Examining the Cacheability of User-Requested Web Resources

Craig E. Wills and Mikhail Mikhailov

Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609
{cew,mikhail}@cs.wpi.edu

WPI-CS-TR-99-01

Abstract

This paper continues work to monitor and better understand the characteristics of resource changes at servers and how these servers report meta data about the resources. In contrast to previous work, which studied selected resources from popular web sites, this work is based on an actual trace of user requests. This approach allows study of a set of resources that users are known to be retrieving.

The results show that there is potential to reuse more cached resources than is currently being realized due to inaccurate and nonexistent directives. For example, over 33% of HTML resources in the study do not change, but contain no last modification time or other cache directive in the response, so these resources cannot be cached and validated with the origin server. In addition, embedded images are often reused, even in pages that change frequently. This result both points to the need to cache such images and to discard them when they are no longer included as part of the page.

The last result of this work is that the inclusion of a cookie as part of a request does not make the response uncacheable. In most cases we obtained identical responses from two requests for the same URL with different cookies. In cases where the responses are not the same, they often differ only in the ad image contained. These results imply such responses can be cached if other cache directives allow for it.

1 Introduction

This paper describes work to monitor and better understand the characteristics of resource changes at servers and how these servers report meta data about the resources. It extends our own previous work [19] in this area using similar methodologies on new data and studying additional issues. The work fits into the long-term goal of our project to both examine the effectiveness of current caching techniques in light of more complete data, and also to investigate the potential of caching if improved techniques are used by Web caches and servers.

Previous work focused on the initial part of our project—characterizing information about Web resources and server responses that is relevant to Web caching [19]. The approach was to study a test set of URLs at popular web sites and gather statistics about the rate and nature of changes correlated with the resource type. In addition, we gathered response header information reported by the servers with each retrieved resource. Other studies used proxy and server logs or network traces of user requests/responses, which constrained the resulting studies to the available data. In contrast, our approach retrieves each resource in the test set at intervals and for a duration appropriate for characterizing the nature of each resource in the test set. In addition, logs and traces are affected by browser and "lower-level" proxy caches, which hide some of the requested resources. Our approach is to disable caching for more complete data gathering.

Results from our previous work indicate that there is potential to reuse more cached resources than is currently being realized due to inaccurate and unavailable cache directives. In addition, the relationships between resources used to compose a page must be considered for caching. Embedded images are often reused, even in pages that change frequently. This result both points to the need to cache such images and to discard them when they are no longer included as part of the page. Finally, while the results show that HTML resources frequently change, these changes can be in a predictable and localized manner. Separating out the dynamic portions of a page into their own resources allows relatively static portions to be cached, while retrieval of the dynamic resources can trigger retrieval of new resources along with any invalidation of already cached resources.

This paper focuses on two new aspects:

- Uses an actual trace of user requests. While the previous study was made with data from popular web sites, the resources retrieved from the site (the home page and its embedded images along with its traversal links and their embedded images) may not be the most frequently retrieved resources at that site. Using an actual proxy trace allows us to concentrate on a set of resources that users are known to be retrieving. We study how results using this new data set relate to our previous results.

- Studies the impact of cookies on caching. In our previous work, we did not investigate differences in resource contents due to cookies contained in the request. In this study we make requests containing different cookie values previously returned by servers in making subsequent requests and study differences in the resulting resource contents.

In the remainder of this paper we describe our study and its results. The following section summarizes the information we are seeking in our study. The next section discusses

the methodology we use in obtaining this information. The middle portion of the paper presents the results from our study on the test sets we use and compares these results to our previous work along with a discussion on possible implications of these results. The paper concludes with a description of related work, a discussion of future work and a summary of our work to date.

2 Study

The general goal of our work is to better understand the nature of how resources change at a collection of servers and how meta information reported by servers reflects those changes. The overriding goal of this work is obtain data that can be used to better understand the potential benefits of caching and whether existing software is reaching this potential. The following summarizes specific directions for investigation from our previous work [19], which are also studied in this work.

- Monitor selected resources to study the frequency at which these resources change. A similar study was done using a packet trace [7], but with our approach we can control what requests are made and test whether resources change using an MD5 checksum of contents to determine when changes occur.

