A Study on Workload Characterization for a Web Proxy Server[♦]

George Pallis*, Athena Vakali*, Lefteris Angelis* and Mohand Saïd Hacid†

*Department of Informatics,
Aristotle University
54124, Thessaloniki, Greece
{gpallis, avakali, lef}@csd.auth.gr

†Université Claude Bernard Lyon 1 - Bâtiment
Nautibus
8, boulevard Niels Bohr
69622 Villeurbanne cedex, France
mshacid@lisi.insa-lyon.fr

Abstract

The popularity of the World-Wide-Web has increased dramatically in the past few years. Web proxy servers have an important role in reducing server loads, network traffic, and client request latencies. This paper presents a detailed workload characterization study of a busy Web proxy server. The study aims in identifying the major characteristics which will improve modelling of Web proxy accessing. A set of log files is processed for workload characterization. Throughout the study, emphasis is given on identifying the criteria for a Web caching model. A statistical analysis, based on the previous criteria, is presented in order to characterize the major workload parameters. Results of this analysis are presented and the paper concludes with a discussion about workload characterization and content delivery issues.

Keywords: Web Technologies, Web Caching, Web Data Workload Analysis.

1. Introduction

World-Wide-Web (WWW) is growing so fast that Web traffic and high server loads are already the dominant workload components for bandwidth consumption. This rapid growth is expected to persist as the number of Web users continues to increase and as new Web applications (such as electronic commerce) become widely used.

Caching is the idea of storing frequently used information in a convenient location so that it can be accessed quickly and easily for future use [1]. The idea of "Web Caching" is to store data at several locations over the Internet. Furthermore, caching is a technology that it has already been familiar in other applications. (Many hardware devices cache frequently used data in order to speed their processing tasks). Despite the fact that there have been great efforts towards this direction, results have shown that the existing solutions are beneficial but need to be improved to accommodate the continuously growing number of Web users and services [2, 3]. More recent results suggest that the maximum cache hit rate that can be achieved by any caching algorithm is usually no more than 50%. This simply means that one out of two documents cannot be found in the cache.

Workload characterization is a basic issue in systems design as it contributes to better understand the current state of the system. It provides a compact description of the load by means of quantitative and qualitative parameters and functions. Measurements have to be collected under varying load conditions.

Recent studies suggest that the workload characterization is basically affected by the daily routine of the users. In particular, a number of workload studies of Web proxies have already been reported [3] and many other studies have examined the workloads of various components of the Web, such as servers, clients and HTTP protocol [4].

In [5] we described the most common architectures which deal with WWW caching, giving more emphasis on proxy caching scheme. Proxy caching has become a well-established technique for enabling effective file delivery within the WWW architecture. Proxy caches can be implemented either as explicit or transparent proxies. By considering, that it is useful to be able to assess the performance of proxy caches, we presented the metrics and factors for evaluating proxy cache performance. In general, several metrics are used when evaluating Web cache performance. The most common of them are described later.

In this paper we focus on the characterization of a Web proxy workload. The purpose of this study is to contribute on a better understanding of today's Web traffic contexts and to set the stage for analysis of system resource utilization as an operation of Web server workload. With an understanding of the workload as it develops gradually over time, we can analyze the demands of users and characterize trends in user behavior over time.

The remainder of this paper is structured as follows: In Section 2, an overview of Web caching servers is presented, with emphasis on Squid proxy server. In Section 3 the most important criteria for a Web caching model are discussed. Section 4 describes the statistical analysis of the workload characterization study based on the identified criteria. Section 5 presents the detailed results of the workload characterization. Finally, Section 6 summarizes the paper, presents our conclusions and discusses future directions in Web proxy caching research.

[♦] This work is supported by the bilateral research programme of cooperation between Greece-France, GSRT, Ministry of Development General Secreteriat for Research & Technology, 2002-2004.

2. Web Caching Servers

The purpose of a Web server is to provide an environment for documents' availability to clients who request them. Web caching servers can be configured to record information about all of the requests and service responses as processed by the server. Web caching is implemented by proxy server applications developed to support many users. A Web proxy server is a special type of Web server, since it is a link between clients' browsers and Web servers over the Internet.

2.1The Squid Proxy Server

The Squid software has been developed as a free version of the Harvest software. The Squid proxy server belongs to the second generation of proxy servers. It is a fast single process server (implements its own "threads" in a select-loop) that uses the **Internet Cache Protocol** (ICP) to cooperate with other proxy servers. ICP is primarily used within a cache hierarchy to locate specific objects in sibling caches.

