In *Proceedings of the Eighth International World Wide Web Conference*, Toronto, Ontario Canada, May 1999.

Towards a Better Understanding of Web Resources and Server Responses for Improved Caching

Craig E. Wills and Mikhail Mikhailov

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

Abstract

This work focuses on characterizing information about Web resources and server responses that is relevant to Web caching. The approach is to study a set of URLs at a variety of sites and gather statistics about the rate and nature of changes compared with the resource type. In addition, we gather response header information reported by the servers with each retrieved resource. Results from the work indicate that there is potential to reuse more cached resources than is currently being realized due to inaccurate and nonexistent cache directives.

In terms of implications for caching, the relationships between resources used to compose a page must be considered. 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 any 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.

Keywords: web caching, characterization, web servers

1 Introduction

There have been many studies to better understand characteristics of the Web [6, 8, 14]. Other studies have proposed improved caching policies and mechanisms [5, 12]. However, work has not been done to specifically understand how changes in Web resources and the meta information reported by servers affect caching by Web browsers and proxy caches. To address this gap we have undertaken a study to monitor and better understand the characteristics of resource changes at servers and how these servers report meta data about the resources. The long-term goal of our project is 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 were used by Web caches and servers.

This paper focuses on the initial part of our project—characterizing information about Web resources and server responses that is relevant to Web caching. The approach is to study a set of URLs at a variety of sites and gather statistics about the rate and nature of changes compared with the resource type. In addition, we gather response header information reported by the servers with each retrieved resource. Previous work 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 is to retrieve each resource in the test set at fixed intervals for a period of time. 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.

We are aware that in generating our own set of resource requests the results may not reflect a realistic mix of requests as is found in a log or a packet trace. Rather, our study focuses on characterizing resources and responses based on content type.

In comparison to previous Web characterization work, our study has two distinguishing aspects: it focuses on issues relevant to Web caching; and it uses a methodology that allows us to study changes to resources in a controlled manner. In the remainder of this paper we describe our study. The following section discusses details of what information we are seeking in our study, followed by a section discussing 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 followed by a discussion on possible implications of these results for Web servers, caches and the HTTP protocol. 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. Our work has many specific directions for investigation:

- Monitor selected resources to study the frequency at which these resources change. A similar study was done using a packet trace [6], 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 last modification time (lmodtime), size, and entity tag (Etag) information. The availability of lmodtime information is important in efficiently validating a cached resource using an If-Modified-Since (IMS) GET request in HTTP. Previous studies have found the percentage of server responses that contain the lmodtime for a resource vary from 50-80% [6, 12, 14]. Etags are an HTTP/1.1 mechanism for servers to provide an "opaque" cache validator [13]. To check the accuracy of the validator information, we calculate MD5 checksums for resource contents and compare the checksums and validators for successive retrievals of a resource. 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?

- Study the predictability and locality of changes to a resource. This is particularly important for resources that change often such as dynamically computed content. Techniques such as delta-encoding [16], HTML pre-processing [7] and active caches [4] have been proposed to allow resources that change frequently, but predictably, to be cached.

- 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 uncachable [3, 9]. 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.

3 Methodology

There are two primary issues in our approach for studying the identified questions: how to determine the test set of resources to monitor and how to do the monitoring. These issues are discussed in this section.

3.1 Test Set

The approach we used in this study is to identify frequently used sites and focus our study on resources at those sites. While such a test set may not be "representative" of a proxy trace, it provides us with a set of resources that are likely to have the most impact on long-term Web usage. We explored different sources for gathering resource usage information such as Media Metrix [15], Keynote Systems [11] and 100hot.com [18]. We use the home page from a set of web sites identified by 100hot.com as a basis for our study.

An alternate approach is to gather a set of URLs from a relatively current proxy log trace. This set of URLs can then be tracked using our methodology. This approach has the advantage of focusing on URLs actually being retrieved by users across a number of different servers and content types. However, it has the disadvantage of being biased by the particular user group encompassed by the trace.