- Examine the availability and accuracy of cache validation information reported by servers for requested resources. The approach is to monitor response headers returned along with a resource to discover lmodtime, size, and entity tag (Etag) information. We also measure the use and accuracy of explicit cache directives returned by servers such as the Expires header along with the Cache-Control header in HTTP/1.1 and Pragma:no-cache header in HTTP/1.0.

- Examine how images and other embedded resources change relative to the HTML resources they are contained in. Prior work indicates that images do not change at the same rate, but how does the use of embedded images change as these container resources change?

More details on these directions are available in [19], which studied them for resources at popular web sites. We repeat our study of these directions in this work for a different set of resources—those that are most frequently requested of a proxy server. In addition to these directions, a new direction of this work is to better understand how servers respond to different types of requests for the same resource. One type of variation is whether servers are supplying cookies that clients are then including as part of subsequent requests. A recent study found that 30% of the requests made in a client trace included cookies, concluding that responses to these requests are uncacheable [3, 10]. This result raises a number of questions. Is there a similar proportion of server replies that contain cookies for our test set? Does the inclusion of a cookie in a request always result in a different resource response than obtained with a request containing no cookie? Do two separate requests with two separate cookies always result in different resource responses? We believe answers to these questions will provide us with a better picture of the impact of cookies on caching. If resource content does

not always change in response to different request cookies (or absence thereof), then such resources could be cached and be used as a base for validation on subsequent requests.

3 Methodology

In our previous study we identified frequently used sites and focused on resources at those sites [19]. In this study we gathered a set of URLs from current NLANR proxy access logs [16]. These logs are from an upper-level cache typically servicing requests not satisfied by caches closer to clients making the requests. This approach has the advantage of focusing on URLs actually being retrieved by users across a number of different servers and content types.

The methodology of the study is to divide the URLs into test sets and perform an unconditional HTTP GET for each of the URLs in a test set on a daily basis. The time between successive retrievals for a URL may be lengthened or shortened as needed, but for this work we used a retrieval interval of one day. For each retrieved resource, we store response headers and calculate an MD5 checksum on the contents. Contents of HTML and text resources are stored if changed from the previous retrieval. Once an HTML resource is retrieved, it is parsed and all embedded images and traversal links are recorded. As described in the following section, we also retrieve all embedded images for one of the test sets and calculate their MD5 checksums.

For additional study, we identified a subset of the URLs returned with a cookie. For these URLs, we did further investigation of the dependence of the retrieved resource on the cookie value. The approach is to initially retrieve each URL twice, recording the cookie returned each time. On subsequent iterations, three retrievals are made, one with no cookie sent, one with the first cookie sent and one with the second cookie sent. In the latter two cases, the value of any new cookie returned by the server is stored for subsequent iterations. Analysis of the three resources returned on each iteration allows us to better understand if and how varying the cookie changes the response.

4 Results

This section gives information about the test sets used in our studies and provides answers to questions raised in Section 2.

4.1 Test Sets

Seven daily proxy traces were obtained from NLANR [16] for the late December, 1998 to early January, 1999 time period. These traces were processed to extract all HTTP GET requests for valid URLs that resulted in 200 or 304 HTTP responses. Over 1.8 million accesses were found in the traces containing a content type. The URLs from these accesses encompassed 214,000 distinct URLs from over 33,000 distinct servers.

We chose to focus our study on non-image URLs in the traces because images are primarily retrieved as embedded images in HTML container pages and can be retrieved as needed by our study. As a consequence, images, accounting for 74% of accesses, were eliminated from

the study set. To further prune this set, we considered only URLs with a reference count of 10 or greater, which resulted in 4122 URLs. Of these we eliminated all queries—URLs containing a "?". Such queries could not be used because all parameters after the question mark were sanitized in the trace data making replication of such requests impossible. A preliminary study of query responses was done in our previous work [19]. We further reduced the resulting set of 3237 URLs by removing all non-existent URLs and URLs referenced fewer than 20 times. The final set contained 1129 URLs.

These URLs were divided into five test sets based upon their content type and reference count.

