Evaluation of Performance of Cooperative Web Caching with Web Polygraph

Ping Du **Jaspal Subhlok**

Department of Computer Science
University of Houston
Houston, TX 77204
{pdu, jaspal}@uh.edu

Abstract

This paper presents a framework for evaluating the performance of cooperative Web cache hierarchies. Web Polygraph is employed to provide an environment for simulation, to generate desired workloads, and to obtain performance statistics from the standpoint of clients and servers. Squeezer, a proxy cache log analysis tool, provides detailed cache statistics including information about cache cooperation. Dummynet toolkit is used to manage network properties and simulate realistic network conditions. The result is a flexible framework that can analyze the performance of a given cache hierarchy on a given workload under given network conditions. All experiments are repeatable and results with different configurations can be compared. We describe our evaluation methodology and report our experience in employing it to compare cache hierarchies built with Squid proxy cache servers.

1 Introduction

Multiple Web caches can cooperate using a variety of protocols to enhance the service they provide to their clients. The value of a particular architecture and configuration for cooperative caching depends on a variety of factors, such as, location of caches, network conditions, user access patterns, hardware and software platforms, and available storage on the caches. For a given scenario, how profitable cache cooperation will be, what protocols are most suitable, and what configuration is the best, are all difficult questions. To our knowledge, no good methods exist to evaluate a cooperative caching setup, and hence decisions about deployment of caches are often left to guesswork.

Analysis of cooperative caching has been addressed by several research efforts and we mention just a few. Cao and Irani [2] and Breslau [1] characterized user access patterns by analyzing traces. Yu [13] et al. studied the benefits of specific cooperation architectures with approximate models of Web clients and servers. Krishnan [10] and Wolman [12] explored the performance of cache cooperation based on simulation with real Web client traces. Chiang [3] and Dykes and Robbins [5] characterized the benefits of several Web caching schemes analytically.

However, none of the research results can be applied to compare the performance of different cooperative caching protocols and hierarchies in realistic Web environments. To achieve this goal effectively, several requirements must be met. The test method should be able to model realistic Web access patterns and network characteristics. The experiments should be repeatable so that different protocols and hierarchies can be compared. Finally, the comparison should include user perceived response times, not just hit ratios.

Analytical approaches cannot accurately model complex practical factors such as the impact of inter-cache queries on cache performance. Simulation with real traces on a real network environment provides realistic results but this approach has several limitations. The experiments are not repeatable, the load on the network due to experiments may not be acceptable, and the approach is not flexible enough to model different situations and workloads.

We present a synthetic workload simulation method to study the performance of cooperative caching. Web Polygraph benchmarking tool [9, 11] was extended to evaluate cooperative caching. Squeezer program [6] was modified to obtain cache and inter-cache statistics. Dummynet toolkit [8] is used to simulate networks of different capa-

bilities between caches, clients and servers. The framework allows us to compare different cache setups and hierarchies under different workloads and different network conditions.

In this paper we describe our framework for evaluation of cooperative cache arrays and report our experience in using it. We demonstrate the tradeoffs between different configurations of caches, and the impact of cache size on the performance of a cache hierarchy. The main result is that our Web Polygraph based framework can evaluate cooperative caching hierarchies effectively and can be used as a tool to assist the deployment of Web caches in the real world.

2 Performance evaluation framework

Our performance evaluation methodology is based on Web Polygraph. We first briefly describe Web Polygraph and then discuss how it is extended to evaluate cache hierarchies.

2.1 Web Polygraph

Web Polygraph [9, 11] is a freely available and widely used benchmarking tool for caching proxies, origin server accelerators and other Web intermediaries. It can be configured to produce a variety of Web traffic workloads and different traffic characteristics can be changed independently. Polygraph can generate Web traffic with different content types, size distributions, object modification and expiration times, cacheability, and reuse patterns. By appropriately configuring these parameters, realistic Web traffic can be simulated.

The Web Polygraph environment consists of *polyclt* processes composed of robots to simulate Web users and *polysrv* processes to simulate Web servers. Requests generated by a *polyclt* can be sent directly to the servers or through a proxy cache. Servers wait for a configurable "think time" before responding. A WAN environment, with configurable network delay and packet loss, is simulated by Dummynet [8]. Web Polygraph generates a detailed performance report including hit ratio, user response time, error rate, and several other aspects of performance.

