Bijective analysis of online algorithms

Christoph Dürr
Sorbonne University, France and CNRS

Mathematical Optimization Theory and Operations Research (MOTOR)
July 2019

Outline

- Models for analyzing online algorithms
- The linear search problem

work done 20 years ago by the community

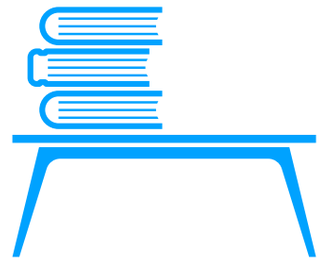
work done last year: Angelopoulos, D, Jin, STACS'2019

Introduction: the paging problem

- Processor generates page requests
- **page hit:** If page is in cache it is served with no cost.
- **page fault:** If page is not in cache it must be fetched at unit cost and placed in cache. Possibly some page from the cache must be evicted in order to make space. But which page? → POLICY

pages := memory units of same size (depicted as books)

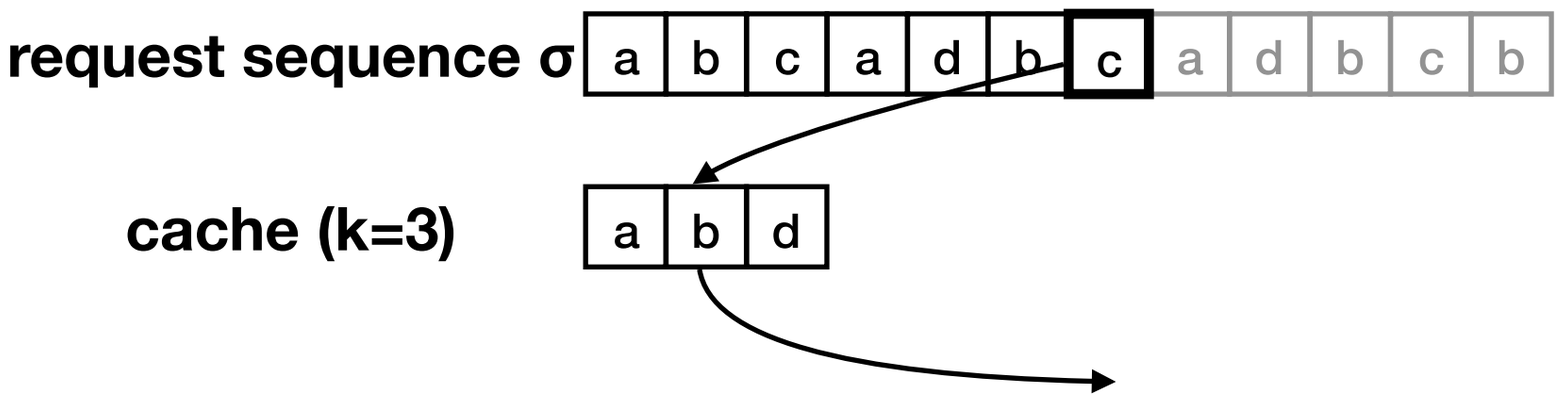
processor



cache
quick access
capacity : k pages

main memory
slow access

Competitive ratio

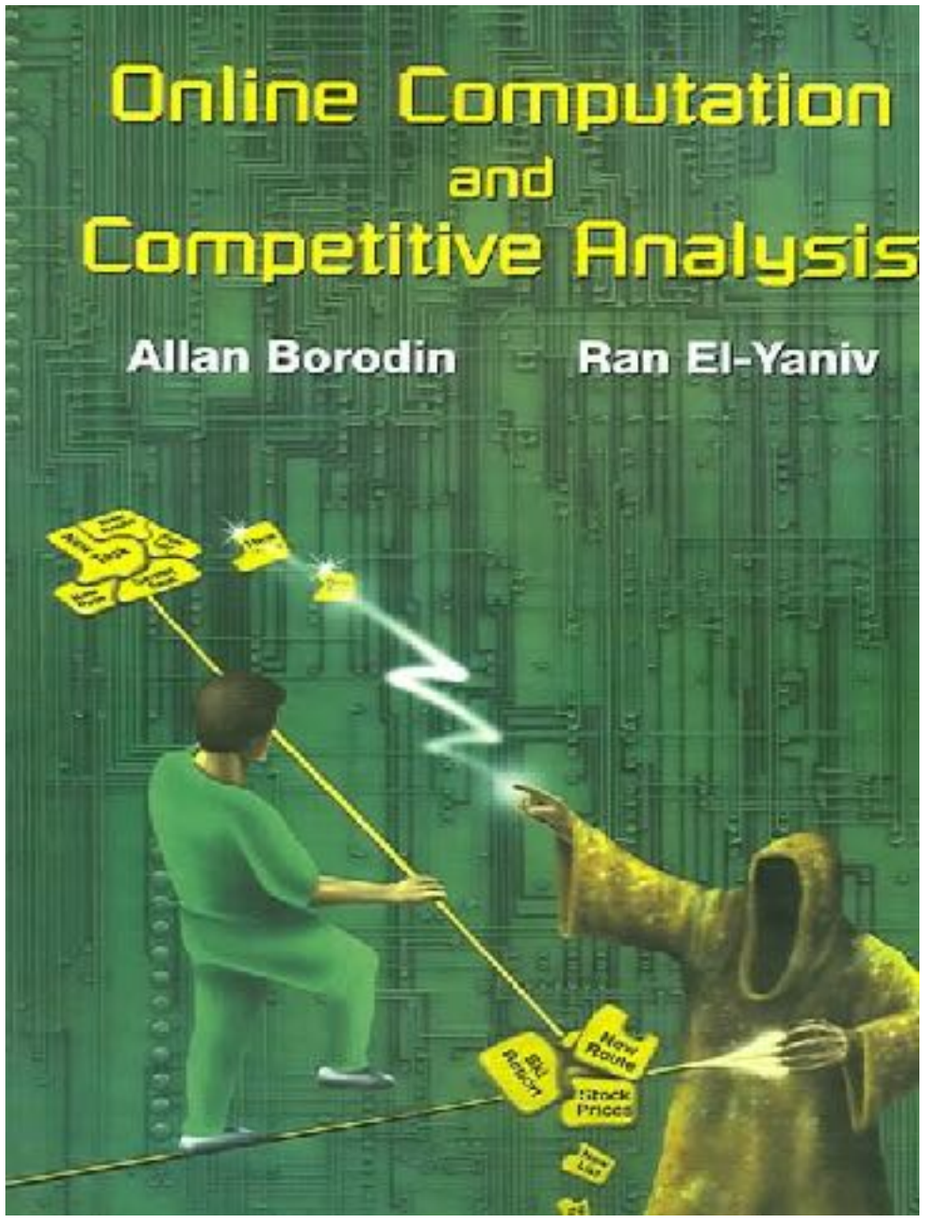


- Optimal offline policy (by tricky exchange argument): [Belady'66]
  LONGEST-FORWARD-DISTANCE:
  evict page whose next request is furthest in the future.

- Online setting: decision has to be done without knowledge of future requests. (see previous talk, *decision making under lack of knowledge*)

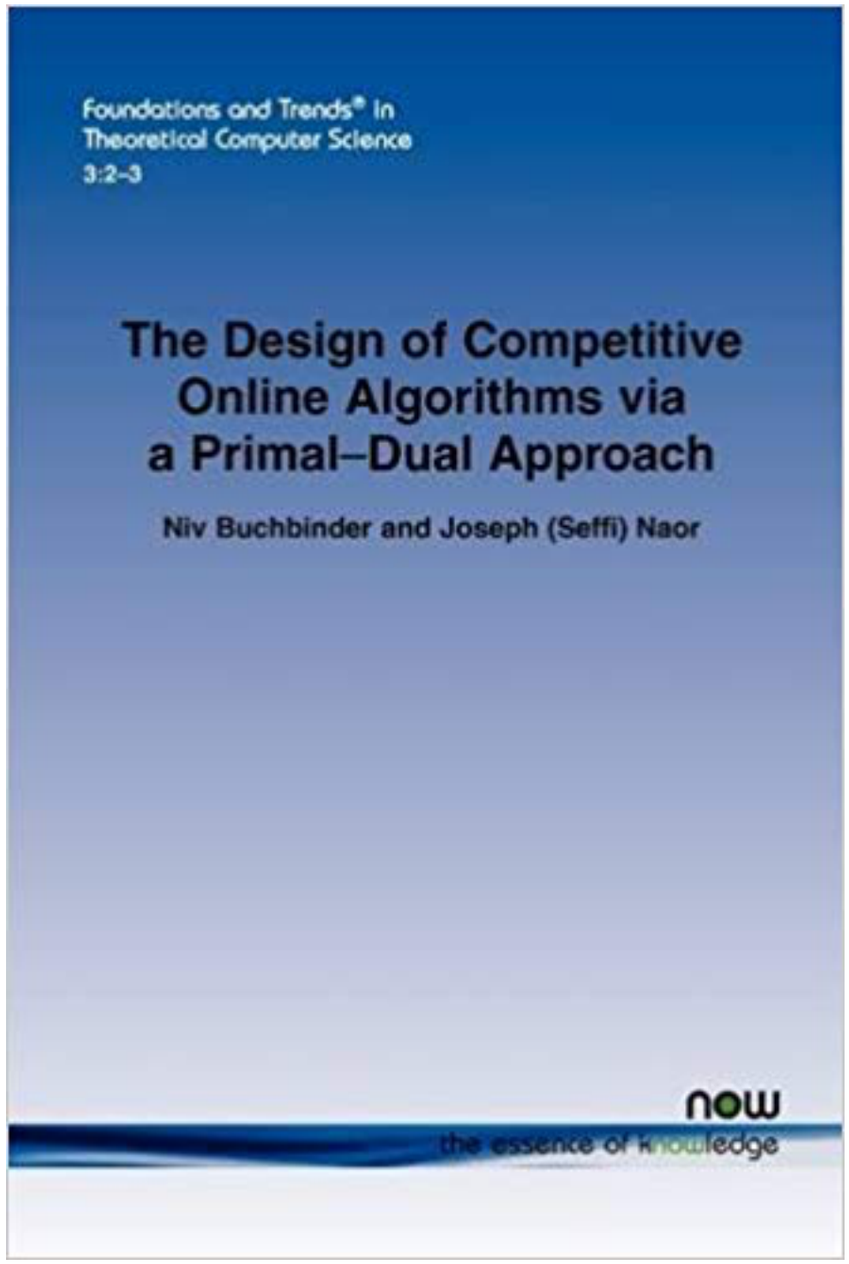
- Typical measure of performance: Competitive ratio of ALG
  := $\sup_\sigma$ ALG(σ) / OPT(σ) sometimes an additive constant is allowed
  = price of not knowing the future

