The Effect of File Caching on U of S - Internet Traffic: Preliminary Investigations

Martin F. Arlitt

Supervisor: Dr. Carey Williamson

Abstract

The volume of network traffic passing over the Internet is increasing rapidly. The main factor in this increase is the growth of Information Resource Discovery Service (IRDS) protocols such as File Transfer (FTP-data), World Wide Web (WWW or HTTP) and Gopher. One approach to reducing this volume of traffic is the use of file caches. This paper looks at two possible scenarios in which a file cache could be implemented at the University of Saskatchewan. A file cache could be used to reduce the network loads and response times to user requests of IRDS protocols. A file cache could also be used to reduce the number of requests that reach an IRDS server. Preliminary results show that a file cache could reduce IRDS network traffic at the U of S by 13%, and reduce the number of requests reaching the Campus Web Server by 97%.

1 Introduction

The volume of network traffic passing over the Internet is increasing rapidly. The major factor in this increase is the growth of Information Resource Discovery Service (IRDS) protocols such as File Transfer (FTP-data), World Wide Web (WWW or HTTP) and Gopher. In December 1994 these protocols were responsible for 11 Terabytes of data being transferred across the NSFNET backbone. With this growth comes an increase in network loads, server loads, and user response times.

One approach for reducing network loads and user response times is the use of file (disk) caches. A network file cache works in the following way. When a user requests an IRDS document, the file cache is checked to see if the document is stored there. If it is, the document can be returned to the user immediately (a cache hit). If the document is not stored in the cache (a cache miss), the request is then sent on to the remote server. When the remote server responds with a copy of the document for the user, a copy is also stored in the file cache. If this file is requested again sometime in the near future by any user, it can be retrieved from the file cache, rather than from the remote server. Each cache hit results in a reduction in response time for the user, as it is much quicker to retrieve a file stored locally than one stored on a remote server. A cache hit also results in a reduction in the number of bytes sent across the network. Every reduction in byte traffic helps to lower the total load on the network. Reducing the network load helps to shorten the response time of requests that need to traverse the network. On a cache miss, the user pays a slight penalty in response time, equal to the time it takes to first check the file cache. However, Glassman [6] reports that this increase is not significant.

The use of file caches can also reduce the loads on IRDS servers. Every time a request for an IRDS document can be satisfied at a file cache, one fewer request must be processed by the server. Furthermore, the number of bytes that must be transferred by the server is reduced. Reducing the load on a server means that any requests that are handled by the server will be dealt with sooner, resulting in shorter response times for the user. By having file caches at a number of locations across a network, particularly at the endpoints of heavily loaded links, the load on the entire network can be reduced, as well as the load on the servers. The fact that a disk cache is relatively inexpensive when compared to the cost of a link or server upgrade makes this approach even more attractive.

There are a number of practical matters that need to be considered when a cache is to be used. First, the cache must be kept consistent. This means that when a document in the cache is requested, the user can be assured that the document is up-to-date (i.e., that it has not been modified at the remote server in the time since a copy was stored in the cache). Second, users demand a certain level of privacy and security. If a confidential document is transferred using an IRDS protocol, there must be a way for the user to avoid having a copy of the document appear in the cache, where other users can access it. While these are important issues, they are deemed to be beyond the scope of this project. The aim of this paper is to determine whether or not users at the U of S could benefit from the implementation of file caching.

The first objective of this paper is to determine the effect that a local disk cache would have on reducing the number of bytes transferred across the University of Saskatchewan's link to the Internet. Zhu [9] showed that, in April 1994, the FTP-data, WWW and Gopher protocols accounted for over 1 Gigabyte of data being transferred between the U of S Campus network and the Internet in just four days. By reducing the number of bytes that are transferred on the U of S - Internet link, the response times for users of IRDS protocols, as well as other protocols, should be improved. Previous work on file caching [5] has focussed on much busier links. The purpose of this work is to determine whether any of the benefits of file caching can be obtained at the edge of a campus network.