We believe there is not a single best test set and that we need to look at different test sets. In the results presented in this paper we use only the first approach, but in subsequent work we obtained proxy traces from NLANR [17] and performed a similar study [22]. Results from this subsequent study are referenced as appropriate in this paper.

3.2 Data Retrieval

The methodology of the study is to perform an unconditional HTTP GET for each of the URLs in the test set on a daily basis using the HTTP request headers shown below for the sample URL `http://owl.wpi.edu/` (the host and path vary for each request).

```
GET / HTTP/1.0
Pragma: no-cache
Accept: */*
Host: owl.wpi.edu
User-Agent: Mozilla/4.03 [en] (WinNT; I)
```

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 a resource is retrieved, it is parsed and all embedded and traversal links are recorded. Embedded images are retrieved and their MD5 checksum is calculated.

This process is repeated for all traversal links in the original URL in the test set. Hence all traversal links in the home page of each site are retrieved along with the embedded images of each of these links. This approach allows us to not only follow the dynamics of individual URLs, but to follow the dynamics of the set of resources used at a site.

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

Four test data sets were constructed using the September, 1998 ratings from 100hot.com. Data from the first test set, "com1," were gathered on a nightly basis for a two-week period during October, 1998. The com1 test set consists of home pages for 19 Web sites identified as the top 10 sites by the 100hot.com (some sites included multiple homes). The specific sites used in this and other test sets are given in [21]. As the test set name implies, all sites in com1 are from the `.com` commercial Internet domain, although in a few cases these sites contain links to URLs not in this domain, particularly in other countries.

The three remaining test sets were studied during a two-week period in November, 1998. The "com2" test set consists of 13 URLs from the next most popular sites from 100hot.com. The "netorg" test set was derived from the set of all non `.com` sites in the 100hot.com top 100. These sites are primarily from the `.net` and `.org` domains. The final test set, "edu," was constructed based on rankings of the `.edu` domain site usage given by 100hot.com along with WPI's home page. Because relatively few queries were included in the four test sets, we added a fifth test set "query" to our study. This test set was studied for six days in November and simply included queries to ten search engines, searching for "search engines." For this test set, the query result was retrieved along with embedded images, but traversal links were not retrieved.

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. Statistics about server software were also gathered, but in our studies we found no specific correlations with server software so these data are not reported.

Table 1: Summary Information for Test Sets

	Test Set				
Item	com1	com2	netorg	edu	query
Number of Base URLs	19	13	11	10	10
Number of Resources	1938	2048	127	110	149
Pragma/Cache-Control	2.6%	17.2%	4.7%	0.0%	0.07%
Expires	50.6%	17.7%	1.6%	0.0%	11.4%
Last-Modified Time	84.5%	90.4%	88.2%	97.3%	85.9%
Etag	9.5%	44.5%	19.7%	50.9%	45.0%
Set-Cookie	17.4%	10.7%	8.7%	0.0%	16.8%
Content-Type: HTML/text	15.8%	10.7%	13.4%	11.8%	6.7%
Content-Type: image	83.5%	89.2%	86.6%	88.2%	93.3%
Number of Repeated Resources	1121	910	113	110	74
Content-Type: HTML/text	21.9%	15.5%	15.0%	11.8%	13.5%
Content-Type: image	77.9%	84.3%	85.0%	88.2%	86.5%

The table shows that retrieving each home page, along with all embedded images in the home page, all traversal links in the home page and all embedded images of the traversal links results in a large number of resources retrieved, particularly for commercial sites.

In examining the top half of Table 1, all test sets show a heavy proportion of image versus HTML responses. The com2 test set shows a relatively high proportion of resources returning a Pragma:no-cache or Cache-Control header while the com1 set shows more than half of the resources returned with an Expires header. The use of cookies is most prevalent in the commercial test sets. While all of our GET requests specified an HTTP version of 1.0, some servers responses indicated they were using the HTTP/1.1 protocol. We found that HTTP/1.1 headers such as Etags were returned by both HTTP/1.0 and HTTP/1.1 servers.

