

Lifetime Behavior and its Impact on Web Caching

Xiangping Chen and Prasant Mohapatra
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011
{xpch, prasant}@iastate.edu

Abstract

The exponential growth of the World Wide Web has made it the most popular information dissemination tool in the world. The growth necessitates caching, prefetching, and replication schemes on the web to alleviate the web server load, conserve the network bandwidth, and reduce the retrieval latency. At the same time, cache consistency should be maintained to avoid returning stale pages to users. The study on the characteristics of web documents helps to determine which schemes to be adopted in a specific web environment. This paper presents characterization of web documents based on their types and environment classes. Nine web server traces are used in this paper, and represent three different classes of web environments: educational, commercial and news. Results indicate significant differences in static as well as dynamic characteristics of documents types in different web site classes. The efficiency of caching can be improved if caching priorities and TTL preferences be given to certain types of web documents. We also provide guidelines for the design and development of caching and prefetching techniques that can exploit the web document characteristics in different environments.

1. Introduction

With the exponential growth of the Internet, the World Wide Web (WWW) has become the most widely used tool for access and dissemination of commercial, educational, and news information on the Internet. The success of the WWW is largely due to its capability of providing quick and easy accesses to a large variety of information from sites all over the world. The retrieval latency of the WWW is however sensitive to overload conditions of the network, which agonizes web users. Caching data copies at client sites (including web proxies) is regarded as a good technique for reducing both the network traffic and the server load, thus improving retrieval latency of web documents. One problem in caching techniques is data consistency. If cached data copies are not consistent with data in the original web server, stale information will be sent back to web users, which impairs the value of web caching.

HTTP specifications provides means of maintaining cache consistency [1] by If-Modified-Since (IMS) and Expires mechanism. When a web cache agent receives a GET request for a web document and finds a copy in its cache, it decides whether to return the

cached copy to the client or to forward the request to a web server and update the cached copy. The cache agent makes this decision based on the *age* and the expiration time of the cached document. The *age* of a document is the amount of time that has elapsed since the document was cached. If the *age* is less than its expiration time, the cache agent return the cache copy; otherwise, the document is reloaded from its original web server and updated in the cache. The IMS mechanism is used by a cache agent to reload documents from an original web server if and only if their contents have changed since the given timestamp. The Expires mechanism is used by a web server to provide *expiration time* of a web document. Using expiration time to maintain cache consistency is called *Time-To-Live* (TTL) based cache consistency scheme.

The setting of the expiration time or TTL of a web document directly affects the performance of TTL-based cache consistency schemes. If the TTL is set too short, conditionally Get requests (Get requests with If-Modified-Since tag) are sent to the original web server frequently, resulting in an increase of server load and unnecessary retrieval latency. On the other hand, if the TTL is set too long, stale data will be sent back to users, which impairs the credibility of retrieved web information. Gwertzman and Seltzer [2] found out that 20% of the web requests got stale data with an fixed expiration time of 7 days.

Previous study in [4] found that most current web documents contained no expiration information, and cache agents have to predict expiration time for documents without server specified expiration time. Some cache agents, e.g., Netscape browsers and CERN proxy cache, allow users to manually configure a default expiration time for cached data. Due to the lack of knowledge on lifetime behavior of web documents, the default expiration time is often set short to reduce the possibility of returning stale information. In an extreme case called *polling-every-time* [8], the expiration time is set to zero to maintain strong consistency. The *validity* of a cached document is checked in every "hit". In another example called *Never-Check* [13], web agents rely on users to manually refresh cached data, otherwise a cached document never expires. Some cache agents use *adaptive TTL* [15] to maintain weak consistency, and set the TTL of a web object to a portion of the time elapsed since its last modification.

Although several studies, see [7] for references, have been done on characterizing web traffic, little work on lifetime behavior and modification characteristics of web documents has been reported in the literature. The authors of [15] periodically sampled 4600 web objects distributed in 2000 web sites and calculated the

Akamai Ex. 1031

Akamai Techs. v. Equil IP Holdings

IPR2023-00329

Page 00001

mean lifetime of these objects. In [2], the author mixed the statistical results of web objects from educational and commercial web sites, which concealed possible influences from the web environments. The rate of change of web objects in two corporate communities was studied in [3].

In the following research, we study the relationship between the characteristics of web objects in different classes of web sites with respect to object types, and the relationship between static and dynamic characteristics of web objects. Characterization of web documents lifetime and modification could provide reference data in the design process of caching and prefetching techniques. An efficient cache consistency policy may reduce unnecessary client polling, thereby reduce server load, network traffic, and retrieval latency. This research shows that the lifetime and modification behavior of a web document is not only influenced by its type, but also by the class of the web site where it is placed.

