Web Traffic Characterization: Models for Generating Synthetic Workloads and for Prediction

Ghaleb Abdulla

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

Edward A. Fox, Chair

Marc Abrams	Ali Nayfeh
Osman Balci	Dennis Kafura

Dec 19, 1997

Blacksburg, Virginia

Keywords: WWW, Web, World Wide Web, Scalability, Modeling, Time Series

Web Traffic Characterization: Models for Generating Synthetic Workloads and for Prediction

Ghaleb Abdulla

Abstract

The rapid growth of WWW usage often is not accompanied by an overall understanding of models of information resources and their deployment strategies. Consequently, the current Web architecture is vulnerable and lacks optimized interaction between applications and network protocols [1]. Performance and reliability are two major concerns for Web users. Although the Web enhances accessibility, users still demand faster response [2]. Performance problems can result from poor protocol design [3, 4], or inadequate servers, clients, or network speed. In addition, the popularity of the Web created other unexpected problems. Scalability, latency, bandwidth, and disconnected operations [5] are some of the important issues that should be considered to make up for the rate of growth in Web usage. The WWW Consortium (W3C) realized these issues and they have launched an effort to design a new protocol that will be able to support future demands [5].

To help solve these problems we attempt to understand the Web architecture, monitor it for a sufficient amount of time, analyze the collected data, and come up with interaction models. Those models can be used to run simulations and come up with usage scenarios based on the studied history. Scalability can be achieved by the collective effort of protocol designers, browser developers, proxy and Web server developers. To reach this stage, however, we need to characterize the current types of interactions over the WWW. We need to look at the long term behavior of Web users, and find similarities and differences in

the way different communities use the Web. The results obtained can be used in generating scenarios for the new HTTP protocol design [6], for building simulation models, and for better understanding Web growth and usage.

The W3C realizes the importance of studying and understanding the system under consideration before attempting to optimize it [7]. We need to characterize the way the Web is being used. Characterization deals with: tasks, documents, access times (e.g., arrival rate of GET commands), file sizes, and other factors that will be discussed in detail in this dissertation.

In our early work we focused on proxies since they provide a good medium for caching, filtering information, payment methods, and copyright management [8]. We collected data from our environment over a period of over two years. We also collected data from other sources such as schools, information service providers, and commercial sites. Sampling time ranges from days to years. Collecting data over a long period of time allows for better understanding of the nature of interactions that appear over the Web. It also allows for caching studies and simulations. One of our findings is that the temporal and spatial locality of reference within a certain community is high, hence caching can be an effective tool to help reduce network traffic and so help in solving the scalability problem. In addition we are utilizing our findings in promoting a smart distribution or push model to cache documents ahead of time before it is accessed.

This work was partially supported by the following grants' numbers: NCR-9627922, CDA-9312611, and IBM

Acknowledgments

I wish to express my sincere thanks to Dr. Fox, my advisor and committee chairman, for his continuous support and assistance. Without his patience and cooperation throughout this research it would not have been possible to finish this dissertation. My special thanks go to Dr. Marc Abrams for his support, comments, and cooperation. His sincere efforts and comments helped me to get through with my work. Also, special thanks for Dr. Ali H. Nayfeh for his comments, advice, and encouragement. I would also like to thank Dr. Osman Balci and Dennis Kafura for their helpful comments and serving on my committee

Special thanks to IBM Corporation for their generous donation of the hardware used for monitoring the network traffic used in this study.

I also would like to thank Laurie Zirkle and Gay Meredith for their help with installing the machines and running the required software for data collection. Also I would like to thank my colleges in the NRG group and the digital library research lab for their support and

comments especially my consultant for proper English Dr. to be Paul Mather.

Special thanks to Jim Pitkow from the Xerox parc and Carry Williamson from University of Saskatchewan for their helpful comments.

I wish to thank my wife Reema Nayfeh for her unlimited support during this difficult period. I also wish to thank my children Amer and Sarah for their joyfulness and brightness.

