Practical Bounds on Optimal Caching with Variable Object Sizes

Daniel S. Berger
Carnegie Mellon University
dsberger@cs.cmu.edu

Nathan Beckmann
Carnegie Mellon University
beckmann@cs.cmu.edu

Mor Harchol-Balter
Carnegie Mellon University
harchol@cs.cmu.edu

ABSTRACT

Many recent caching systems aim to improve miss ratios, but there is no good sense among practitioners of how much further miss ratios can be improved. In other words, should the systems community continue working on this problem?

Currently, there is no principled answer to this question. In practice, object sizes often vary by several orders of magnitude, where computing the optimal miss ratio (OPT) is known to be NP-hard. The few known results on caching with variable object sizes provide very weak bounds and are impractical to compute on traces of realistic length.

We propose a new method to compute upper and lower bounds on OPT. Our key insight is to represent caching as a min-cost flow problem, hence we call our method the *flow-based offline optimal* (FOO). We prove that, under simple independence assumptions, FOO's bounds become tight as the number of objects goes to infinity. Indeed, FOO's error over 10M requests of production CDN and storage traces is negligible: at most 0.3%. FOO thus reveals, for the first time, the limits of caching with variable object sizes.

While FOO is very accurate, it is computationally impractical on traces with hundreds of millions of requests. We therefore extend FOO to obtain more efficient bounds on OPT, which we call *practical flow-based offline optimal* (PFOO). We evaluate PFOO on several full production traces and use it to compare OPT to prior online policies. This analysis shows that current caching systems are in fact still far from optimal, suffering 11–43% more cache misses than OPT, whereas the best prior offline bounds suggest that there is essentially no room for improvement.

The full version of this abstract appears in ACM POMACS [5].

ACM Reference Format:

Daniel S. Berger, Nathan Beckmann, and Mor Harchol-Balter. 2018. Practical Bounds on Optimal Caching with Variable Object Sizes. In *SIGMETRICS '18 Abstracts: ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems Abstracts, June 18–22, 2018, Irvine, CA, USA*. ACM, New York, NY, USA, 3 pages. <https://doi.org/10.1145/3219617.3219627>

1 INTRODUCTION

Caches are pervasive in computer systems, and their miss ratio often determines end-to-end application performance. For example, content distribution networks (CDNs) depend on large, geo-distributed caching networks to serve user requests from nearby datacenters, and their response time degrades dramatically when the requested

content is not cached nearby. Consequently, there has been a renewed focus on improving cache miss ratios, particularly for web services and content delivery [6, 7, 10–12, 14, 16, 18, 20]. These systems have demonstrated significant miss ratio improvements over least-recently used (LRU) caching, the de facto policy in most production systems. *But how much further can miss ratios improve? Should the systems community continue working on this problem, or have all achievable gains been exhausted?*

2 THE PROBLEM: FINDING THE OFFLINE OPTIMAL

To answer these questions, one would like to know the best achievable miss ratio, free of constraints—i.e., the offline optimal (OPT). Unfortunately, very little is known about OPT with variable object sizes. For equal-sized objects, computing OPT is simple (i.e., Belady [4, 19]), and it is widely used in the systems community to bound miss ratios. But object sizes often vary widely in practice, from a few bytes (e.g., metadata [21]) to several gigabytes (e.g., videos [6, 14]). We need a way to compute OPT for variable object sizes, but unfortunately this is known to be NP-hard [9].

There has been little work on bounding OPT with variable object sizes, and all of this work gives very weak bounds. On the theory side, prior work gives only three approximation algorithms [2, 3, 15], see Table 1. The best approximation is only provably within a factor of 4 of OPT. Hence, when this algorithm estimates a miss ratio of 0.4, OPT may lie anywhere between 0.1 and 0.4. This is a big range—in practice, a difference of 0.05 in miss ratio is significant—, so bounds from prior theory are of limited practical value. From a practical perspective, there is an even more serious problem with the theoretical bounds: they are simply too expensive to compute. The best approximation takes 24 hours to process 500 K requests and scales poorly, while production traces typically contain hundreds of millions of requests.

Since the theoretical bounds are incomputable, practitioners have been forced to use conservative lower bounds or pessimistic upper bounds on OPT. The only prior lower bound is an infinitely large cache [1, 10, 14], which is very conservative and gives no sense of how OPT changes at different cache sizes. Belady variants (e.g., Belady-Size) are widely used as an upper bound [13, 14, 17, 22], despite offering no guarantees of optimality.

