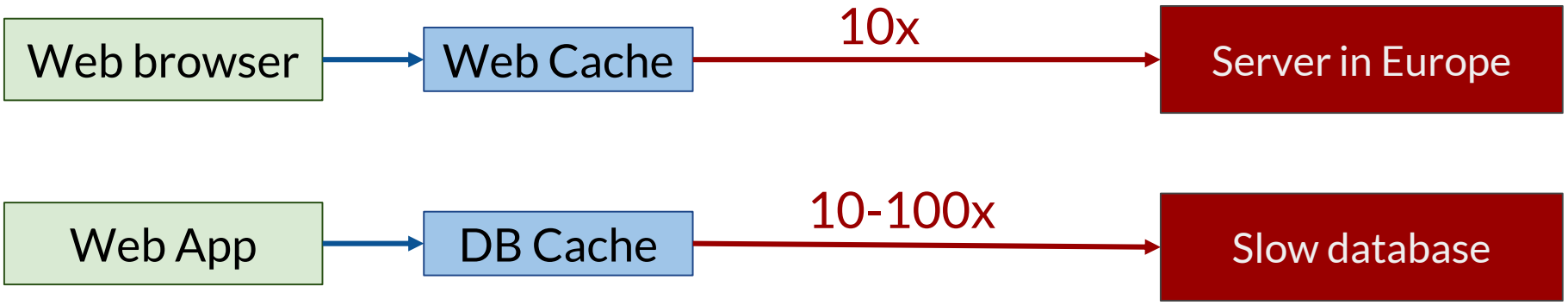
Practical Bounds on Optimal Caching with Variable Object Sizes

Daniel S. Berger *Nathan Beckmann* *Mor Harchol-Balter*

Carnegie Mellon University

ACM Sigmetrics, Irvine, June 19, 2018.

