§6 Online Algorithms

Ski Rental Problem: The skiing season begins but is not known how long to last: Each day the weather either • allows for skiing or • does not. You are then to decide whether to (i) rent ski at \$1/day or (ii) buy for \$$D$>1 until end of season.

Breakeven algorithm: rent D-1 days, then buy. cost $\min(L,2D-1) \le (2-1/D) \times$ optimum $\min(L,D)$.

Fix any algorithm $\mathcal{A}$, run on ∞ season, let X=#days it rents before buying, abort season on day #X+1: cost $X+D \ge (2-1/D) \cdot \min(X+1,D)$ optimum. *adversary*

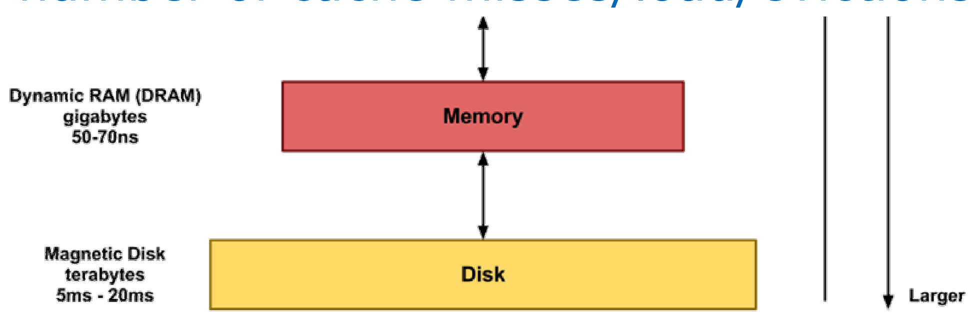
Analyze *online* algorithm output in comparison to the *offline* optimum: **competitive ratio**.

Online Paging

k pages of *fast* memory caching $K \gg k$ *slow* pages.

For sequence $a_1,\ldots,a_N \in \{1,\ldots K\}$ of disk accesses, minimize number of cache misses/load/evictions.

- **LRU**
- **LFU**
- **FIFO**
- **MIN**

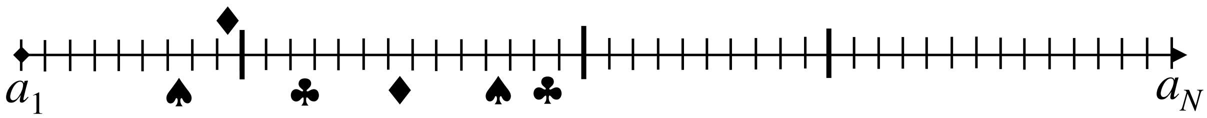


Input is revealed gradually; *online* algorithm makes decisions with only partial knowledge.

Analyze *online* algorithm output in comparison to the *offline* optimum: **competitive ratio**.

Online Paging: LRU

Lemma: Consider optimal algorithm O, starting with same initial page content.
In each phase, O incurs at least <u>one</u> page fault.



Proof: Divide $1,\ldots,N$ into *phases* $1<t_0<t_1<\ldots<t_M=N$ s.t. LRU incurs <u>precisely</u> k faults in $(t_{m-1}\ldots t_m]$ and $(0\ldots k]$ faults in $[1\ldots t_1]$

> *Whenever a new page is accessed, evict the one **L**east **R**ecently **U**sed.*

fault

Theorem: LRU has competitive ratio k.

Analyze *online* algorithm <u>output</u> in comparison to the *offline* optimum: **competitive ratio**.

Online Paging: LFU

KAIST CS500 M. Ziegler

k pages of *fast* memory caching $K \gg k$ *slow* pages.
For sequence $a_1,\ldots,a_N \in \{1,\ldots K\}$ of disk accesses, minimize number of cache misses/load/evictions.

repeat m-times

$2,3,\ldots k,\ 2,3,\ldots k,\ 2,3,\ldots k,\ \ldots\ldots\ldots,\ 2,3,\ldots k,$

$1,\ k+1,\ 1,\ k+1,\ 1,\ k+1,\ \ldots\ldots\ldots,\ 1,\ k+1$

repeat m-times

> *Whenever a new page is accessed, evict one **L**east **F**requently **U**sed.*

Theorem: LFU has no (finite) competitive ratio!

Proof: Compare LFU to the optimal algorithm on the above access sequence with $K=k+1$.

Deterministic Online Paging

KAIST
CS500 M. Ziegler

k pages of *fast* memory caching $K \gg k$ *slow* pages.

For sequence $a_1,\ldots,a_N \in \{1,\ldots K\}$ of disk accesses, minimize number of cache misses/load/evictions.

Theorem: No deterministic algorithm $\mathcal{A}$ has competitive ratio $<k$.

"adversary"

Proof: Let $K:=k+1$ and consider the access sequence $a_1,\ldots,a_N$ where a_n is defined inductively as the page currently not in $\mathcal{A}$'s cache.

Then $\mathcal{A}$ suffers a page fault on every request.

Yet an *offline* algorithm can choose to evict a page not to be accessed in the next k-1 steps. ■

Randomized Online & Adversaries

KAIST
CS500 M. Ziegler

Ski Rental Problem revisited**:** decide daily whether to (i) rent for \$1/day or to (ii) buy at \$$D$ for good.

Breakeven is $(2–1/D)$-competitive, and best possible: Fix any algorithm $\mathcal{A}$, run on ∞ season, let X=#days it rents before buying, abort season on day #X+1.

Randomized Ski Rental: Guess X with distribution

$$\mathbb{P}[X=x] := (1\text{-}1/D)^{D\text{-}1\text{-}x} \cdot D^{D\text{-}1} / (D^D - (D\text{-}1)^D), \quad x=0\ldots D\text{-}1$$

Then the expected cost is $\leq e/(e\text{-}1)$ × optimum.

No randomized online algorithm can do better.

Here *oblivious* adversaries, not *adaptive* ones.