The second objective of this paper is to determine the effectiveness of a file cache in reducing the load on the U of S Web Server. Braun and Claffy [3] found that a relatively small portion of the NCSA Web Server file set could be cached, resulting in savings of up to 70% of the bytes transferred from the central server. The U of S Web Server does not handle loads of the same magnitude as the servers at NCSA. The purpose of this investigation is to determine whether the U of S Web Server could benefit from the presence of a file cache.

The remainder of this paper is organized as follows. Section 2 investigates the effects of file caching on the network load between the U of S Campus and the Internet. Section 3 discusses the effects of file caching on the Campus Web Server. Section 4 briefly describes related work. Finally, the conclusions of the paper are presented in Section 5.

2 The Effect of File Caching on Network Load

To determine the effect of a file cache on reducing IRDS traffic at the University of Saskatchewan, a trace-driven simulation was conducted. The trace data used in this simulation was a subset of the trace data obtained by Zhu [9] in April 1994. The cache simulation was used to determine how effective a file cache would be in reducing the number of IRDS requests that go off-campus, and the number of files that are sent to the U of S in response. Section 2.1 describes how the traces were collected. Section 2.2 discusses the analysis of the data. Section 2.3 explains how the cache simulation was conducted. Section 2.4 presents the results of the simulation.

2.1 Trace Collection

Traffic information about FTP-data, WWW and Gopher protocols was extracted from a large collection of conversations between hosts at the University of Saskatchewan and hosts on the Internet. This trace was collected in April 1994 by Zhu, as part of her M.Sc. thesis. The trace collected only the SYN, FIN and RST packets of TCP connections. Using the SYN packet, which marks the start of a TCP connection, and the FIN packet, which marks the end of a TCP connection, the source host, destination host, size and duration of a connection can be derived. By collecting only the SYN, FIN and RST packets, a substantial reduction in the size of the trace files occurs. Zhu collected the traces on a SUN Sparcstation 1+, using the TCPDUMP program. A total of 617,806 SYN, FIN and RST packets were collected over a four day period. Characteristics of the trace are given in Table 1. A more detailed description of how the trace was collected is given in [9].

Table 1: Summary of Zhu's Trace

Dates	Duration	Total SYN, FIN and RST Packets	Dropped Packets
Apr. 24-28, 1994	96 hr	617,806	0

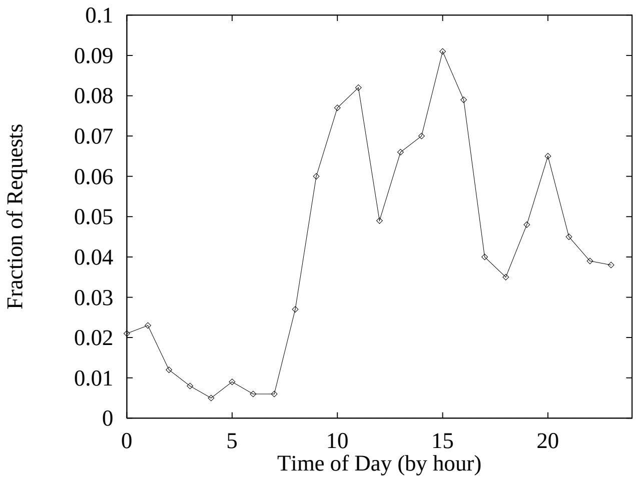

Figure 1: Distribution of IRDS Requests by Time of Day

2.2 Data Analysis

From the trace collected by Zhu, all packets that used either the FTP-data, WWW or Gopher protocols were extracted. These IRDS protocols accounted for 33% of the network traffic in Zhu's trace, and form the basis for the following analysis.

Figure 1 shows the distribution of IRDS requests by time of day. The graph shows that the requests are most common during the late morning and early afternoon hours. Figure 1 also indicates significant usage during the evening hours, likely caused by people "surfing the net" after work or school, or people trying to take advantage of lighter network loads.