Caches are Everywhere

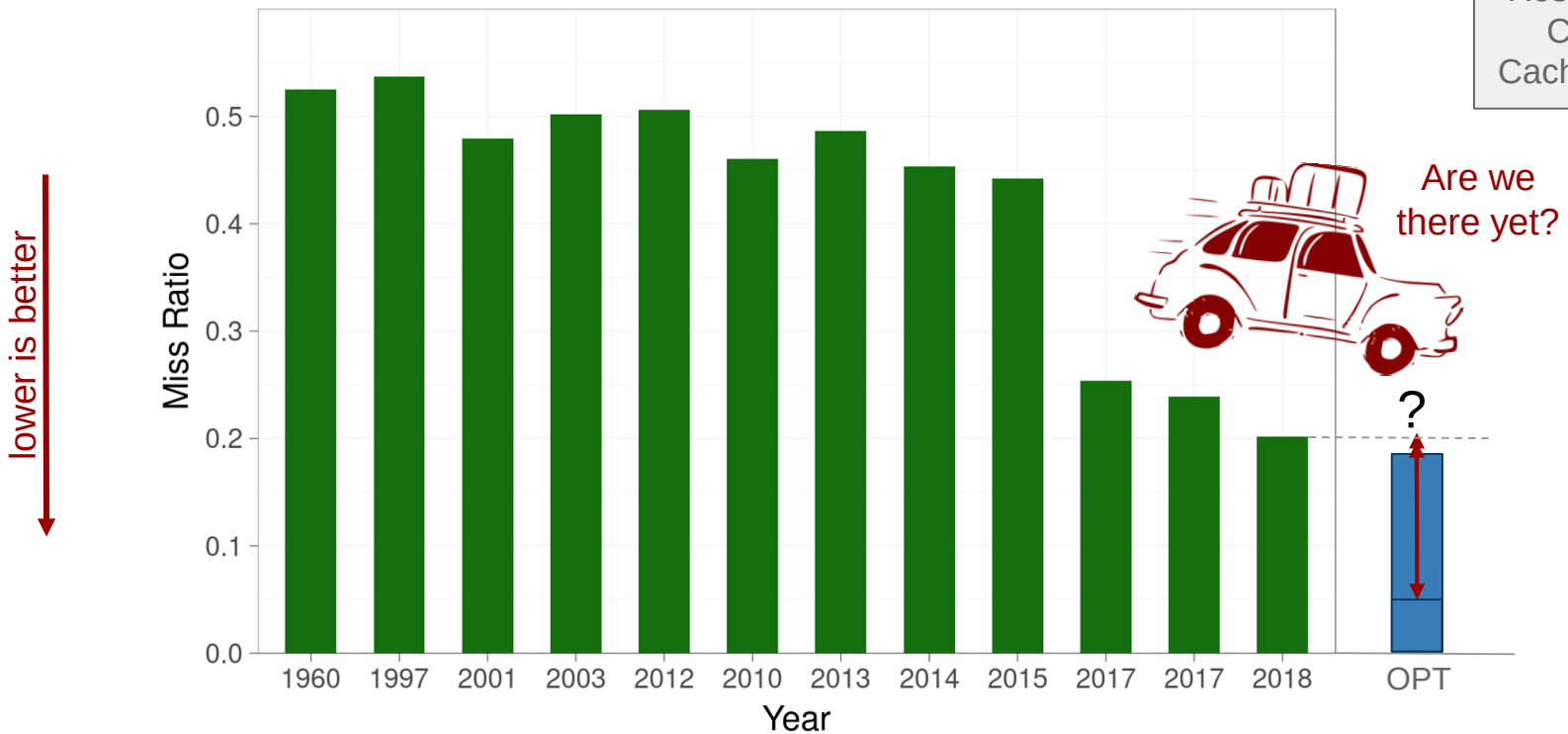


Goal: minimize cache miss ratio

$$\text{miss ratio} = \frac{\text{\# requests not served from cache}}{\text{total \# requests}}$$

Everyone Has Worked in Caching

Results on 2016 CDN trace. Cache size: 4GB.



Key Question: how much further can miss ratios be improved?

Def: OPT = lowest miss ratio possible on a given trace

Defining OPT

<u>Def:</u> OPT = lowest miss ratio possible on a given trace

= offline optimal miss ratio on a given trace

a b c b d a c d a b c d

Constraints:

1. Limited cache size
2. Gets to see full request trace
3. No prefetching

   ⇒ admit an object only when it is requested

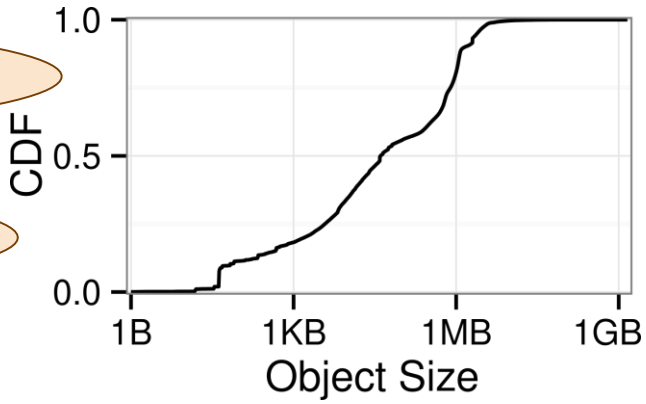
Finding OPT

What is OPT?

Belady?

No! Belady assumes equal sizes!

9 orders of magnitude variability!



So, can we find OPT?

Unfortunately, NP-hard

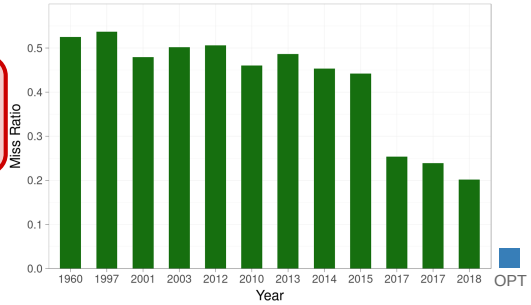
In fact, strongly NP complete

Can we approximate OPT?