The Squid proxy server has been installed in many academic institutions such as the **Aristotle University** (AUTH) and it is one of the top proxy servers in the Greek Universities. Aristotle University has installed Squid proxy cache for main and sibling caches and supports a Squid mirror site. The data used in our statistical analysis come from this Squid proxy server.

In general, Squid supports log files which are a valuable source of information about Squid workloads and performance. The logs include not only access information, but also system configuration errors and resource consumption (i.e. memory, disk space). There are several log files maintained by Squid. Some have to be explicitly activated during compile time; others can safely be deactivated during run-time. We will give emphasis on store.log and access.log files which deal with our analysis. In Table 1, we present an access.log and a store.log entry that usually consists of (at least) 10 columns separated by one or more spaces:

Store.log	Timestamp, File Number, HTTP Reply Code, Date, Lastmod, Expires, Type, Sizes, Read-Len, Method, Key
Access.log	Timestamp, Duration, Client Address, Result Codes, Bytes, Request Method, URL, rfc931, Hierarchy Data/Hostname, Type

TABLE 1: Summary of access.log and store.log files

3. Criteria for characterizing a Web Caching Model

Earlier research efforts have shown that it is not simple to determine a universal model for Web traffic. In any case, it is required to acquire very good knowledge about several Web caching issues and experience in finding mathematical models. As a first step it is very important to decide on the total recording of requests and their statistical analysis. The statistical analysis of proxy data will help to log the capacity of files that were requested by the users and will lead off to a mathematic model. This model will become the base for a new replacement policy.

3.1 Identifying user request patterns

In the past, there have been numerous attempts to deduce a universal model for Web traffic [6, 7, 8]. These are usually based on statistics collected from various servers. The primary goal of such a modeling is to better understand the overall functionality of the Web and to develop synthetic Web workloads. In designing a universal Web caching model it is crucial to estimate access probabilities and predict user request patterner. Therefore, effective Web caching models are based on more than one features. For example, a model is useful for caching when we characterize Web accesses and Web re-accesses. Any methodology must be able to model multiple features adequately. Here, we categorize on earlier research efforts for designing a model for Web caching. More specifically, we identify the factors that are immediately related with characterizing future user accesses. These factors include:

Size of the documents: A re-access probability is derived in [8] as a function of cache full size, the number of past accesses and the time since last access. The probability is evaluated by manually fitting an exponential function to empirical data. Let *N* be the total number of different documents and D_i be the set of documents accessed at least *i* times, and $\|D_i\|$ the size of the set. The parameter $P(i)=\frac{\|D_{i+1}\|}{\|D_i\|}$, corresponds to the probability that a document is accessed again after the *i-th* access. $P_{i=1}(i)$ is a direct indication of the percentage of documents for which caching is useful in any circumstance. If *t* is the time from the last access, *i* is the number of previous access and *s* is the document size, it is estimated that:

$$P_r(i,t,s)=\{ \begin{matrix} P_{i=1}(i,s)(1-D(t)) \\ P(i)(1-D(t)) \end{matrix} ,$$

where D(t) is the exponential function to empirical data and P_r is the probability that the document is accessed again in the future. P_r is different for each document and is time-dependent. Here, the results have shown that the probability of re-access appears to depend heavily on the size of the documents.

Number of references: A mathematical model presented in [6] is based on the distance between successive accesses to cache objects. According to this model, the authors consider a Web server with *N* different documents. Let r_t be a HTTP request at time *t* and let D_t be the number of the document referenced by an HTTP request at time *t*. It is also defined an LRU stack $stack_t$, which is an ordering of all N documents. Let $stack_t=[D_1, D_2, ..., D_N]$, where $D_1, D_2, ..., D_N$ are the documents of the server (D_1 is the most popular document). Whenever a reference is made to a document, the stack is updated. Supposing that l_t is the stack distance of the document referenced at time *t* and $r_{t+1} = D_{dist}$, we have the following relation:

If $r_{t+1} = D_{dist}$, then l_{t+1} = *dist*, where *dist* is the distance in $stack_t$.

Thus, to any HTTP request the string $\rho = r_1, r_2, ..., r_t$ corresponds to a distance string $\delta = l_1, l_2, ..., l_t$. This distance string reflects the pattern in which users request documents from a server. So, this model is mainly based on number of references.

