Relative Worst Order Analysis

Joan Boyar

University of Southern Denmark, Odense

Currently visiting UCI

Joint work with

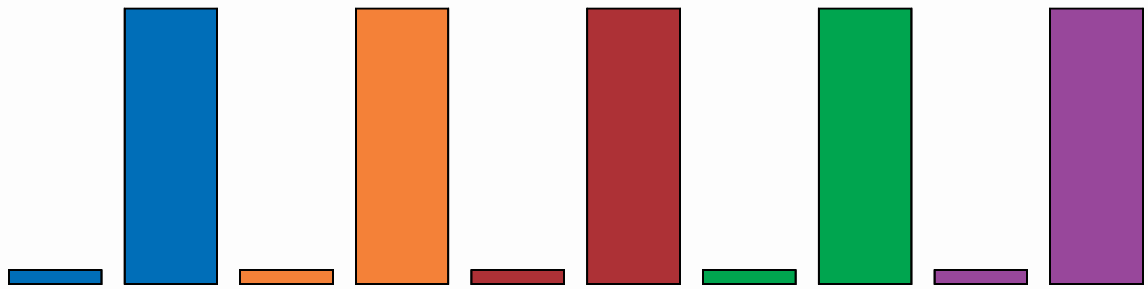
Lene M. Favrholdt

Kim S. Larsen

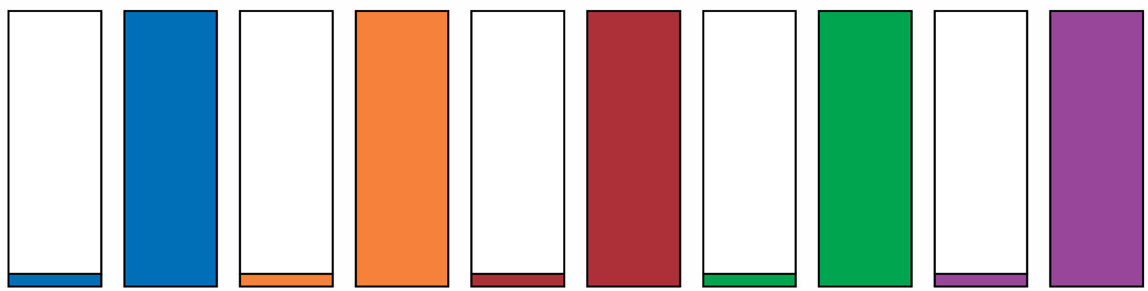
University of Southern Denmark

On-Line Bin Packing

Item sizes: $5 \times [\epsilon, 1]$
Bin size: 1



Result by Next-Fit:



Competitive Ratio

$\mathbb{A}$ is c-competitive if for any input seq. I,

$$\mathbb{A}(I) \leq c \cdot \mathsf{OPT}(I) + b.$$

optimal off-line algorithm

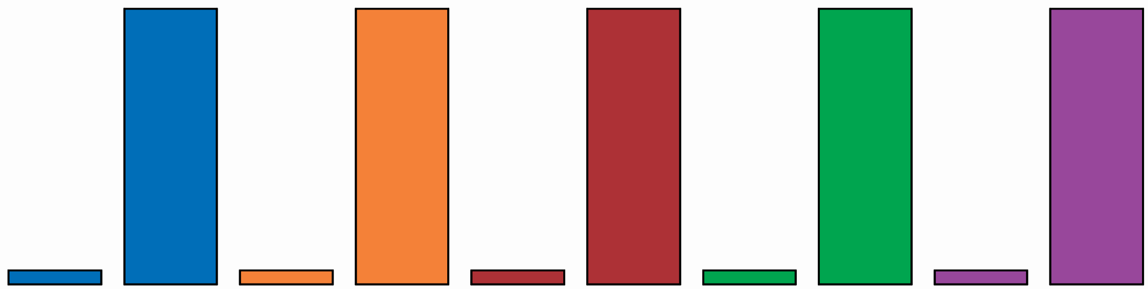
constant

The competitive ratio of $\mathbb{A}$ is

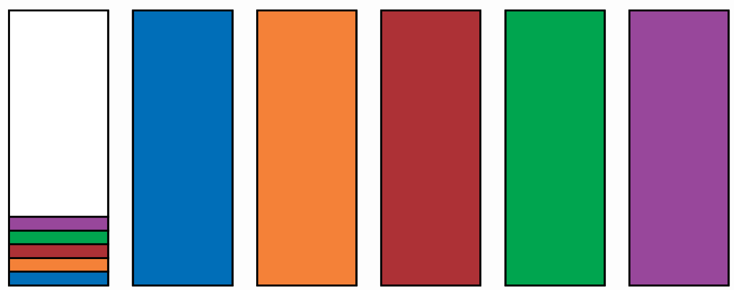
$$\mathsf{CR}_{\mathbb{A}} = \inf \{ c \mid \mathbb{A} \text{ is } c\text{-competitive} \} .$$

Compare to OPT

Item sizes: $5 \times [\epsilon, 1]$
Bin size: 1



Result by optimal off-line algorithm, OPT:



$\text{CR}_{\text{Next-Fit}} = 2$ [Johnson 1974]

Any-Fit Algorithms

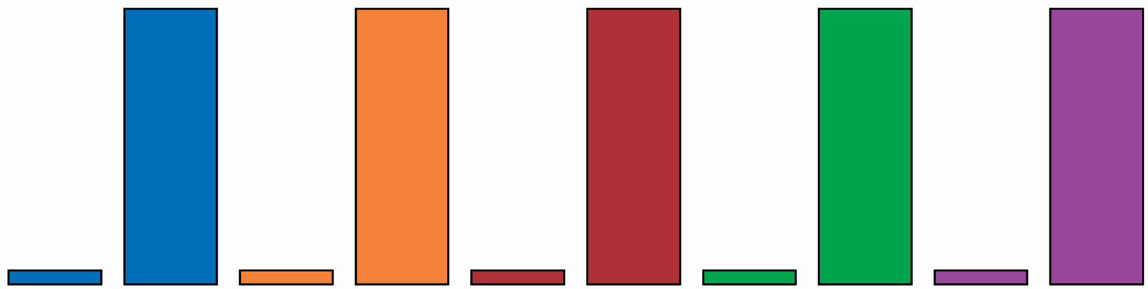
Any-Fit algorithms only open if necessary.

First-Fit: put item in first bin where it fits
Best-Fit: put item in most full bin where it fits
Worst-Fit: put item in least full bin where it fits

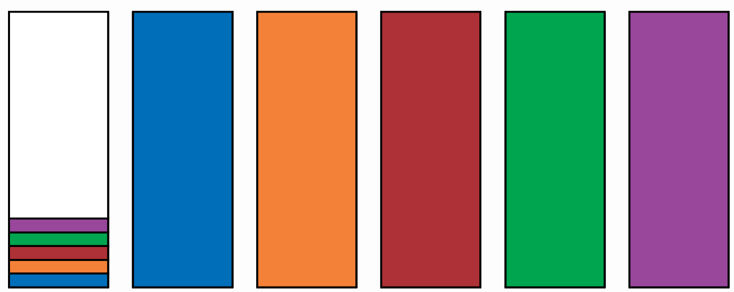
$\text{CR}_{\text{First-Fit}} = 1.7$ [Johnson, et.al. 1974]
$\text{CR}_{\text{Best-Fit}} = 1.7$ [Johnson, et.al. 1974]
$\text{CR}_{\text{Worst-Fit}} = 2$ [Johnson 1974]

Any-Fit Algorithms

Item sizes: $5 \times [\epsilon, 1]$
Bin size: 1



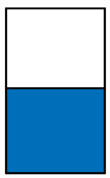
Result:



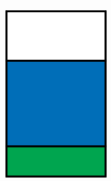
Next-Fit vs. Worst-Fit

$\text{CR}_{\text{Worst-Fit}} = \text{CR}_{\text{Next-Fit}} = 2.$

Consider any item sequence I:
Suppose Worst-Fit opens bin t now:

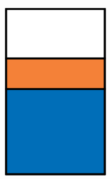


Inductively, assume Next-Fit uses bin $t' \geq t$.

