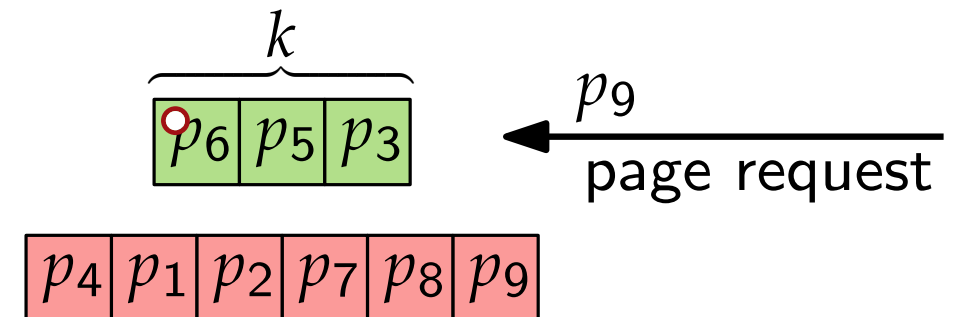


Advanced Algorithms

Online Algorithms

Ski-Rental Problem and Paging



Ski-Rental Problem

Winter has started...



...and the skiing season is back!



- But what if there is not always enough snow? Or snow but “bad” weather?
- Is it worth **buying** new skis?
- Or should we rather **rent** them?
- We don't know the weather (much) in advance.

Ski-Rental Problem – Definition

Behavior.

- Every day when there is “good” weather, you go skiing.
 - We call this is a **good** day.
- Each morning, we can check if today is a good day, but we can't check any earlier.

Costs.

- Renting skis for 1 day costs 1 [Euro].
- Buying skis costs M [Euros] and you have them forever.
- In the end, there will have been T good days.

(When) Should we buy skis?

Task.

- Not knowing T , devise a strategy if and when to buy skis.

Ski-Rental Problem – Strategies I and II

Renting costs 1 per day
Buying costs M
 T good days

Strategy I: Buy on the **first** good day

- Imagine this was the only good day the whole winter.
- Then we have paid M ; optimally, we would have rented and paid 1.
- So Strategy I is M times worse than the optimal strategy.

→ for arbitrarily large M arbitrarily bad

Strategy II: never buy, **always rent**

- Suppose there are many good days, i.e., $T > M$.
- Then we have paid T .
Optimally, we would have bought on or before the first good day and paid M .
- Strategy II is T/M times worse than the optimal strategy.

competitive ratio

→ for arbitrarily large T arbitrarily bad

Ski-Rental Problem – Strategy III

Renting costs 1 per day
Buying costs M
 T good days

Is there a strategy that cannot become arbitrarily bad? – Yes!

Strategy III: buy on the M -th good day

- Observation: the optimal solution pays $\min(M, T)$
- If $T < M$, the competitive ratio is 1. Otherwise, it is $\frac{2M-1}{M} = 2 - \frac{1}{M} \stackrel{M \rightsquigarrow \infty}{=} 2$.

$\Rightarrow$ Strategy III is deterministic and 2-competitive.

Theorem 1. No det. strategy is better than 2-competitive (for $M \rightsquigarrow \infty$; in general: $2 - \frac{1}{M}$).

Proof Idea.

- Any det. strategy can be formulated as “buy on the X -th day of rental” for a fixed X .
- $X = 0$ and $X = \infty$ are arbitrarily bad; assume $X \in \mathbb{N}^+$. Observe: w.c. is $T = X$.
- $\frac{c_{\text{det}}}{c_{\text{OPT}}} = \frac{X-1+M}{\min(X,M)} \geq \min \left(\underbrace{\frac{X-1+X+1}{X}}_{\text{case } X < M}, \underbrace{\frac{M-1+M}{M}}_{\text{case } M \leq X} \right) = \min \left(2, 2 - \frac{1}{M} \right) = 2 - \frac{1}{M} \stackrel{M \rightsquigarrow \infty}{=} 2$

Ski-Rental Problem – Strategy IV

Renting costs 1 per day
Buying costs M
 T good days

Can we get below this bound using randomization? – Let's try!

