Dynamic Hierarchical Caching for the World Wide Web

Martha H Greenberg
AUP Supervisor: David R. Karger

February 6, 1997

The World Wide Web is a system for distributing documents, including hypertext, images, sound files, plain text, audio, animations and applications. One of the most visible problems for the web today is slow response time due to overloaded networks and servers. Very popular web sites get too many queries at once and respond very slowly or not at all.

A web document is uniquely identified by a Uniform Resource Locator (URL). The URL for a document contains the protocol that should be used to ask for the document, the name of the machine which has the document (the content server), and the pathname by which the machine can identify the document. In the simplest web scheme, a client will directly request the document from the content server every time the user requests to see the document.

Caching is the storage of retrieved documents locally for further use. Therefore, it won't be necessary to connect to the content server the next time that document is requested. This technique reduces the number of requests the content server must process. A proxy server is a machine that receives requests from a number of machines; it then either serves the request itself, denies the request or forwards the request to the real server. A proxy cache server is a proxy server that caches the information that went through the proxy. Since the proxy server is usually closer (in terms of network throughput) and less loaded than the content server, the client sees a faster response to their query. Caching also saves network bandwidth to the content server.

In current hierarchical caching schemes, the hierarchy of machines is fixed. Neighbors are other caches which are on the same level of the hierarchy. Parents are on a higher level, and children on a lower level. A cache will make a request to another cache, usually its parent, if it does not have a valid copy of the document being requested. If no other caches have the document, it will request the document from the content server. Machines higher in the hierarchy are responsible indirectly for more clients and will be asked for more distinct documents.

In the Harvest/Squid scheme, each machine knows a list of neighbors and parents to whom it can make requests. The network administrator must give the cache a list of what other caches it can ask, and whether they are neighbors or parents. This list is the same for all documents, so if the cache receives multiple requests for different documents that it does not have, it asks the same set of caches for all of them.

A major problem with current hierarchical caching schemes is that caches near the top of the hierarchy are responsible for a very large number of distinct documents and may become overloaded processing queries for those documents. The ultimate goal of this project is to set up a system in which it is very unlikely for a single caching server to become overloaded with queries relative to other caching servers.

I will be working for David R. Karger in the Algorithms groups of the Theory of Computation Group of the Laboratory for Computer Science. He has designed an algorithm that may solve this problem. My project is intended to determine if this algorithm will work in the current Internet, and if it will be more efficient than the schemes that are currently in use.

The proposed new algorithm differs in that there is a unique hierarchy of parents for each URL. The cache builds this hierarchy in a number of steps. First, an abstract tree of degree d is constructed. This tree and the degree d depend on only on the number of caching servers. Then a hashing function maps `(tree node, URL)` pairs to caching servers. When the cache gets a request for a document, it picks a random leaf in the abstract tree, computes the path from the leaf to the root, and then maps the abstract tree nodes onto actual caching servers using the URL. The cost to the new algorithm is that the cache must have a near-recent list of all the proxy servers in the cache system. This is a benefit, since no caching server is at the top of any fixed hierarchy, all caching servers will be responsible for about the same number of distinct documents and queries.

I will implement this algorithm using the Squid caching and proxying system from the National Laboratory for Applied Network Research (developed from the Harvest project at the University of Colorado). In order to do this, I will also have to extend the currently used cache protocol to allow the machines to communicate which machines on the network are using this algorithm. They will also have to pass along where they are in the hierarchy for a particular document, so that the next machine does not have to recompute the entire hierarchy.

I will have to implement this algorithm for determining what cache to ask for a particular document. I will also need to integrate it with Squid so that Squid uses this algorithm for determining what other caches to ask for a document, rather than using its fixed hierarchy.

References

[1] Bowman, C., Peter B. Danzig, Udi Manber and Michael F. Schwartz. *Scalable Internet Resource Discovery: Research Problems and Approaches*. Communications of the ACM, 37(8), pp. 98-107, August 1994.

[2] Chankhunthod, Anawat, Peter B. Danzig, Chuck Neerdaels, Michael F. Schwartz and Kurt J. Worrell. *A Hierarchical Internet Object Cache*. Technical Report 95-611, Computer Science Department, University of Southern California, Los Angeles, California, March 1995.

[3] Hardy, Darren R., Michael F. Schwartz, Duane Wessels. *Harvest User's Manual*. Available at `http://harvest.transarc.com/afs/transarc.com/public/trg/Harvest/user-manual/`

[4] Karger, David R., et al, *Relieving Hot Spots on the World Wide Web*. Available at `http://theory.lcs.mit.edu/~karger/Papers/web.ps.gz`

[5] Wessels, Duane. *A Distributed Testbed for National Information Provisioning*. Available at `http://www.nlanr.net/Cache/`

[6] Wessels, Duane. *Intelligent Caching for World-Wide Web Objects*. Interdisciplinary Telecommunications Program, University of Colorado - Boulder, 1995. M.S. Thesis.

[7] Wessels, Duane. *Internet Cache Protocol (ICP) version 2*. Available at `http://www.nlanr.net/Cache/ICP/ICP-id.txt`

[8] Wessels, Duane. *Squid Internet Object Cache*. Available at `http://www.nlanr.net/Squid/`