(See it as a game



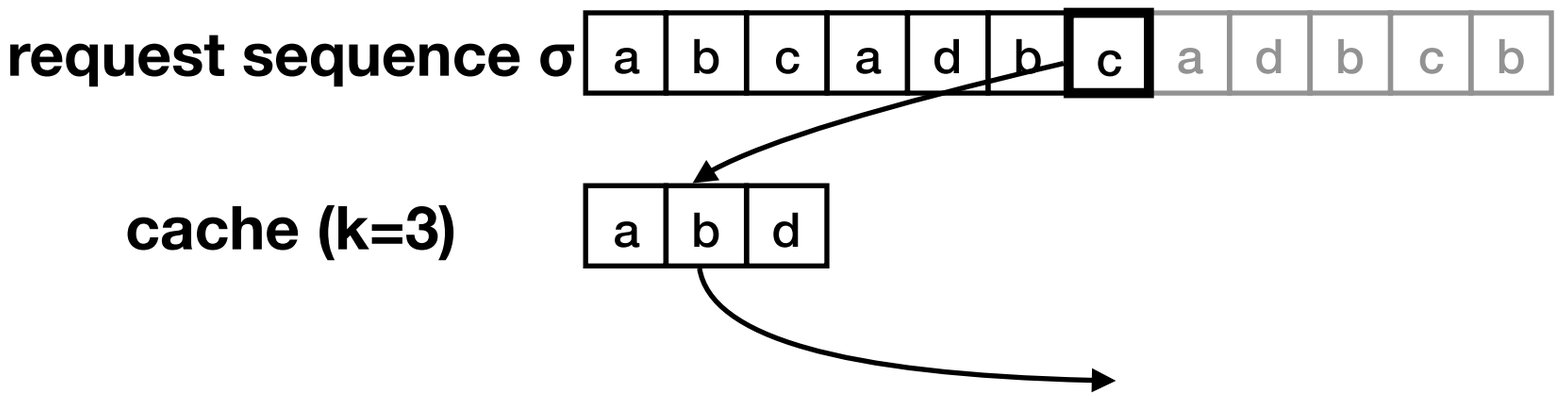
- Played between an algorithm and an adversary.
- The adversary can choose the next page. He wants to maximize the ratio.
- The algorithm has to decide which page to evict. He wants to minimize the ratio.

A more recent book)



- Unifying approach using linear programming.
- Model problem as an LP.
- Adversary generates constraints.
- Algorithm needs to increase variables to satisfy them.
- see Tutorial by Alexander Kononov (Tue, room B, 15:40)

Some online policies



- LEAST-RECENTLY-USED (LRU):
  evict page whose latest request is furthest in the past.

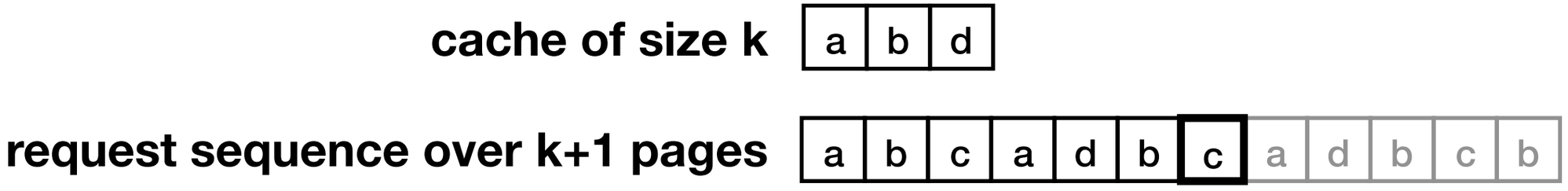
- FIRST-IN-FIRST-OUT (FIFO):
  evict page which entered the cache the earliest.

- FLUSH-WHEN-FULL (FWF):
  evict all pages when cache is full.

- LEAST-FREQUENTLY-USED (LFU):
  evict page which smallest number of requests so far.

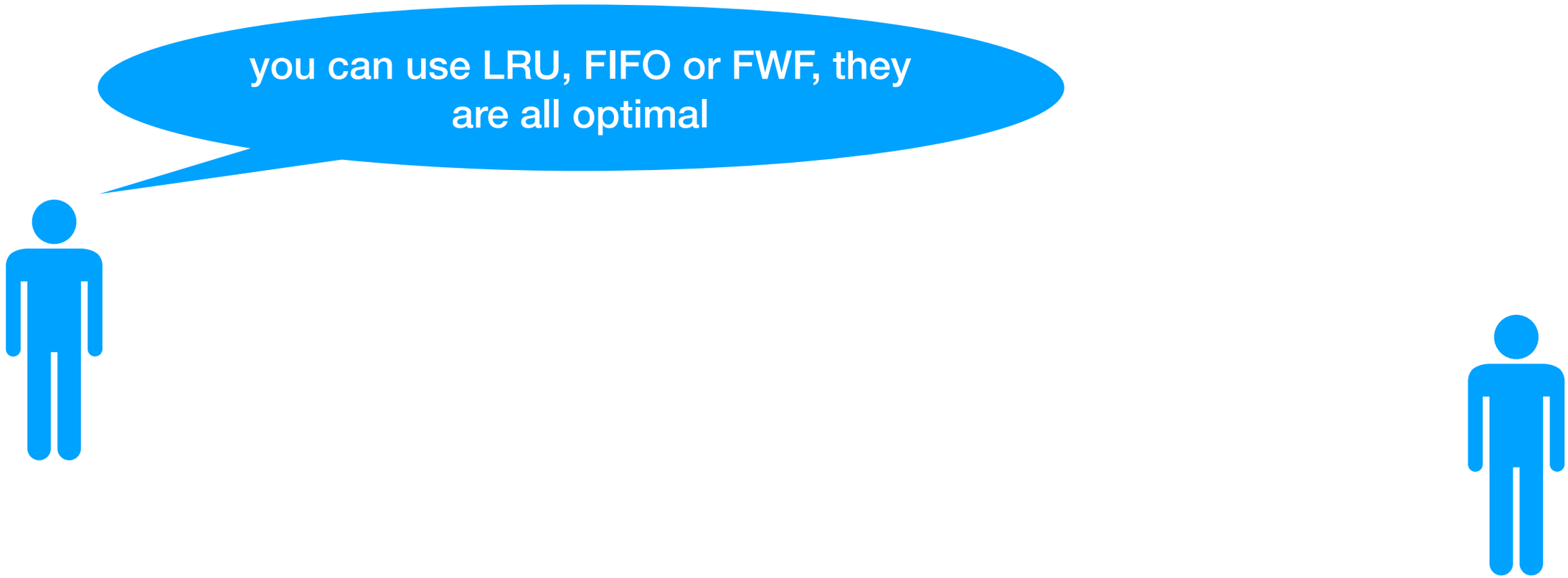
- These policies have all competitive ratio k, which is optimal for deterministic online policies.
  Except LFU, which has unbounded ratio.

Easy lower bound

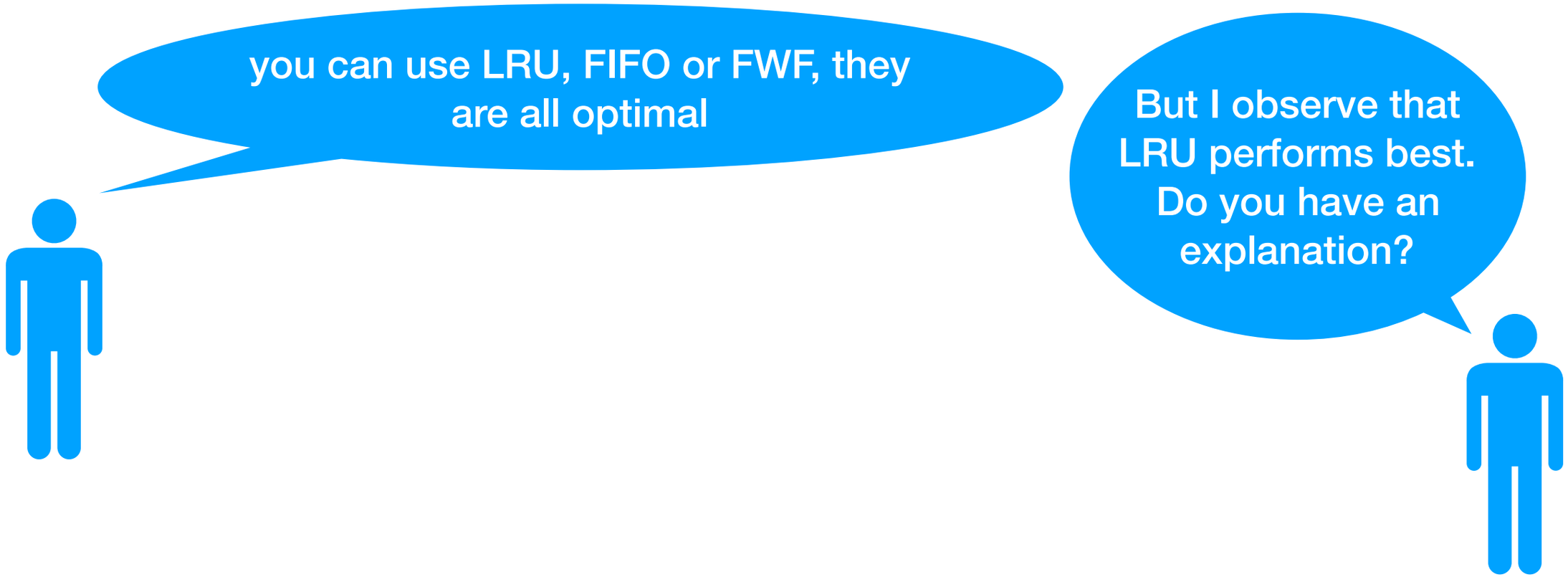


- Fix arbitrary deterministic online algorithm.
- Adversary always requests *the* page which is not in cache.
- Algorithm makes a page miss every request.
- page misses of *Longest-Forward-Distance* are distant at least k.