Finally special thanks to my mother Sarah Abdulla for her support, guidance, and encouragement. This work is dedicated to the memory of my father Muhammad Abdulla who supported me all my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years the Internet in general, and World Wide Web (WWW or Web) in particular, has grown rapidly as a dissemination tool for different kinds of information resources. Frequently, the Web is used for deployment of educational and commercial material. Educators are using the Web to post course notes, syllabi, homework assignments, and even exams and quizzes. Companies are using the Web for advertising, publicity, and to sell products. Recent literature shows that the number of clients and servers is increasing rapidly and that Web traffic displays exponential growth [1]. As a result, the Web has changed the fundamental dynamics of network usage. A URL broadcast on a popular television program

(such as during a commercial in the Superbowl game in the United States) or in a historic event (such as the Shoemacker-Levy comet encounter with Jupiter) can produce a sudden spike in demand for a particular server. This phenomenon was termed the "flash crowd" by Jakob Nielsen [10], when winter sports fans tried to access the latest results of the 1994 Winter Olympics in Lillehammer that were posted on the Web by the Norwegian Oslonett.

The rapid growth of WWW usage often is not accompanied by an overall understanding of models of information resources and their deployment strategies. Consequently, the current Web architecture is vulnerable and lacks optimized interaction between applications and network protocols [1]. Performance and reliability are two major concerns for Web users. Although the Web enhances accessibility, users still demand faster response [2]. Performance problems can result from poor protocol design [3, 4], or inadequate servers, clients, or network speed. In addition, the popularity of the Web created other unexpected problems. Scalability, latency, bandwidth, and disconnected operations [5] are some of the important issues that should be considered to make up for the rate of growth in Web usage. The WWW Consortium (W3C) realized these issues and they have launched an effort to design a new protocol that will be able to support future demands [5]. Spero [4] did a careful analysis of the HTTP protocol and showed that it is not optimized for fast interactions. One of the major problems with the current protocol is that it has to establish a new connection for each retrieved document whether it is an embedded image or a stand alone document. This negotiation process introduces delay in retrieving documents and

overloads the network with unnecessary packets.

The dynamic nature and flexibility of the Web introduce other challenges. Documents are frequently edited, updated, or replaced and some pages are dynamically generated. The situation become even more complicated when users in large numbers exploit the flexibility inherent in the Web. Any person regardless of age or education can get to a browser and in no time he or she can access any document anywhere in the world. This flexibility is a major challenge to scalability and network bandwidth.

To help solve these problems we attempt to understand the Web architecture, monitor it for a sufficient amount of time, analyze the collected data, and come up with interaction models. Those models can be used to run simulations and come up with usage scenarios based on the studied history. Scalability can be achieved by the collective effort of protocol designers, browser developers, proxy and Web server developers. To reach this stage, however, we need to characterize the current types of interactions over the WWW. We need to look at the long term behavior of Web users, and find similarities and differences in the way different communities use the Web. The results obtained can be used in generating scenarios for the new HTTP protocol design [6], for building simulation models, and for better understanding Web growth and usage.

The W3C realizes the importance of studying and understanding the system under consideration before attempting to optimize it [7]. We need to characterize the way the Web

is being used. Characterization deals with: tasks, documents, access times (e.g., arrival rate of GET commands), file sizes, and other factors that will be discussed in detail in this dissertation.

In our early work we focused on proxies since they provide a good medium for caching, filtering information, payment methods, and copyright management [8]. We collected data from our environment over a period of over two years. We also collected data from other sources such as schools, information service providers, and commercial sites. Sampling time ranges from days to years. Collecting data over a long period of time allows for better understanding of the nature of interactions that appear over the Web. It also allows for caching studies and simulations. One of our findings is that the locality of reference within a certain community is high, hence caching can be an effective tool to help reduce network traffic and so help in solving the scalability problem. Recent tests and studies by our group, however, showed that proxies increase latency [9], so some type of engineering balance is needed involving network traffic quantity and time for response latency.

1.2 The problem

The central problem we address is managing the unprecedented growth and rapid evolution of the WWW in an affordable and scalable manner that leads to good performance. Accordingly, there is a great need for studies that deal with Web traffic characterization and

a methodology to continue sampling the Web traffic for future studies. Several published studies undertook partial characterization of the Web traffic, however there is a need for study that includes different user groups and spans over a sufficient amount of time. In addition the analytical models used in the literature are not accurate enough and they ignore the fact that the Web traffic is highly correlated on different time scales and displays long-range dependency.