Strategy IV: throw a coin; **HEADS:** buy on the M -th good day
TAILS: buy on the αM -th good day ($\alpha \in (0, 1)$)

■ Observation: worst case can only be $T = M$ or $T = \alpha M$

■ Case $T = M$:
$$\frac{E[c_{\text{Strategy IV}}]}{c_{\text{OPT}}} = \frac{\frac{1}{2} \cdot (2M-1) + \frac{1}{2} \cdot ((1+\alpha)M-1)}{M} = \frac{3+\alpha}{2} - \frac{1}{M} \stackrel{M \rightarrow \infty}{=} \frac{3+\alpha}{2}$$

■ Case $T = \alpha M$:
$$\frac{E[c_{\text{Strategy IV}}]}{c_{\text{OPT}}} = \frac{\frac{1}{2} \cdot \alpha M + \frac{1}{2} \cdot ((1+\alpha)M-1)}{\alpha M} = 1 + \frac{1}{2\alpha} - \frac{1}{2\alpha M} \stackrel{M \rightarrow \infty}{=} 1 + \frac{1}{2\alpha}$$

■ The w.c. ratio is minimum if $\frac{3+\alpha}{2} = 1 + \frac{1}{2\alpha} \Rightarrow \alpha = \frac{\sqrt{5}-1}{2}$

$\Rightarrow$ Strategy IV (with $\alpha = \frac{\sqrt{5}-1}{2} \approx 0.62$) is 1.81-competitive, randomized, and better than any deterministic strategy.

■ With a more sophisticated probability distribution for the point in time when we buy skis, we can get an expected competitive ratio of even $\frac{e}{e-1} \approx 1.58$.

Online vs. Offline Algorithms

Online Algorithm

- No full information available initially (*online problem*)
- Decisions are made with incomplete information.
- The algorithm may get more information over time or by exploring the instance.

in the worst case (deterministic algo.)

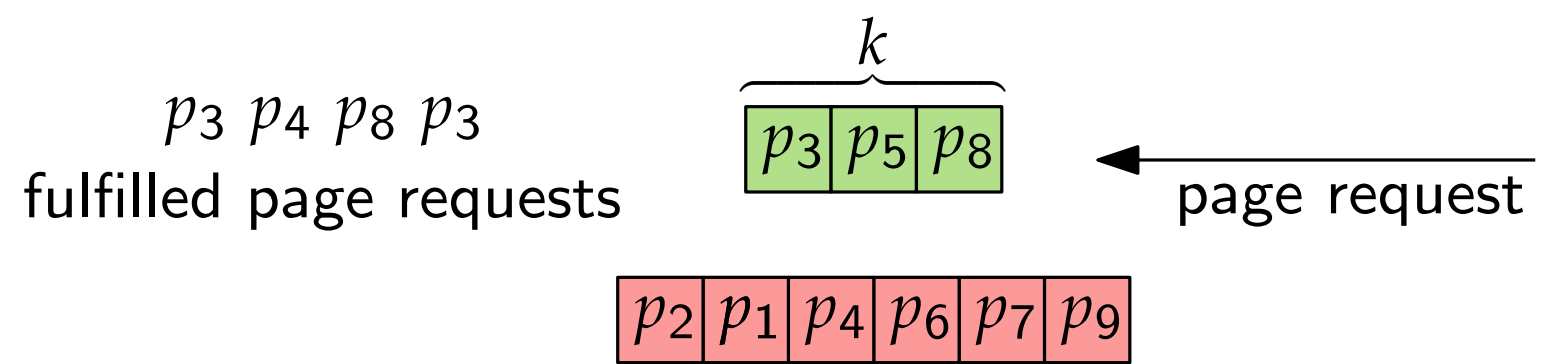
in the worst *average* case (randomized algo.)

