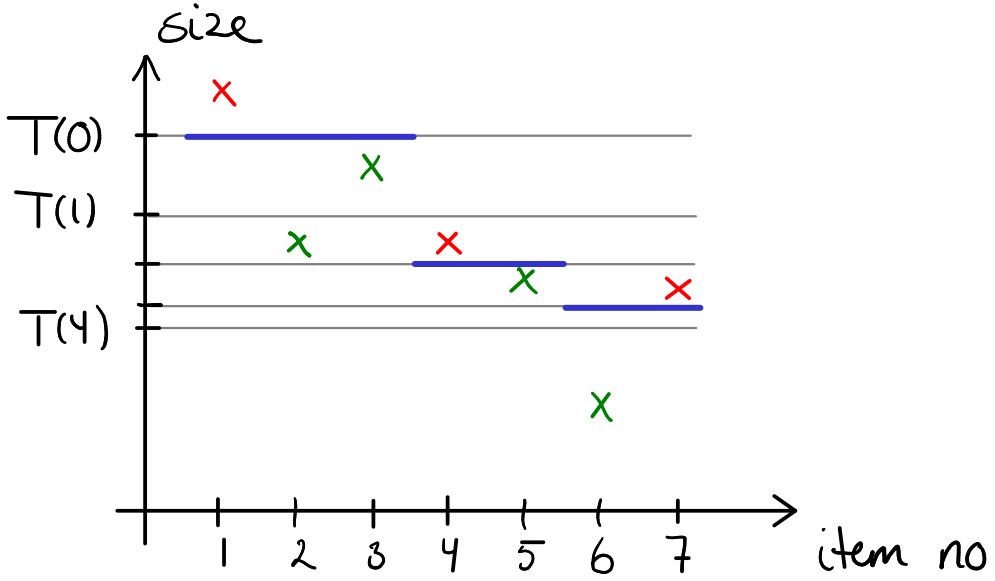
$t = \min\{ T(i) \mid i \text{ items} \geq T(i) \text{ in knapsack} \}$

If current item fits in knapsack and has size $\leq t$

Accept

$\frac{e}{e-1}$ - competitive

Optimal among alg. knowing only a



[BFK22]:
Boyar, Favrholdt, Larsen
Online Unit Profit Knapsack with Untrusted Predictions
SWAT 2022

Unit Profit Knapsack: Maximize #items packed

Average item size in OPT → $a = r \cdot \hat{a}$ ← predicted value of a

Threshold Algorithm [BFK22]

Prediction: average item size, $\hat{a}$, in OPT

$$T(i) = \sqrt{\frac{\hat{a}}{2i}}$$

$$t = \min\{ T(i) \mid i \text{ items} \geq T(i) \text{ in knapsack} \}$$

If current item fits in knapsack and has size $\leq t$

Accept

$$\text{c.r.} = \begin{cases} 2r, & r \geq 1 \\ \frac{2}{r}, & r \leq 1 \end{cases}$$

No alg. knowing only $\hat{a}$ can be better for both $r > 1$ and $r < 1$.

Bin Packing

$1.54 < C.R. < 1.58$

Advice: m = # medium sized items $\left(\frac{1}{2} < \text{size} \leq \frac{2}{3}\right)$

$C.R. = 1.5$ [BKLL-O 16]

Advice: k-bit approx. of $\frac{m}{m+t}$, t = # bins used for tiny items

$C.R. \approx 1.47$ [ADKRR 18]

[BKLL-O 16]: Boyar, Kamali, Larsen, Lopez-Ortiz
Online Bin Packing with Advice
Algorithmica 16

[ADKRR 18]: Angelopoulos, Dürr, Kamali, Renault, Rosén
Online Bin Packing with Advice of Small Size
ToCS 18

Bin Packing

Advice: m = # medium sized items $\left(\frac{1}{2} < \text{size} \leq \frac{2}{3}\right)$

Advice: k-bit approx. of $\frac{m}{m+t}$, t = # bins used for tiny items

Prediction: k-bit approx. of $\frac{m}{m+t}$ [ADJKR 20]

$\alpha \in [0,1]$ is a parameter of trust

$(1.5 + f_c(\alpha, k))$ - consistent (decreasing in α)

$(1.5 + f_r(\alpha))$ - robust (increasing in α)

[ADJKR 20]: Angelopoulos, Dürr, Jin, Kamali, Renault
Online Computation with Untrusted Advice
ITCS 20

Summing up

Advice Complexity

How much extra information does it take to
- be (offline) optimal ?
- beat best online algorithm?

Algorithms may be stepping stones to
- semi-online algorithms
- parallel solutions
- algorithms with predictions

Lower bounds give impossibility results for
- randomized online algorithms (sometimes)
- semi-online algorithms
- parallel solutions

(Online) algorithms with predictions

- New and rapidly evolving field
- Some keywords: Predictions, error measures, consistency, robustness

Summing up

Advice Complexity

How much extra information does it take to
- be (offline) optimal ?
- beat best online algorithm?

Algorithms may be stepping stones to
- semi-online algorithms
- parallel solutions
- algorithms with predictions

Lower bounds give impossibility results for
- randomized online algorithms (sometimes)
- semi-online algorithms
- parallel solutions

THANK YOU !!

(Online) algorithms with predictions

- New and rapidly evolving field
- Some keywords: Predictions, error measures, consistency, robustness