Size and number of references: In [7] a logistic regression model is presented as a simple repeatable method of analysis. According to this model, the probability *(P)* of an object to re-access at least once in the next W_F accesses, where W_F is a defined number, is given. This model is based on the following predictors: (1) Size of object (X_1), (2) Type of object (X_2), (3) Number of times it has been accessed (X_3), (4) Time since last access (X_4).

The probability *(P)* is given from the relation $P(Y=g \mid 1, X_1, ..., X_k) = f(z)$, where *Y* is the binary result of the *ith* access which takes on the value of one or zero, according to whether or not a specific event occurs during the study period. This model depends heavily both on the number of references and on the size of the documents.

3.2 Proxy Performance Estimates

Definition 1
Hit Ratio (HR) is the percentage of the number of requests that are served by the proxy cache content over the total number of requests.

If the objects are homogeneous in size, the above measure may be a reasonable feature of effectiveness, i.e. a high HR reflects an effective cache policy. If the objects are of varying sizes, the byte hit ratio (BHR) is a better performance metric.

Definition 2
Byte Hit Ratio (BHR) is the percentage of the number of bytes that correspond to the requests served by the proxy cache over the total number of bytes requested.

The bandwidth utilization is another significant measure where the obvious objective is to reduce the amount of bandwidth consumed. It is the percentage of available bandwidth used over a given time interval. Another criterion for Web caching model is the average speedup in retrieving documents, which is achieved by the cache and it is also discussed in [8]. In general, the retrieval time of a given document *d* of size s_d depends on its presence in cache. We can make here a distinction between cached and non cached documents. For cached documents the retrieval time is $\frac{s_d}{B_d}$, where B_d is the actual bandwidth between the cache and the client. For no-cached documents (the document must be fetched from the origin server) the retrieval time is $\frac{s_d}{\min(B_d, b_d)}$, where b_d is the actual bandwidth to the server providing the document. In most of the cases the clients have a high bandwidth to the proxy. So, we can assume that $b_d \ll B_d$. After some analysis it is estimated in [21] that

$$\textbf{\textit{Speedup}} = \frac{1}{(\bar{b}/\bar{B})BHR + 1 - BHR} \approx \frac{1}{1 - BHR}$$

because $\bar{b} \ll \bar{B}$.

From the final equation we observe that a high BHR reflects to a high average speedup in retrieving documents, which is achieved by the cache.

In [5] we described the most common metrics and factors for evaluating proxy cache performance. Here, we presented more metrics in order to deduce a universal model for Web traffic. These metrics can be classified into three categories: (1) Re-access probabilities, (2) Distances between successive accesses, (3) Cache's Speed-up.

These metrics are highly depended on other factors such as the size and the type of the objects, the byte hit rate, the time since last access and the number of references. For example the average speedup, which is achieved by the cache, is depended on the byte hit rate.

4. Statistical Analysis and Model Definition

This section describes the statistical analysis of the workload characterization study based on the criteria presented in the earlier section. The present paper uses the access logs data from AUTH cache installation for experimentation. Due to the extremely large access logs created by the proxy, we needed to separate the period which we collected the data into three different time periods. We selected three weeks randomly, a necessary convention, due to storage constraints and to ensure that our workload analyses could be completed in a reasonable amount of time. Despite these gaps in the data set, we have a relatively complete view of the proxy workload for an extended period of time. Analytically, these logs were collected on a daily basis from 4th February until 11th February and from 4th March until 11th March and finally from1st April until 8th April 2001. The statistical package "SPSS 10.0" and a few Perl scripts are the tools which we mainly used for this analysis.

As referred in previous sections, each entry in an access log contains information on a single request received by the Squid proxy from a client. However, not all information of interest is available in the access log files. Table 2 contains the summary of the statistics for the raw data sets used. The access logs contain a total of 4850661 requests for a total of 39.9 GB content data. Finally, the Squid's cache size is among 30 GB and 50 GB.