By matching the SYN and FIN packets of IRDS traffic, information about each TCP connection was obtained. There were a total of 70,857 such connections. These connections were then divided into three groups. These groups consisted of:

1. requests from a user at the U of S to a server on the Internet
2. responses from a server on the Internet to the user at the U of S
3. connections (requests and responses) between users on the Internet and servers at the U of S.

The files that could be cached can be determined from the connections in group 2. These are the files that were sent by a remote server in response to a request from a user at the U of S. This collection of files was then examined in more detail.

Figure 2 indicates that a high percentage of the IRDS files are quite small. For example, 80% of all Gopher files requested were less than 4 KB in size, and 80% of all WWW files requested were less than 12 KB in size. FTP files tended to be slightly larger, as 80% of the FTP files were 24 KB or less. The file size distribution is important since it affects how many files can fit in a finite-sized file cache.

Prior to simulating a network file cache, an attempt was made to identify all unique files that were requested in the trace. Glassman [6] recommends representing a file by its Uniform Resource Locator, or URL. However, the trace data available does not contain the URL. Instead, an identifier was created for each file, created from the protocol used to transfer the file, the name of the server that sent the file, and the size of the file, in bytes. These three items were combined in this order

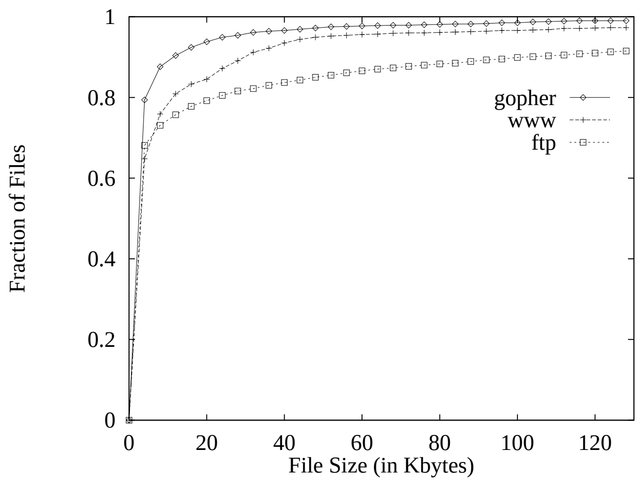

Figure 2: Cumulative Distribution of File Sizes by IRDS Protocol

to follow as closely as possible the form of a URL. For example, consider a 737 byte file delivered from the server *gowinnt.microsoft.com* using the FTP-data protocol. The resulting file identifier is *ftp-data/gowinnt.microsoft.com/737*. While this is not guaranteed to uniquely identify a file, it was deemed acceptable for an initial analysis. Each "file" in group 2 was given an identifier that was created in this manner. The results are shown in Table 2. There were 22,991 files totalling 784 MB identified as IRDS responses. From this set, using the identification method described above, 15,871 unique files totalling 690 MB were identified. Table 2 also presents the results for each protocol.

Table 2: Breakdown of IRDS Requests by Protocol Using File Identification Approach

	All Files	FTP-data	WWW	Gopher
Number of Files in Group	22,991	11,449	5,124	6,418
Number of Unique Files	15,871	7,879	3,811	4,181
Size of Requested Files (in MB)	784	649	84	51
Size of Unique Files (in MB)	690	575	74	41

2.3 Cache Simulation

The simulation of a network file cache focussed on two parameters. First, a replacement algorithm is needed, to determine which file should be removed from the cache in order to make room for a new file. Two replacement algorithms were used in this simulation: Least Recently Used (LRU) and Least Frequently Used (LFU). Second, the size of the cache must be set before the simulation. For this simulation, a number of cache sizes were used, from 1 MB up to 700 MB, which was large enough to hold all of the unique files seen in the trace.