The bottom half of Table 1 focuses on the resources that were retrieved more than once in our tests. Because only the base set of URLs is fixed in our measurements, the actual set of images and links can and obviously did change over the course of the study. Only about 50% of the resources were retrieved more than once for the commercial and query test sets while this ratio was much higher for the other two test sets. For multiply retrieved resources, the ratio of HTML resources is a bit higher than for all resources. As part of the study, 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 [21].

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 [6]. Our calculations are based upon the MD5 checksum computed for a returned resource and not on cache validation information such as lmodtimes or Etags reported by the server. Figure 1 shows the results for HTML and images for each of the test sets. The images for all test sets show virtually no change as found in [6] while the HTML resources show much variation in change characteristics. The netorg and edu results show 60-70% of the HTML resources did not change (comparable to the HTML results in [6]), but the HTML resources for the commercial sets show much more volatility. Only 10-20% of these resources did not change during the study while 70-80% of these resources changed on each retrieval. 100% of the query HTML resources changed on each retrieval. As a comparison, 40-50% of HTML resources never changed in our subsequent study [22].

4.3 Cache Validation Information

We next examined the availability and accuracy of cache validation information returned by Web servers for a resource. Table 2 shows the data for three potential cache validators: last modification time, entity tags and content-length for the commercial test sets. 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.

As the table shows, the lmodtime is generally available and generally corresponds to whether or not the resource changes. However, there are instances (1.53% and 9.36%) where the resource does not change, but the lmodtime does. If validation was used in these cases, a new resource would be retrieved, but the contents of the new resource would be identical to

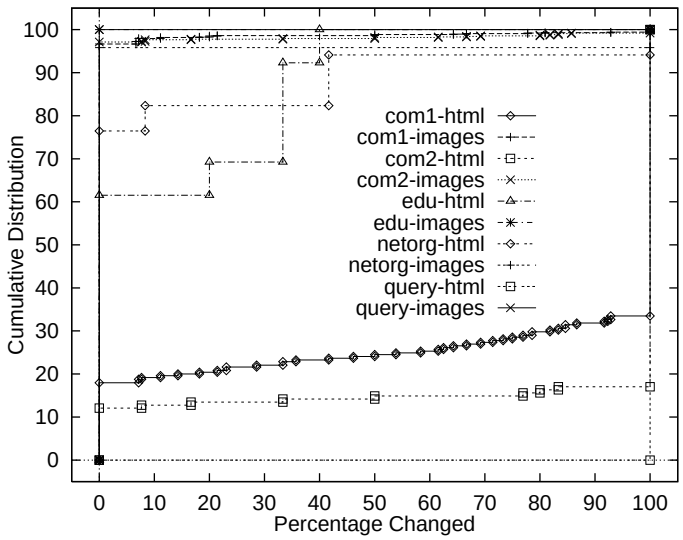

Figure 1: Cumulative Distribution of Test Set Change Ratio Grouped by Content Type

Table 2: Comparsion of HTTP Validators to MD5 Content Changes for All Com1 and Com2 Resources (%)

Test		LModTime			Etag			Content-Length		
Set	MD5	chng	noch	unav	chng	noch	unav	chng	noch	unav
Com1	chng	1.97	0.32	15.26	0.60	0.00	16.94	11.22	0.52	5.80
	noch	1.53	78.20	2.73	6.05	6.32	70.10	0.00	80.94	1.52
Com2	chng	1.38	0.03	13.39	0.74	0.00	14.05	1.93	0.12	12.74
	noch	9.36	74.46	1.39	37.26	15.29	32.66	0.00	74.46	1.39

the old. Even more problematic are a relatively few instances (0.32% and 0.03%) where the resource has changed, but the lmodtime does not. Finally, the results show that 14-18% of the resources have no lmodtime making it impossible to conditionally validate the currently cached contents of a resource. While most of these cases show that the resource has indeed changed, there are a few cases where the resource has not. Similar results were obtained in our subsequent study, although in a test set containing only HTML content types 65% of the resources included no lmodtime information as part of the response [22].