The rest of the paper is organized as follows. The structure and the model of the WWW environment are introduced in the following section. The experimental study of web access traces is presented in Section 3. Section 4 compares and analyzes the web objects characterization results. The inferences derived from the study and their impact on the design issues are outlined in Section 5, followed by the concluding remarks in Section 6.

2. WWW Environment and Traces

Web clients and servers communicate using the Hyper-Text Transfer Protocol (HTTP) on top of the TCP layer. When a client makes a request to a particular web server, a TCP connection is established between the client and the server. Once the connection is established, the client sends a request to the server. The server parses the request and issues a response. After the response completes, the TCP connection is closed.

Access logs of nine web servers and proxy servers are used in the lifetime study in this research. Each line in a access log contains information on a single request for a document. A common log entry is in the form of:

HostName - - (*UserName*)¹ [*Timestamp*] "*Request_Method* *URL* (*HTTP-Version*)" *Status_Code* *Size*

HostName is the name of a remote computer sending out requests as a client. *UserName* is a authenticated user name when the web **User Authentication** is used. Following are timestamp of the connection. There are three *Request_Methods* in the access logs: GET, POST, and HEAD. A GET request requests a document from the server or triggers the execution of a program which produces the output sent to the client. It is used for standard page requests. The POST method is used to send out formatted input data to a program in the server. The execution of the program produces output sent back to the client. The HEAD method is used for testing. *Status_Code* is a three digital number and describes the retrieval status of a transaction. Success codes are in the 200s; redirect codes are in the 300s; failure codes are in the 400s; and server error codes are in the 500s. *Size* is the size of the retrieved document.

¹Contents in () are optional.

3. Experimental Study

We have collected trace files from three classes of web servers: educational (EDU), commercial (COM) and news (NEWS). Our objective in this study is to find out influence of the usage environment on the static and dynamic characteristics of different types of web objects. The classes of usage environment are described as follows.

EDU class: The three traces used for the EDU class are: *Calgary* (access logs from a CS department web server), *Saska* (access logs from a university web server), and *ISU* (access logs from a ECpE department web server). Table 1 summarizes traces in the EDU class.

COM class: The traces to three commercial web servers are extracted from NLANR proxy traces due to the difficulty of collecting data from commercial web servers directly. *MS* (www.microsoft.com), *Compaq* (www.compaq.com) and *Disney* (www.disney.com) were chosen as representatives of commercial web servers. Table 2 summarizes traces in the COM class.

Table 1. Summary of logs from EDU class.

Item	<i>Calgary</i>	<i>Saska</i>	<i>ISU</i>
Log Duration	353 days	214 days	97 days
Start Date	10/24/94	06/01/95	02/03/98
Successful Req.	696,619	2,385,866	1,189,376
Avg. Req./Day	1,974	11,149	12,262
Distinct Req.	8,384	8,097	8,603
Traffic (MB)	7,758	12,637	7,338
Bytes/Day (MB)	21.98	59.05	75.65
Bytes/Req. (KB)	11.13	5.30	6.17

Table 2. Summary of logs from COM class.

Item	<i>MS</i>	<i>Compaq</i>	<i>Disney</i>
Log Duration	68 days	68 days	68 days
Start Date	03/14/98	03/14/98	03/14/98
Success Req.	347,589	103,486	91,397
Avg. Req./Day	5,112	1,522	1,344
Distinct Req.	30,716	22,726	17,514
Traffic (MB)	2,096	665	960
Bytes/Day (MB)	30.82	9.78	14.12
Bytes/Req. (KB)	6.03	6.43	10.51

Table 3. Summary of logs from NEWS class.

Item	<i>CNN</i>	<i>Usa</i>	<i>Weather</i>
Log Duration	68 days	68 days	68 days
Start Date	03/14/98	03/14/98	03/14/98
Success Req.	127,203	147,104	73,267
Avg. Req./Day	1,871	2,163	1,077
Distinct Req.	9,906	7,841	2,594
Traffic (MB)	947	834	571
Bytes/Day (MB)	13.93	12.26	8.40
Bytes/Req. (KB)	7.45	5.67	7.80

NEWS class: Traces used for the news class also come from the proxy traces of NLANR. *CNN*

(www.cnn.com) and *Usa* (www.usatoday.com) are two popular online news web sites. *Weather* (www.weather.com) provides online national weather service. Table 3 summarizes traces in the NEWS class.

3.1. Document Types

The traces are categorized into 10 different types according to the nature of web documents. The type of a document is distinguished by its URL or the request method to it. The 10 types used in our study are listed in Table 4.

Table 4. Types of Web documents.

Index	Type	Explanation
0	PTXT	Plain text
1	HTML	HTML files
2	ARC	Formatted txt files or archives
3	AUD	Audio files
4	GIF	GIF files ²
5	JPG	JPEG and other image files
6	FORM	Form input and responses
7	VDO	Video files
8	EXE	Exe and script files
9	OTHR	Others

Table 5. Not-modified vs. Get retrieval.