2.2 Cooperative caching evaluation

Web Polygraph was designed to evaluate a single caching system. The performance of a cache is measured by the statistics obtained from the Polygraph processes. However, Polygraph does have some features that help in the evaluation of cache hierarchies. Polygraph can generate multiple request streams within the same global URL space. Multiple client and server processes executing on different machines can exist in the same environment, and each robot in a client process can access any object from any server.

Figure 1 shows a simple setup to evaluate a hierarchy of proxy caches. It consists of machines running *polyclt* and *polysrv* processes, cooperating proxy caches, and a network connecting them. The proxy cache hierarchy is a black box from Polygraph's viewpoint; Polygraph does not know about the configuration or cache cooperation. However, statistics from Polygraph and the caches can be combined for detailed performance analysis.

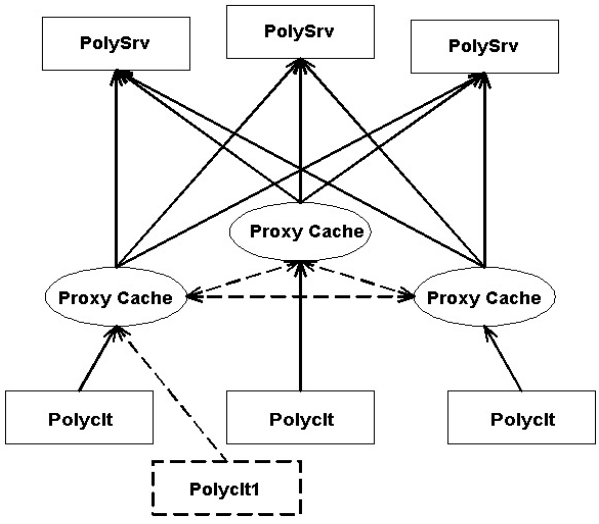

Figure 1. Evaluation of a cooperative cache hierarchy with Web Polygraph

Each *polyclt* process can be configured to point to a different cache. Four parameters determine the number of robots on each *polyclt*: total request rate, maximum client load, maximum robot load, and number of client hosts. Normally the load is divided evenly among *polyclts*. An unbalanced workload can be achieved by configuring different number of *polyclts* to point to different proxy caches as illustrated with the additional process

polyclt1 in Figure 1. A fine grain mechanism to achieve unbalanced load is to specify different numbers of robots on different *polyclts*. The most recent version of Polygraph also allows individual robots to point to different caches. Polygraph processes can be run on multiple machines, which makes this approach scalable.

Cache hierarchy evaluation should be performed in the kind of environment where the caches will be deployed. Dummynet toolkit [8], which can control bandwidth, latency, and delay on per node and per link basis, is employed to simulate different network environments. This makes it possible to simulate expected network conditions between caches, clients, and servers.

2.3 Performance metrics

Polygraph provides a comprehensive report of Web cache performance that includes throughput, mean response time, miss response time, hit response time, hit ratio, error rate, queuing of requests, connection length, object size, object class, etc. However, there is no way for Polygraph to get cache cooperation information since it considers the entire cache hierarchy as a black box. If the only goal of analysis was to obtain aggregate cache hierarchy performance information, then the reports provided by Polygraph will be sufficient. However, information about cache cooperation, such as hit ratios and response times associated with sibling and parent hits and misses, are critical for understanding the behavior of a cache hierarchy.

In order to evaluate cache hierarchies, it is clear that information from proxy servers must be used in conjunction with Polygraph statistics. This is a challenge since the meaning of common metrics is different for Polygraph and proxy caches. As an example, throughput is the rate at which client requests are served from Polygraph's standpoint. However, for a proxy cache, it is the rate at which HTTP requests are served by one particular cache including requests from peer caches. Similarly, response time and hit ratio have different meanings since Polygraph defines them from the standpoint of clients and servers, which is different from the standpoint of proxy caches in a hierarchy.

Fortunately, most Web cache systems generate statistics and logs that can be used to complete the necessary analysis. Summary statistics are not sufficient to get a detailed picture of the performance of a cache hierarchy. For instance, it is often not possible to get statistics on sibling traffic, or distinguish between phases of request streams from Polygraph. Both of these are critical for our evaluation. We decided to work with Squid caches [10] and took the approach of analyzing the per request logs produced by Squid. We modified the Squeezer [6] profiling tool in two ways: first, cache cooperation information was added, and second, start and end times could be defined as command arguments to get performance results for specific Polygraph phases. This modified tool provides information such as hit ratios and response times, separately for local, sibling and parent hits, and for different phases of Polygraph.