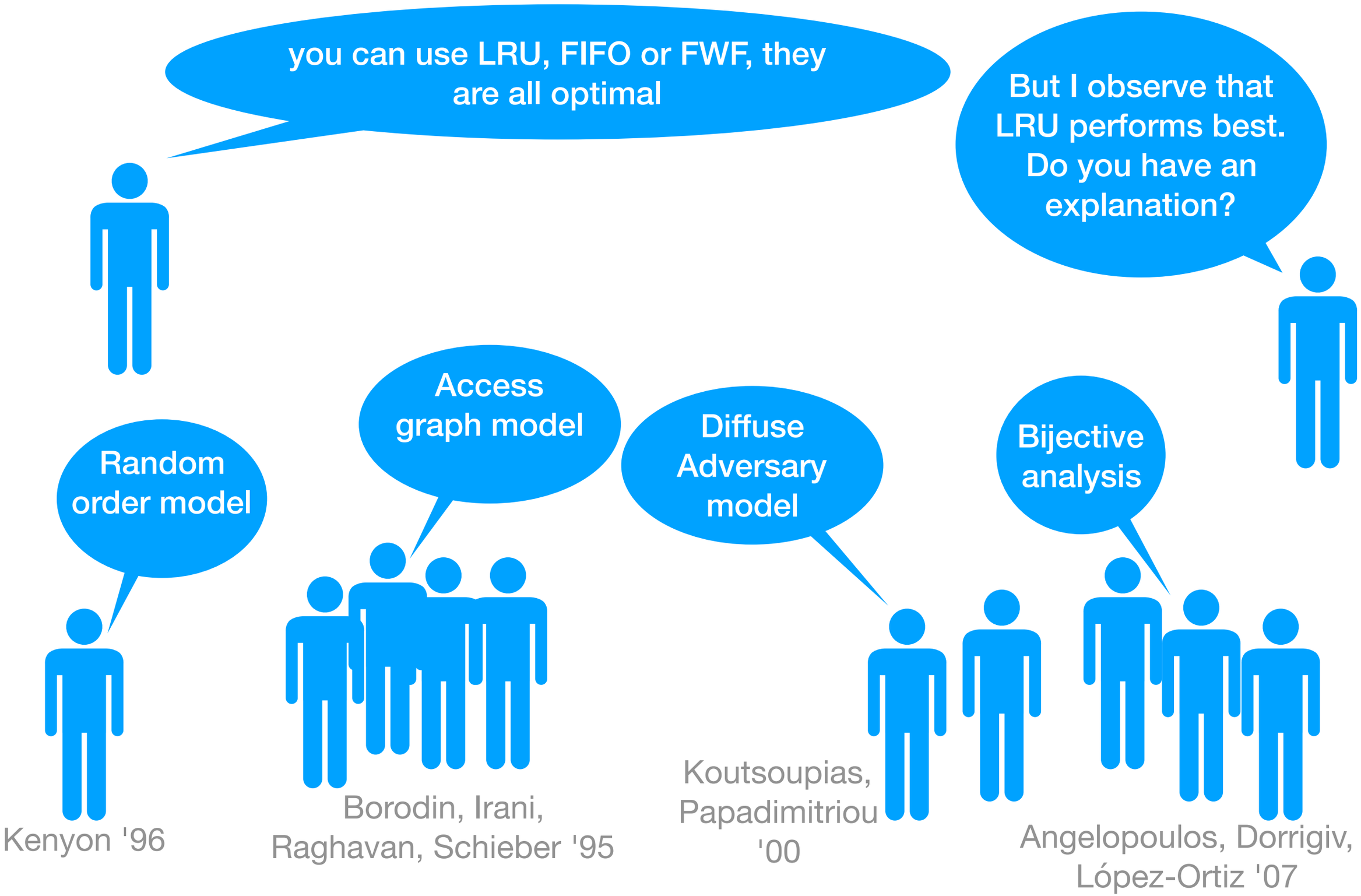
Alternative measures



Alternative measures



Alternative measures



Many models have been proposed

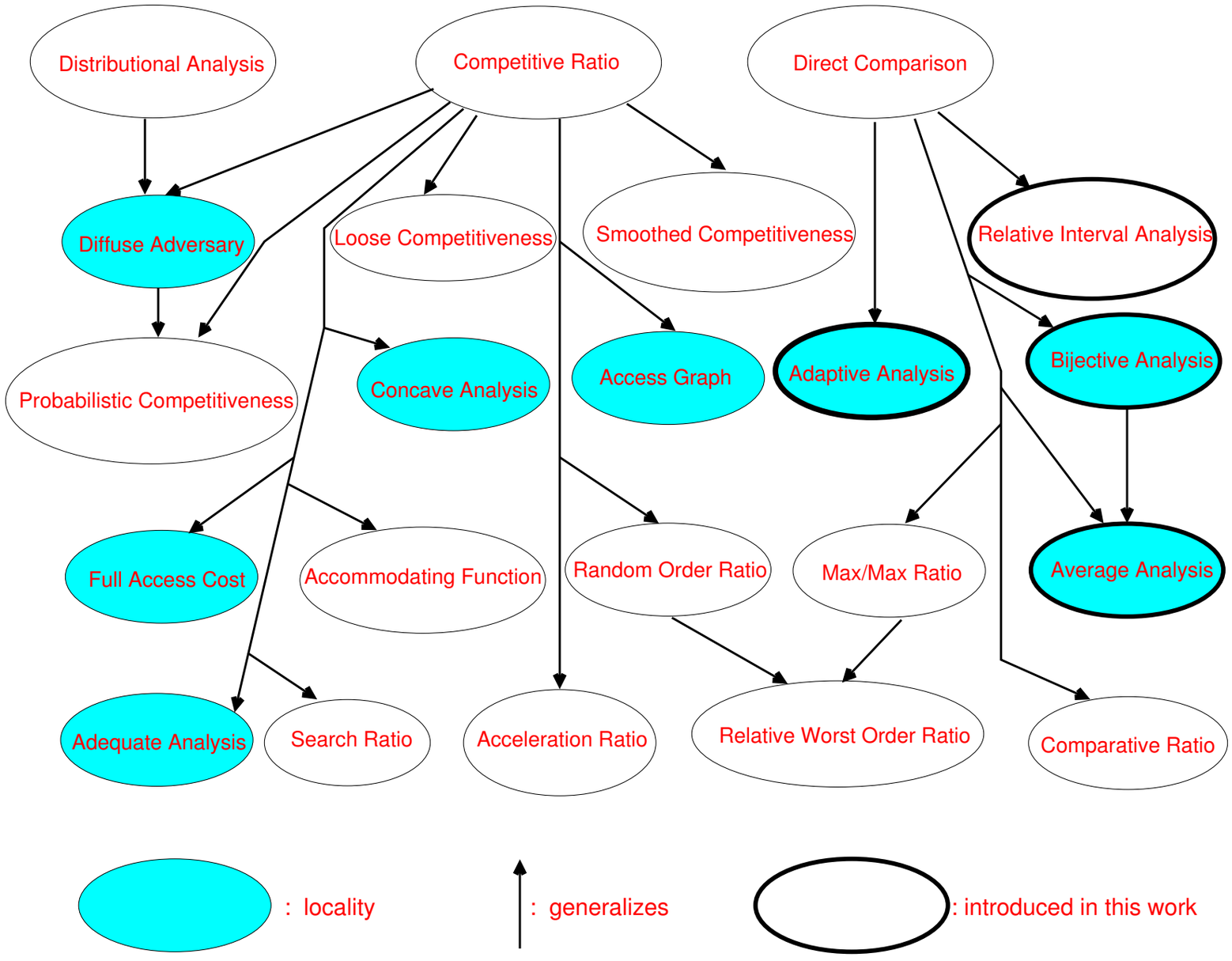
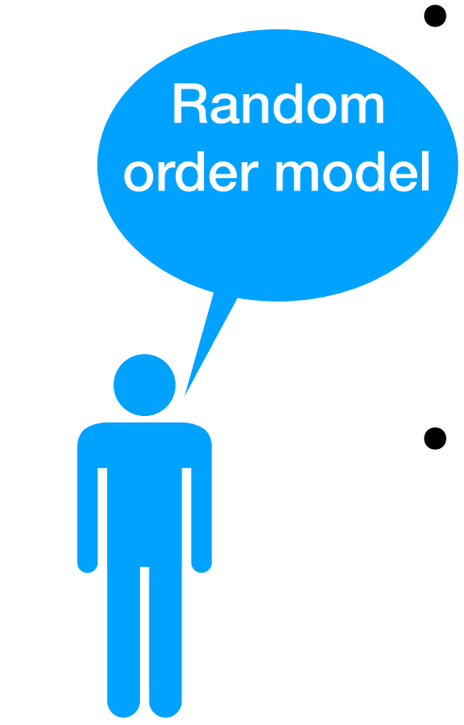


Figure 3.1: Relationships among different alternative measures.

Random order model

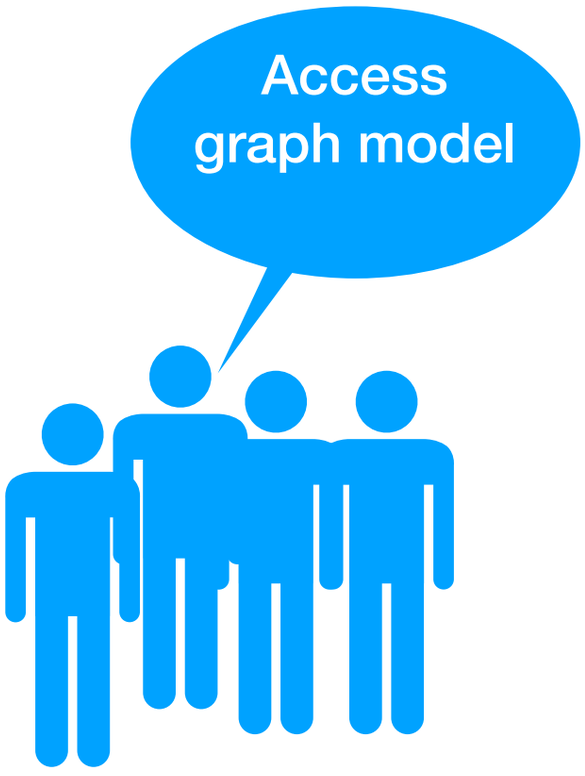
- Adversary chooses the requests
- Which are presented to the algorithm in uniform random order
- Was used to analyse the performance of Best-Fit for the Bin packing problem, and explain its good behavior.