	Server	NM #	Get #	NM/Get	Avg
E	Calg.	97,560	566,833	17.2 %	23%
D	Saska	151,607	1,032,966	14.7 %	
U	ISU	303,474	846,442	35.9 %	
C	Comp.	37,834	65,002	58.2 %	53%
O	MS	128,236	208,765	61.4 %	
M	Disn.	24,802	62,851	39.5 %	
N	CNN	54,835	67,045	81.8 %	56%
E	Usa	27,106	119,557	22.7 %	
W	Weath.	27,248	44,025	61.9 %	

Tables 5, 6, and 7 list the proportion of accesses to different document types in three server classes. **Acc%** is the access percentage of each type of files, and **Doc%** is the percentage of unique URLs of each type of documents in a web server. **AccFreq** is the relative access frequency of a web document with respect to the average access frequency of HTML files. The reason for using relative measurement of **AccFreq** is to improve the comparability between a busy and a less busy web server, and get a fair average result. **AvgTransBytes** is the average transfer size of each type, and **AvgDocSize** is the average size of each type.

HTML, GIF, and JPG are the top three popular types. The document (Unique URLs) percentage and the access percentage for the top three file types, however, are not the same in each web server class. Compared to the other two classes, there are more unique HTML documents in the EDU class, more unique GIF documents in the COM class, and the JPG documents in the NEWS class are much more than in other classes. The average HTML document size in the NEWS class

is about twice of that in the COM class, which in turn is about twice the size of that in the EDU class.

These differences suggest that, even web documents of the same types have various average sizes in different server environments. In the following experiment, we also observe that the HTML files in the NEWS class are much more mutable than the other two classes. Here the document's size and lifetime characteristics indicate the same trend of caching gain and thus a size-based caching technique could be adopted [17]. However, our study on the size distribution in different classes of web sites show that this inference is not true for all the cases. The study reported here provides more insight to the caching and removal policies.

3.2. Lifetime Calculation

In the lifetime calculation, an access request is treated as a client polling, and the corresponding access log records the result of the polling. A status code of **304** indicates that a requested document has not been modified since last time the same client requested it. A status code of **200** indicates that either the data copy in the client's local cache is stale or has been swapped out from the cache, or the client requests the document for the first time; the document content is sent back to the client successfully. Since modification timestamp of a document are unavailable from standard access logs, it is difficult to tell directly whether a cached data copy is stale. File size is the only information about a web document in a log besides the URL. We rely on the assumption that most of the modifications result in a change of file sizes. To validate this assumption, modifications of web documents in a departmental web server at ISU were monitored over one month's period. The status of each web document was polled twice on a daily basis. Timestamps of last modification and file sizes were compared. Statistical results showed that more than 95% of modifications resulted in a change of file sizes. Using file size as an indicator of modification is thus a reasonable estimation.

If a web document is requested frequently, it is assumed that its size change could be indicated in access logs in a short period. This assumption has high credibility in the EDU class, since traces in the EDU class are access logs from original web servers, where the most frequent polling records are available. For traces in the COM and NEWS classes, high frequencies are ensured by the combination of multiple users' requests and selection of "hot" web servers. The assumption is based on the fact that the ratio of accesses with status *Not-Modified* (NM) to accesses with status of *successful retrieval* (Get) is generally high, especially in proxy trace files. High percentages of *Not-Modified* responses means that the access frequency and the default expiration time of web documents are much shorter than the real lifetime of them. In other words, any change of a web document could be found in a short period of time by frequent client polling. Table 8 shows the percentage of NM responses to successful Get responses. **Avg%** is the average NM/Get retrievals of web servers in the same class.

A web document lifetime is approximated to the two continuous retrievals of the same document with a different size. Here the same document refers to the documents having the same URL, and a lifetime of a

Table 6. Access pattern of different types in EDU class.

Type	PTXT	HTML	ARC	AUD	GIF	JPG	FORM	VDO	EXE	OTHR
Acc %	0.3	41.5	0.6	0.0	45.6	5.4	5.7	0.0	0.1	0.7
Doc %	1.3	49.0	3.4	0.1	34.3	4.0	0.8	0.4	0.3	6.3
AccFreq	0.25	1.00	0.16	0.43	1.79	1.35	50.0	0.11	0.59	0.13
AvgTransBytes (K)	43.67	5.66	246.90	72.82	6.28	20.86	2.23	621.82	5.97	58.76
AvgDocSize (K)	25.37	4.63	278.60	170.56	14.51	43.09	4.72	751.39	46.66	58.76

Table 7. Access pattern of different types in COM class.