In order to get more accurate results, the cache was "warmed-up" for a period before statistics were collected. During the "warm-up" period, responses are added to the cache as they come in. When the cache fills up, files are replaced according to the replacement algorithm being used. When the "warm-up" period ends, collection of statistics begins. Every request that can be satisfied by the cache is termed a *cache hit*, while every request that cannot be satisfied is termed a *cache miss*. We allow the cache to "warm-up" for a period before collecting statistics, because when the cache is first started, there will be an abnormally high miss rate, as the cache is empty. To obtain accurate results, the cache should reach a state of equilibrium before collecting any statistics. However, it is difficult to

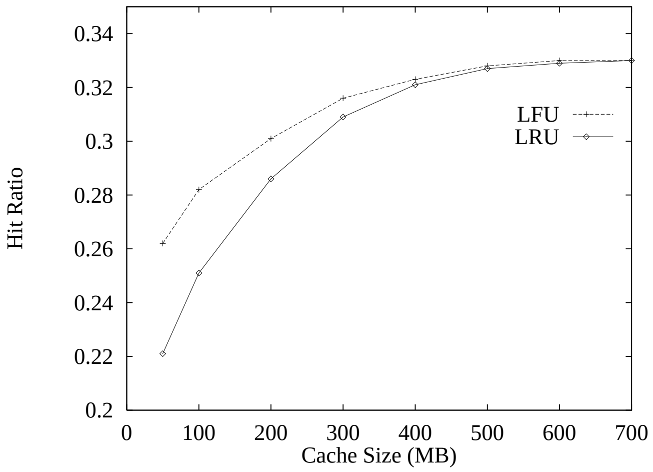

Figure 3: Cache Hit Ratios from Caching IRDS Responses

determine when this state has been reached. For this simulation, the first day of trace data was used to warm up the cache, and the remaining three days of trace data to generate statistics.

2.4 Simulation Results

The initial cache simulations used a single file cache for the files of all three IRDS protocols. Simulations were conducted to evaluate cache replacement policies, and to determine cache size.

Figure 3 shows that for small caches (50-300 MB), the LFU replacement policy performs better than LRU, but both policies perform the same for larger cache sizes. The maximum hit rate observed was approximately 33%, for a cache size of 600 MB. A cache of this size was able to hold almost the entire set of unique files. However, even when the entire set of unique files is cached, the amount of traffic generated by the IRDS protocols is not reduced significantly. Figure 4 shows that a file cache would not reduce the amount of IRDS traffic by more than 13%. The reason that the volume of IRDS data traffic was not significantly reduced is that a small number of large files are responsible for most of the bytes transferred. These files are requested very infrequently, so it does not help to cache them.

A network file cache was also simulated for the files of each IRDS protocol independently. This step was taken to see if any one protocol would show more promise than the others. Since both replacement policies performed about the same in the initial simulations, only one policy, LRU, was used for each of these simulations. The results for FTP-data traffic only were almost identical to the results in Figures 3 and 4, since FTP-data made up the bulk of the traffic. The maximum hit rate achieved was about 33%, for a file cache of 500 MB. A reduction in data traffic of 13% was also achieved with this size of cache. The file cache performed slightly better for the Gopher files, achieving a maximum hit rate of 36%, while reducing the volume of traffic by 21%, for a 40 MB cache. WWW, on the other hand, performed worse than the other protocols. The simulations using only the WWW files gave a maximum hit rate of 27%, and a 15% reduction in the volume of traffic. These rates occurred using a 60 MB file cache.

2.5 Summary

This section considered the use of a file cache to reduce the IRDS traffic at the University of Saskatchewan. IRDS trace data obtained from a trace conducted in April 1994 was used to drive a file cache simulation. The results showed that a 600 MB file cache would reduce the IRDS traffic by 13%.