3 Experimental results

We report on the usage of the framework for evaluating cache hierarchies of Squid cache servers. The hardware configuration of the Squid machines is 800 MHz Pentium III, 512 MB RAM and four 30.7 GB SCSI disks. Squid servers and Polygraph machines are connected by 100 Mbps switched Ethernet. The Squid cache on each machine uses at most three disks. All machines run FreeBSD 4.1.1 and Squid 2.4. DEVEL4. For Web Polygraph, version 2.5.4 was used.

The test framework can be used to explore different cooperative caching scenarios. Variables that can be changed are as follows:

1. Architecture of cooperative caches: parent and sibling configurations.
2. Cooperation protocol: ICP, cache digest, etc.
3. Cache size: memory and disk cache.
4. Cache replacement and refreshment policy.
5. Workload characteristics: sharing pattern, content type, request rate, etc.
6. Network condition: bandwidth, latency, etc.
7. Client and server characteristics: client number, server think time, etc.

We report on some experiments to demonstrate the evaluation framework. More details are available in [4]. The workload is based on Polymix-3 [11] and consists of separate phases to fill up the cache and for actual evaluation. Polygraph servers

were configured with 80 millisecond delay for communication with other machines in the test bench. Server think times are normal distributed with a mean of 2.5 seconds. Caches use the ICP protocol for cooperation among peers. All experiments use a *public interest* of 50% which is a Polygraph parameter reflecting commonality of objects in client request streams. The total HTTP request rate is 90 and the total disk cache size is 9 GB unless otherwise specified. No network latency is configured between the caches.

3.1 Performance of different hierarchies

We compare the performance of different cache hierarchies built with 2 or 3 Squid caches. Figure 2 lists the name and topology of the hierarchies. Experimental results are shown in Figure 3.