Type	PTXT	HTML	ARC	AUD	GIF	JPG	FORM	VDO	EXE	OTHR
Acc %	0.0	21.1	0.1	0.3	68.0	6.2	0.0	0.1	2.6	1.6
Doc %	0.1	36.6	0.3	0.5	50.6	8.8	0.1	0.2	2.2	1.2
AccFreq	0.66	1.00	0.43	0.65	2.44	1.22	0.22	0.56	1.79	2.05
AvgTransBytes (K)	54.97	13.00	388.24	111.10	3.16	10.87	1.99	964.62	77.56	33.11
AvgDocSize (K)	46.90	10.47	346.41	186.57	5.61	14.81	2.13	1320.05	239.58	76.49

Table 8. Access pattern of different file types in NEWS class.

Type	PTXT	HTML	ARC	AUD	GIF	JPG	FORM	VDO	EXE	OTHR
Acc %	1.6	16.5	0.0	0.0	65.0	13.5	0.1	0.1	1.9	1.4
Doc %	0.0	40.2	0.0	0.1	35.7	21.1	0.1	0.4	0.8	1.4
AccFreq	50.0	1.00	2.08	0.61	5.0	2.38	0.88	0.36	5.0	1.82
AvgTransBytes (K)	1.05	19.04	38.52	265.11	4.59	10.16	8.22	904.66	31.25	13.72
AvgDocSize (K)	1.49	18.10	34.02	235.40	7.93	9.77	7.48	1002.64	103.17	27.44

document is an interval between two successive modifications of the same URL. A lifetime of a document i , denoted as LT_{ij} , can be obtained from, where MT_{ij} denotes the j th modification of document i .

$$LT_{ij} = MT_{(i+1)j} - MT_{ij} \quad (1)$$

A couple of factors appear to distort the results in our study. Some documents might never change during the period of trace recording. The longest observed lifetime of a document is made equal to the duration of trace files, which might be shorter than the real lifetime of a document. Average lifetime would thus be shortened by the limit of trace duration. However, the observed lifetime distribution is not affected by that limit. Another factor is that the accuracy of the results are influenced by the user reference patterns. The results of frequently accessed documents are more accurate than the less accessed ones. These two distorting factors are compensatory in nature. Furthermore, experimental results of the average lifetime of web documents in the EDU class are consistent with the previous findings in [2], which suggests that the influences of above two factors are negligible.

4. Results

In this section, we characterize the nature of different types of web documents with respect to three classes: EDU, COM, and NEWS.

4.1. Average Lifetime

The average TTL of the web documents in different classes were compared and are summarized in Table 9. The results show that the documents in the EDU class are much more stable than the other two classes. The average lifetime of a document is about 60 days in the EDU class. Web documents in the NEWS class are the most mutable, only 2~3 days. Lifetime for a typical web document in the COM class is about 8~10 days.

There is no obvious difference of HTML and GIF average lifetime in the EDU and COM classes respectively. In the NEWS class, the average lifetime of GIF files is about 50% longer than that of HTML files. However, average access interval of GIF files is 2 to 5 times of interval of HTML files, which indicates that the average access frequency of a GIF file during its lifetime is 2 to 5 times of that of an HTML file. Contrary to our expectations, results show that the lifetime characteristics of audio/video files do not differ significantly from other types of files.

4.2. Modification Distribution

The relationship between a document lifetime and its access frequency is examined. Average lifetime, modification times, and accesses during lifetime of *the most popular files* are collected. Web documents are sorted by their access frequency, *the most popular files* are defined as a small set of files that are responsible for 80% of retrieval requests. The distribution of modification times versus document percentage and access percentage are collected and shown in Figures 1, 2, and 3. In each of these figures, there are four diagrams.

Table 9. Average TTL (unit days).

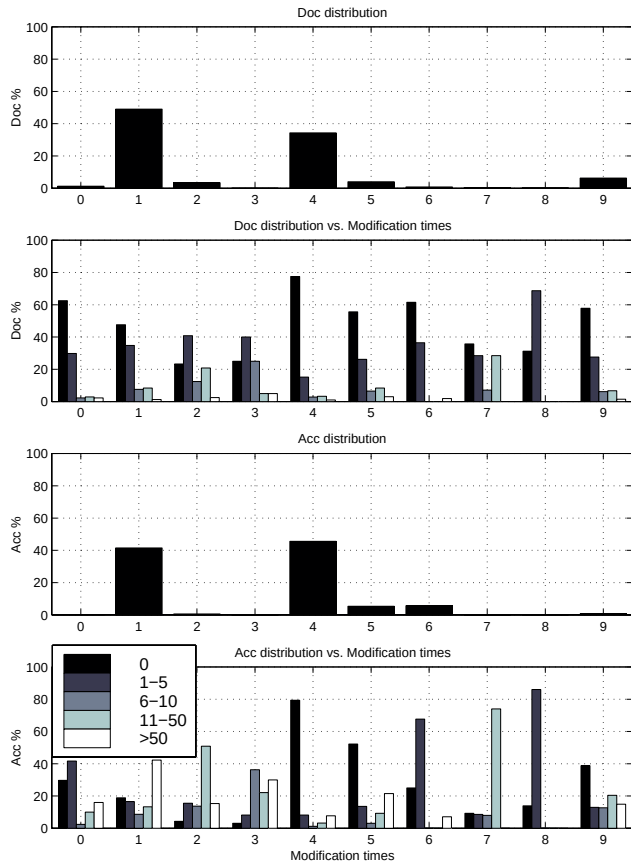
File-Type	PTXT	HTML	ARC	AUD	GIF	JPG	FORM	VDO	EXE	OTHR
EDU	56.9	61.5	21.8	13.6	63.4	55.6	N/A	10.3	29.7	45.6
COM	10.2	8.5	3.9	9.6	8.6	9.3	N/A	9.2	7.7	10.0
NEWS	8.5	2.9	2.8	2.2	4.4	2.6	N/A	2.0	7.6	4.8