1. cnt100: resources with content type text (html, css and plain) in the list with 100 and more references. All embedded images for this set are retrieved in addition to the resource itself.

2. cnt20: resources with content type text (html, css and plain) in the list with 20-99 references. Embedded images for this set are not retrieved to save on system resources and because previous work has shown images rarely change [7, 19].

3. audio: resources with content type of audio.

4. appldata: resources with application content type of either octet-stream or zip.

5. appltext: other resources with application content type, primarily javascript.

Summary statistics about all test sets are given in Table 1. While headers from all responses were saved and catalogued, the table focuses on statistics related to caching and content type.

Table 1: Summary Information for Test Sets

	Test Set				
Item	cnt100	cnt20	audio	appldata	appltext
Number of Base URLs	122	927	7	10	63
Number of Resources	1131	927	7	10	63
Pragma/Cache-Control	3.7%	12.4%	0.0%	0.0%	12.7%
Expires	5.3%	12.0%	0.0%	0.0%	0.0%
Last-Modified Time	79.3%	33.0%	100.0%	100.0%	54.0%
Etag	31.8%	22.0%	100.0%	40.0%	28.6%
Set-Cookie	9.1%	21.6%	0.0%	10.0%	31.7%
Content-Type: HTML/text	10.8%	100.0%	0.0%	0.0%	0.0%
Content-Type: image	89.2%	0.0%	0.0%	0.0%	0.0%
Number of Repeated Resources	754	927	7	8	63
Content-Type: HTML/text	15.8%	100.0%	0.0%	0.0%	0.0%
Content-Type: image	84.2%	0.0%	0.0%	0.0%	0.0%

In examining the top half of Table 1, only the cnt100 test set shows images because embedded images are retrieved. The ratio of images is comparable to our previous study for popular web sites [19]. Other results are also comparable. The cnt20 and appltext test sets show a relatively low percentage of resources that include a last-modified time (lmodtime) in the response. Further examination of cnt100 HTML resources shows that only about 30% of them provide lmodtime information. This situation hinders caching as it does not allow cached resources to be validated with the origin server.

The cnt100 test set contains more resources than the base set because embedded images are retrieved as well. Some of these embedded images are contained in only one instance of the container resource hence the smaller number of repeated resources in the lower portion of Table 1. These repeated resources are used to study changes in resource contents. In our previous work we tried to further classify HTML resources as "static" or "dynamic" based on applying heuristics to the resource name, but found little difference in the characteristics of resources in the sub-categories. We found a similar "non-result" in this work and do not show any results based on this sub-categorization.

4.2 Rate of Change

Our first step in analyzing the data was to repeat the rate of change calculations as done by Douglis, et al [7] and our own previous work [19]. Figure 1 shows the results for all test sets with the cnt100 test set broken down into text/html resources and images. The images, audio and appldata show little or no change. The HTML resources exhibit less change than the HTML resources in our prior study. Here between 40-50% of such resources do not change while this value was between 10-20% for the commercial web sites in the previous study [19].