- The objective value of the returned solution divided by the objective value of an optimal (offline) solution is the *competitive ratio*.
- Examples (problems & algorithms):
Ski-Rental Problem, searching in unknown environments, Cow-Path Problem, Job-Shop Scheduling, Insertion Sort, Paging (replacing entries in a memory)

Offline Algorithm

- Full information available initially (*offline problem*)
- Decisions are made with complete information.

Paging: Definition



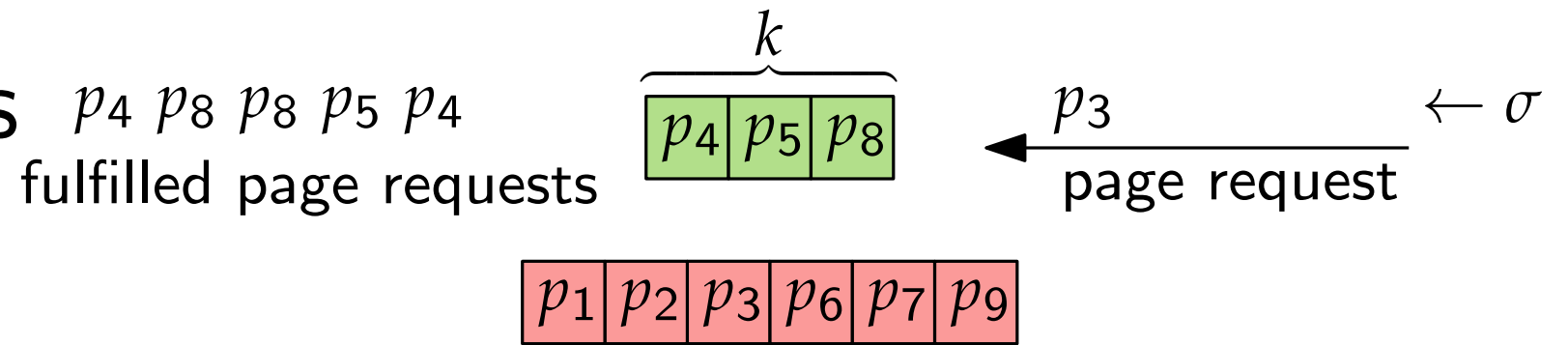
Given (offline/online):

- Fast access memory (a cache) with a capacity of k pages
- Slow access memory with unlimited capacity
- If a page is requested, but it is not in the cache (*page fault*), it has to be swapped with a page in the cache. A page request is fulfilled if the page is in the cache.
- Sequence σ of page requests that need to be fulfilled in order. / where we just see one request and have to fulfill that request before we see the next request.

Objective value:

- Minimize the number of page faults while fulfilling σ .

Paging: Det. Strategies



- On a page fault, a paging algorithm chooses which page to evict from the cache.

Deterministic Strategies: Evict the page that has ...

- Least Frequently Used (LFU): ... the lowest number of accesses since it was loaded.
- Least Recently Used (LRU): ... been accessed least recently.
- First-in-first-out (FIFO): ... been in the cache the longest.

Which of them is – theoretically provable – the best strategy?

Theorem 2. LRU & FIFO are k -competitive. No deterministic strategy is better.

Paging: Deterministic Strategies – Analysis

Theorem 2. LRU & FIFO are k -competitive. No deterministic strategy is better.

Proof. (only for LRU, FIFO is similar)

MIN: optimal strategy
 σ : sequence of pages

- Initially, the cache contains the same pages for all strategies.
- We partition σ into phases $P_0, P_1, \dots$, s.t. LRU has at most k faults in P_0 and exactly k faults in each other phase.
- We show next: MIN has at least 1 fault in each phase.
- Clearly, MIN also faults in P_0 ; consider P_i ($i \geq 1$) and let p be the last page of P_{i-1} .
- Show: P_i contains k distinct page requests different from p (implies a fault for MIN).
- If the k page faults of LRU in P_i are on distinct pages (different from p), we're done.
- Assume LRU has in P_i two page faults on one page q . In between, q has to be evicted from the cache. According to LRU, there were k distinct page requests in between.
- Similarly, if LRU faults on p in P_i , there were k distinct page requests in between.