The first one shows the document distribution of all file types; the second shows the percentage of documents with different modification frequency for each type of files; the third shows the access distribution of all file types; and the last one shows the percentage of accesses with different modification frequency for each file type. The figures indicate the distribution mode of each web object type.

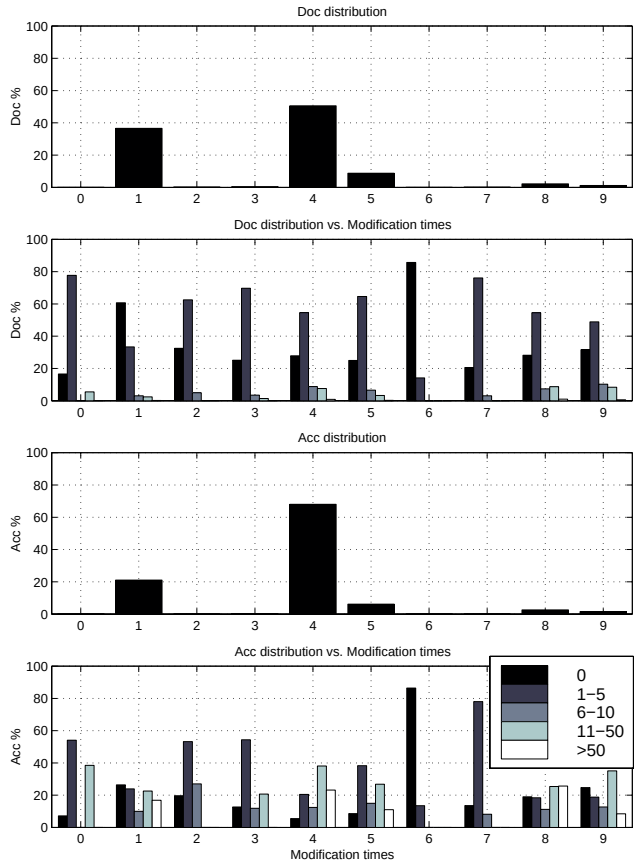
Figure 1 suggests that some popular documents, especially those GIF documents, are seldom modified in the EDU class. Similar observation was reported by Bestavros in [20]. The average lifetime is about 2 months for a web document in the EDU class. On the other hand, 1.3% of the HTML files that are frequently updated accounted for 42% of the access requests. Users have tendency to access more frequently modified data. Figure 2 shows that more than 94% of

10 days in the COM class. However, the access percentage to those documents are less than 50%. On the other hand, 0.2% of HTML documents are modified in daily base, and the access requests to those documents are about 17% of the total accesses. This observation suggests that, Bestavros' conclusion [20] about the popularity of a document being inversely proportional to the possibility of its modification is not valid for the HTML documents in the COM class.

GIF documents in the COM class also show that more popular documents were more mutable. The documents modified 11~50 times are responsible for 38% of access requests, and 1% of documents modified more than 50 times are responsible for 1/4 of total requests. Figure 3 shows that most references in the NEWS class

**Figure 1. Modification distribution in EDU class.**

HTML documents have average lifetime of more than

**Figure 2. Modification distribution in COM class.**

are to the documents with modifications more than 50 times. Considering the total trace collection period of 68 days, The documents with lifetime of 1 day or

less accounts for most of requests. This behavior is expected of the NEWS class. 2.7% of HTML documents in the NEWS class are modified more than 50 times, and they constitute 50% of the access requests. Although there are some rarely updated HTML documents, access times do not seem to increase with the lifetime of those documents. For the GIF documents in the NEWS class, the more popular a document is, the shorter lifetime it has. There is no GIF document which is never modified during the trace collection period.