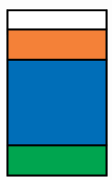


Next-Fit vs. Worst-Fit

If Worst-Fit puts more in bin t
before opening bin $t+1$:

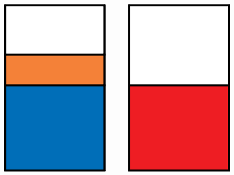


Next-Fit uses bin $t'' \geq t'$.

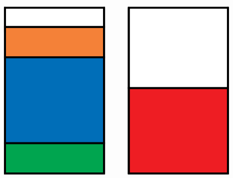


Next-Fit vs. Worst-Fit

When Worst-Fit opens bin $t+1$:

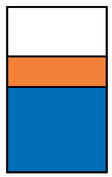


Next-Fit uses bin $t''' \geq t' + 1$.

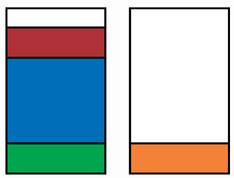


Next-Fit vs. Worst-Fit

If Worst-Fit puts more in bin t before opening bin $t+1$:

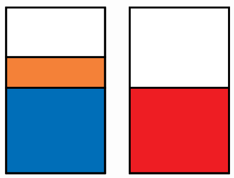


Next-Fit uses bin $t'' \geq t'$.

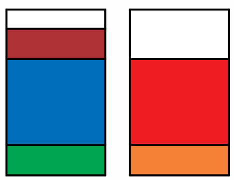


Next-Fit vs. Worst-Fit

When Worst-Fit opens bin $t+1$:



Next-Fit uses bin $t''' \geq t' + 1$.



Inductively, Next-Fit uses at least as many bins as Worst-Fit.
But $\text{CR}_{\text{Worst-Fit}} = \text{CR}_{\text{Next-Fit}} = 2.$

Refinements of competitive analysis

Long list...

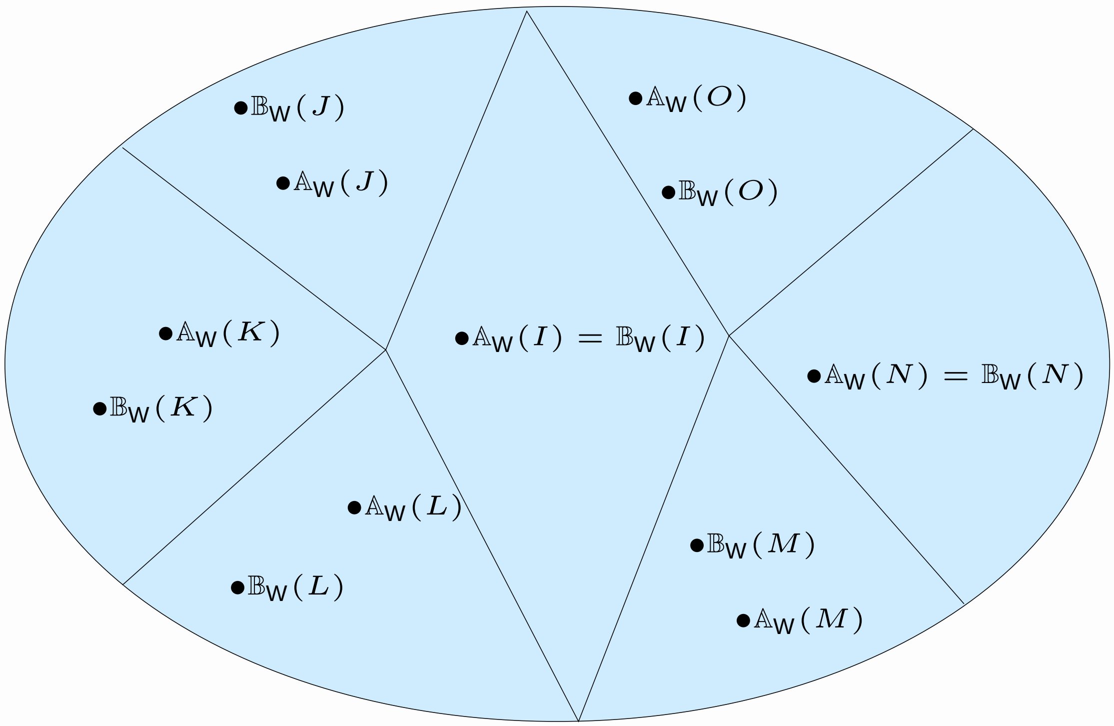
Max/Max Ratio
[Ben-David, Borodin 94]
Compares $\mathbb{A}$ to OPT
on worst sequences of length n.