Access log duration	**4-11 February 2001**	**4-11 March 2001**	**1-8 April 2001**
Total Requests	1508620	2130403	1211638
Total Content Data	13.5 GB	17.3 GB	9.1 GB

TABLE 2: Summary of access log data

4.1 Protocols

Our first analysis focuses on the protocol (e.g. HTTP, FTP, SSL) of each URL in the data set. Table 3 shows that HTTP is the dominant protocol, accounting for over 98% of all requests and almost 93% of the content data. The only other protocol responsible for any significant amount of activity is FTP. In spite the fact that FTP have been seen in only 0.2% of requests, it accounted for a large amount of the total content data. The primary purpose of FTP in this workload is to provide clients quick access to very large files.

	TCP	**TCP BYTES**
Protocol	# Requests (%)	Content Data (%)
HTTP	1481424 (98.9%)	12489.02 MB (93%)
SSL	13790 (0.9%)	-
FTP	2769 (0.2%)	963.54 MB (7%)

TABLE 3: Breakdown of requests by protocol for the time period between 4/3/01 and 11/3/01

4.2 Analysis of Document Types and Result Codes

In order to infer more results from the access.log data, the access logs need some kind of "clean-up" before analysis can be performed. Each entry in an access log contains information on a single request received by the proxy. As a first step, we filtered the raw Squid access logs by removing four fields of each request. More specifically, we removed the IP address of the connecting client, the requested URL, the *rfc931* field and the hierarchy data/hostname field. We kept the time of the request, the number of bytes that delivered to the client, the result codes, the content type of the objects, the request methods and finally the elapsed time of the request.

In order to process the remaining information we made a coding of data in access logs according to their characteristics. Specifically, we classified the types of files being requested by clients. We placed each file into one of seven categories: (1) Application Files, (2) Audio Files, (3) Image Files, (4) Text Files, (5) Video Files, (6) Query Files, (7)All Others. The 'All others' category contains all files that could not be classified based on their extension, although they could belong in one of the defined categories.

Secondly, we categorized the result codes into three categories: (1) MISS, (2) DENIED, (3) HIT.

Subsequently¸ we used the following attributes:

X_1 = size of each document s_d (in bytes)

X_2 = type of document with the following categories: 1:Image files, 2:Text files, 3:Audio files, 4:Application files, 5:Video files, 6:Query files, 7:All others files

Table 4 shows the results of the file type analysis for the requests. This table shows also the size of all transfers from the proxy for the above mentioned period.

Type of File	**# Requests (%)**	**Content Data (%)**	**Hit Rate**	**Byte Hit Rate**
Application Files	107635 (5%)	6182.11 MB (35.7%)	18%	~4%
Audio Files	1076 (~0%)	15.06 MB (~0%)	27%	13%
Image Files	1016757 (48%)	3540.76 MB (20.4%)	60%	41%
Text Files	90758 (4%)	888.32 MB (5.1%)	34%	24%
Video Files	1211 (~0%)	1091.81 MB (6.3%)	20%	11%
Query Files	716746 (33.8%)	2186.88 MB (12.6%)	2%	6%
All Others	183359 (9.2%)	3414.81 MB (19.9%)	29%)	17%

TABLE 4:Unique file size information for the time period between 4/3/01 and 11/3/01

These results indicate that images and queries files account for most of the requests. Another observation from these results is that there does not appear to be a significant number of requests for multimedia files (video and audio). This may be due in part to a lack of multimedia objects on popular Web sites.

Additionally, image and application files account for just over half of the content data transferred from the proxy to the clients. We can observe, also, that the percentage of image and text files are less than the percentage of requests that these types receive. This is because the most of the text and image files are quite small. The image files have also very high hit rate and byte hit rate. On the other hand, the transfer of larger file types such as video impacts the content data heavily. While these types make up only 0.1% of the requests, they accounted, the most times, for 10 % of the content data traffic.

4.3 Hit Rate and Byte Hit Rate Evaluation

In general, several metrics are used when evaluating Web cache performance. The most common of them were described analytically in the previous sections. In order to determine the proxy cache hits and misses, we should examine the proxy and the origin server response codes.

Period of time	**Hit Rate**	**Byte Hit Rate**	**Miss Rate**	**Byte Miss Rate**	**Denied Rate**	**Byte Denied Rate**
4/2/01-11/2/01	50.2%	20.6%	49.4%	79.3%	0.4%	0.1%
4/3/01-11/3/01	34.9%	16%	64.6%	83.5%	0.5%	0.5%
1/4/01-8/4/01	49.7%	16%	49.5%	83.3%	0.8%	0.7%

TABLE 5: Summary of result codes

Earlier, in Section 3 we identified that the speedup in retrieving documents (achievable with a cache) can be expressed as

$$Speedup=\frac{1}{1-BHR}$$

So, in our data collection sets, we result in *Speedup=1.26 (BHR=0.206)* for the first period and *Speedup=1.19 (BHR=0.16)* for the other periods of time.