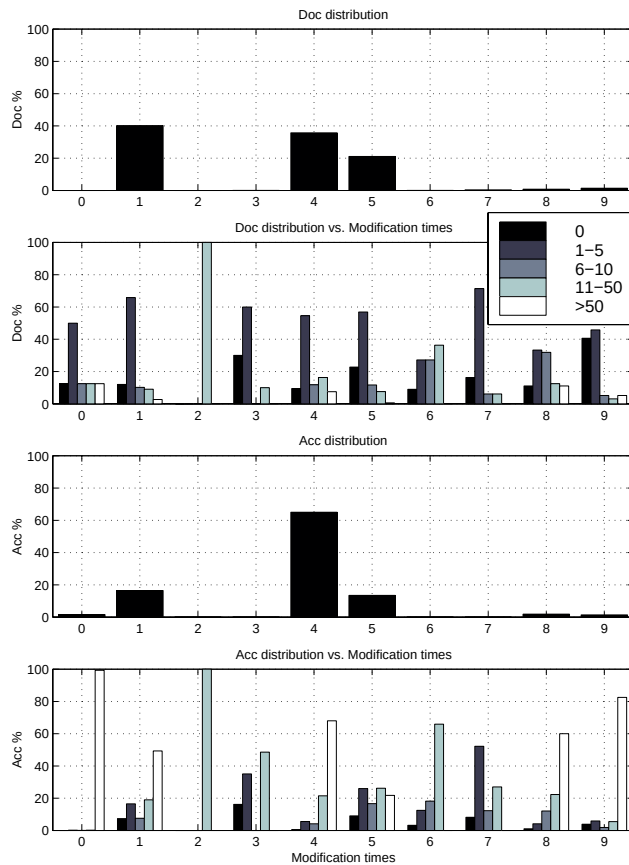


Figure 3. Modification Distribution in NEWS class.

5. Design Issues

5.1. Document Classification

Based on the access, lifetime and modification characteristics, web documents can be classified into four categories: *highly mutable documents*, *stable documents*, *short life documents*, and *others*.

A *highly mutable document* is defined as a document with frequent modifications, e.g., the front page of some web sites. They are usually accessed more frequently than other documents, see Table 10. Conventional caching policies such as LRU or LFU might

be insensitive in distinguishing highly mutable documents, and lead to high percentage of stale data in caches. Instead, prefetching or server-pushing based data dissemination might be a suitable option for retrieving documents from web sites with high percentage of highly mutable documents.

Table 10. Highly mutable document ratio.

Environment	Doc %	Acc %
EDU	10	20
COM	3	25
NEWS	33	80

A *stable document* is in steady state with relatively long lifetime. Some of them are accessed frequently, while most of stable documents have relatively low access rate. In the EDU class, 44% of the HTML and 78% of the image documents never change during the whole trace period, 20% of the HTML file accesses and 80% of the image file accesses are to those stable documents. Nearly 30% of HTML documents in the COM class were never modified during the trace period and responsible for 18% accesses.

Short life documents are accessed or existed in traces for only 1 or 2 days. Nearly 1/3 of the documents in the NEWS class are Short life documents. There are about 20% of short life documents in the COM class. The short life documents should be distinguished and removed from the local cache to save space for potentially useful data.

5.2 A Two-State TTL Algorithm

Keeping highly mutable and short life objects in a cache sometimes does not help to increase the cache hit rate, rather it could cause potentially useful (to be retrieved in the near future) data copies being deleted from the cache and thereby increase the number of requests to the original server. Most web caching policies fail to efficiently deal with all classes of web documents. For example, LFU is not suitable for distinguishing short life documents [6], LRU is not effective for highly mutable documents.

A good cache consistency mechanism helps to prevent a cache from returning stale data as well as improve the cache efficiency. For instance, highly mutable and short life documents are granted a short TTL. In case of cache replacement, they are treated as expired data and discarded to save space for potentially useful data. This is especially important where clients are personal computers with limited cache space. At the same time, the TTL for stable documents should not be set too short to introduce bulks of “conditional get” requests, i.e., GET requests with If-Modified-Since fields.

Here, we present a two-state TTL consistency algorithm. When a new data copy is cached for the first time, it is set in a *transient* state with a short TTL, say less than 1 hour. If the data copy is still valid after the short TTL, it is switched into a *steady* state with a relatively long TTL. This long TTL can be decided based on statistic measurement, or a portion of time elapsed since the last modification time.

We set up a simulation using a cache simulator from Virginia Tech [17], and adapted it by extending three consistency, i.e., adaptive TTL, fixed TTL, and two-state TTL algorithms into caching policies. In the two-state TTL algorithm, the cache space is divided into two equal areas for transient and steady state documents respectively. New caching copies only replace documents in transient state, older steady state documents are replaced by newer steady state documents in cache replacement. The traces from EDU³, COM and NEWS⁴ classes are replayed to roughly check the effectiveness of the two-state TTL algorithm. Cache hit rate, conditional get requests with not-modified response, and the amount of stale data returned are collected and compared.