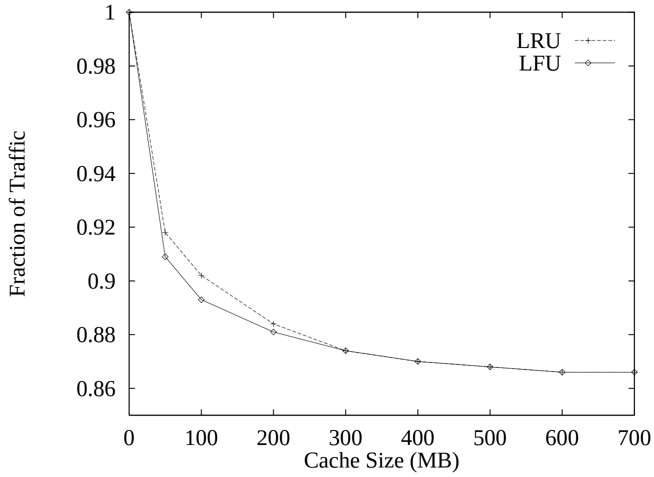

Figure 4: Effect of File Caching on IRDS Traffic Volume

3 The Effect of File Caching on a Web Server

File caching can also be used to reduce the number of requests processed by a Web server. The assessment of caching documents for the Web Servers at the University of Saskatchewan followed the methods used by Braun and Claffy [3]. The access logs of the Campus Web Server were analyzed to determine whether caching a small number of documents could substantially reduce the number of requests that must be handled by the server. The geographic sources of the requests were also looked at, to determine the area which has been served. Section 3.1 describes the Campus Web Server, Section 3.2 discusses the analysis of the Campus Web Server access logs, Section 3.3 describes the cache simulations, and Section 3.4 presents the simulation results.

3.1 Background

The Web Server for the University of Saskatchewan is a DECstation 5000/133, running NCSA httpd version 1.3. This software keeps a log of all requests made to the server for Web pages or documents. This log stores the name of the host requesting the document, the date and time of the request, the name of the requested file, and the size of the file in bytes. The Campus Web Server is maintained by the Department of Computing Services. This server is also the main campus Gopher and X.500 Server, and the DNS Name Server. At times it has trouble handling the load. This server could benefit from a file cache that reduces the number of requests that need to be handled by the server.

3.2 Data Analysis

For this project, the access logs of the Campus Web Server for the months of December 1994, January 1995, and February 1995 were analyzed. Each line of an access log contains information about one request. A normal request is of the form:

```
hostname - - [dd/mmm/yyyy:hh:mm:ss tz] request status bytes
```

where hostname is the name of the host making the request; dd/mmm/yyyy is the day, month and year of the request; hh:mm:ss is the time of the request; tz represents the timezone that the server is in; request is the desired document; status indicates whether the request was successful or not; and bytes is the size of the file requested, if applicable. An example of a normal request is:

```
alfonso.usask.ca - - [15/Sep/1994:13:50:05 -0600] "GET / HTTP/1.0" 200 1265
```

This request came from alfonso.usask.ca at 1:50:05pm CST on September 15, 1994. The requested document was the home page of the server, which was 1265 bytes in size. The status code of "200" means that the request was fulfilled by the server.

During the initial analysis of the log files, it became apparent that there were several problems. First, some of the lines contained two requests instead of just one. In this case, the problem could be fixed by placing the second request on a separate line. Second, some lines did not contain all the information about the request. Approximately 1000 requests were missing a host name. In this case, a dummy name, *foo.sk* was inserted, so that the request could still be processed. Finally, a number of the requests were missing too much information, and had to be removed. Table 3 provides a breakdown of the number of requests seen each month, and the number of requests remaining after the corrupted requests were removed. Table 3 shows that the number of requests per month is growing quite rapidly. Since the number of corrupted log entries also increases as the number of requests increases, it suggests that the load the server is faced with is causing these errors in the log file to occur. A significant number of the errors occurred when two requests arrived at the same time. These errors could possibly be avoided if the granularity of the timestamp was increased.

Table 3: Breakdown of Campus Web Server Requests by Month

Month	Total Requests	Usable Requests	Percentage
Dec 1994	44,631	42,675	95.6
Jan 1995	65,180	61,762	94.8
Feb 1995	97,204	88,794	91.3
Totals	207,015	193,231	93.3

Once the errors had been removed from the access logs, the remaining valid entries were analyzed. Figure 5 shows the distribution of request arrivals at the Campus Web Server by time of day, averaged over the entire three month period. The graph clearly shows that the heaviest load occurs during the early afternoon, which is likely the same time that the other loads that the Campus Web Server handles peak. This is another indication that this server could benefit from the implementation of a file cache.