While these offline bounds are easy to compute, we will show that they are in fact far from OPT. They have thus given practitioners a false sense of complacency, since existing online algorithms often achieve similar miss ratios to these weak offline upper bounds.

3 OUR APPROACH: FLOW-BASED OFFLINE OPTIMAL (FOO)

We propose a new approach to compute bounds on OPT with variable object sizes, which we call the *flow-based offline optimal* (FOO). The key insight behind FOO is to represent caching as a min-cost

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SIGMETRICS '18 Abstracts, June 18–22, 2018, Irvine, CA, USA

© 2018 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-5846-0/18/06.

<https://doi.org/10.1145/3219617.3219627>

Technique	Time	Requests / 24hrs	Approximation
OPT	NP-hard [9]	<1 K	1
LP rounding [2]	$\Omega(N^{5.6})$	50 K	$O\left(\log \frac{\max_i \{s_i\}}{\min_i \{s_i\}}\right)$
LocalRatio [3]	$O(N^3)$	500 K	4
OFMA [15]	$O(N^2)$	28 M	$O(\log C)$
FOO ^a	$O(N^{3/2})$	28 M	1
PFOO ^b	$O(N \log N)$	250 M	≈ 1.06

Notation: N is the trace length, C is the cache capacity, and s_i is the size of object i .

^aFOO's approximation guarantee holds under independence assumptions.

^bPFOO does not have an approximation guarantee but its upper and lower bounds are within 6% on average on production traces.

Table 1: Comparison of FOO and PFOO to prior bounds on the optimal caching policy (OPT) with variable object sizes. Computing OPT is NP-hard. Prior bounds [2, 3, 15] provide only weak approximation guarantees, whereas FOO's bounds are asymptotically tight. PFOO is a faster variant of FOO that performs well empirically and can be calculated for hundreds of millions of requests.

flow problem. This formulation yields a lower bound on OPT by allowing non-integer decisions, i.e., letting the cache retain fractions of objects for a proportionally smaller reward. It also yields an upper bound on OPT by ignoring all non-integer decisions. Under simple independence assumptions, we prove that the non-integer decisions become negligible as the number of objects goes to infinity, and thus the bounds are asymptotically tight (Theorem 2 in the full version of this paper [5]).

Our proof is based on the observation that an optimal policy will strictly prefer some requests over others, forcing integer decisions. We show such preferences apply to almost all requests by relating such preferences to the well-known coupon collector problem.

While FOO is very accurate, it is too computationally expensive to apply directly to production traces containing hundreds of millions of requests. To extend our analysis to such traces, we develop more efficient upper and lower bounds on OPT, which we call *practical flow-based offline optimal (PFOO)*. PFOO enables the first analysis of optimal caching on traces with hundreds of millions of requests, and reveals that there is still significant room to improve current caching systems.

4 EVALUATION OF FOO & PFOO

We evaluate FOO and PFOO extensively on eight production traces across cache capacities from 16 MB to 64 GB. Our traces span major Internet companies and are gathered from CDNs, web applications, and storage systems.

We find that FOO's upper and lower bounds are nearly identical, as predicted by our analysis. FOO's error across all CDN and web application traces remains below 0.15%. Even on storage traces, which have highly correlated requests that violate our proof's independence assumptions, FOO's error remains below 0.3%.

Given FOO's high accuracy, we use FOO to estimate the error of the other offline bounds. Specifically, we assume that OPT lies in the middle between FOO's upper and lower bounds. Since the

Figure 1: The maximum error in miss ratio of FOO, PFOO, and of existing upper bounds (above) and lower bounds (below) on representative traces of CDNs, Web applications, and storage systems. We note that FOO's error is too small to be seen. PFOO is also highly accurate, with errors of less than 0.03. In contrast, existing bounds lead to errors in the range 0.04 to 0.57.

difference between the bounds is so small, this adds negligible error (less than 0.15%) to all other results.

Figure 1 shows the maximum error of FOO, PFOO, and prior bounds on the first ten million requests of three traces representative of CDNs, web applications, and storage systems. Of the prior theoretical upper bounds, only OFMA runs in a reasonable time at this trace length (Table 1). We also compare to four bounds that do not have approximation guarantees: three upper bounds based on Belady (Belady, Belady-Size, and Freq/Size) and an infinite-cache lower bound (Infinite-Cap).