Random Order Ratio
[Kenyon 95]
Compares $\mathbb{A}$ to OPT
on random ordering of same sequence.

Relative Worst Order Ratio

$\mathbb{A}_W(I)$: $\mathbb{A}'s$ performance on worst permutation of I wrt. $\mathbb{A}$

Intuitively: $\mathrm{WR}_{\mathbb{A},\mathbb{B}} =$ worst-case $\frac{\mathbb{A}_W(I)}{\mathbb{B}_W(I)}$ on long I



Relative Worst Order Ratio

[B.,Favrholdt 03], [B.,Favrholdt,Larsen 07]

Formally:

Given $\mathbb{A}$ and $\mathbb{B}$,

$$c_{\mathsf{l}}(\mathbb{A},\mathbb{B}) \;=\; \sup\,\{c \mid \exists b\colon \forall I\colon \mathbb{A}_{\mathsf{W}}(I) \geq c\,\mathbb{B}_{\mathsf{W}}(I) - b\}$$

$$c_{\mathsf{u}}(\mathbb{A},\mathbb{B}) \;=\; \inf\,\{c \mid \exists b\colon \forall I\colon \mathbb{A}_{\mathsf{W}}(I) \leq c\,\mathbb{B}_{\mathsf{W}}(I) + b\}$$

Relative worst-order ratio $\mathsf{WR}_{\mathbb{A},\mathbb{B}}$ of $\mathbb{A}$ to $\mathbb{B}$:

$$c_{\mathsf{l}}(\mathbb{A},\mathbb{B}) \geq 1 \Rightarrow \mathsf{WR}_{\mathbb{A},\mathbb{B}} = c_{\mathsf{u}}(\mathbb{A},\mathbb{B})$$

$$c_{\mathsf{u}}(\mathbb{A},\mathbb{B}) \leq 1 \Rightarrow \mathsf{WR}_{\mathbb{A},\mathbb{B}} = c_{\mathsf{l}}(\mathbb{A},\mathbb{B})$$

Relative Worst Order Ratio

Values of $\mathrm{WR}_{\mathbb{A},\mathbb{B}}$:

	minimization	maximization
$\mathbb{A}$ better than $\mathbb{B}$	< 1	> 1
$\mathbb{B}$ better than $\mathbb{A}$	> 1	< 1

$\mathrm{WR}_{\mathbb{A},\mathbb{B}} < 1 \Rightarrow$ $\mathbb{A}$ and $\mathbb{B}$ are comparable in $\mathbb{A}$'s favor.

$\mathrm{WR}_{\mathbb{A},\mathbb{B}} > 1 \Rightarrow$ they are comparable in $\mathbb{B}$'s favor.

$\mathrm{WR}_{\mathbb{A},\mathbb{B}}$ bounds how much better.

Next-Fit vs. Worst-Fit

Shown: Next-Fit$(I) \geq$ Worst-Fit$(I)\ \forall\ I$

$\Rightarrow$ Next-Fit$(I_{\text{WF}}) \geq$ Worst-Fit(I_{WF})

$\Rightarrow$ Next-Fit$(I_{\text{NF}}) \geq$ Next-Fit$(I_{\text{WF}}) \geq$ Worst-Fit(I_{WF})

So $\text{WR}_{\text{Next-Fit},\text{Worst-Fit}} \geq 1$.