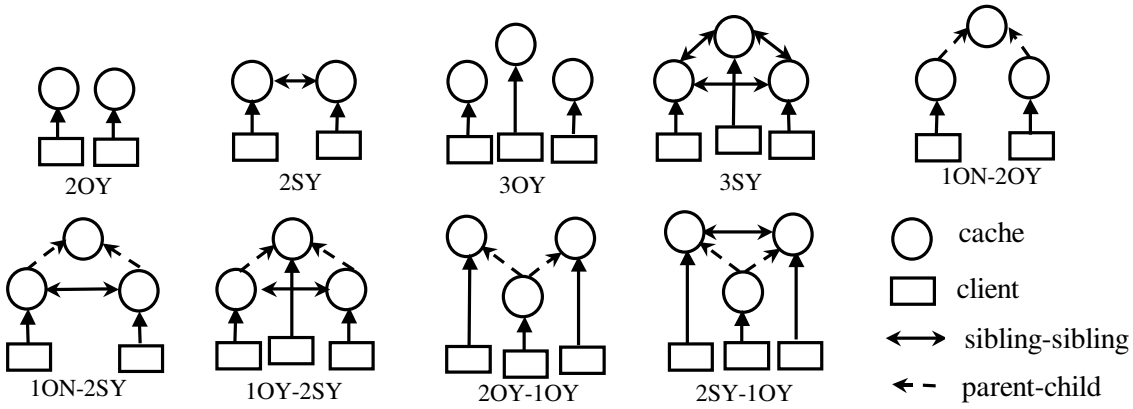

Figure 2. Topologies of tested Squid cache hierarchies

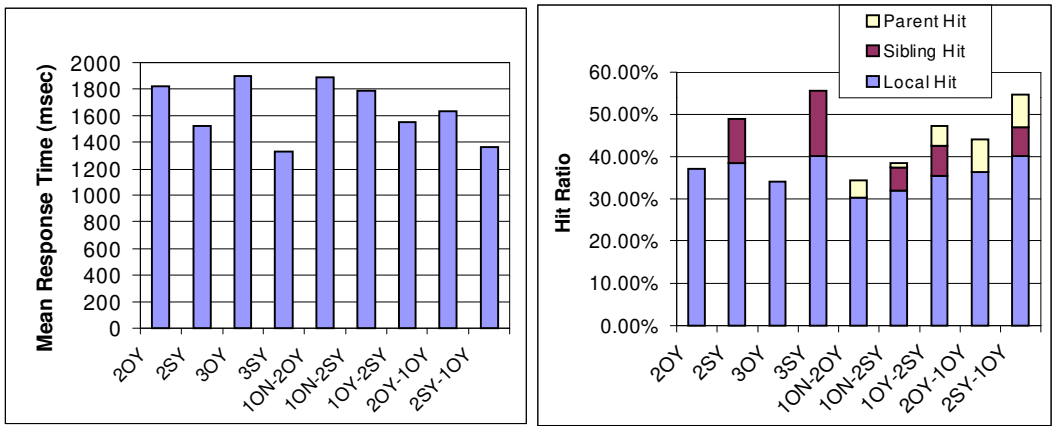

Figure 3. Comparison of performance of different Squid cache hierarchies

We point out some interesting observations. Comparing the hit ratios of separate caches (2OY and 3OY) with corresponding all sibling hierarchies (2SY and 3SY), we observe that the overall hit ratio increases significantly – from around 33% to 55%, which underlines the benefits of peering. A more interesting observation is that the local hit ratios are also higher when there are siblings. The reason is that the popular objects are more likely to stay in a cache because of remote hits, which improves the local hit ratio also. The average response time for all cooperative hierarchies is better than separate caches. This shows that the impact of improved hit ratios due to parents and siblings outweighs the delays and overheads of cooperative caching in this setup. However, this situation can change for higher delay between caches.

The overall response time performance is best for an all sibling hierarchy 3SY for 3 caches. It

appears that the role of siblings is more crucial than parents in our experiments since all the higher ranked hierarchies have sibling configurations.

Normally, sibling caches are part of a single organization, whereas parent caches are located closer to the Internet backbone. Parents are also expected to have more storage and CPU power, and have several children. None of these is true for our setup. So additional experiments are necessary to explore the functions of parents fairly.

3.2 Performance for different cache sizes

Another important question in Web cache usage is the extent of performance improvement with increasing disk space, and whether that depends on cache cooperation. For this experiment, we set up two independent and sibling caches and varied the available disk size per cache from 1.5 GB to 24 GB. The total memory cache size per cache is 350 MB. The measured results are charted in Figure 4.

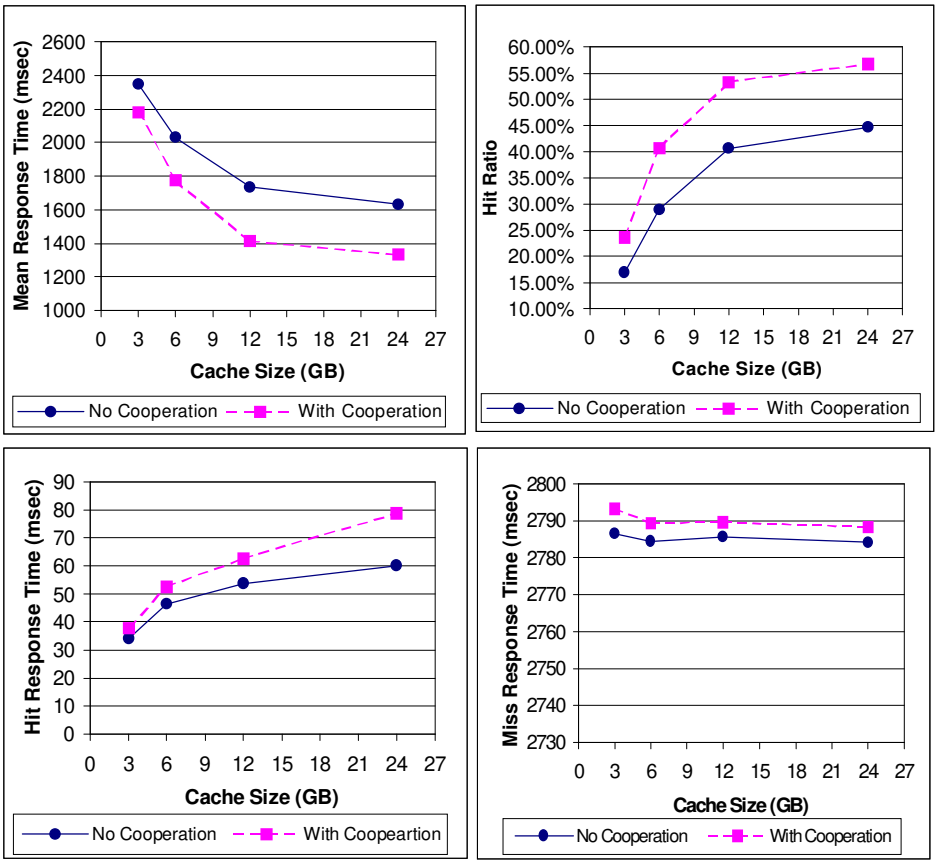

Figure 4. Performance of hierarchies with different disk space on caches

Both hit and miss response time are higher for cooperative caching. Miss response time increases because all caches have to be checked before going to the server and hits include remote hits, which take longer. The miss response time is fairly stable for the experiment, but the hit response time increases with cache size. The reason is that a larger fraction of hits are disk hits and not memory hits as the disk size increases.