The geographic sources of the requests were also looked at, in an attempt to determine how wide spread the sources are. In total, 57 countries and 8 other domains have sent requests to the U of S Web Server. The top ten domains are listed in Table 4. Slightly more than half of all requests originated from a host on campus, while another 18.5% originated from hosts within Canada. The remainder are scattered throughout the world.

The file size distribution for Web server requests was then studied. Based on the 193,231 valid requests, 562 MB of data were transferred by the server. However, a total of only 4,366 unique documents were referenced. These unique documents totalled only 30 MB, or 5.3% of all bytes transferred. The most requested document, the image that accompanies the U of S home page, is only 2,273 bytes. However, this one document accounts for 11.8% of all requests, and 8.8% of all bytes transferred by the server. Figure 6 reveals that a significant number of these unique files are also quite small: approximately 90% are less than 4 KB.

The 193,231 valid requests were then divided according to the status of the request, as shown in Table 5. The requests that are usable for a cache simulation are the fulfilled requests and the requests for unmodified files. Thus, 153,686 of the requests can be used in the simulation. The remaining 39,545 requests cannot be used, as they do not contain any information about cachable files.

3.3 Cache Simulation

The cache simulation involving the Web server followed the same approach as the network file caching simulation in Section 2.3. Again, two replacement policies, LRU and LFU, were used. As well, a number of cache sizes were simulated. The main difference between this simulation and the previous

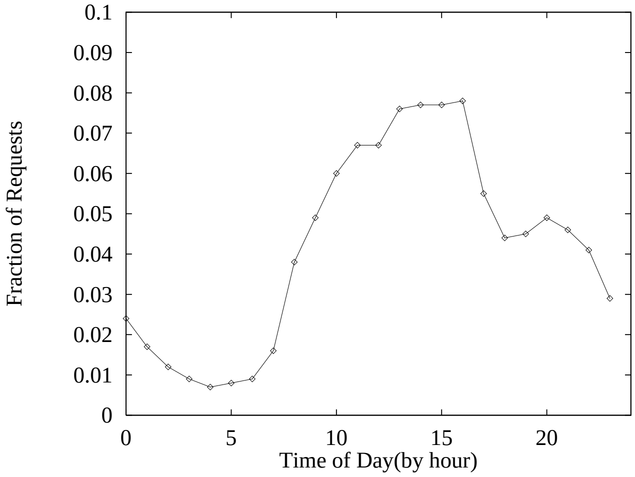

Figure 5: Requests to Campus Web Server by Time of Day

Table 4: Distribution of Requests by Domain

Domain	Percentage
.usask.ca	54.9
.ca	15.7
.com	10.8
.edu	6.0
.net	3.2
.sk.ca	2.8
.uk	0.7
.org	0.7
unknown	0.5
.au	0.4
Others	4.3

one is that the access logs clearly specify the name of the file, the path to the file, and the size of the file. From this information we can accurately determine if the requested document is in the cache. If a requested file is in the cache, but the size of the file has changed, we assume that the file has been modified. If a file has been modified, the request is marked as a cache miss, and the cached version of the file is replaced with the new version.

Once again the cache was "warmed" before collecting the statistics about the cache hit ratio and the volume of traffic transferred by the server. The requests from the month of December were used to "warm" the cache. Statistics were then computed using the requests from January and February.

3.4 Simulation Results

The simulation revealed some interesting results. Figure 7 shows that with a file cache of only 16 MB, 97% of the requests could be satisfied at the cache, and would not have to proceed to the server. At the same time, the server would only have to transfer 10% of the bytes that it currently transfers in response to these requests. Figure 8 shows this reduction in volume of data that must be delivered by the server. Clearly, Figures 7 and 8 illustrate that file caching would be beneficial to servers that are struggling to keep up with their workload.