Next-Fit vs. Worst-Fit

Shown: Next-Fit$(I) \geq$ Worst-Fit(I) $\forall\ I$

$\Rightarrow$ Next-Fit$(I_{\text{WF}}) \geq$ Worst-Fit(I_{WF})

$\Rightarrow$ Next-Fit$(I_{\text{NF}}) \geq$ Next-Fit$(I_{\text{WF}}) \geq$ Worst-Fit(I_{WF})

So $\text{WR}_{\text{Next-Fit,Worst-Fit}} \geq 1$.

Recall example:

Next-Fit used $2k$ bins

Worst-Fit used $k+1$ bins

So $\text{WR}_{\text{Next-Fit,Worst-Fit}} \geq 2$.

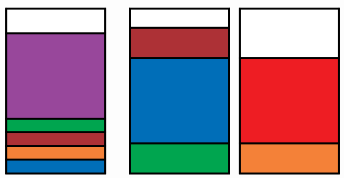
Theorem: $\text{WR}_{\text{Next-Fit,Worst-Fit}} = 2$.

Proof: $\text{WR}_{\mathbb{A},\mathbb{B}} \leq \text{WR}_{\mathbb{A},\text{OPT}} \leq \text{CR}_{\mathbb{A}}$.

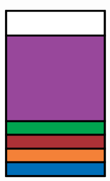
Worst-Fit vs. First-Fit

Claim: $\text{WR}_{\text{Worst-Fit,First-Fit}} \geq 1$.

Consider First-Fit's packing of any item sequence I:



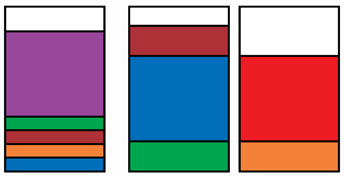
Give these items bin-by-bin to Worst-Fit:



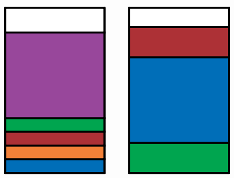
Worst-Fit vs. First-Fit

Claim: $\text{WR}_{\text{Worst-Fit,First-Fit}} \geq 1$.

Consider First-Fit's packing of any item sequence I:



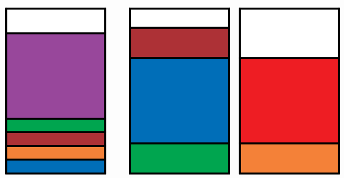
Give these items bin-by-bin to Worst-Fit:



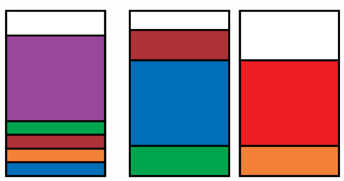
Worst-Fit vs. First-Fit

Claim: $\mathrm{WR}_{\text{Worst-Fit,First-Fit}} \geq 1$.

Consider First-Fit's packing of any item sequence I:



Give these items bin-by-bin to Worst-Fit:

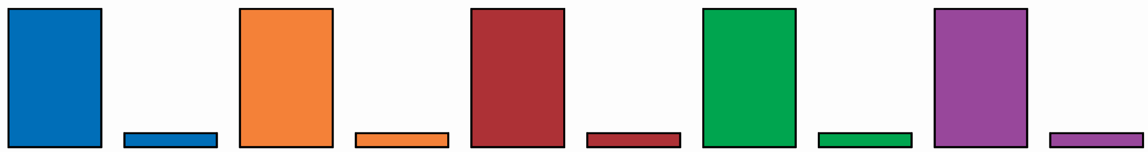


Worst-Fit uses as many bins as First-Fit.

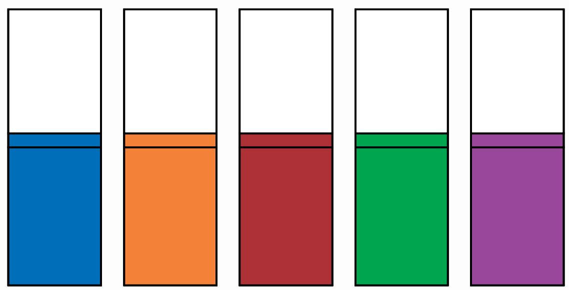
Worst-Fit vs. First-Fit

Claim: $\text{WR}_{\text{Worst-Fit,First-Fit}} \geq 2$.