Paging: Deterministic Strategies – Analysis

Theorem 2. LRU & FIFO are k -competitive. No deterministic strategy is better.

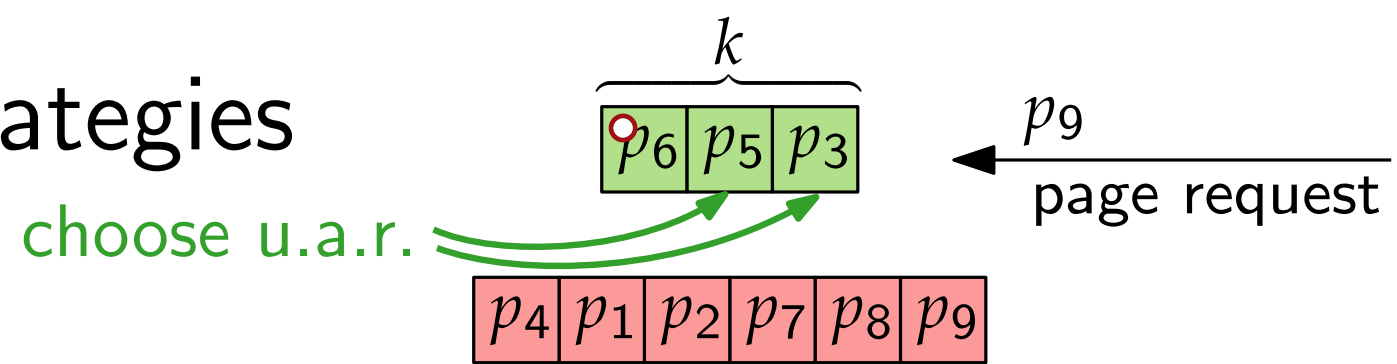
Proof. (only for LRU, FIFO is similar)

- Remains to prove: No deterministic strategy is better than k -competitive.
- Assume that the memory system consists of $k + 1$ pages (i.e., cache + 1 page).
- For any deterministic strategy, there is a worst-case page sequence σ^* always requesting the page that is currently not in the cache.
- Let MIN have a page fault on the i -th page of σ^* .
- Then the next $k - 1$ requested pages are in the cache already & the next fault occurs on the $(i + k)$ -th page of σ^* the earliest. Until then, the det. strategy has k faults.

$\Rightarrow$ The competitive ratio cannot be better than $\frac{|\sigma^*|}{\left\lceil \frac{|\sigma^*|}{k} \right\rceil} \stackrel{|\sigma^*| \rightsquigarrow \infty}{=} k.$



Paging: Randomized Strategies



Randomized strategy: MARKING

Phase P_2

- Proceeds in phases
- At the beginning of each phase, all pages are unmarked.
- When a page is requested, it gets **marked**.
- A page for eviction is chosen **uniformly at random** from the unmarked pages.
- If all pages are marked and a page fault occurs, unmark all and start new phase.

Theorem 3. MARKING is $2H_k$ -competitive.

Remark.

$H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$ is the k -th harmonic number. For $k \geq 2$: $H_k < 1 + \ln k$.

Paging – Rand. Strategy – Analysis

Theorem 3. MARKING is $2H_k$ -competitive.

We consider phase P_i .

Proof.

- A page is *stale* if it is unmarked, but was marked in P_{i-1} .
- A page is *clean* if it is unmarked, but not stale. Hence, it is not in the cache!
- S_{MARK} (S_{MIN}): set of pages in the cache of MARKING (MIN)
- d_{begin} : $|S_{\text{MIN}} \setminus S_{\text{MARK}}|$ at the beginning of P_i
- d_{end} : $|S_{\text{MIN}} \setminus S_{\text{MARK}}|$ at the end of P_i
- c = number of clean pages requested by σ in P_i
- MIN has $\geq \max(c - d_{\text{begin}}, d_{\text{end}})$ faults.