Kenyon '96

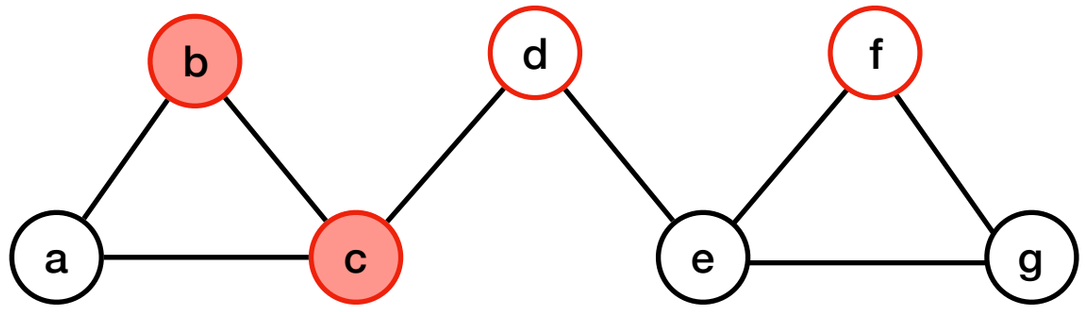
- Ratio := Expected cost of algorithm is compared with the optimal solution using the optimal order
- Model makes sense if requests come from uncoordinated sources.

Access graph model

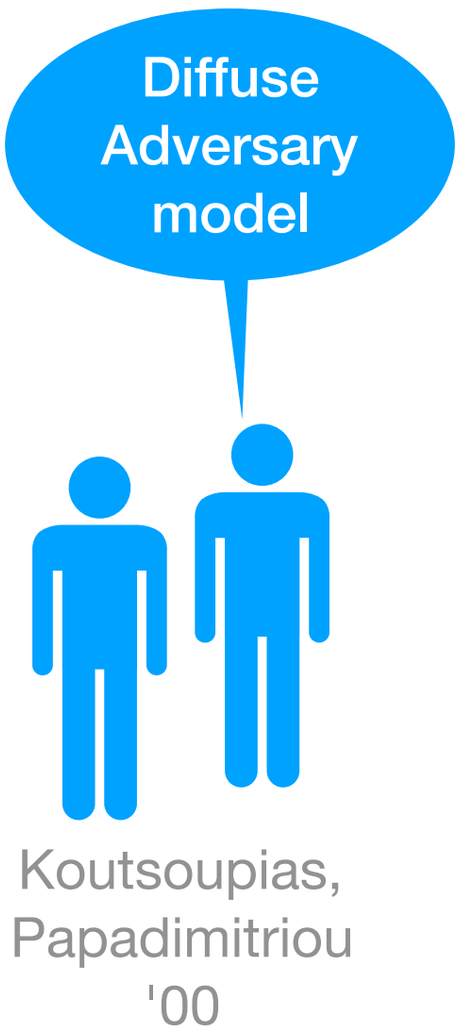


Borodin, Irani, Raghavan, Schieber '95

- Pages are vertices of a graph G. Encodes locality of reference.
- Adversary walks on the graph
  → request sequence
- Borodin et al. can compute a lower bound on competitive ratio given G.
- And propose algorithm FAR: page hit? mark the page. page miss? evict unmarked page furthest from the marked pages.
- Irani, Karlin, Phillips '96 show algorithm is optimal up to a multiplicative constant



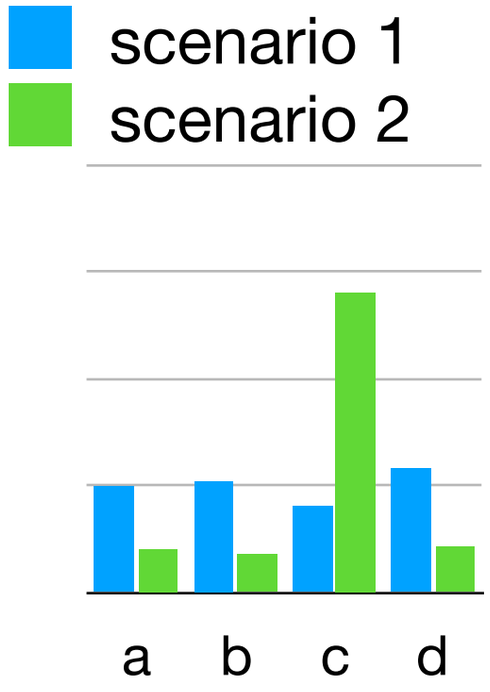
Diffuse adversary model



Koutsoupias, Papadimitriou '00

- There is know class of distributions.
- Adversary chooses a distribution.
- Requests are generated from this distribution.
- Ratio := expected cost of algorithm over optimal cost
- Results on all distributions with probabilities ≤ ε

- LRU is optimal.
- Young'00 shows
  - For $\varepsilon<1/k$, ratio is $\Theta(1)$
  - For $\varepsilon\simeq1/k$, ratio is $\Theta(\ln k)$
  - For $\varepsilon>1/k$, ratio is $\Theta(k)$



Bijective analysis

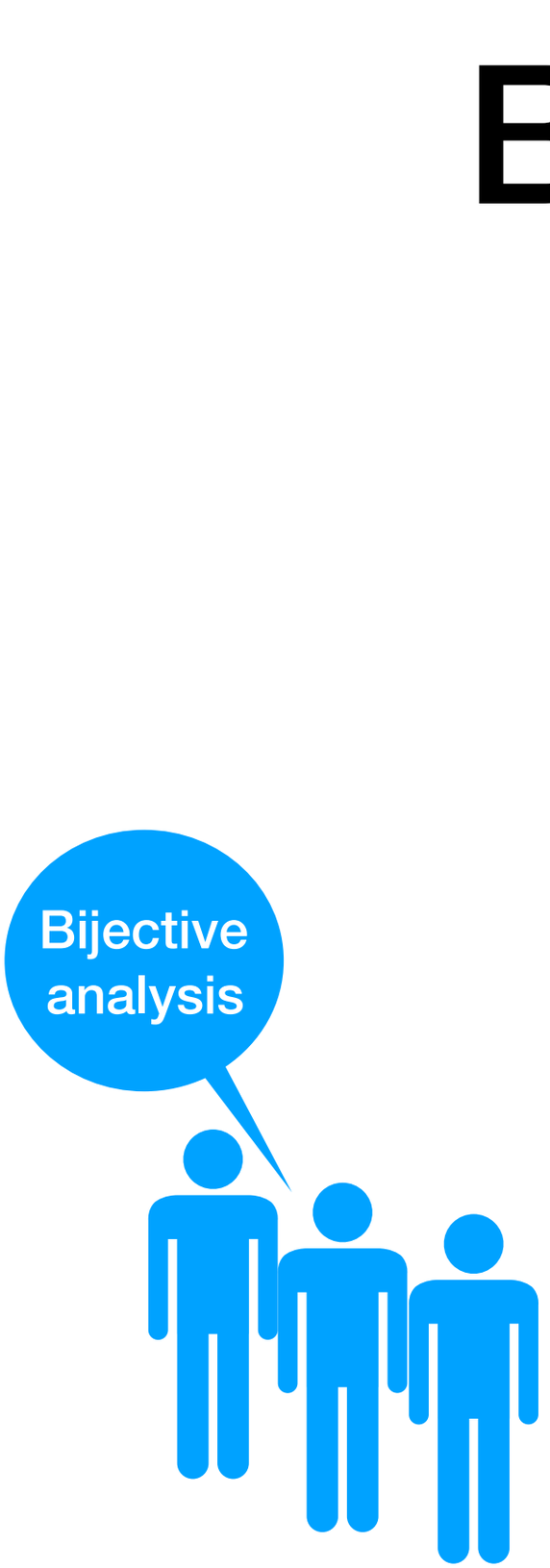
- Consider all instances of size n (#requests)

- Instead of comparing algorithm A with OPT, and B with OPT, then comparing the ratios,

  we could compare directly A with B.

- Problem: may be incomparable. A is good on some instances, B on others.

σ	A(σ)	B(σ)	OPT(σ)
σ_1	5	6	3
σ_2	7	10	4
...	...	...	...
σ_{m^n}	8	7	4



Angelopoulos, Dorrigiv, López-Ortiz '07

Bijective analysis

- Two approaches:

- **Average analysis**:
  $A \preccurlyeq_a B$ if
  $\exists\ n_0\ \ \forall\ n \geq n_0$
  $\Sigma\sigma \in I_n\ A(\sigma) \leq$
  $\Sigma\sigma \in I_n\ B(\sigma)$

Angelopoulos, Dorrigiv, López-Ortiz '07

I_n

σ	A(σ)	B(σ)
σ_1	5	6
σ_2	7	10
...	...	...
σ_{m^n}	8	7
	130	**142**

Bijective analysis

- Two approaches:

- **Average analysis**:
  $A \preccurlyeq_a B$ if
  $\exists\ n_0\ \ \forall\ n \geq n_0$
  $\Sigma\sigma \in I_n\ A(\sigma) \leq$
  $\Sigma\sigma \in I_n\ B(\sigma)$

- **Bijective analysis**:
  $A \preccurlyeq_b B$ if
  $\exists\ n_0\ \ \forall\ n \geq n_0$
  $\exists$ bijection b
  $\forall\ \sigma \in I_n\ A(\sigma) \leq B(b(\sigma))$

Angelopoulos, Dorrigiv, López-Ortiz '07

σ	A(σ)		B(b(σ))
σ_1	5	≤	6
σ_2	7	≤	7
...	...		... ...
σ_{m^n}	8	≤	10

I_n

Bijective analysis

- **Bijective analysis**:
  $A \preccurlyeq_b B$ if $\exists\ n_0\ \ \forall\ n \geq n_0\ \exists$ bijection $b\ \forall\ \sigma \in I_n\ A(\sigma) \leq B(b(\sigma))$

- **Results:**
  LRU $\prec_b$ FWF. All lazy algorithms are equivalent $\equiv_b$

Angelopoulos, Dorrigiv, López-Ortiz '07

- **Definition:** Algorithm is *lazy* if it evicts only one page per page fault.