5.Experimentation – Results

5.1 Zipf-Law distribution

In several earlier research efforts it has been observed that the relative frequency with which documents are requested follows Zipf's law [9]. Consider a cache that receives a stream of requests for Web documents. Let N be the total number of Web documents in the universe. Let all the documents be ranked in order of their popularity where document d is the d^{th} most popular document. Zipf's law states that the relative probability of a request for the d^{th} most popular document is proportional to Ω/d^{a}, with $0<a<1$, where Ω is a normalizing constant and is defined as

$$\Omega=\left(\sum_{d=1}^{N}\frac{1}{d^{a}}\right)^{-1}$$

The strict Zipf's law has $a=1$.

Parameters of Zipf-like Distribution	Values
Total Request of Objects	1505517
Total Unique Objects	95294
Ω	957.746
a	-0.617

TABLE 6: Parameters of Zipf-like distribution

After some analysis, Table 6 results that the request distribution does not follow the strict Zipf's law because $\alpha<0$. Another result of this analysis, is that the percentage of the total unique objects to the total request objects is 6.3%. Unique objects are the objects that are requested only once during the observed time.

5.2 File Size Distribution

Figure 1 shows the cumulative file size distribution. Most files appear to be larger than 1000 bytes. Figure 5 shows that the distribution of file sizes is heavy-tailed. A distribution is considered to be heavy-tailed if the asymptotic shape of distribution is hyperbolic [10]. According to this paper, the simplest heavy-tailed distribution is the *Pareto* distribution.

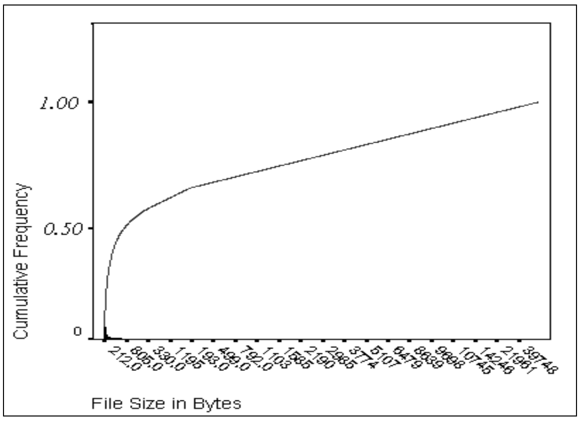

FIGURE 1: Distribution of File Sizes

5.3 Time Analysis

A characteristic registered in log files is the timestamp, which logs the time when the client socket is closed. The format is "Unix time" (seconds since Jan 1, 1970) with millisecond resolution. Here we focus on the time between current time and when the client socket is closed. Firstly, we compute the $difftime_i$, which is defined as

$$difftime_i = time_i - time_{i-1}$$

where the $time_i$ represents the time when the client socked is closed. The $time_{i-1}$ returns the previous value of the variable $time_i$. We examined various probabilities distributions and found out that when taking the logarithm of *difftime* (ln(*difftime)*), it appears to follow the normal distribution quite closely. Figure 2 depicts this observation. Figure 2 is a Q-Q Plot, which is a well-known graphical diagnostic for the goodness-of-fit of a variable to the normal distribution. As the points cluster around the straight line, we can conclude that the variable ln(*difftime)* follows a normal distribution.

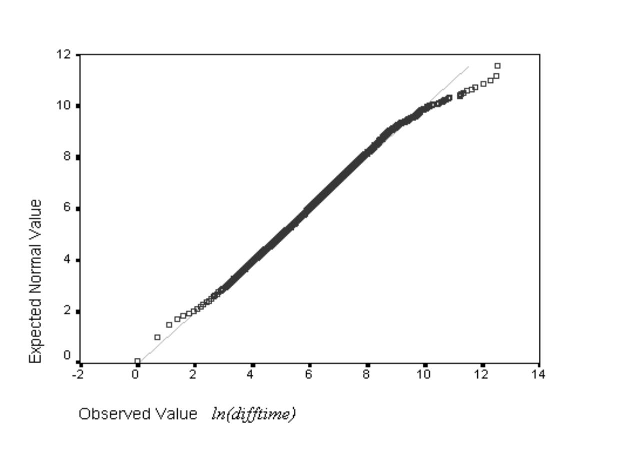

FIGURE 2: Distribution of ln(difftme)

6. Conclusions – Future Work

The WWW can be considered as a large distributed information system that provides access to shared data objects. While the Web continues its exponential growth, researches have shown that the size of static pages increases approximately 15% per month, and Web users are suffering from network congestion and server overloading. Web caching is the best solution to reduce the Internet traffic and bandwidth consumption.