MARKING produces a fault for each of the c clean pages in phase P_i . Among these, MIN can avoid only those faults that correspond to pages that were initially in $S_{\text{MIN}} \setminus S_{\text{MARK}}$.

Each page in $S_{\text{MIN}} \setminus S_{\text{MARK}}$ at the end of P_i has caused MIN to produce a fault since all k pages in S_{MARK} have been requested by σ during P_i .

Paging – Rand. Strategy – Analysis

Theorem 3. MARKING is $2H_k$ -competitive.

We consider
phase P_i .

Proof.

- A page is *stale* if it is unmarked, but was marked in P_{i-1} .
- A page is *clean* if it is unmarked, but not stale. Hence, it is not in the cache!
- S_{MARK} (S_{MIN}): set of pages in the cache of MARKING (MIN)
- d_{begin} : $|S_{\text{MIN}} \setminus S_{\text{MARK}}|$ at the beginning of P_i
- d_{end} : $|S_{\text{MIN}} \setminus S_{\text{MARK}}|$ at the end of P_i
- c = number of clean pages requested by σ in P_i
- MIN has $\geq \max(c - d_{\text{begin}}, d_{\text{end}}) \geq \frac{1}{2}(c - d_{\text{begin}} + d_{\text{end}}) = \frac{c}{2} - \frac{d_{\text{begin}}}{2} + \frac{d_{\text{end}}}{2}$ faults.
Over all phases, all $\frac{d_{\text{begin}}}{2}$ and $\frac{d_{\text{end}}}{2}$ cancel out, except the first $\frac{d_{\text{begin}}}{2}$ and the last $\frac{d_{\text{end}}}{2}$.
- Since the first $d_{\text{begin}} = 0$, MIN has at least $\frac{c}{2}$ faults per phase.

Paging – Rand. Strategy – Analysis

Theorem 3. MARKING is $2H_k$ -competitive.

Proof.

- For the clean pages, MARKING has c faults.
- For the stale pages, there are $s = k - c \leq k - 1$ requests.
- For requests $j = 1, \dots, s$ to stale pages, consider the expected number of faults $E[F_j]$.
- $c(j)$: # clean pages requested in P_i at the time the j -th stale page is requested
 $s(j)$: # pages that were stale at the beginning of P_i and have not been requested
- $E[F_j] = \frac{s(j)-c(j)}{s(j)} \cdot 0 + \frac{c(j)}{s(j)} \cdot 1 \leq \frac{c}{k+1-j}$ $s(j) = k - (j - 1)$
- $E \left[\sum_{j=1}^s F_j \right] = \sum_{j=1}^s E[F_j] \leq \sum_{j=1}^s \frac{c}{k+1-j} \leq \sum_{j=2}^k \frac{c}{j} = c \cdot (H_k - 1)$
- So the competitive ratio of MARKING is at most $\frac{c + c(H_k - 1)}{c/2} = 2H_k \in O(\log k)$ $\square$

Reminder.

No deterministic strategy is better than k -competitive.

MARKING is $O(\log k)$ -competitive.

$\Rightarrow$ **Exponential improvement!**

Discussion

- Online algorithms operate in a setting different from that of classical algorithms. However, this setting of incomplete information is very natural and occurs often in real-world applications. Can you think of further examples?
- We can also transform a classical problem with incomplete information into an online problem. E.g.: Matching problem for ride sharing.
- Randomization can help to improve our behavior on worst-case instances. You may also think of: we are less predictable for an adversary.

Literature

Main source:

- Sabine Storandt's lecture script "Randomized Algorithms" (2016/2017)

Original papers:

- [Belady '66] "A Study of Replacement Algorithms for Virtual-Storage Computer."
- [Sleator, Tarjan '85] "Amortized Efficiency of List Update and Paging Rules."
- [Fiat, Karp, Luby, McGeoch, Sleator, Young '91] "Competitive Paging Algorithms."