OPT Approximation Algorithms

Technique	Time Complexity	Approximation Guarantee
OFMA [STOC'97]	$O(N^2)$	O(log cache size)
LP rounding [SODA'99]	$O(N^5)$	$O(\log(max_{size}/min_{size}))$
LocalRatio [JACM'01]	$O(N^3)$	**4**

State-of-the-art 4-approximation not practical



OPT Approximation Algorithms

Technique	Time Complexity	Approximation Guarantee
Best prior approximation	$O(N^3)$	4

Traces are not adversarial in practice

⇒ Probabilistic assumptions

Independent Reference Model (IRM)

Large systems: #objects, cache size

Our Main Result

Technique	Time Complexity	Approximation Guarantee
Best prior approximation	$O(N^3)$	4
Flow-offline optimum (FOO)	**$O(N^2 \log^2 N)$**	**1**

On trace with strong correlations: error < 0.14%

Traces are not adversarial in practice

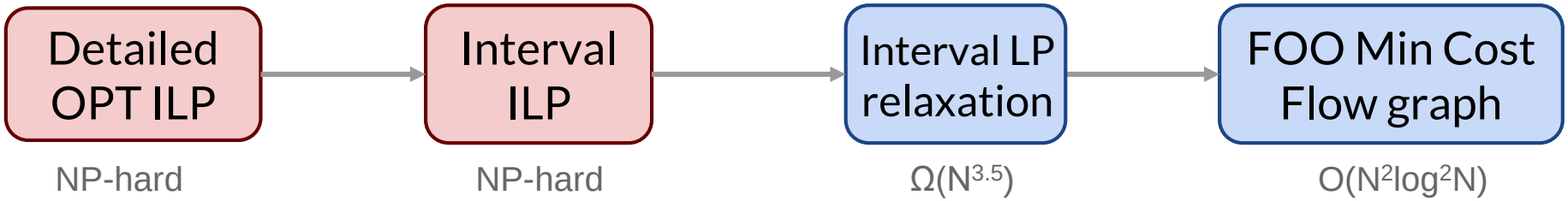
⇒ Probabilistic assumptions

Independent Reference Model (IRM)

Large systems: #objects, cache size

How does FOO attain OPT for large systems?

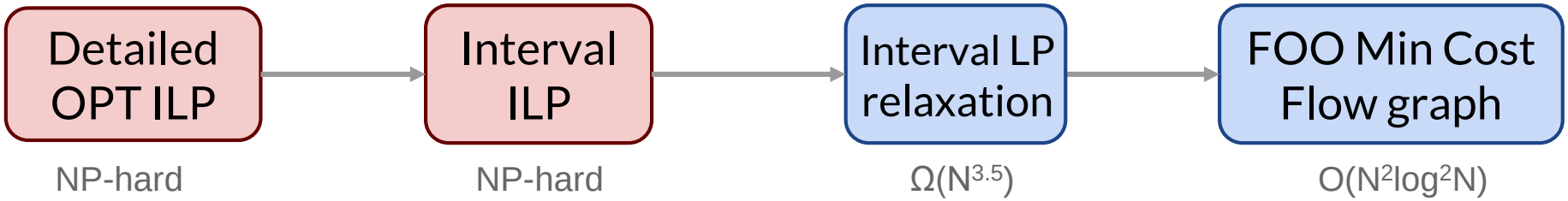
<u>How to get OPT fast:</u>



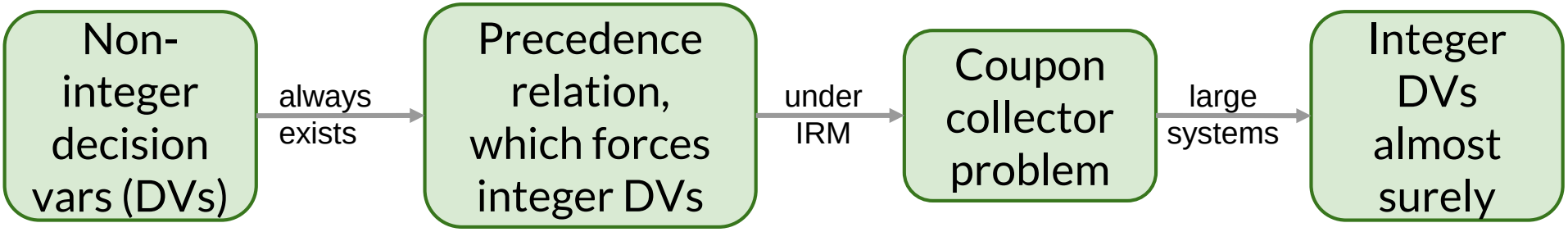
Trace:	a	b	c	b	a
DVs for object a:	$x_{(a,1)}$	$x_{(a,2)}$	$x_{(a,3)}$	$x_{(a,4)}$	$x_{(a,5)}$

How does FOO attain OPT for large systems?