1.3 Contribution of this dissertation

This dissertation focuses on Web traffic and task characterizations and lays the foundation for future research that will help in solving the scalability problem. In particular its contributions include the following:

- a methodology for Web traffic collection and filtering over a long period of time and a library of Web traffic data from different sources;
- a comprehensive characterization study of Web proxy traffic that has been used as a guide in answering questions raised from the W3C protocol design group;
- a detailed empirical model of arrival rates of Web traffic to a proxy server and an analysis of the long-range dependency observed in Web traffic, along with its causes;

- a study to characterize usage of the Web for information retrieval as a sample task that is performed by Web users;
- a study to characterize interactions with a digital library server that is used to deliver Computer Science course material from Virginia Tech over the Web [11]; and
- a simple WWW model with scenarios to show how we can help scale up the current Web architecture.

The core of the work in this dissertation deals with workload characterization. Network monitoring and collecting data on the internet scale is a major challenge. In this dissertation, however, we show how to sample data from user communities and how to use the data to reach useful conclusions. The Web traffic observed on the internet is mostly the result of the interactions that occur between clients and servers on intranets. We say *mostly* because much of the traffic is local or sometimes managed through caching and will not appear on the internet. For example, in all the locally collected data sets only 40%-60% of the accesses are to remote servers.

The first part of the dissertation will discuss the relation between traffic that appears between different Web *objects* such as a proxy, a client, a network, and a server, and through the data collection and analysis process. The next part of the dissertation will introduce a comprehensive Web proxy traffic characterization study. The results of the study include general statistics and comprehensive statistical and empirical models that

can be used for simulation studies and to understand the nature of Web interactions in order to help in optimizing the network and the HTTP protocol for faster responses.

The third part of the dissertation focuses on characterizing two tasks that are of great importance for Web users. The first task is the use of search tools to find information over the Web. The second task is the use of the Web as a tool to distribute information through digital libraries.

The last part of the dissertation will introduce a simple model of the WWW and will show how to use some of the results obtained from the characterization process to come up with scenarios of future Web usage. The model in this part is far from being complete, however; it serves as an example of how to make use of the identified characteristics in constructing such models and it shows some possible future usage of the Web. The purpose of the model is to identify factors that will help in solving the Web scalability problem.

1.4 Overview of this dissertation

This chapter presents the motivation to this work, the problem under study, and gives a high level overview of the structure of the dissertation. Chapter 2 is divided into two parts; the first part describes the Web architecture and methods to make it scalable. The second part describes the data collection points and the sampling method that we are using. In addition

it describes the data sets that we have collected. The third chapter describes a detailed characterization study on proxies and some implications on caching and HTTP protocol design. Chapter 4 shows how to use Fourier analysis to build a realistic model of arrival rate of Web GET commands to proxy servers and it explains the long-range dependency and strong correlation observed in the literature. Chapters 5 and 6 are two studies that characterize two different tasks that the Web is used for. Chapter 5 presents a study to characterize interactions with a digital library server that is used by the department of Computer Science and others for content and course delivery. Chapter 6 presents a study to characterize the activities while using the Web for searching and retrieving information. Chapter 7 introduces several different scenarios that show how to scale the Web for future demands. Finally, chapter 8 presents discussion and suggests future work.

Chapter 2

Background and sources of Web traffic

2.1 Introduction

The Web is a distributed set of servers and clients connected via networks. Information, which is stored on servers, can be accessed easily by any client over the network using a client application called a *browser*. The browser issues requests to the server to retrieve documents which can be identified by a network identifier called a *URL* (Uniform Resource Locator). The server responds by sending the requested document. Web documents contain formating and rendering information that is written in a language called HTML (Hyper Text Markup

Language). The browser is responsible for displaying the retrieved document using the HTML commands embedded in it. The protocol that controls transactions between clients and servers is called the HTTP protocol or Hyper Text Transfer Protocol.