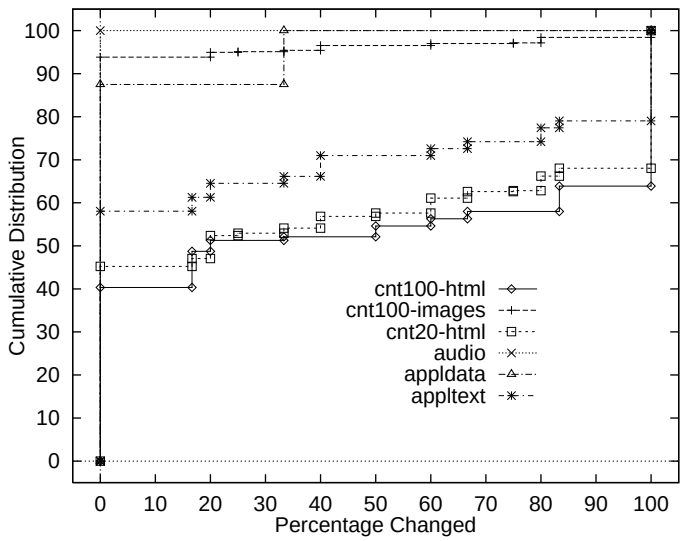

Figure 1: Cumulative Distribution of Test Set Change Ratio

4.3 Cache Validation Information

We examined the availability and accuracy of cache validation information returned by Web servers for a resource. Tables 2 and 3 show the data for three potential cache validators: last modification time, entity tags and content-length. The lmodtime is currently the most common validator for a cached resource in combination with the "If-Modified-Since" header sent with a GET request. Etags are an available validator for HTTP/1.1. Content-Length is not explicitly used for validation, but has been used in prior simulation studies based only on proxy or server logs as a means to determine when resources change.

Table 2: Correlation of HTTP Validators to MD5 Content Changes for All Cnt100 Resources (%)

	LModTime			Etag			Content-Length		
MD5	chng	noch	unav	chng	noch	unav	chng	noch	unav
chng	3.64	0.05	6.81	1.13	0.00	9.38	5.34	0.07	5.10
noch	1.04	79.81	8.64	9.43	26.16	53.90	0.00	83.50	6.00

Table 3: Correlation of HTTP Validators to MD5 Content Changes for All Cnt20 Resources (%)

	LModTime			Etag			Content-Length		
MD5	chng	noch	unav	chng	noch	unav	chng	noch	unav
chng	13.09	0.73	28.73	8.75	0.02	33.78	14.39	2.14	26.02
noch	5.93	14.18	37.35	7.61	5.42	44.42	0.20	22.16	35.10

As Table 2 shows, the lmodtime is generally available and generally corresponds to whether or not the resource changes for the cnt100 test set, which contains a high percentage of images. As in our previous study [19], there are some cases where the lmodtime changes, but the resource does not and even a few cases where the lmodtime does not change, but the resource does. Another problem is the large percentage of resources that contain no validators, particularly for the cnt20 resources with only HTML content type. With no validator information, a cache cannot conditionally validate the contents of a cached resource.

Correlation of MD5 with the three potential validators does not take into account no-cache directives and explicit directives on the amount of time a resource can be cached. In the former case we include retrieved resources with the "pragma:no-cache," "cache-control:no-cache," "cache-control:no-store" and "cache-control:private" headers and in the latter case we include resources with the "expires" header. Tables 4–6 show the results for the cnt100 (divided into HTML and image resources) and cnt20 test sets by first considering whether a resource can be cached, then if it has an expiration time and finally if its lmodtime is available. In each case, we correlate the results against content changes using the MD5 checksum.

Table 4: Correlation of HTTP Cache Directives to MD5 Content Changes for Cnt100 HTML (%)

	No	Expires							LModTime		
MD5	Cache	past	<1h	<1d	<1w	<1m	<1y	>1y	chng	noch	unav
chng	7.45	3.50	1.26	0.84	1.26	0.00	0.00	0.00	5.60	0.00	26.69
noch	6.83	0.70	0.00	0.00	0.00	0.00	0.84	0.00	1.68	9.52	33.81

Table 5: Correlation of HTTP Cache Directives to MD5 Content Changes for Cnt100 Images (%)

	No	Expires							LModTime		
MD5	Cache	past	<1h	<1d	<1w	<1m	<1y	>1y	chng	noch	unav
chng	0.00	0.00	0.16	0.00	0.60	0.00	0.00	0.00	1.86	0.06	0.93
noch	0.31	0.00	0.00	0.16	0.19	0.00	1.57	0.00	0.74	90.25	2.69

Table 6: Correlation of HTTP Cache Directives to MD5 Content Changes for Cnt20 HTML (%)

	No	Expires							LModTime		
MD5	Cache	past	<1h	<1d	<1w	<1m	<1y	>1y	chng	noch	unav
chng	5.96	2.69	0.76	0.25	0.09	0.00	0.00	0.02	9.76	0.33	23.11
noch	3.98	2.13	0.39	0.15	0.02	0.00	0.22	0.09	5.23	11.66	33.61

The expiration results are broken down into time periods based on whether the given expiration time is in the past, whether it is less than one hour, whether it is greater than one hour, but less than a day and continuing for one week, month and year. The results show that some resources have explicit no-cache directives and relatively few contain expiration time information. The most noticeable statistic for caching is the large percentage of HTML resources that contain no cache control directive, expiration time nor lmodtime. Over 33% of the resources in each HTML set do not contain any cache directive and do not change.