<u>How to get OPT fast:</u>



<u>How to prove FOO's correctness:</u>



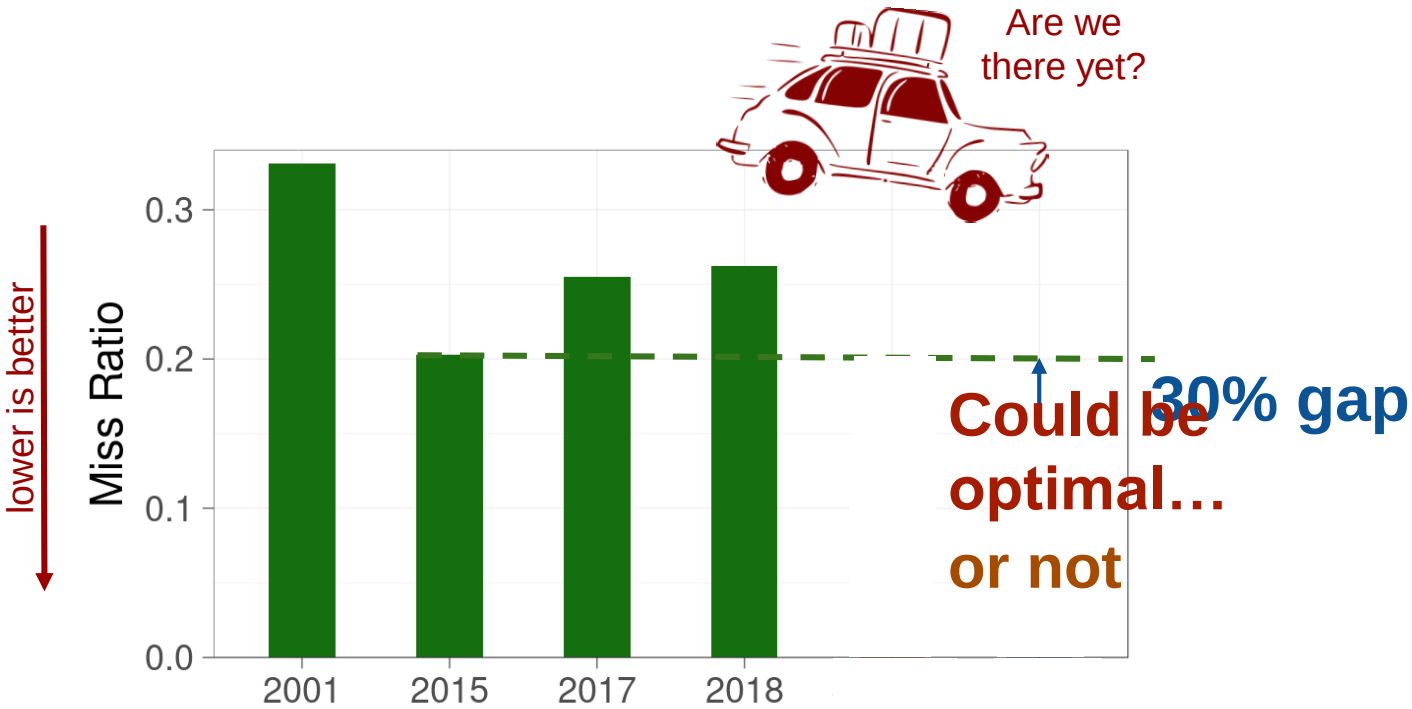
Our Main Result

Technique	Time Complexity	Approximation Guarantee
Best prior approximation	$O(N^3)$	4
Flow-offline optimum (FOO)	**$O(N^2 \log^2 N)$**	**1**