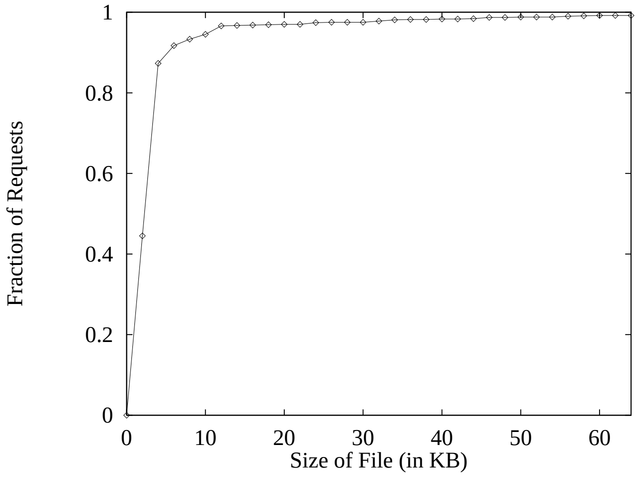

Figure 6: Cumulative Distribution of File Sizes of Requests to Campus Web Server

Table 5: Breakdown of Requests by Request Status

Status	Number of Requests
Fulfilled Request	136,856
Relocated File	37,836
Unmodified File	16,830
Not Found	1445
Bad Requests	235
Forbidden Requests	18
Internal Errors (Server)	11
Total	193,231

3.5 Summary

This section assessed the effectiveness of a file cache in reducing the number of requests processed by a Web server. Access logs of the University of Saskatchewan Campus Web Server were used to drive a file cache simulation. The results showed that a 16 MB file cache would satisfy 97% of the requests to the server, and reduce the number of bytes transferred by the server by 90%.

4 Related Work

Currently, there is a substantial amount of interest in file caching for IRDS services [3, 5, 6]. For example, in January 1994, a caching relay was set up at DEC in Palo Alto, California [6]. This caching relay caches both WWW and Gopher files. DEC established the relay to reduce external network traffic. Early results indicate that 30-50% of all WWW and Gopher requests are satisfied by the file cache [6]. Response times are reduced by up to 80% when the document is retrieved locally, rather than from the remote server.

File caching also plays an important role in an information retrieval system called Harvest [1]. What is quite interesting about this system is it attempts to set up a hierarchy of file caches. When a document is not present in one file cache, the next cache up the hierarchy is checked. This is much more complex than the file cache considered in this paper, although potentially much more powerful.