LFU
(LEAST-FREQUENTLY-USED)

FWF
(FLUSH-WHEN-FULL)

FIFO
(FIRST-IN-FIRST-OUT)

LRU
(LEAST-RECENTLY-USED)

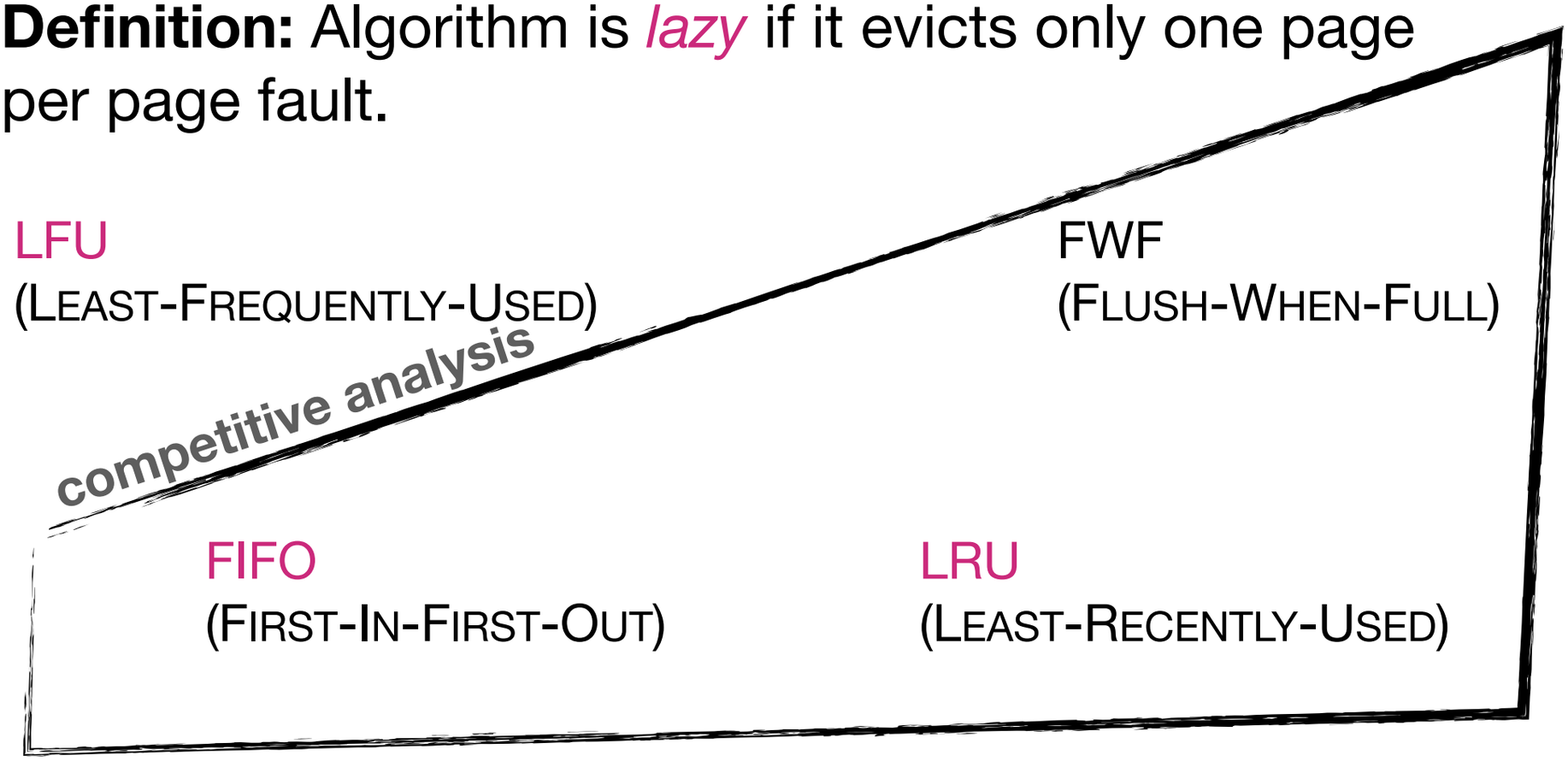
Bijective analysis

- **Bijective analysis**:
  $A \preccurlyeq_b B$ if $\exists\ n_0\ \ \forall\ n \geq n_0\ \exists$ bijection $b\ \forall\ \sigma \in I_n\ A(\sigma) \leq B(b(\sigma))$

- **Results:**
  LRU $\prec_b$ FWF. All lazy algorithms are equivalent $\equiv_b$

- **Definition:** Algorithm is *lazy* if it evicts only one page per page fault.

Angelopoulos, Dorrigiv, López-Ortiz '07



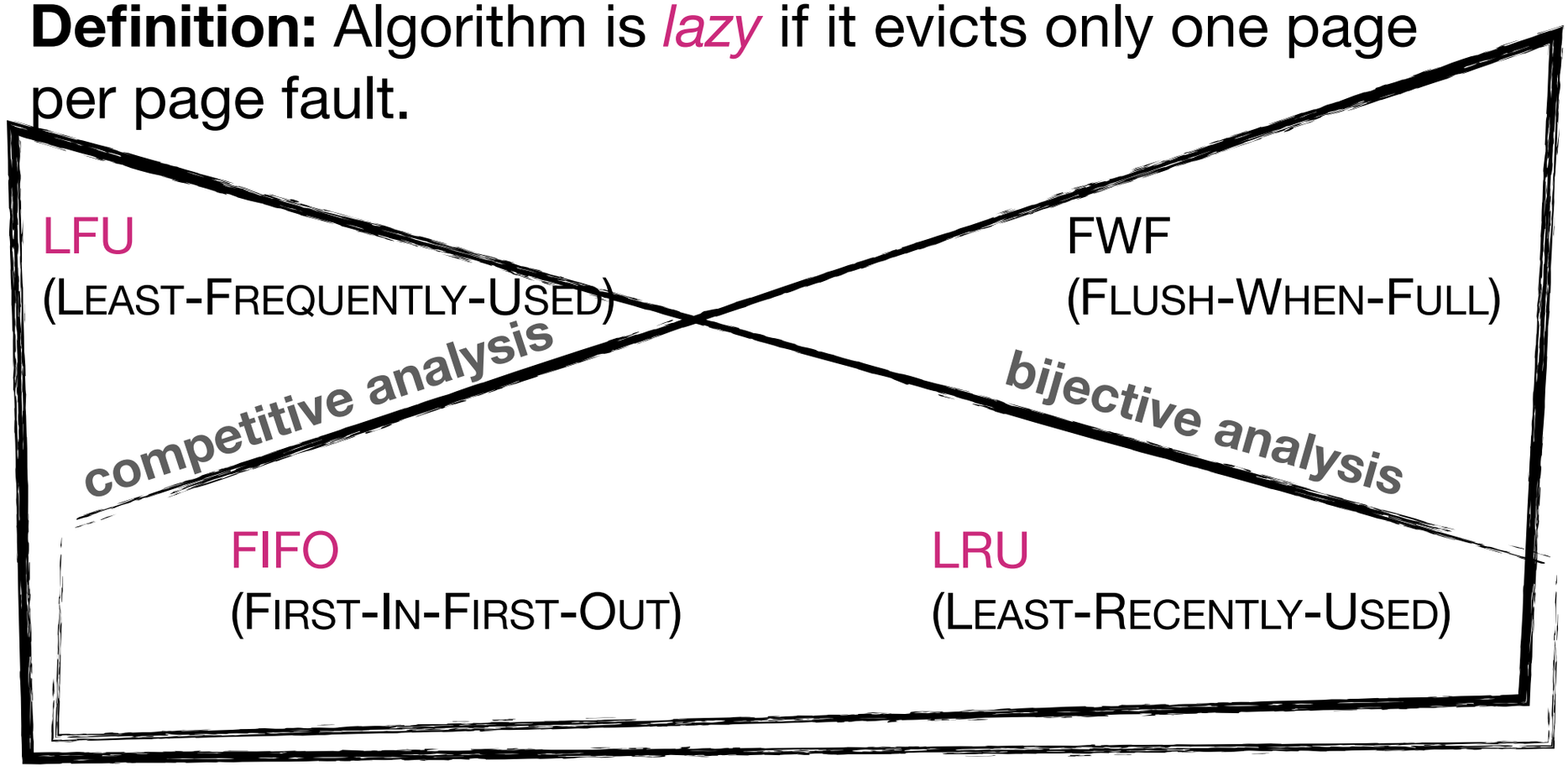
Bijective analysis

- **Bijective analysis**:
  $A \preccurlyeq_b B$ if $\exists\ n_0\ \ \forall\ n \geq n_0\ \exists$ bijection $b\ \forall\ \sigma \in I_n\ A(\sigma) \leq B(b(\sigma))$

- **Results:**
  LRU $\prec_b$ FWF. All lazy algorithms are equivalent $\equiv_b$

Angelopoulos, Dorrigiv, López-Ortiz '07

- **Definition:** Algorithm is *lazy* if it evicts only one page per page fault.