Analysis for the use of Etags shows that fewer servers currently respond with Etags. One problem with Etags highlighted in the results is that in some cases (6.05% and 37.26%) the Etags change between successive responses, but the resource contents do not. Investigation of this phenomenon found that it often occurred for images at sites using multiple servers. We found these sites often use the Apache server, which forms Etags by concatenating the inode, lmodtime and size of a file. We observed that for sites with multiple servers, the inode varied depending on the server that served the resource. This observation shows a poor interplay between how the server software is written and how it is being used at multi-server sites.

Comparison 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. Table 3 shows the results for the commercial test sets of 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 compare the results with content changes using the MD5 checksum.

Table 3: Comparison of HTTP Cache Directives to MD5 Content Changes for Com1 and Com2 Content Types (%)

Test		No	Expires						LModTime		
Set	MD5	Cache	past	<1h	<1d	<1w	<1m	>1m	chng	noch	unav
Com1	chng	2.73	0.74	0.06	2.42	25.07	0.00	0.00	5.61	1.14	37.26
HTML	noch	1.35	2.11	1.17	0.00	0.26	0.00	0.00	0.00	14.88	5.19
Com1	chng	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.85	0.09	0.39
Image	noch	0.43	0.07	0.05	0.23	0.00	0.11	58.88	1.79	35.94	1.27
Com2	chng	71.73	2.01	0.49	0.24	0.00	0.00	0.00	0.00	0.00	11.47
HTML	noch	5.57	0.12	0.00	0.47	0.00	0.00	0.00	0.00	6.38	2.01
Com2	chng	0.28	0.00	0.00	0.10	0.16	0.00	0.00	1.11	0.03	0.29
Image	noch	9.98	1.94	0.79	1.30	0.15	0.00	0.00	1.74	85.61	0.02

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 and month. The results show that a large proportion of com1 images have an expiration time greater than one month (actually greater than one year). These resources come from `yahoo.com` servers. A high proportion

of the com2 HTML resources have explicit directives not to cache. All of these resources come from `www.geocities.com`. The results also show that few image resources have no cache directive (no cache control, or expires or lmodtime), but there are still a relatively large number of HTML resources in this category.

As a final direction for studying the availability and accuracy of cache directives we use the data in Table 3 along with similar data for the netorg, edu and query test sets to calculate the current and potential reusability of cached resources. For this calculation, we first determine which columns of Table 3 can be cached and sum the percentages in these columns. These columns are the <1m and >1m 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 resources where the lmodtime is unavailable. Using these assumptions the second column in Table 4 shows the reusability of cached resources for each category.

Table 4: Current and Potential Reusability of Cached Resources (%)

Resources	Current Reuse	Stale Reuse	Additional Reuse	Potential Reuse
com1 HTML	16.28	-1.14	+9.82	24.96
com2 HTML	6.85	-0.00	+7.80	14.65
netorg HTML	30.39	-0.00	+23.04	53.43
edu HTML	65.13	-0.00	+22.56	87.69
query HTML	0.00	-0.00	+0.00	0.00
com1 image	95.16	-0.09	+3.61	98.68
com2 image	87.17	-0.03	+14.47	98.03
netorg image	95.83	-0.00	+0.00	95.83
edu image	100.00	-0.00	+0.00	100.00
query image	89.69	-0.00	+10.31	100.00

The results show zero to a small amount of reuse available for the query and commercial HTML resources, a larger amount for the HTML resources of the other test sets and a high reusability for the images of all test sets. The third column of Table 4 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 (with small errors due to mixed classification of resources) 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 impediments to reuse are cases with missing lmodtimes for unchanged resources.

We found the same impediments in our subequent study and determined a potential reuse of over 50% for HTML resources in that study [22].