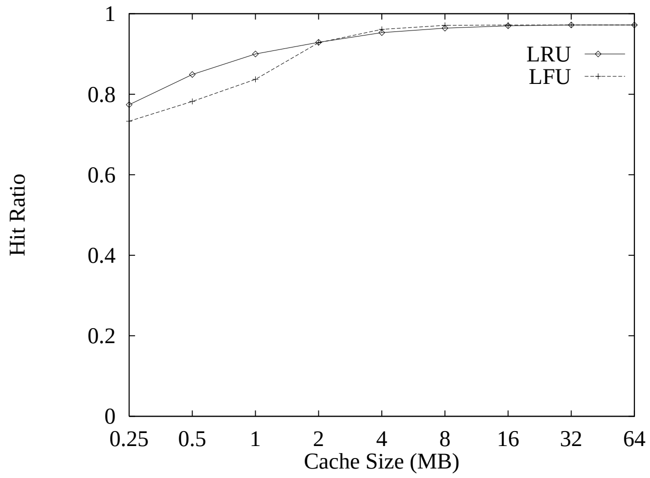

Figure 7: Cache Hit Ratio from Caching Requests to Campus Web Server

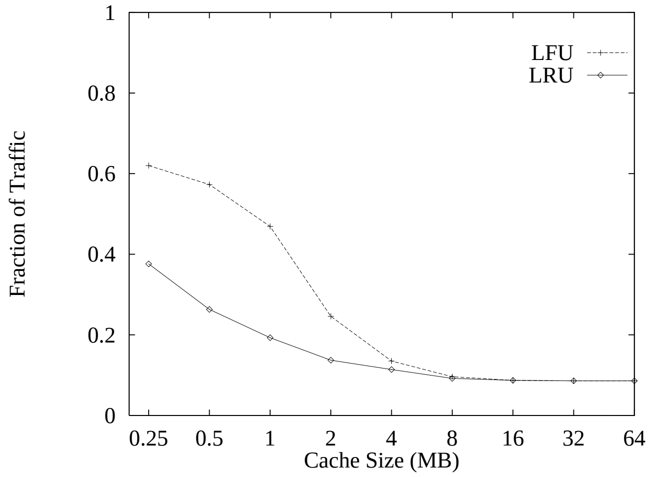

Figure 8: Effect of File Caching on IRDS Traffic Volume

5 Conclusions

This paper has presented a preliminary analysis of the effect of file caching on U of S - Internet traffic, particularly for IRDS protocols.

The initial results presented in this paper show that a file cache at the University of Saskatchewan might reduce the network traffic for incoming IRDS responses slightly. Although there is a large number of small files requested, the working set of files is quite large, which causes a low cache hit rate. Furthermore, most of the bytes transferred are due to a few large files. These files are not requested frequently, so caching would not help. The main benefit to reduced network traffic would be a decrease in user response times. However, when issues such as cache consistency and the administrative complexity are considered, the benefit of such a cache appears to be minimal.

On the other hand, initial results indicate that a small file cache could significantly reduce the number of requests that a Web server must handle, as well as the number of bytes that it must transfer. The implementation of file caches could significantly improve the performance of Web servers that are faced with a heavy load.

While the results for network file caching do not appear to be promising, the results might have been different if newer data had been available. The traces used in this investigation were taken during April 1994. Since that time, the traffic profile of the NSFNET has changed considerably. Similar changes are likely to appear in the traffic profile at the U of S. This change in traffic profile may make file caching a more appealing option. New traces should be taken to verify this hypothesis.

The results from the study of Web server requests are much more encouraging. Further investigation, such as how and where to implement a file cache to relieve an overloaded server remain to be done.

References

[1] Bowman, C. M., Danzig, P. B., Hardy, D. R., Manber, U., and Schwartz, M. F. Harvest: A Scalable, Customizable Discovery and Access System. Tech. Rep., CU-CS-732-94, Department of Computer Science, University of Colorado-Boulder, 1994.

[2] Bowman, C. M., Danzig, P. B., Manber, U., and Schwartz, M. F. Scalable Internet Resource Discovery: Research Problems and Approaches. *Commun. ACM 37*, 8(Aug. 1994), pp. 98-107.

[3] Braun, H.-W., and Claffy, K. Web Traffic Characterization: an Assessment of the Impact of Caching Documents rom NCSA's Web Server. In *Proceedings of the Second International World Wide Web Conference*, (Chicago, Illinois, October 1994).

[4] Dahlin, M. D., Mather, C. J., Wang, R. Y., Anderson, T. E., and Patterson, D. A. A Quantitative Analysis of Cache Policies for Scalable Network File Systems. In *Proceedings of ACM SIGMETRICS '94* ACM, Nashville, 1994, pp. 150-160.

[5] Danzig, P. B., Schwartz, M. F., and Hall, R. S. A Case for Caching File Objects Inside Internetworks. In *Proceedings of ACM SIGCOMM '93*, ACM, New York, 1993, pp. 239-248.

[6] Glassman, S. A Caching Relay for the World Wide Web. In *Proceedings of the First International World Wide Web Conference*, (Geneva, Switzerland, May 1994).

[7] NSFNET Statistics. Data available by anonymous ftp from nic.merit.edu/statistics/nsfnet.

[8] Worrell, K. J. Invalidation in Large Scale Network Object Caches. M.Sc. Thesis, Department of Computer Science, University of Colorado-Boulder, 1994.

[9] Zhu, W. M. Characterizing Wide Area Conversations on the Internet. M.Sc. Thesis, Department of Computational Science, University of Saskatchewan, July 1994.