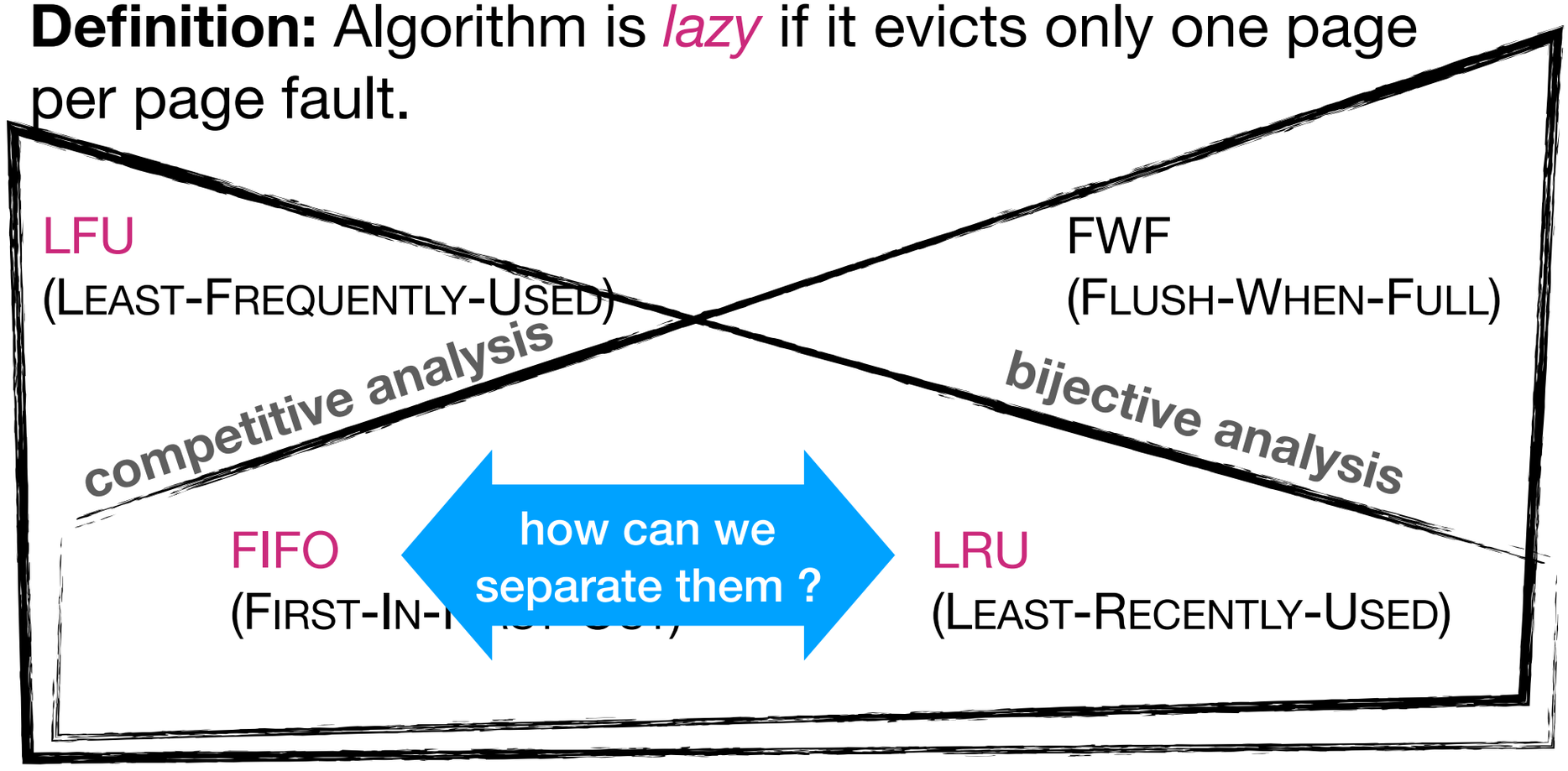
Bijective analysis

- **Bijective analysis**:
  $A \preccurlyeq_b B$ if $\exists\ n_0\ \ \forall\ n \geq n_0\ \exists$ bijection $b\ \forall\ \sigma \in I_n\ A(\sigma) \leq B(b(\sigma))$

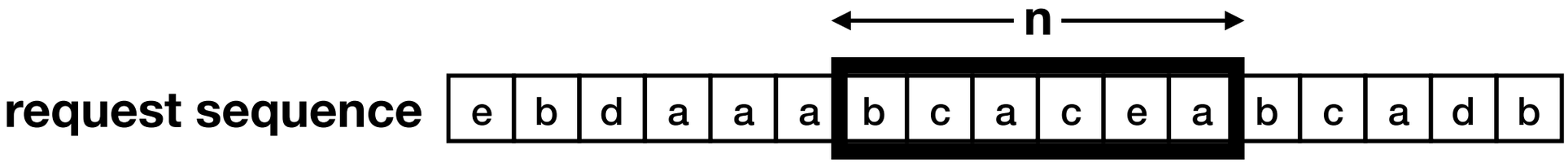
- **Results:**
  LRU $\prec_b$ FWF. All lazy algorithms are equivalent $\equiv_b$

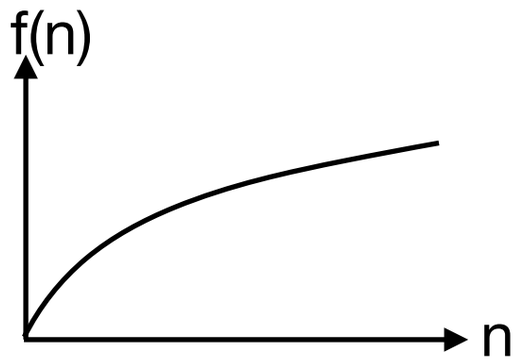
Angelopoulos, Dorrigiv, López-Ortiz '07

- **Definition:** Algorithm is *lazy* if it evicts only one page per page fault.



Locality of reference





- For a fixed convex fonction f
  (f(1)=1, f(n+1)-f(n)≥f(n+2)-f(n-1))
- sequence is *compatible with f* if for every window of size n, number of distinct pages ≤ f(n).
  I_n^f := all sequences of n requests compatible with f
  Albers, Favrholdt, Giel '05
- Domination: $A \preccurlyeq_a^f B$ if $\exists\ n_0\ \ \forall\ n \geq n_0$
  $\Sigma\sigma\in I_n^f A(\sigma) \leq \Sigma\sigma\in I_n^f B(\sigma)$

- **Results:**
- For any algorithm A:
  $LRU \preccurlyeq_a^{f`} A$
- No algorithm A (≠LRU) dominates
  $A \preccurlyeq_a^{f`} LRU$

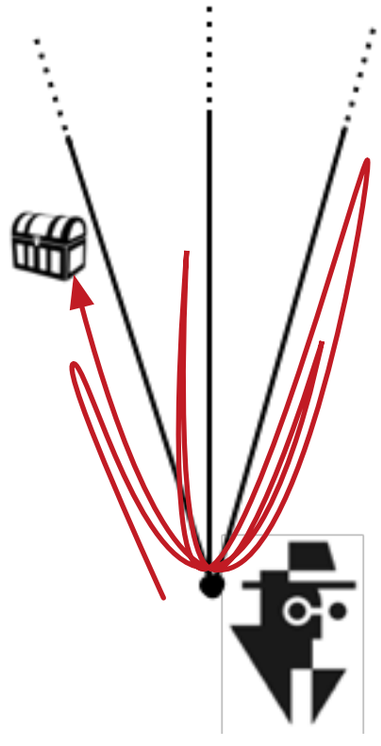
Outline

- Models for analyzing online algorithms
- The linear search problem

Online search

- Search space with an origin
- **Immobile hider**:
  at unknown position,
  (technical assumption:
  distance at least 1 from origin)
- **Mobile searcher**:
  initially at origin. Many cost models (turning cost, exploring known area is free)
- Question:
  how to **measure efficiency** of a search strategy ?

- This is a problem with hidden information, even though not an online problem. Techniques are similar.

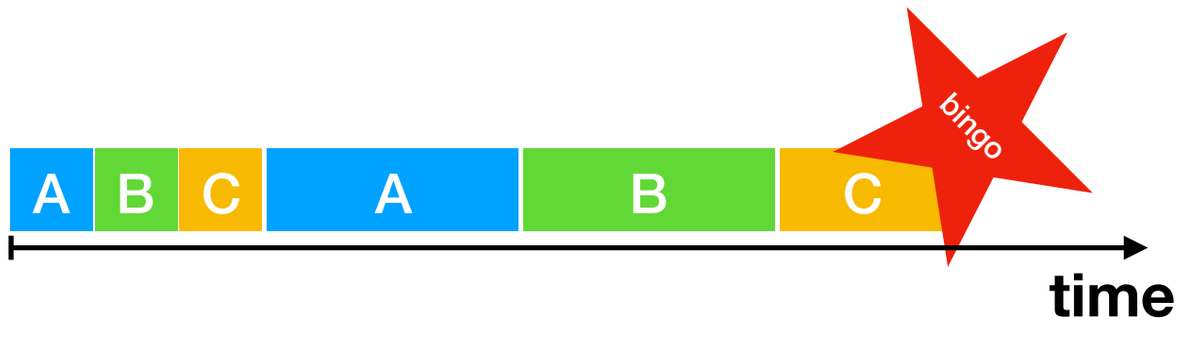


Application: demining

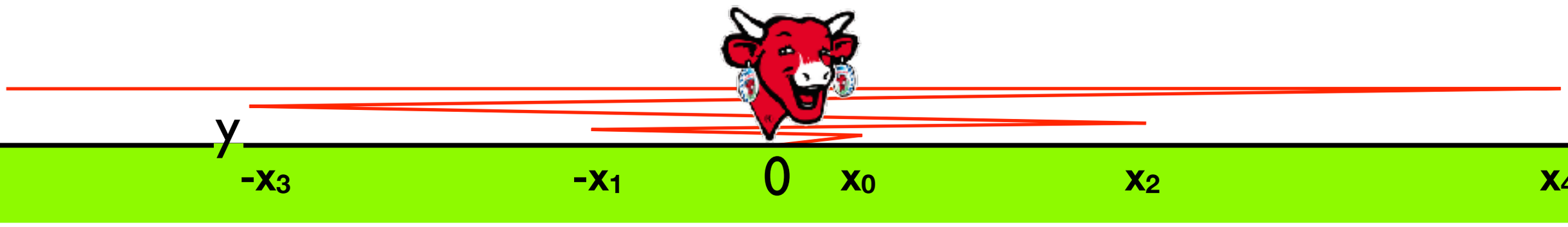
- You know: A single mine is set in some crossing of a road network
- You don’t know: where
- Demining a road segment takes some time
- Walking along demined roads takes no time
- In what order should you proceed ?

Application: scheduling heuristics

- You are given a problem instance I
- and a fixed number of heuristics for this problem, with the promise that one will eventually succeed on I.
But you don't know which one, and after how many CPU cycles.