Computable with up to 10^7 requests

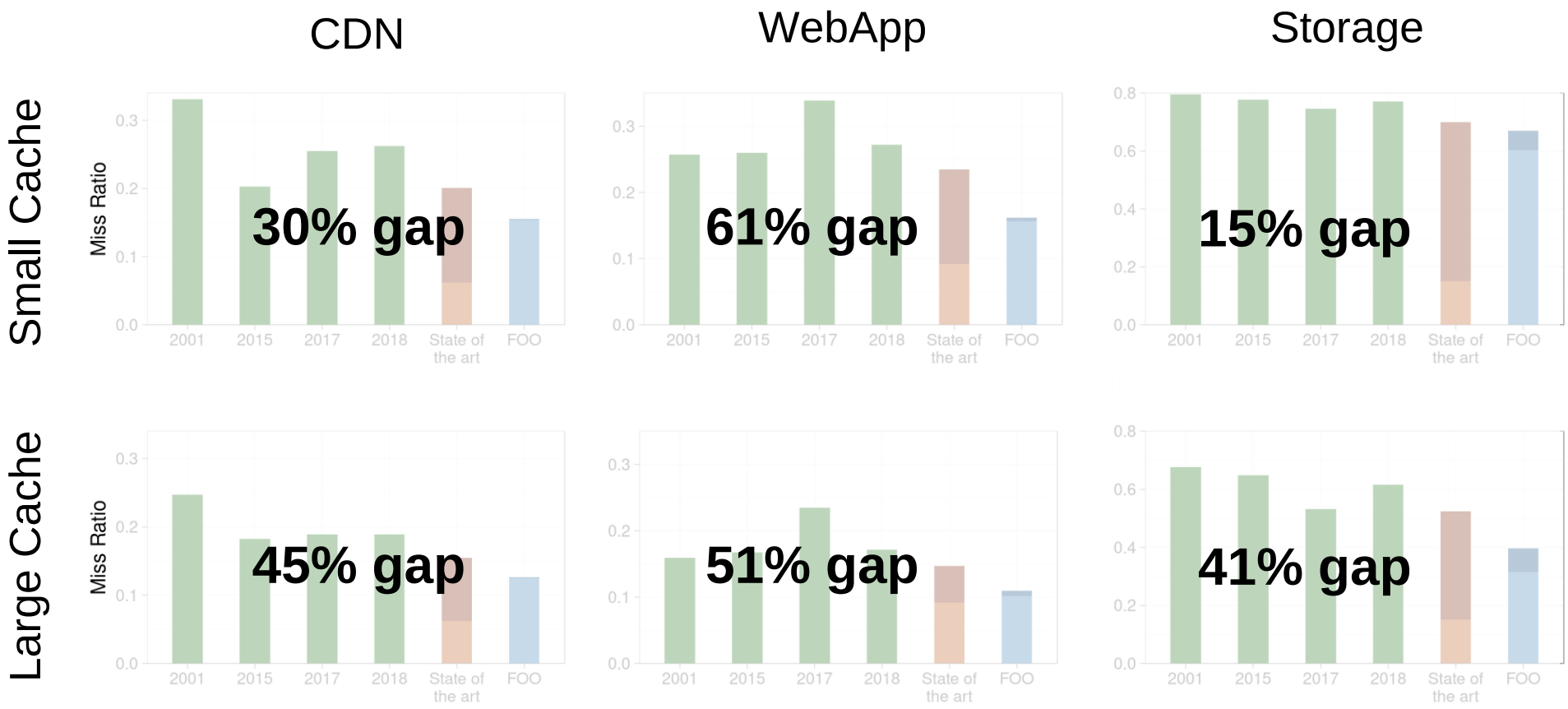
Empirical Results

<u>Key Question:</u> how much further can miss ratios be improved?



Results for Other Configurations

Key Question: how much further can miss ratios be improved?



Conclusion

Technique	Time Complexity	Approximation Guarantee
Best prior approximation	$O(N^3)$	4
Flow-offline optimum (FOO)	**$O(N^2 \log^2 N)$**	**1**

Actually can do: O(N log N)

<u>Implication:</u> large potential for new caching policies
⇒ e.g., 60% improvement possible for WebApps

Practical Bounds on Optimal Caching with Variable Object Sizes

Daniel S. Berger *Nathan Beckmann* *Mor Harchol-Balter*

Carnegie Mellon University

Source code and data: available at GitHub/dasebe/optimalwebcaching

BSD-2-Clause

ACM Sigmetrics, Irvine, June 19, 2018.