In this paper, we presented a detailed workload characterization study of a busy Web proxy server. Throughout this study, emphasis was given on identifying the criteria for a Web caching model. Understanding the nature of the workloads and system demands related to WWW users is important to properly designing and provisioning of Web services. The effectiveness of caching scheme relies on the presence of temporal locality in Web reference streams and on the use of appropriate cache management policies for Web workloads. The results of this study revealed many characteristics that can impact the performance of a Web cache, such as file size, client utilization, distribution of time and frequency of reference. Finally, we concluded that the usage behavior is heavily affected by the daily routine of the users.

Despite the great efforts, which are made the last years upon Web caching schemes, we notice that there are still open problems in Web caching. Such as proxy placement, dynamic data caching, cache routing, security etc. Finally, the last year a great interest on content delivery networks has recently evolved. On the Web, content delivery is the service of copying the documents of a Web site to geographically dispersed servers and, when a document is requested, dynamically identifying and serving document content from the closest server to the user, enabling faster delivery [11]. Content delivery can also be used for specific high-traffic events such as live Web broadcasts by continually dispersing content from the originating server to other servers via satellite links. The content delivery networks (CDNs) have many possibilities to dominate in the near future.

7. Acknowledgments

The authors would like to thanks P. Junakis (System administrator), S. Anastasiadis and T. Tsiaperas (technical staff) of the Network Operation Center at the Aristotle University, for providing access to the Squid cache traces.

References

[1] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, T. Jin, *Evaluating Content Management Techniques for Web Proxy Caches*, ACM SIGMETRICS Performance Evaluation Review, Vol. 27, No. 4, pp. 3-11, Mar. 2000.

[2] J. Dilley, M. Arlitt, *Improving Proxy Cache Performance: Analysis of Three Replacement*, IEEE Internet Computing, Vol. 3, No. 6, pp. 44-50, Nov./Dec. 1999.

[3] E. P. Markatos, C. E. Chronaki, *A Top-10 Approach to Prefetching the Web*, Proc. of INET' 98 (The Internet Summit), Geneva, Switzerland, July 1998.

[4] M. Arlitt, C. Williamson, *Internet Web Servers: Workload Characterization and Performance Implications*, IEEE/ACM Transactions on Networking, Vol. 5, No 5, pp. 631-645, Oct. 1997.

[5] A. Vakali, G. Pallis: *A Study on Web Caching Architectures and Performance*, Proc. of the 5th World Multiconf. on Systemics, Cybernetics and Informatics (SCI 2001), Jul. 2001.

[6] V. Almeida, A. de Oliveira, *On the Fractal Nature of WWW and its Application to Cache Modeling*, Boston Univ. Tech. Report 96-004, Boston University, CS Dept, Boston, February 1996.

[7] A. P. Foong, Y. Hu, D. M. Heisey, *Essence of an Effective Web Caching Algorithm*, Proc. of the Int. Conf. on Internet Computing, Las Vegas, Jun. 2000.

[8] P. Lorenzetti, L. Rizzo, L. Vicisano, *Replacement Policies for a Proxy Cache*, Tech. Report LR-960731, University of Pisa, 1997.

[9] L. Breslau, P. Cao, L. Fan, G. Phillips, Scot Shenker, *Web Caching and Zipf-Like Distributions: Evidence and Implications*, Proc. of IEEE Infcom '99, pp. 126-134, New York, NY, Mar. 1999.

[10] M. Crovella, A. Bestavros : *Explaining World Wide Web Traffic Self Similarity*, Proc. of the SIGMETRICS Conf. on the Measurement and Modeling of Computer Systems, Philadelphia, pp. 160-169, 1996.

[11] M. Rabinovich, O. Spatsheck : *Web Caching and Replication.* (Indianapolis, Adison Wesley, 2002).