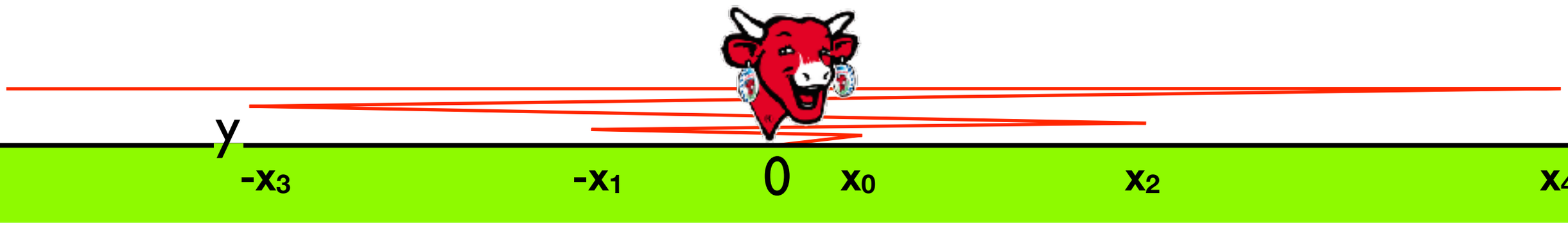


Considered variant: cow path problem:



- there is juicy grass on the other side of the fence
- cow is at position 0,
  fence has an opening at position y with $|y| \geq 1$
  (sign(y) is unknown to the cow)
- strategy: $1 \leq x_0, x_1, x_2, x_3, \ldots$ with $x_i \geq x_{i-2}$
- **Standard measure**:
  competitive ratio := distance walked by cow / |y|

Considered variant: cow path problem:



- strategy: $1 \leq x_0, x_1, x_2, x_3, \ldots$ with $x_i \geq x_{i-2}$
- worst case: $y = x_n + \varepsilon$ for some n
- distance walked by cow = $2(x_0+\ldots+x_{n+1})+x_n+\varepsilon$
- *competitive ratio* is at least 9
  doubling strategy ($x_i=2^i$) achieves 9

Problem has long history

Competitive ratio

- Initially proposed by Bellman and Beck (1963) in a Bayesian context
- First solved by Beck and Newman : optimal competitive ratio of 9
- Many papers by Beck in 70’s, 80’s, 90’s
- [Gal 72, 74] : solution to the m-ray problem
- Rediscovered in [Baeza-Yates et al, 96]
- Many variants :
    - Multiple searchers [Lopez-Ortiz and Schuierer 02]
    - Turn cost [Demaine et al 04], [Angelopoulos et al 17]
    - Randomized strategies [Kao et al 96]
    - Probabilistic search [Jaillet and Stafford 01]
    - Weighted targets [Angelopoulos and Panagiotou 16]
    - Distance bounds [Bose et al 13] .... and many others

Set of optimal strategies

Competitive ratio

- Characterization of search strategies with optimal competitive ratio

$$1 \leq x_0 \leq 4, \quad x_1 \geq 1 \quad and \quad x_n \leq 3x_{n-1} - \sum_{i=0}^{n-2} x_i, \quad for\ all\ n \geq 1.$$

doubling $x_i=2^i$

agressive $x_i=4(i+1)2^i$

Σ_9

Σ

you can use any solution to this LP,
they are all optimal

Set of optimal strategies

Competitive ratio

- Characterization of search strategies with optimal competitive ratio

$$1 \leq x_0 \leq 4, \quad x_1 \geq 1 \quad and \quad x_n \leq 3x_{n-1} - \sum_{i=0}^{n-2} x_i, \quad for\ all\ n \geq 1.$$

doubling $x_i=2^i$

agressive $x_i=4(i+1)2^i$

Σ_9

Σ

you can use any solution to this LP, they are all optimal

Can't we distinguish among them?

Set of optimal strategies

Competitive ratio

- Characterization of search strategies with optimal competitive ratio

$$1 \leq x_0 \leq 4, \quad x_1 \geq 1 \quad and \quad x_n \leq 3x_{n-1} - \sum_{i=0}^{n-2} x_i, \quad for\ all\ n \geq 1.$$

doubling $x_i=2^i$

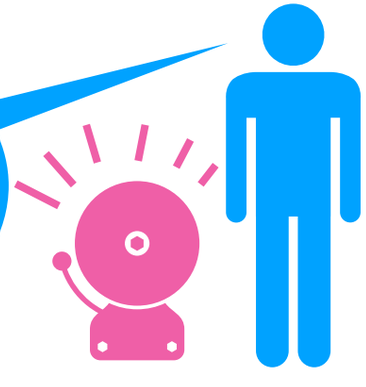
agressive $x_i=4(i+1)2^i$

Σ_9

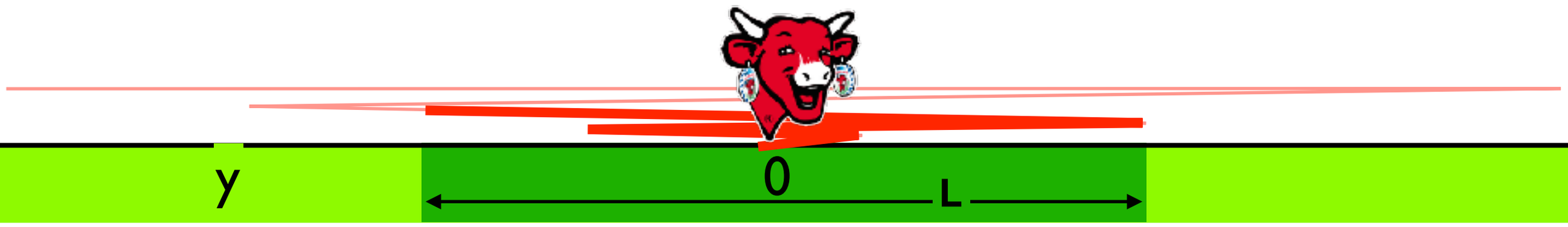
Σ

you can use any solution to this LP, they are all optimal

Can't we distinguish among them?



Our new measure



- **Discovery time**
  D(x,L) := time it takes for strategy x to discover a segment of length L

- Intuitively we prefer strategies with *small* discovery times. Inversely they *explore more* in the same time.

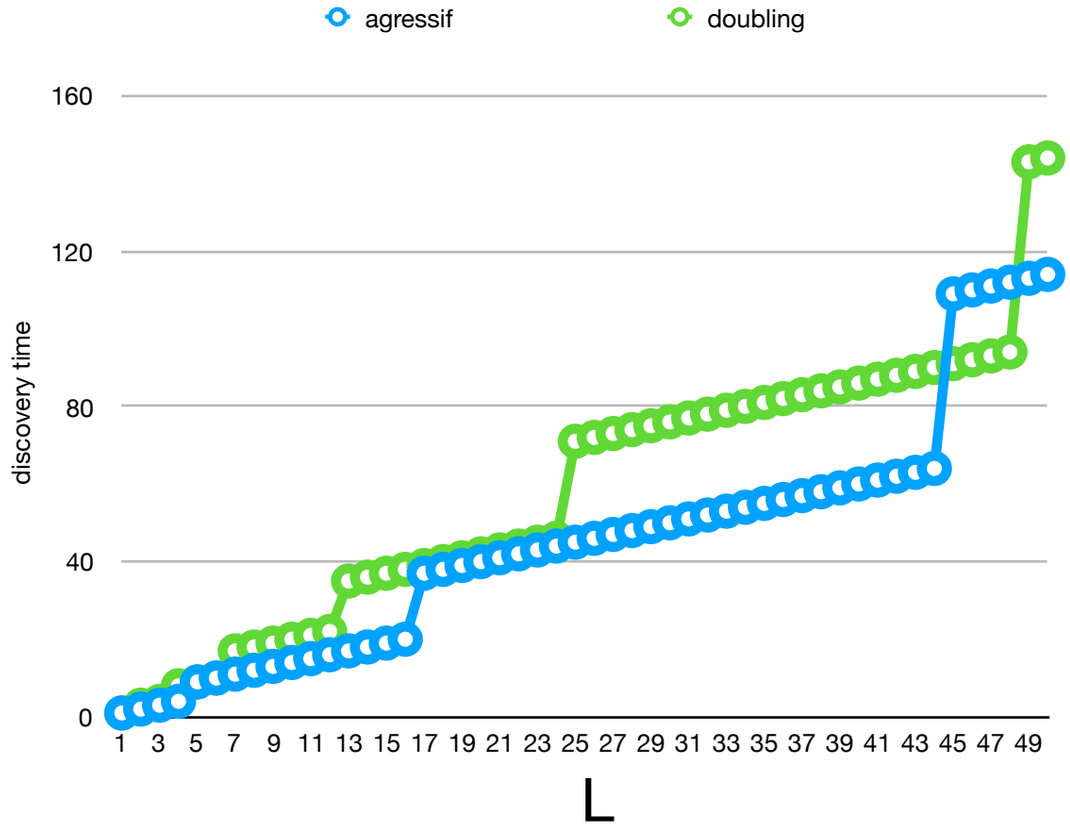
- Should help us to distinguish among optimal strategies with respect to the competitive ratio

Σ_9

How to compare two strategies under this measure ?

- In this example *agressif* seems to dominate *doubling*, but not for all lengths L.

- What happens for L∈[44,48], is it seldom?



No

- d*(L) is the optimal discovery time among strategies in Σ_9. It is reached by the strategy $x_i=t(i+1)2^i$ for some optimized t.