Item sizes: $n \times [1/2, \epsilon]$



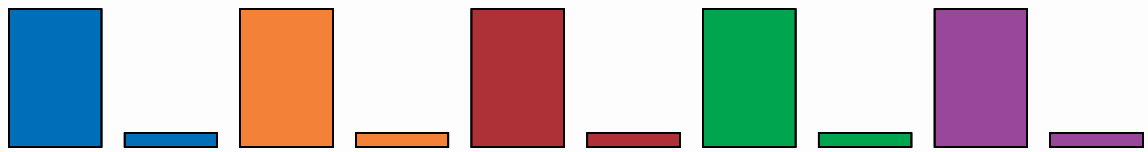
Result by Worst-Fit:



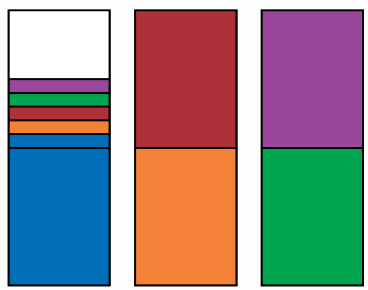
Worst-Fit vs. First-Fit

Claim: $\mathrm{WR}_{\text{Worst-Fit,First-Fit}} \geq 2$.

Item sizes: $n \times [1/2, \epsilon]$



Result by First-Fit:



Worst-Fit vs. First-Fit

Theorem: $\text{WR}_{\text{Worst-Fit,First-Fit}} = 2$.

Proof: $\text{WR}_{\mathbb{A},\mathbb{B}} \leq \text{CR}_{\mathbb{A}}$.

Compare to:

$\text{CR}_{\text{First-Fit}} = 1.7$ [Johnson, et.al. 1974]

$\text{CR}_{\text{Worst-Fit}} = 2$ [Johnson 1974]

Paging Problem

- Cache: k pages
- Slow memory: $N > k$ pages

- Request sequence: sequence of page numbers
- Fault: page requested not in cache
- Cost: 1 per fault to bring page into cache
- Goal: minimize cost

Algorithms: LRU vs. FWF

LRU – Least Recently Used
FWF – Flush When Full
Both have competitive ratio k.

Example sequence, $k = 5$:

$$\langle 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1, 2, 3, 4, 5, 6, 5, 4, 3, 2 \rangle$$

Total cost LRU = 8
Total cost FWF = 20

FWF vs. LRU

I_{LRU} – worst ordering of I for LRU

$\forall I \quad \text{FWF}_W(I) \geq \text{FWF}(I_{\text{LRU}}) \geq \text{LRU}_W(I)$

Thus, $\text{WR}_{FWF,LRU} \geq 1$ holds.

FWF vs. LRU

$$I^n = \langle 1, 2, .., k, k+1, k, ...3, 2 \rangle^n$$

$\text{FWF}_W(I^n) = 2kn$

Worst ordering for LRU:

$$\langle 2, ..., k, k+1, 1 \rangle^n, \langle 2, ..., k \rangle^n$$

$\text{LRU}_W(I^n) = n(k+1) + k - 1$

Theorem: $\text{WR}_{\text{FWF,LRU}} \geq \frac{2k}{k+1}$

In fact: $\text{WR}_{\text{FWF,LRU}} = \frac{2k}{k+1}$

Look-Ahead

Model: $\mathbb{A}$ sees request + next l requests:
Look-ahead(l)

On-line $\longrightarrow$ Look-ahead(l) $\longrightarrow$ OPT

Fact: k is still best possible competitive ratio, even with look-ahead l.