We find that the errors of PFOO's upper and lower bounds are less than 0.002 and 0.03, respectively. In contrast, the error of both OFMA bounds is always more than 0.26, the error of the Belady variants ranges between 0.04 and 0.32, and the error of Infinite-Cap is between 0.04 and 0.57.

We also compare PFOO to online caching policies (LRU, GDSF [8], and AdaptSize [6]) and the fast Belady-based offline bounds on full production traces with hundreds of millions of requests. The theoretical bounds, including both prior work and FOO, are too slow to run on traces this long. Figure 2 plots the miss ratio obtained on a representative CDN trace. The figure illustrates the two problems with prior practical bounds and how PFOO improves upon them. First, prior bounds on OPT are very weak. The miss ratio of Infinite-Cap is 60% lower than Belady-Size, whereas the gap between PFOO's upper and lower bounds is less than 1%. Second, prior bounds on OPT are misleading. Comparing GDSF and Belady-Size, which are also within 1%, one would conclude that online policies are nearly optimal. In contrast, PFOO gives a much better upper bound and shows that there is in fact a 22% gap between state-of-the-art online policies and OPT.

Figure 2: Miss ratios on a representative production CDN trace for a 4GB cache. Prior to our work, the best prior upper bound on OPT is within 1% of online algorithms on this trace, leading to the false impression that there is no room for improvement. The only prior lower bound on OPT (Infinite-Cap) is 60% lower than the best upper bound on OPT (Belady-Size). By contrast, PFOO provides nearly tight upper and lower bounds, which are 22% below the online algorithms. PFOO thus shows that there is actually significant room for improving current caching algorithms.

5 SUMMARY AND CONCLUSION

In summary, we make the following contributions: We present *flow-based offline optimal (FOO)*, a novel formulation of offline caching as a min-cost flow problem. FOO exposes the underlying problem structure and enables our remaining contributions.

We prove that FOO is asymptotically exact under simple independence assumptions, giving the first tight, polynomial-time bound on OPT with variable object sizes.

We present *practical flow-based offline optimal (PFOO)*, which relaxes FOO to give fast and nearly tight bounds on real traces with hundreds of millions of requests. PFOO gives the first reasonably tight lower bound on OPT on long traces.

We perform the first extensive evaluation of OPT with variable object sizes on eight production traces. Our techniques reveal that prior bounds for OPT lead to qualitatively wrong conclusions about the potential for improving current caching systems. Prior bounds indicate that current systems are nearly optimal, whereas PFOO reveals that misses can be reduced by up to 43%.

Moreover, since FOO and PFOO yield constructive solutions, they offer a new path to improve caching systems. While it is outside the scope of this work, we plan to investigate adaptive caching systems that tune their parameters online by learning from PFOO, e.g., by recording a short window of past requests and running PFOO to estimate OPT's behavior in real time. For example, AdaptSize, a caching system from 2017, assumes that admission probability should decay exponentially with object size. This is unjustified, and we can learn an optimal admission function from PFOO's decisions.

This paper gives the first principled way to evaluate caching policies with variable object sizes: FOO gives the first asymptotically exact, polynomial-time bounds on OPT, and PFOO gives the first practical and accurate bounds for long traces. Furthermore, our

results are verified on eight production traces from several large Internet companies, including CDN, web application, and storage workloads, where FOO reduces approximation error by 4000×. We anticipate that FOO and PFOO will prove important tools in the design of future caching systems.

6 ACKNOWLEDGMENTS

We thank Jens Schmitt, our shepherd Niklas Carlsson, and our anonymous reviewers for their valuable feedback.

Daniel S. Berger was supported by the Mark Stehlik Postdoctoral Teaching Fellowship of the School of Computer Science at Carnegie Mellon University. Nathan Beckmann was supported by a Google Faculty Research Award. Mor Harchol-Balter was supported by NSF-CMMI-1538204 and NSF-XPS-1629444.