In the fixed TTL algorithm, the default TTL is set to be 1/4 of average lifetime in that class. In the adaptive TTL algorithm, TTL is set to be 1/2 of elapsed time since last modification. In the two-state TTL algorithm, transient TTL is set to be 15 minutes in NEWS class, 12 hours in COM class, 24 hours in EDU class, and adaptive TTL is used in steady states.

The results indicate that less than 1% of stale data are returned in the EDU class irrespective of the consistency policy. In the COM and NEWS class, using fixed TTL returns about 19.8% of stale data, and 10.9% of conditional get requests with not-modified response. Adaptive TTL performs the best in keeping cache consistency, returns about 7.3% of stale data, while 25.4% of unnecessary conditional get requests. Two-state TTL decreases 3.1% of unnecessary get requests from adaptive TTL algorithm at the cost of returning 0.9% more stale data. Furthermore, two-state TTL increases cache hit rate by average 2.8% compared to the other two TTL consistency algorithms, in a cache size of 5 MBytes. It proves that the two-state TTL algorithm do help to improve caching efficiency. Detailed performance evaluation can be done by varying the configuration and workload.

5.3 Caching and Prefetching

Due to the explosive growth of the Internet, lots of efforts have been directed towards implementation of client/proxy caching [4, 9, 10, 15, 24], sever caching [5, 11, 14], prefetching [19, 21], and replication [12, 22] techniques to reduce network latency and the server load. However, there has been limited studies reported on the integration of workload characterization and the design of caching techniques in the web environment.

In the context of caching, there exists significant differences between an web environment and a traditional distributed computing system. The workload characteristics in the two environments differ significantly. Most of the web caching algorithms are transplanted from the distributed file systems that do not consider the static and dynamic characteristics of documents on the web. For example, the number of documents on the web are much higher than traditional distributed file systems. A large quantity of short life documents exist on the web and accesses to the web documents

tend to be bursty [23]. Caching algorithms should be developed targeting the web documents characteristics and for specific classes of environments.

Our study has shown that significant differences exist in both static and dynamic characteristics of web documents in different web environments. Caching algorithms should be adjusted with the characteristics to gain better performance.

The average lifetime of documents in the educational web sites are significantly higher than the commercial and news web sites. A high percentage of stable image files with high access rate exists on the educational web sites. They are the best candidates of the conventional caching algorithms. Special priority should be given to graphic documents, since more graphic documents are seldom changed compared to the other types of documents, and users' tendency of retrieving small graphic documents improves cache efficiency. In [17], the authors have presented a good evaluation of caching algorithms suited for documents in the educational web sites.

For documents in the commercial web sites, About 15% of the GIF documents and nearly 30% of the HTML files have short lifetime with high access frequencies. LRU combined with two-state expiration time filters out these documents effectively. In a commercial web site, there are also some frequently updated documents with very high access frequency. The assumption that the popular files are modified less frequently is not valid here. Prefetching with server invalidation should be adopted to deal with these highly mutable documents. Two possible types of prefetching schemes reported in [21] could be examined in these situations. The access frequency of documents on the commercial web sites tend to decrease with the total lifetime for most stable documents. In such cases, LRU might outperform LFU replacement schemes, which can explain why some trace driven simulation results are not consistent in web caching studies [6, 17, 24].

Most documents in the news web sites are highly mutable and many are accessed only in a short period. Except for large proxy servers, news documents are "not worthing caching" for personal web users. Instead, server initiated push caching as in [22] might be used to gain better performance. For the less mutable documents in the news web sites, their TTL should be set significantly shorter than which from the other classes of web sites.

6 Concluding Remarks

Different types of web objects from different sites have different static and dynamic characteristics. In this paper, we have reported an extensive study on the size, access, lifetime and modification behavior of web documents from various web sites, and tried to find out some rules in improving caching schemes with the knowledge of web environments dynamics. Some of the major observations derived from our study are itemized as follows.

- The document size, access pattern and modification behavior are dependent on the document type as well as the server environments. The TTL estimation algorithm should be made adopted to the document type and/or the server environment. A two-state TTL weak consistency algorithm based on multi-mode characteristics of web documents introduced in this study might outperform current fixed TTL or adaptive TTL algorithms.

³We use the web client traces from Virginia Tech representing the EDU class.

⁴We use the NLANR proxy traces representing the COM and the NEWS classes.

- A generalized technique for caching and prefetching technique may not be effective for a diverse WWW environment. Rather, server-specific or data-specific caching and prefetching techniques should be adopted. Caching consistency policy should be taken into consideration in caching schemes to improve the cache efficiency.
- Several additional interesting behavior of different types of web documents in different classes of servers have been reported in the study. In COM and NEWS classes, there is a strong trend that users access highly mutable documents more frequently than other documents.