As a final direction for studying the availability and accuracy of cache directives we use the data in Tables 4–6 along with similar data for the audio, appldata and appltext test sets to calculate the current and potential reusability of cached resources. For this calculation, we first determine which columns of Tables 4–6 can be cached and sum the percentages in these columns. These columns are the <1m, <1y and >1y expiration periods along with the lmodtime validations that are unchanged. A cached resource from any of these columns would be reused by the cache. Due to our testing methodology, resources were retrieved in a little under and a little over a day. Thus we consider that unchanged resources in the <1d and <1w columns to be reported correctly and reusable. We assume the remaining cache "buckets" cannot be reused for retrieval after a day—resources with explicit no-caching directives, resources with past or short expiration times, resources where the lmodtime has changed or the lmodtime is unavailable. Using these assumptions the second column in Table 7 shows the reusability of cached resources for each category.

Table 7: Current and Potential Reusability of Cached Resources (%)

Resources	Current Reuse	Stale Reuse	Additional Reuse	Potential Reuse
cnt100 HTML	10.36	-0.00	+41.34	51.70
cnt100 image	92.23	-0.06	+3.43	95.60
cnt2 HTML	12.49	-0.35	+45.34	57.48
audio	100.00	-0.00	+0.00	100.00
appldata	95.83	-0.00	+0.00	95.83
appltext	46.56	-0.00	+22.85	69.41

The results show a small amount of reuse available for the HTML resources, a larger amount for appltext (javascript) resources and a high reusability for other resources. The third column of Table 7 shows the percentage of stale resources that would be returned, where the cached resource is considered current using the cache directive, but in fact the resource has changed. Similarly, the fourth column shows the percentage considered not reusable when in fact the resource did not change. The summation of the three columns yields the fifth column which is the potential reusability for each of the categories. It demonstrates there is potential improvement in the accuracy of current cache directives. The biggest impediment to reuse are cases with missing lmodtimes for unchanged resources.

4.4 Characteristics of Embedded Images

The rate of change results in Section 4.2 and reusability results in Section 4.3 indicate that HTML resources change frequently. However, what these results do not indicate is the nature and degree of changes. Because HTML resources are often "containers" for embedded images we examine the number of embedded images that remain in an HTML resource between successive retrievals. Table 8 provides results on the number of images that remain between successive retrievals of an HTML page in the two relevant test sets.

Table 8: Number of Embedded Images and Traversal Links Remaining in an HTML Page Between Successive Retrievals

	Test Set	
Item	cnt100	cnt20
Number of HTML Pages	119	920
Ave Number of Embedded Images Per Page	10.31	17.06
Ave Number of Remaining Embedded Images	6.61	8.03
Ave Number of Links Per Page	43.97	42.39
Ave Number of Remaining Links	37.92	34.49

The results show that the percentage of images remaining is a little over and a little under a half. Similar to our previous results [19], these results have two significant implications for caching:

1. despite the fact that HTML resources change frequently there is a significant amount of reuse of images, and

2. cache replacement policies need to associate an image with its container resource so that if the resource no longer contains the image then it should be garbage collected and removed from the cache.

In a fashion similar to embedded images, Table 8 also shows the frequency at which traversal links remain the same between successive retrievals. While not having direct implications for caching, the results show a significant ratio of links remain between retrievals.

4.5 Cookies

Our final analysis of changes to resources examines the nature of changes as they relate to the use of cookies as part of the request. Our previous study and prior work in this study did not include a cookie as part of a request, even if returned from the previous request. For this analysis we used the 94 resources from cnt100 test set that contained a "Set-Cookie" header as part of their response on our initial retrieval. This cookie test set contains 38 HTML and 56 image resources. To better understand the overall change characteristics of these resources, independent of changes due to cookies, we used our daily data retrieval results to calculate resuability information as was done in Table 7. The reusability results

for these 94 resources are shown in Table 9. The results show that these resources are a bit less reusable when compared to the corresponding cnt100 statistics in Table 7 both using current cache directives and when adding in additional reuse due to inaccurate or unavailable cache directives.