The overall hit ratio increases quickly with disk space for both cooperative and separate caches and then stabilizes. The average response time decreases rapidly with increased space and then stabilizes in both cases. It is noteworthy that the benefit of cooperative caching, in terms of the average response time, increases as the available disk space increases. We conjecture that the overheads of cooperative caching are fixed but the benefits increase when a sibling cache is larger.

4 Conclusions

We introduce a framework for evaluation of cooperative caching based on Web Polygraph proxy cache benchmarking tool, Squeezer cache trace analysis tool, and Dummynet. The framework allows extensive evaluation of Web cache hierarchies, just as Polygraph does for single caches. We have illustrated the usage of the framework by comparing several configurations of Squid proxy caches. Distinguishing features of our approach are as follows:

- Web traffic with different workload characteristics can be easily specified.
- The experiments are performed on a real network whose properties can be controlled by Dummynet to simulate different conditions.
- Tests are repeatable and results with different configurations can be compared.
- Both hit ratio and user response time based metrics are used.
- Cooperative caching overheads, such as the overhead due to inter-cache queries, directly impact measured performance.

There are several future directions of this r esearch. While there are no fundamental limitations to the scalability of the approach, validation for large systems is needed. Analytical techniques may be combined with this approach to study large hierarchies efficiently. Finally, updating to the most recent version of Polygraph and improving the user interface are critical to making this project useful for the community.

This research was motivated by discussions with our industrial partners that indicated that solving the caching needs of large organizations was a bigger challenge than building better individual caches. This framework can help in the evaluation and deployment of caching solutions.

5 Acknowledgements

This research was sponsored by the Texas Advanced Technology Program under grant number 003652-0424, and by the Texas Learning and Computation Center. Compaq Inc. loaned us the Polygraph testbed for this project. We thank Dr. Martin Herbordt at Boston University and Mr. Kevin Leigh at Compaq for their advise and help.

References

[1] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. "Web Caching and Zipf-like Distribution: Evidence and Implications". In *Proceedings of IEEE INFOCOM'99*, March 1999.

[2] P. Cao and S. Irani. "Cost-Aware WWW Proxy Caching Algorithms". In *Proceedings of the 1997 USENIX Symposium on Internet Technology and Systems (USITS'97)*, December 1997.

[3] C. Chiang, M. Ueno, M. Liu, M. Muller. "Modeling Web Caching Hierarchy Schemes". Technical Report, OSU-CISRC-6/99-TR 17.

[4] P. Du. "Evaluating of Cooperative Web Caching with Web Polygraph". Master thesis. University of Houston, Department of Computer Science.

[5] S. Dykes and K. Robbins. "A Viability Analysis of Cooperative Proxy Caching". In *Proceedings of IEEE INFOCOM 2001*, April 2001.

[6] M. Koziński. "Squeezer: a Tool for Profiling Squid Web Cache Server". http://www.geocities.com/maciej_zinski/w3cache/squeezer.html.

[7] P. Krishnan and B. Sugla. "Utility of Co-operating Web Proxy Caches". In *Proceedings of the Seventh International World Wide Web Conference*, April 1998.

[8] L. Rizzo. Dummynet. http://info.iet.unipi.it/~luigi/ip_dummynet/.

[9] A. Rousskov ans D. Wessels. "High Performance Benchmarking with Web Polygraph". http://polygraph.ircache.net/doc/papers/paper01.ps.gz.

[10] Squid Internet Object Cache. http://www.squid-cache.org/.

[11] Web Polygraph: a Proxy Performance Benchmark. http://www.Web-polygraph.org/.

[12] A. Wolman, G. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy. "On the Scale and Performance of Cooperative Web Proxy Caching". In *Proceedings of the 17th ACM symposium on Operating Systems Principles (SOSP'99)*, 16-31, December 1999.

[13] P. Yu and E. MacNair. "Performance Study of a Collaborative Method for Hierarchical Caching in Proxy Servers". In *Proceedings of the Seventh International World Wide Web Conference*, April 1998.