Other Models of Look-Ahead

Resource-bounded look-ahead [Young 91]

Strong look-ahead [Albers 93]

Natural look-head [Breslauer 98]

Look-ahead

LRU(ℓ):

- Sees current page and next l pages.
- Avoids evicting pages it sees.
- Evicts l.r.u. among others in cache.

First show $\text{WR}_{\text{LRU},\text{LRU}(\ell)} \geq 1$.

Theorem. For any sequence I,
$\text{LRU}_W(I) \geq \text{LRU}(\ell)_W(I)$.

LRU vs. LRU(ℓ)

Sequence I. Partition into phases:
LRU(ℓ) faults $k+1$ times per phase.
Suppose $\leq k$ distinct pages in phase P.

$$\langle \ldots \underbrace{p_1, \ldots, p, \ldots, q, \ldots, p, \ldots, p_s}_{\text{phase } P;\ k+1 \text{ faults for LRU}(\ell)}, p_{s+1}, \ldots \rangle$$

Page p evicted when q requested.

Least recently used not among next ℓ.

LRU vs. LRU(ℓ)

Case p not among next ℓ:

$$\langle \ldots p_1, \ldots, p \underbrace{, \ldots, q, \ldots,}_{P' \subset P} p, \ldots, p_s, p_{s+1}, \ldots \rangle$$

P' has q and $\geq k-1$ distinct pages.

Phase P has $\geq k+1$ distinct pages.

LRU vs. LRU(ℓ)

Case p not among next ℓ:

$$\langle \ldots p_1, \ldots, p \underbrace{, \ldots, q, \ldots,}_{P' \subset P} p, \ldots, p_s, p_{s+1}, \ldots \rangle$$

P' has q and $\geq k-1$ distinct pages.
Phase P has $\geq k+1$ distinct pages.

Case p among next ℓ:

$$\langle \ldots p_1, \ldots, p, \ldots, q \underbrace{, \ldots,}_{P'' \subset P} p, \ldots, p_s, p_{s+1}, \ldots \rangle$$

$\geq k-1$ distinct in P''; $\quad \geq k+1$ in P.

LRU vs. LRU(ℓ)

Process I by phases.
Example sequence, $k = 5$ and $\ell = 2$:

$$\langle 1, 2, 3, 4, 5, 6, \quad || \quad 5, 7, 1, 8, 4, 2, 5, 9, 3 \rangle$$

Reorder phase with new pages first;
others in order from last phase.

$$\langle 1, 2, 3, 4, 5, 6, \quad || \quad 7, 8, 9, 1, 2, 3, 4, 5, 5 \rangle$$

LRU faults on $\geq$ as many as LRU(ℓ).

LRU vs. LRU(ℓ)

Consider $I^n = \langle 1, 2, .., k, k+1 \rangle^n$.
I^n has only $k+1$ pages.
LRU faults on every page.

Suppose $l \leq k-1$.
Whenever LRU(ℓ) faults (after first k faults),
it doesn't fault on next l requests.

Suppose $l \geq k$.
LRU(ℓ) faults on $\leq$ 1 page out of k.

Theorem. $\text{WR}_{\text{LRU},\text{LRU}(\ell)} \geq \min\{l+1, k\}$.

Results for Paging

1. All conservative algorithms equivalent.
2. RW is transitive:
   so FIFO and LRU(ℓ) better than FWF.
3. (Randomized algorithm)
   MARK better than LRU.
4. New algorithm: $\mathrm{WR}_{\mathrm{LRU,RLRU}} \geq \frac{k+1}{2}$.
5. LRU-2 and LRU are asymptotically comparable in LRU-2's favor [B., Ehmsen, Larsen]

Results with Relative Worst Order Ratio

1. Dual Bin Packing:
   First-Fit better than Worst-Fit.
2. Scheduling: minimizing makespan, 2 related machines, a post-greedy algorithm is better than scheduling all jobs on the fast machine [Epstein, Favrholdt, Kohrt].
3. Bin coloring: a natural greedy-type algorithm is better than just using one open bin at a time [Kohrt].
4. Proportional price seat reservation: First-Fit better than Worst-Fit [B.,Medvedev].

Future Work

Apply to other problems?

Many open problems!