Table 9: Current and Potential Reusability of Cached Resources Returning Cookies (%)

Resources	Current Reuse	Stale Reuse	Additional Reuse	Potential Reuse
cookie HTML	5.70	-0.00	+33.60	39.30
cookie image	58.97	-0.00	+30.04	89.01

In comparing changes to these resources based on cookies, we initialized the cookie test set by retrieving each resource twice and recording the two cookies (cookie1 and cookie2) obtained with each response. At a later time we made three retrievals for each resource: one with cookie1, one with cookie2 and one with no cookie. We then compared the MD5 checksums of each resource content returned. The results of this comparison are shown in Table 10 for all resources in the test set and for each content type. The tests were repeated a few days later with similar results.

Table 10: Comparison of Retrieved Content for Different Requests with Cookies

	All Resources		HTML Resources		Image Resources	
Comparison	Same	Different	Same	Different	Same	Different
cookie1 vs. cookie2	76%	24%	57%	43%	87%	13%
cookie vs. no-cookie	73%	27%	51%	49%	87%	13%

The results show that a majority of HTML and most image resource responses are the same for requests with two different cookies or for a cookie and non-cookie request. Further examination of the HTML source code for resources with different responses for two different cookies showed that only ad images changed in most of these cases. These results indicate that responses with cookies can be cached and in most cases the cached content can be reused for subsequent requests. As indicated in Table 9, better use of caching directives by servers would allow more of this potential reuse to be realized.

Our results indicate that assumptions made in previous work about the uncacheability of responses with cookies may not be valid [3, 10]. Rather a more appropriate approach with such responses is for a proxy to cache the resource contents and use a conditional GET request for subsequent requests with different cookies. Such an approach is used in the newest version of the Squid proxy cache, although the previous version did not allow responses with cookies to be cached [18].

Aside from analysis of the resource contents, we also examined the "Set-Cookie" header in responses to requests with cookies. These results are shown in Table 11 for all resources (there was little variation based on resource type).

Table 11: Cookie Responses for Different Requests with Cookies

	Request Type	
Cookie Response	with Cookie	with No-Cookie
No cookie	67%	10%
Same cookie, same value	0%	0%
Same cookie, new value	20%	0%
New cookie	13%	90%

The results show that the presense of a cookie in the request resulted in no cookie being returned in two-thirds of the cases. In other cases, responses either contained a cookie with the same name and path, but different value, or a new cookie altogether. These results indicate problems for basing caching decisions on the response headers alone. Many requests with cookies will not result in a response with a cookie even though that response may be specific to the cookie.

5 Implications for Web Caching

The results of this study generally coincide with those from our prior work and hence the implications for better realizing the potential of caching discussed in that work still hold. The results show that in many cases there are problems in using the set of validators currently available in HTTP. Lack of accuracy causes a few stale resources to be reused and many more unchanged resources to be unnecessarily retrieved. An even larger problem found in this study is the large number of cases where the last modification time is not available making validation of such resources difficult in the absence of other cache directives.

One alternative is the use of Etags, which could be generated by a server as appropriate for a resource. Unfortunately the results show that when Etags currently are used they do change even in cases where the resource contents do not. For long-term caching effectiveness, Etags need to be used correctly and could incorporate reliable validators such as MD5 checksums.

Results from Section 4.4 show the importance of taking into account the relationships between resources when caching them. Specifically, a cache needs to maintain a set of "pointers" to an embedded image from one or more container HTML resources. If the HTML resources change and no longer contain the image, then this image should be garbage collected and removed from the cache.

In addition, new results from this work indicate little variation in resource contents based on the presence or absence of cookies in the request. In contrast to previous studies that considered responses with cookies as uncacheable [3, 10], these results imply such responses may be cached and reused, perhaps after validation with the origin server. In cases where the origin server is simply using cookies to track the activity of individual users, the proxy cache could immediately return the cached contents to the client and forward on the cookie request to the origin server. This approach minimizes response time for the client and keeps the server informed of user interests.