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. In this section we examine one issue from the standpoint that HTML resources are often "containers" for embedded images: the frequency at which embedded images remain in an HTML resource between successive retrievals. Table 5 provides results on the number of images that remain between successive retrievals of an HTML page from each test set.

Table 5: Number of Embedded Images and Traversal Links Remaining in an HTML Page Between Successive Retrievals

	Test Set				
Item	com1	com2	netorg	edu	query
Number of HTML Pages	245	141	17	13	10
Ave Number of Embedded Images Per Page	9.24	31.72	8.45	16.34	25.03
Ave Number of Remaining Embedded Images	4.67	17.57	7.32	11.39	5.64
Ave Number of Links Per Page	73.74	63.77	14.31	28.10	75.50
Ave Number of Remaining Links	64.10	41.39	13.52	26.16	53.69

The results show that the percentage of images remaining is a little over half for the commercial test sets (a similar percentage was found for frequently requested URLs in [22]). The results for the netorg and edu test sets show that for 70-80% of resources the set of images remains the same between retrievals. The query test set yields the least amount of reuse on average, although the median is close to 50%. 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 an image is no longer used by any container resource then it should be garbage collected and removed from the cache.

Table 5 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 Changes to HTML Resources

We also examined the extent of changes to HTML resources. If the changes to a resource are relatively small then techniques such as delta-encoding [16] or other techniques become

relevant. Rather than study changes to HTML resources using automated techniques as done in [16], our initial work has been to manually examine a selected set of HTML resources that change frequently. For each resource in the case study we first retrieved the resource, then retrieved it again to compare for changes that occur over a short period of time. We then retrieved the resource again after a longer period of time (a few hours to a day) to compare long term changes. What we found was that in many cases the only changes in the short-term were changes to the ads contained on the page (see [21] for more details). Longer term changes included new news stories, new dates and some changes in the set of images. However, the structure of the page remained largely the same.

While these results are preliminary, and in need of more extensive study, they indicate that many changes to an HTML resource are predictable and localized. These preliminary results indicate that potential gains can result from techniques such as delta-encoding [16] or cachelets [4]. Caching improvements can also be made if the dynamic portions of a page can be separated from the static and treated differently for retrieval. We discuss such an approach in Section 5.3.

4.6 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. This test was not done as part of our original study, but was done in February, 1999. For this study, we started with a list of com1 resources from our original study set that included a "Set-Cookie" header in the response. We pruned from this list all resources that no longer existed or no longer returned cookies. The resulting cookie test set contained 169 image and 19 HTML resources.

In comparing changes to these resources based on cookies, we initialized the cookie test set by requesting (with no cookie) 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 6 for all resources in the test set and for each content type. The tests were repeated a day later with similar results.

Table 6: 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	94%	6%	47%	53%	100%	0%
cookie vs. no-cookie	94%	6%	50%	50%	99%	1%

The results show that about 50% of HTML and virtually all image resource responses are the same for requests with two different cookies or for a cookie and non-cookie request. Our other study found 57% of the HTML resources and 87% of the image resources to be the same for requests with two different cookies [22]. In that study, we manually examined resources that did show a difference and found that only ad references changed in most of

those cases. These results indicate that responses with cookies can be cached and in most cases the cached content can be reused for subsequent requests.

Our results indicate that assumptions made in previous work about the uncacheability of responses with cookies may not be valid [3, 9]. 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 [20].

5 Implications for Web Caching

The primary results of this work are a better understanding of how resources change and how meta information is reported by servers. This improved understanding points at the potential for a variety of improvements in Web cache performance. The following are ideas to better realize the potential of caching.

5.1 Choice of Validator

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. One problem is that the last modification time is not always 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 the current generation of Etags is not always a good match for multi-server sites as different Etags are returned by different servers for the same, unchanged resource. For long-term caching effectiveness, Etags need to be served correctly and could incorporate reliable validators such as MD5 checksums.

5.2 Relationship Between Resources

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.