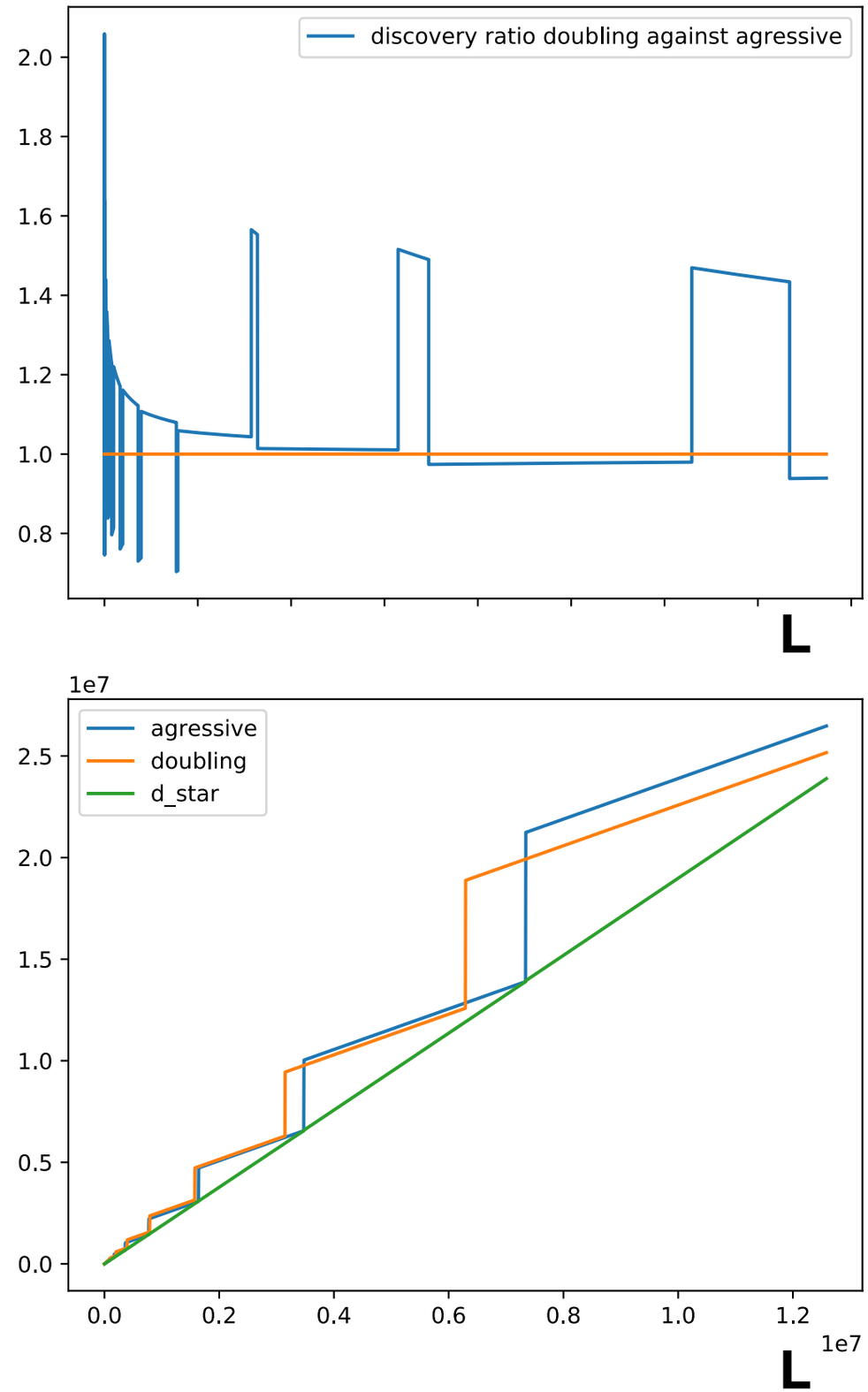
- There is no strategy in Σ_9 that is optimal for *all* L.

- Denote D(x,L) := discovery time := time for x until it discovers a segment of length L

- Discovery ratio of a given strategy x against y is the ratio D(x,L) / D(y,L) maximized over L

- **Discovery ratio DR(x,S)** of a given strategy x is the ratio of x against y, maximized over all strategies $y \in S$.

- where S=set of strategies, either all (Σ), or only strategies with optimal competitive ratio (Σ_9)

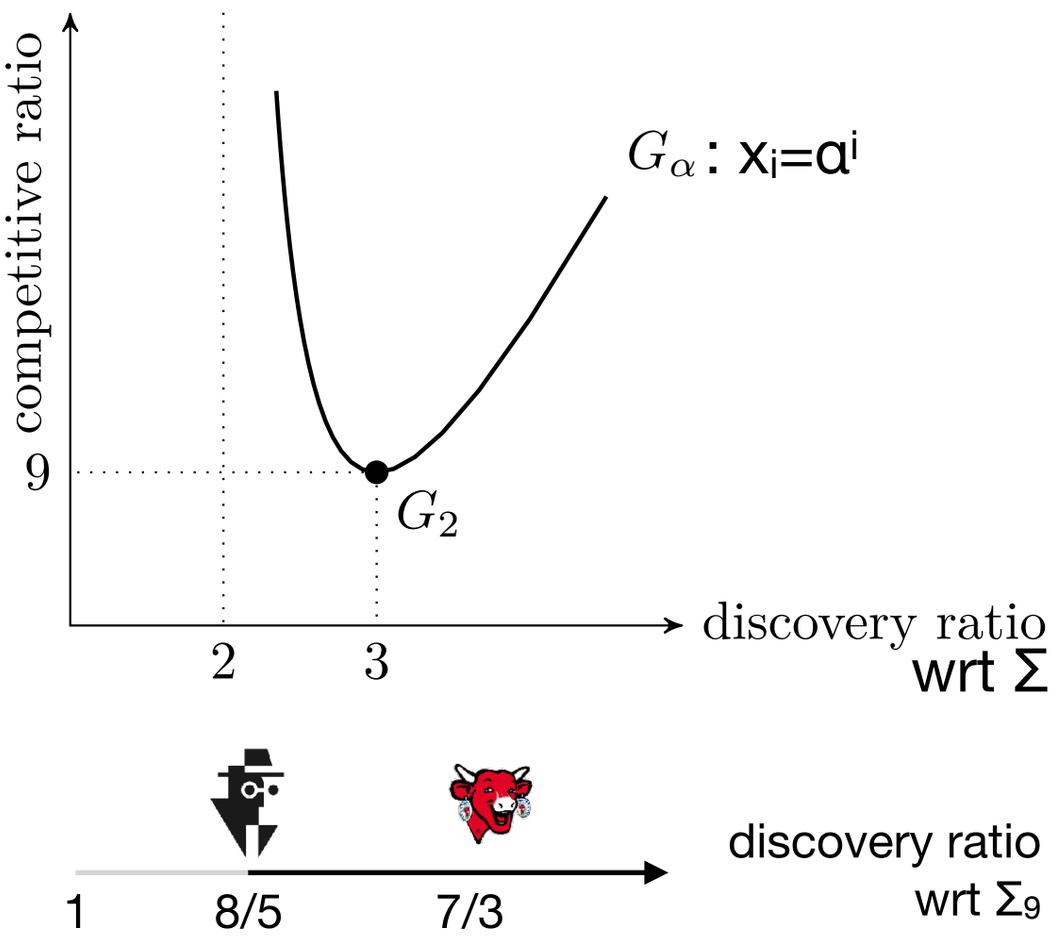


Our results

- **Discovery ratio DR(x,S)** of a given strategy x is the ratio of x against y, maximized over all strategies y∈S.

- **Results**:
  DR(x,Σ_9) ≥ 8/5 for all x∈Σ_9
  DR(aggressive,Σ_9) = 8/5
  DR(doubling,Σ_9)=7/3

- If DR(x,Σ_9)=8/5, then x coincides with agressive on the first 5 steps, but not always on the 6th



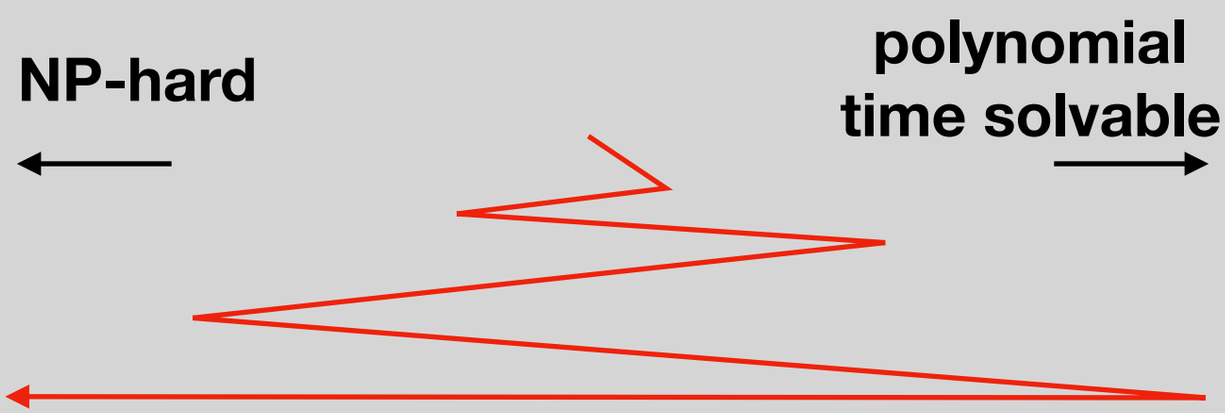
Summary

- Competitive ratio focuses on single worst case instances
- This measure is to rough in some settings
- Alternative measures permit to distinguish more algorithms
- and can describe better performance in practice

Summary

- Competitive ratio focuses on single worst case instances
- This measure is to rough in some settings
- Alternative measures permit to distinguish more algorithms
- and can describe better performance in practice

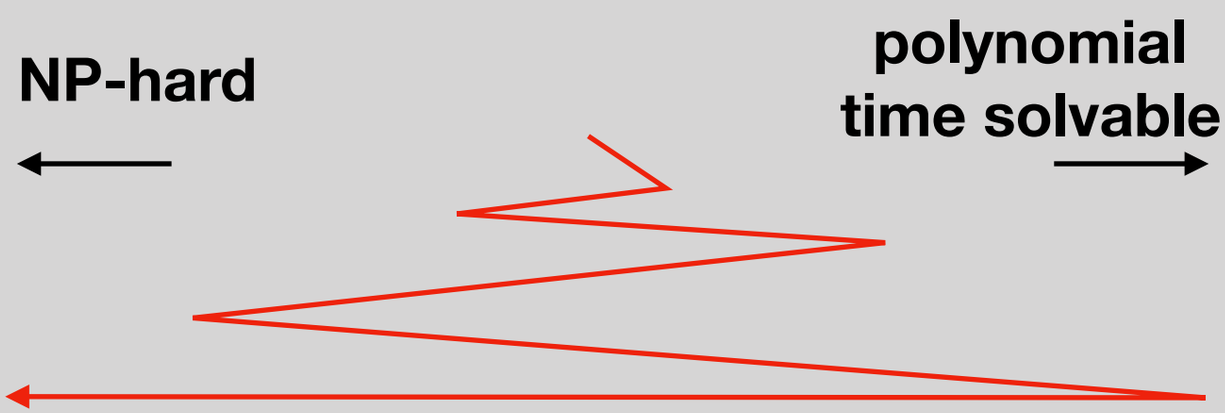
Suggested research agenda for a new problem



Summary

- Competitive ratio focuses on single worst case instances
- This measure is to rough in some settings
- Alternative measures permit to distinguish more algorithms
- and can describe better performance in practice

Suggested research agenda for a new problem



thank you