6 Related Work

There has been much related work on both web characterization and caching, but none that has focused specifically on characterization for improved caching. We have drawn on other web characterization studies in trying to understand and classify our results [1, 9, 11, 14, 17].

Previous work we and others have done on web caching [3, 5, 12] has motivated this work in trying to better understand the potential of web caching. Kroeger, et al [13] published a previous study on the potential of caching and prefetching in reducing web latency. That study was based solely on data from a proxy log, which limits the type of information available.

Other work has examined approaches to cache HTML resources that change frequently, but in a localized and predictable manner. Delta encoding [6, 7] and cachelets [4] are two approaches for handling such changes. We and others have proposed separating the static and dynamic portions of Web page to allow caching of the static portions [8, 19].

7 Future Work

There are a number of directions for work that we plan to pursue, with three of these directions receiving the most interest. In our previous study [19], we did some monitoring of the characteristics of query results and tracking actual changes to resources. The results from that study indicate that even frequently changing pages often exhibit deterministic and predictable changes. While techniques such as delta-encoding [15] and cachelets [4] have been proposed as possible solutions for dealing with such changes, we believe an approach that structures pages into resources of similar characteristics shows promise. The idea is to compose pages as a set of resources where each resource has similar characteristics—not just type, but the frequency at which it changes. Other work [8] has proposed a similar idea. We would like to explore these ideas more completely.

Another direction for investigation is a closer study of ideas for caching improvements. A proxy log or a workload generator such as SURGE [2] might serve as a basis for a simulation study to be performed. Such a study could be used to investigate how improvements in caching mechanisms translate into better cache hit rates and reduced latency.

A final direction for study is the impact of mirror sites on caching. Casual observation of the URLs with 10 or more accesses showed a number of sites with similar server names. We also found 69 cases where the same non-trivial path (containing at least two components) was found at two or more servers. While we did not examine the respective contents of these potential duplicates, preliminary results indicate that identification of mirror sites could be beneficial to improving caching.

8 Summary

In summary, we believe our work makes important contributions by using a methodology that focuses on Web characteristics as they relate to caching. The advantage of this approach is that we can study the characteristics of a set of user-requested resources without being constrained by the data from a set of logs or packet traces. One of the results of our work is

a detailed study on the availability and accuracy of existing cache directives. These results indicate that there is potential to reuse more cached resources than is currently being realized due to inaccurate and nonexistent directives.

In terms of implications for caching, the relationships between resources used to compose a page must be considered. As in previous work, we found embedded images are often reused, even in pages that change frequently. This result both points to the need to cache such images and to discard them when they are no longer included as part of the page. Current caches treat each resource separately and may unnecessarily continue caching an embedded image long after it has been removed from its page.

The last result of this work is that the inclusion of a cookie as part of a request does not make the response uncacheable. In most cases we obtained identical responses from two requests for the same URL with different cookies. In cases where the responses are not the same, they often differ only in the ad image contained. These results imply such responses can be cached if other cache directives allow for it.

9 Acknowledgements

This work would be impossible but for the logs provided by the National Laboratory for Applied Network Research supported by National Science Foundation grants NCR-9616602 and NCR-9521745. Thank you.

References

[1] M. Arlitt and C. Williamson. Web server workload characterization. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, May 1996.

[2] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. In *Proceedings of the ACM SIGMETRICS '98 Conference*. ACM, June 1998.
`http://www.cs.bu.edu/faculty/crovella/paper-archive/sigm98-surge.ps`.

[3] Ramon Caceres, Fred Douglis, Anja Feldmann, Gideon Glass, and Michael Rabinovich. Web proxy caching: the devil is in the details. In *Workshop on Internet Server Performance*, Madison, Wisconsin USA, June 1998.
`http://www.cs.wisc.edu/~cao/WISP98/final-versions/anja.ps`.

[4] P. Cao, J. Zhang, and K. Beach. Active cache: Caching dynamic contents (objects) on the web. In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98)*, The Lake District, England, September 1998.
`http://www.cs.wisc.edu/~cao/papers/active-cache.html`.