The benefits of considering relationships between resources for caching was explored in a more general way in [10]. Our results clearly indicate that policies for caching resources from the same site should take into account the inherent structure of the resulting pages and to discard unneeded images when the structure changes. We do need to explore how the set of embedded images for a container HTML resource changes over time. For example, if a set of images are rotated within a container resource then the next retrieval for the resource may show that the image has been removed from the page, but that it will reappear in a

subsequent retrieval. An effective cache replacement policy needs to be able to discover this situation and handle it appropriately.

5.3 Embedded Resources

The preliminary results from Section 4.5 indicate that even frequently changing pages often exhibit deterministic and predictable changes. While techniques such as delta-encoding [16] 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, such as proposed in [7]. The idea is already present in the use of embedded images, which change infrequently and are of a different type than the HTML page they are contained in. We envision a further division of HTML portions of a page into distinct resources based on change frequency. A prime example, which showed up in many of the resources studied in Section 4.5, is the rotation of an ad link and a banner. The ad link and the banner can be treated as a separate embedded resource that changes on each access as part of a larger container resource that changes infrequently. Not only does this approach allow much of the page to be cached, but it involves the origin server in retrieving new ads, which allows the server to monitor ad hits—a concern of content providers.

How such embedded resources are provided for in HTML or its successor can vary. One approach is to simply extend the notion of embedded images to more generalized objects. Another idea is "client-side includes," which provides the same functionality as server-side includes, already available in most server implementations, but exposes the structure of the overall page to the client and allows for static portions to be cached. Some browser implementations allow for a similar directive in HTML. For example, the `<ilayer>` tag introduced by Netscape allows a page to be composed of multiple resources [19]. The `<iframe>` tag proposed by Microsoft embeds one resource within another, and is similar to the `<img>` tag. Douglis, et al propose an HTML pre-processor where the static portion of a page contains macro-instructions for inserting dynamic information [7].

This approach of dividing a page into more resources, potentially increases the number of resources that must be retrieved for a page. However, clients and servers can exploit the association between dynamic and static resources of a page to both retrieve new resources and validate cached resources. When clients retrieve the new contents of a dynamic resource, they can include a validator for the resource. While it is expected that the resource itself has changed, servers can additionally use the validator as a version indicator to perform two important functions:

1. pipeline new resources needed to render the entire contents of the page, and

2. piggyback invalidations of any resources that continue to exist in the page, but have actually changed [12].

The first function allows all needed resources to be retrieved with a single HTTP request over a persistent connection in HTTP/1.1. This approach assumes that preexisting resources

have been cached and that only resources that are new to the page since the given validator are retrieved. The second function allows clients to avoid validating embedded resources and to have servers send invalidations, which are expected to be infrequent, when such resources do change. It treats the set of resources for a page as a volume, which are maintained coherent using piggybacked server invalidations [12]. The result is a reduction in the number of unneeded validation checks for these resources.

5.4 Access Count-Based Expirations

One continuing problem with a more general notion of embedded resources is that resources such as ads must still be retrieved from the origin server on each access—incurring latency for the page. An alternate approach for cache lifetimes is to combine the existing time-based expiration approach with a count-based approach. This approach would allow cached resources such as ads to expire after a pre-determined number of accesses. In the simplest case, the count could be one, which still requires the client to retrieve a new ad on each access of the page.

However, the advantage of such a count-based expiration is that it can be combined with local prefetching where a browser or a proxy cache can prefetch such ads immediately after they expire. The newly retrieved and cached ads then have an expiration count of one. The next access of the same ad is serviced by the cache, which itself can trigger a new copy of the ad resource to be retrieved for the next access. Because the ad is retrieved from the origin server, the server can still maintain hit counts for each ad. Larger access counts could be used for improved performance, but will result in coarser granularity of hit counts.