The results suggested that the caching performances are influenced by different classes of web servers as well as by different types of web documents. Thus, caching, prefetching and server initiated replication techniques should be designed such that they exploit these varied characteristics. Our future work will concentrate on the design, development, and implementation of a flexible, efficient, and adaptive caching and prefetching technique for the World Wide Web.

References

- [1] A. Dingle, "Cache Consistency in the HTTP 1.1 Proposed Standard," in *ICM Workshop on Web Caching*, (Warsaw, Poland), September 1996.
- [2] J. Gwertzman and M. Seltzer, "World-Wide Web Cache Consistency," in *Proceedings of the 1996 Usenix Technical Conference*, (San Diego, CA), January 1996.
- [3] F. Douglass, A. Feldmann, *et. al.*, "Rate of Change and other Metrics: A Live Study of the World Wide Web," in *Proc. USENIX Symp. on Internet Technologies and Systems*, December 1997.
- [4] S. Glassman, "A Caching Relay for the World Wide Web," in *First International Conference on the World Wide Web*, (Geneva, Switzerland), May 1994.
- [5] R. Tewari, H. Vin, *et. al.*, "Resource Based Caching for Web Servers," in *Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN98)*, (San Jose, CA), 1998.
- [6] J. Pitkow and M. Recker, "A Simple Yet Robust Caching Algorithm Based on Dynamic Access Patterns," in *Second International World Wide Web Conference*, (Chicago, IL), October 1994.
- [7] James E. Pitkow, "Summary of WWW Characterization," in *Computer Networks and ISDN Systems*, vol. 30, pp. 551–558, 1998.
- [8] P. Cao and C. Liu, "Maintaining Strong Cache Consistency in the World-Wide Web," *IEEE Transactions on Computers*, vol. 47, pp. 445–457, April 1998.
- [9] P. Cao and S. Irani, "Cost-Aware WWW Proxy Caching Algorithms," in *Proceedings of the 2nd Web Caching Workshop*, (Boulder, Colorado), June 1997.
- [10] R. Wooster, M. Abrams, "Proxy Caching that Estimates Page Load Delays," in *Sixth International World Wide Web Conference*, (Santa Clara, CA), April 1997.
- [11] A. Reddy, "Evaluation of Caching Strategies for an Internet Server," in *International Conference in Multimedia Computing and Systems*, (Ottawa, Ontario, Canada), June 1997.
- [12] M. Baentsch, *et. al.*, "Introducing Application-level Replication and Naming into today's Web," in *Fifth International World Wide Web Conference*, (Paris, France), May 1996.
- [13] A. Dingle and T. Partl, "Web Cache Coherence," in *Fifth International World Wide Web Conference*, (Paris, France), May 1996.
- [14] E. Markatos, "Main Memory Caching of Web Documents," in *Fifth International World Wide Web Conference*, (Paris, France), May 1996.
- [15] A. Chankhunthod, P. Danzig, *et. al.*, "A Hierarchical Internet Object Cache," in *Proceedings of the 1996 USENIX Technical Conference*, (San Diego, CA), January 1996.
- [16] M. Arlitt and C. Williamson, "Web Server Workload Characterization: The Search for Invariants," in *SIGMETRICS 96*, pp. 126–137, May 1996.
- [17] S. Williams, M. Abrams, *et al.*, "Removal Policies in Network Caches for World-Wide Web Documents," in *ACM SIGCOMM*, (Stanford, CA), pp. 293–305, August 1996.
- [18] T. T. Kwan, R. E. McGrath, and D. A. Reed, "User Access Patterns to NCSA's World Wide Web Server," CS Tech Report UIUCDCS-R-95-1934, University of Illinois at Urbana-Champaign, February 1995.
- [19] A. Bestavros, "Using Speculation to Reduce Server Load and Service Time on the WWW," CS Tech Report BUCS-TR-95-006, Boston University, February 1995.
- [20] A. Bestavros, "Demand-Based Document Dissemination to Reduce Traffic and Balance Load in Distributed Information Systems," in *Proc. of the 7th IEEE Symp. on Parallel and Distributed Processing*, (San Antonio, TX), October 1995.
- [21] E. Markatos and C. E. Chronaki, "A Top-10 Approach to Prefetching on the Web," Tech. Rep. 173, ICS-FORTH, Heraklion, Crete, Greece, August 1996.
- [22] A. Bestavros and C. Cunha, "Server-Initiated Document Dissemination for the WWW," *IEEE Data Engineering Bulletin*, vol. 19, pp. 3–11, September 1996.
- [23] V. Almeida, A. Bestavros, *et. al.*, "Characterizing Reference Locality in the WWW," in *Proceedings of the PDIS'96*, (Miami Beach, FL), December 1995.
- [24] A. Bestavros, *et. al.*, "Application Level Document Caching in the Internet," in *IEEE SDNE'96, The Second International Workshop on Services in Distributed and Networked Environments*, Whistler, British Columbia, June 1995.