REFERENCES

- [1] Marc Abrams, C. R. Standridge, Ghaleb Abdulla, S. Williams, and Edward A. Fox. 1995. *Caching Proxies: Limitations and Potentials*. Technical Report. Virginia Polytechnic Institute & State University Blacksburg, VA.
- [2] Susanne Albers, Sanjeev Arora, and Sanjeev Khanna. 1999. Page replacement for general caching problems. In *SODA*. 31–40.
- [3] Amotz Bar-Noy, Reuven Bar-Yehuda, Ari Freund, Joseph Naor, and Baruch Schieber. 2001. A unified approach to approximating resource allocation and scheduling. *J. ACM* 48, 5 (2001), 1069–1090.
- [4] L. A. Belady. 1996. A study of replacement algorithms for a virtual-storage computer. *IBM Systems journal* 5 (1996), 78–101.
- [5] Daniel S. Berger, Nathan Beckmann, and Mor Harchol-Balter. 2018. Practical Bounds on Optimal Caching with Variable Object Sizes. *ACM POMACS* (2018). To appear, June 2018.
- [6] Daniel S. Berger, Ramesh Sitaraman, and Mor Harchol-Balter. 2017. AdaptSize: Orchestrating the Hot Object Memory Cache in a CDN. In *USENIX NSDI*. 483–498.
- [7] Aaron Blankstein, Siddhartha Sen, and Michael J. Freedman. 2017. Hyperbolic Caching: Flexible Caching for Web Applications. In *USENIX ATC*.
- [8] Ludmila Cherkasova. 1998. *Improving WWW proxies performance with greedy dual-size-frequency caching policy*. Technical Report. Hewlett-Packard Laboratories.
- [9] Marek Chrobak, Gerhard J. Woeginger, Kazuhisa Makino, and Haifeng Xu. 2012. Caching is hard—even in the fault model. *Algorithmica* 63 (2012), 781–794.
- [10] Asaf Cidon, Assaf Eisenman, Mohammad Alizadeh, and Sachin Katti. 2016. Cliffhanger: Scaling Performance Cliffs in Web Memory Caches. In *USENIX NSDI*. 379–392.
- [11] Gil Einziger and Roy Friedman. 2014. Tinylfu: A highly efficient cache admission policy. In *IEEE Euromicro PDP*. 146–153.
- [12] Nicolas Gast and Benny Van Houdt. 2015. Transient and steady-state regime of a family of list-based cache replacement algorithms. In *ACM SIGMETRICS*. 123–136.
- [13] Peter Hillmann, Tobias Uhlig, Gabi Dreo Rodosek, and Oliver Rose. 2016. Simulation and optimization of content delivery networks considering user profiles and preferences of internet service providers. In *IEEE Winter Simulation Conference*. 3143–3154.
- [14] Qi Huang, Ken Birman, Robbert van Renesse, Wyatt Lloyd, Sanjeev Kumar, and Harry C. Li. 2013. An analysis of Facebook photo caching. In *ACM SOSP*. 167–181.
- [15] Sandy Irani. 1997. Page replacement with multi-size pages and applications to web caching. In *ACM STOC*. 701–710.
- [16] Conglong Li and Alan L. Cox. 2015. GD-Wheel: a cost-aware replacement policy for key-value stores. In *EUROSYS*. 1–15.
- [17] Suoheng Li, Jie Xu, Mihaela van der Schaar, and Weiping Li. 2016. Popularity-driven content caching. In *IEEE INFOCOM*. 1–9.
- [18] Bruce M. Maggs and Ramesh K. Sitaraman. 2015. Algorithmic nuggets in content delivery. *ACM SIGCOMM CCR* 45 (2015), 52–66.
- [19] et al Mattson, Richard L. 1970. Evaluation techniques for storage hierarchies.. In *IBM Systems journal*, Vol. 9. 78–117.
- [20] Giovanni Neglia, Damiano Carra, and Pietro Michiardi. 2018. Cache policies for linear utility maximization. *IEEE/ACM Transactions on Networking* (2018).
- [21] Rajesh Nishtala, Hans Fugal, Steven Grimm, Marc Kwiatkowski, Herman Lee, Harry C. Li, Ryan McElroy, Mike Paleczny, Daniel Peek, Paul Saab, et al. 2013. Scaling memcache at facebook. In *USENIX NSDI*. 385–398.
- [22] Samta Shukla and Alhussein A. Abouzeid. 2017. Optimal Device-Aware Caching. *IEEE Transactions on Mobile Computing* 16, 7 (2017), 1994–2007.