5.5 Use of Cookie-Based Responses

Results from this and related 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, 9], 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, 8]. 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 [14] published a previous study on the potential of caching and prefetching in reducing web latency. This study was

based on a proxy log, which limits the type of information available. Specifically, CGI-generated resources were treated as uncacheable, last-modified timestamps were unavailable in about half the entries and no information was available about the nature of a change.

Work by Douglis, et al [6] has influenced our work on determining the frequency at which resources change. Many of these researchers were also responsible for a study on delta encoding [16], which allows servers to transmit changes to a resource as a list of differences. Delta encoding is an alternate to using embedded resources in a page. It is more general, but in cases where portions of a page change frequently, but predictably the embedded resources can receive updates without the server having to explicitly compute a difference between the new and old versions. Cao, et al [4] have proposed a more general method for cachelets to handle small changes to cached resources, but the results show this approach incurs non-trivial computational costs at a proxy cache. Our proposal of local prefetching and access count expiration limits allows a proxy cache to provide current ads while allowing the origin server to maintain hit counts that reflect actual use. Douglis, et al [7] propose an idea for separating static and dynamic portions of a page, which fits into our plans for future study.

7 Future Work

An obvious direction for future work is to do a more complete study on the nature of changes to a resource. In addition, our test sets contained only HTML and image content types. We plan an extended study to more fully investigate the nature of changes for different content types. As part of this study, we need to investigate varying the intervals and durations appropriate for characterizing the nature of each resource. On a longer term basis, our approach provides a platform for monitoring evolution of the Web by focusing on characteristics of URLs at frequently used sites.

Another direction for investigation is closer study of the ideas given 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 these improvements in caching mechanisms translate into better cache hit rates and reduced latency.

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 resources from a variety of servers 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. A subsequent study using a set of frequently accessed URLs has yielded similar results.

In terms of implications for caching, the relationships between resources used to compose a page must be considered. 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 any 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 our work is that while HTML resources frequently change, these changes are often in a predictable and localized manner. We have suggested composing such pages into resources with not only similar type characteristics, but also similar change characteristics where static and dynamic portions of a page are grouped into separate resources. This approach 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. These improvements could lead to better cache hit rates and reduced retrieval latency.

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.
`ftp://ftp.cs.usask.ca/pub/discus/paper.96-3.ps.Z`.

[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] 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`.

[7] 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`.

[8] 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`.

[9] 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`.

[10] John H. Hine, Craig E. Wills, Anja Martel, and Joel Sommers. Combining client knowledge and resource dependencies for improved world wide web performance. In *Proceedings of the INET '98 Conference*, Geneva, Switzerland, July 1998. Internet Society.
`http://www.mcs.vuw.ac.nz/~hine/INET98paper/index.htm`.

[11] Keynote business 40 internet performance index.
`http://www.keynote.com/measures/business/business40.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] Balachander Krishnmurthy, Jeffrey C. Mogul, and David M. Kristol. Key differences between HTTP/1.0 and HTTP/1.1. In *Eighth International World Wide Web Conference*, Toronto, Canada, May 1999.

[14] 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`.

[15] Media metrix. `http://www.mediametrix.com`.

[16] 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`.

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

[18] 100hot.com. http://www.100hot.com.

[19] Thomas A. Powell. *HTML: The Complete Reference*. Osborne McGraw-Hill, 1998.

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

[21] 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.
http://www.cs.wpi.edu/~cew/papers/tr98-27.ps.gz.

[22] Craig E. Wills and Mikhail Mikhailov. Examining the cacheability of user-requested web resources. In *Proceedings of the 4th International Web Caching Workshop*, San Diego, CA, March/April 1999.
http://www.cs.wpi.edu/~cew/papers/wcw99.ps.gz.

Vitae

Craig E. Wills is an associate professor in the Computer Science Department at Worcester Polytechnic Institute. His research interests include distributed computing, operating systems, networking and user interfaces.

Mikhail Mikhailov is a Ph.D. student in the Computer Science Department at Worcester Polytechnic Institute. His research interests include distributed computing, operating systems, networking and computer system performance evaluation.