The Web is one of the most successful ideas in the last decade and it has changed the way people think of computers. Computers are used not only for scientific calculations or programming, but also for information, education, and communication. Information can be shared through different formats and methods and it can be manipulated, updated, created or deleted easily. In addition, the Web introduced new users to the computer world. Home users are consuming a significant percentage of network bandwidth. Educational organizations, governments, and commercial companies have invested in this environment and will keep investing in the future. This huge success leads to new challenges that almost every Web user is willing to help solve so as to keep the Web alive.

Major challenges to the Web are scalability, latency, bandwidth, and disconnected operations [5]. Scaling issues are a major concern with the current Web architecture. To help with such issues the protocol that carries Web interactions should be redesigned to be able to meet the new challenges. Before we can do that, however, we have to understand the nature of the current interactions that appear over the network. In the following sections we define scalability, discuss some promising techniques that will help in scaling the current Web architecture, and then introduce a taxonomy of sources of Web traffic that help in

identifying data collection points for further research and exploration studies.

2.2 Scalability

Scalability can be defined as the ability to increase the size of the problem domain with a small or negligible increase in the solution's time and space complexity. Scalability of the World Wide Web architecture is the ability to increase the number of servers, clients, users, data types, data size, and the ability to handle servers in widely spread geographic locations, with minimum change in the quality of service. In this chapter we discuss methods and techniques that will help in scaling up the Web. We start by identifying techniques that improve scalability other than the brute force solutions of increasing network bandwidth and server throughput. These techniques are: use of caching on the server side (e.g., [12]), on the client side (e.g., caches built into Web browsers), and in the network (known as "Proxy caching") (e.g., [13]); creating better Web citizens; protocol enhancement; compression; and finally delta encoding.

2.2.1 Caching

One impediment to scalability is use of the wrong protocol for a given type of document delivery. For example, the aforementioned flash crowd phenomena consumes Internet band-

width and server capacity because HTTP delivers a separate document copy to each of many readers whereas multicast methods, for example, have one shared copy sent as far as possible for multiple users. The convenience of Web browsers, however, made misuse of HTTP inevitable. Nevertheless, in many cases, caching can alleviate bandwidth consumption.

The Web design, although inherently un-scalable, can be helped by widespread migration (e.g., push) of document copies from servers to points closer to users. Migration can follow a distribution model, in which servers control where document copies are stored prior to client requests, or a cache model, in which copies automatically migrate in response to user requests. Distribution or replication is popular with commercial users that want to protect material with copyrights, and will only trust certain sites to keep copies of such material, rather than relinquishing control over who has a copy of the material.

Cache placement

Caching can be implemented in three places: at servers, in the network itself, and at clients. Caching on the server side can be implemented by replicating the file system and the HTTP server and connecting the replicated servers with a high speed network [12]. This is similar to server mirroring, where the data on the server is copied into several other servers to reduce the load on the network and the original server. However in server mirroring the servers are not placed in one location; they can be separated by long distances. Server

caching is used mainly to reduce the load on the server and enhance its response time and throughput.

Caching on the client side will reflect user interests. The cache contents will change according to the user access pattern and the cache size. Caching on the network side will reflect the access pattern of a group of users who share the cache. The effectiveness of the network cache can increase by placing it where we know that a group of users have a high degree of similarity of interest (or locality of reference), and by implementing multiple [14] or hierarchical proxy caching. Multiple proxy caching is when many clients share many caches and a cache that misses can query other caches. In two-level caching we have several network caches connected to another parent or first level network cache with a larger cache size. If the document is not found in the first cache level, the second level caches are accessed, and if the document is not in the second cache level then it must be accessed from the source.

Caching will not provide a magical solution for the scalability problem but it will help in solving it. Combined with other techniques, such as data compression, server replication, and better networks we may be able to provide a usable Web architecture for the future.

Obstacles to caching

First, caching only works with static and infrequently changing documents. Some suggests that their number is declining due to the trend among commercial content providers toward dynamically generating Web pages from databases. Second, in HTTP 1.0 there is no reliable method to identify whether a Web document is cacheable (e.g., one cannot reliably distinguish a static text file from a script generated file). The HTTP 1.1 draft specification provides a response message pragma to request no caching, that can address this problem. Third, there is no accepted method in the Web for keeping cached copies consistent. Finally, copyright laws could legislate caching proxies out of existence, unless they incorporate a pay-per-view method.