[5] Pei Cao and Sandy Irani. Cost-aware WWW proxy caching algorithms. In *Symposium on Internet Technology and Systems*. USENIX Association, December 1997. `http://www.usenix.org/publications/library/proceedings/usits97/cao.html`.

[6] Mun Choon Chan and Thomas Woo. Cache-based compaction: A new technique for optimizing web transfer. In *Proceedings of the IEEE Infocom '99 Conference*, New York, NY, March 1999. IEEE.

[7] Fred Douglis, Anja Feldmann, Balachander Krishnamurthy, and Jeffrey Mogul. Rate of change and other metrics: a live study of the world wide web. In *Symposium on Internet Technology and Systems*. USENIX Association, December 1997. `http://www.usenix.org/publications/library/proceedings/usits97/douglis_rate.html`.

[8] Fred Douglis, Antonio Haro, and Michael Rabinovich. HPP: HTML macro-preprocessing to support dynamic document caching. In *USENIX Symposium on Internet Technology and Systems*, Monterey, California, USA, December 1997. USENIX Association. `http://www.research.att.com/~douglis/papers/hpp.ps`.

[9] Brad Duska, David Marwood, and Michael J. Feeley. The measured access characteristics of World Wide Web client proxy caches. In *USENIX Symposium on Internet Technology and Systems*, Monterey, California, USA, December 1997. USENIX Association. `http://www.usenix.org/publications/library/proceedings/usits97/duska.html`.

[10] Anja Feldmann, Ramon Caceres, Fred Douglis, Gideon Glass, and Michael Rabinovich. Performance of web proxy caching in heterogeneous bandwidth environments. In *Proceedings of the IEEE Infocom '99 Conference*, New York, NY, March 1999. IEEE. `http://www.research.att.com/~anja/feldmann/papers/infocomm99_proxim.ps.gz`.

[11] Steven D. Gribble and Eric A. Brewer. System design issues for internet middleware services: Deductions from a large client trace. In *USENIX Symposium on Internet Technology and Systems*, Monterey, California, USA, December 1997. USENIX Association. `http://www.usenix.org/publications/library/proceedings/usits97/gribble.html`.

[12] Balachander Krishnamurthy and Craig E. Wills. Piggyback server invalidation for proxy cache coherency. In *Seventh International World Wide Web Conference*, pages 185–193, Brisbane, Australia, April 1998. Published in Computer Networks and ISDN Systems (30)1-7 (1998) pp. 185–193. `http://www.cs.wpi.edu/~cew/papers/www7/www7.html`.

[13] Thomas M. Kroeger, Darrel D.E. Long, and Jeffrey C. Mogul. Exploring the bounds of web latency reduction from caching and prefetching. In *Symposium on Internet Technology and Systems*. USENIX Association, December 1997. `http://www.usenix.org/publications/library/proceedings/usits97/kroeger.html`.

[14] Stephen Manley and Margo Seltzer. Web facts and fantasy. In *Symposium on Internet Technology and Systems*. USENIX Association, December 1997.
`http://www.usenix.org/publications/library/proceedings/usits97/manley.html`.

[15] Jeffrey C. Mogul, Fred Douglis, Anja Feldmann, and Balachander Krishnamurthy. Potential benefits of delta-encoding and data compression for HTTP. In *ACM SIGCOMM'97 Conference*, September 1997.
`http://www.acm.org/sigcomm/sigcomm97/papers/p156.html`.

[16] NLANR. Proxy cache log traces, January 1999.
`ftp://ircache.nlanr.net/Traces/`.

[17] James E. Pitkow. Summary of WWW characterizations. In *Seventh International World Wide Web Conference*, Brisbane, Australia, April 1998.

[18] Squid internet object cache.
`http://squid.nlanr.net/Squid`.

[19] Craig E. Wills and Mikhail Mikhailov. Towards a better understanding of web resources and server responses for improved caching. Technical Report WPI-CS-TR-98-27, Computer Science Department, Worcester Polytechnic Institute, December 1998. Submitted for publication.
`http://www.cs.wpi.edu/~cew/papers/tr98-27.ps.gz`.