2.2.2 Create better Web citizens

The network and the WWW are shared resources and will continue as such. To help scalability we should increase the awareness of this fact and we should teach users to use the network effectively to avoid wasting network bandwidth. Browsing rapidly and aimlessly through large multimedia files can result in transferring huge numbers of bytes and slowing the access of someone else. Searching if done effectively can reduce the amount of browsing required for a person with a specific information need [15]. Effective search can be achieved by having effective Web Information Retrieval Systems and by teaching users

to submit effective queries.

2.2.3 HTTP protocol enhancement

The HTTP protocol suffers from several problems; it was not designed for optimized interaction [1, 5]. One problem with the original protocol is the use of a new connection for every single document retrieved over the network. This creates extra packets over the network and causes network congestion.

Another problem with HTTP 1.0 is that it does not follow a distribution model and whenever a document is accessed by a client it has to be fetched from the original server. This increases the load on servers and increases network traffic. Mirroring using Web servers is done currently by manually creating a copy of documents on another server in another location. Proxies also are used to cache the accessed documents, however, there is still a need to notify mirrors or caches about document changes.

A third problem with the current HTTP protocol is that it is not able to carry and deliver different kinds of real time multimedia information with adequate quality of service. One specific example is video and audio delivery over the network. The current method is to use another protocol for such transactions. This introduces the overhead of starting a new communication protocol. The HTTP protocol could be designed to utilize available functionalities in such protocols for faster delivery. Work to enhance the HTTP protocol

is done by several groups; see for example Nielsen et. al. [16].

2.2.4 Compression

Compression works well with text documents but can be less effective with binary files especially if they are already compressed (such as JPEG or GIF). Using the percentage of compressible files and the average entropy in each of those types of files we can estimate how much bandwidth we can save through compression.

2.2.5 Delta encoding

The idea of caching is very appealing and experiments showed that it can save network bandwidth [17]. Caching usually assumes that the entire document should be retrieved whenever it changes. However, some documents might change because of a typo or a minor revision and retrieving the whole new document will be a waste of bandwidth. The idea behind delta encoding is to retrieve only the parts of the document which have changed [18].

Using delta encoding over the Web was suggested in [19]. The work reported by Mogul et al. [20] shows that delta encoding can provide remarkable improvements in response size and delay for certain types of Web documents. The research in this field triggered

recommendations to extend the HTTP protocol to include delta encoding in its future versions [20, 18, 21].

2.3 Web traffic, data collection and analysis

Characterizing interactions and kinds of tasks that the Web is used for is an essential step to help design a better protocol. Consequently, the WWW Consortium created a working group to collect and review the literature, gather and analyze data, and make use of available Web characterization results. The work described in this dissertation was one of the major sources that helped in answering some of the questions raised by the W3C protocol design group [22, 23, 24, 25].

Braun and Claffy pointed out the difficulty of tracking Web statistics on a large scale-e.g., the entire Internet [1]. A more reasonable approach is to study Web statistics collected from representative Web objects which are responsible for initiating, responding to, and carrying Web transactions. In this section we will describe sources of Web traffic data and how to collect data from those sources.

2.3.1 Taxonomy of Web traffic data sources

Network traffic

The lower part of Figure 2.1 shows a situation where n clients, a server, and a proxy are connected to a LAN. One or more other Web servers (or proxies) can be connected to the same LAN. However, for simplicity and without loss of generality we will assume that there is only one Web server and one proxy connected to this LAN. The upper half of the figure shows m clients, a server, and a proxy connected to another LAN.

We can think of Web traffic as transactions ordered by time between clients and servers. Let T_{LAN} be the set of all Web transactions that appear over the LAN in the lower part of the figure. Considering transaction source and destination we can define subsets of Web traffic over the LAN; the defined subsets are defined in Table 2.1.

Using the definitions in Table 2.1, then

$$T_{LAN} = T_{c,p} \cup T_{c,rs} \cup T_{c,s} \cup T_{rc,s} \cup T_{p,rs} \tag{2.1}$$

To capture this traffic we use *tcpdump* [26] with filters that we have developed to extract relevant data [27].

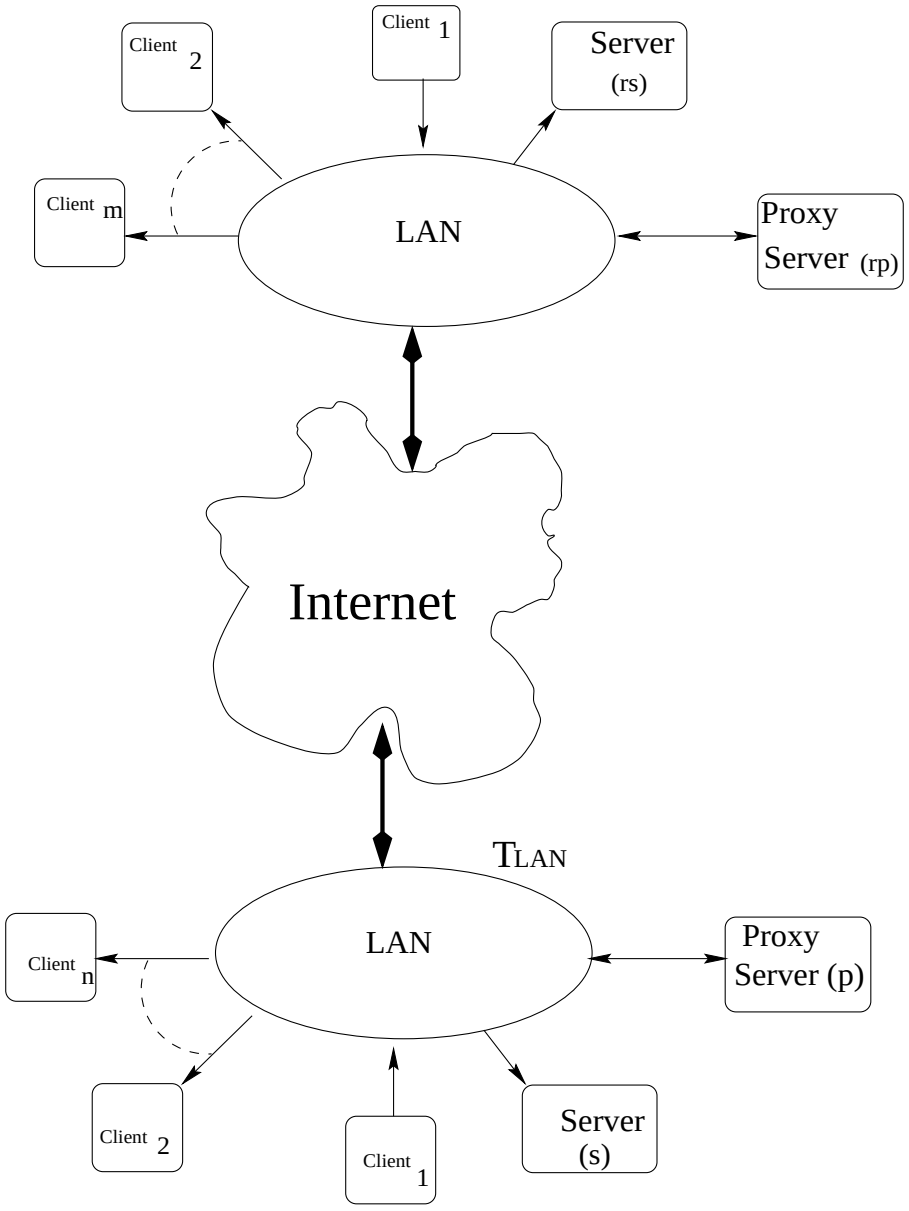


Figure 2.1: Web objects and the network model

Table 2.1: Subsets of Web traffic that appear over the LAN

Symbol	Definition
$T_{c,p}$	the set of Web transactions between local clients and local proxy server p
$T_{c,s}$	the set of Web transactions between local clients and local server s
$T_{c,rs}$	the set of Web transactions between local clients who do not use the proxy server, and remote servers rs
$T_{rc,s}$	the set of Web transactions between remote clients and local server s
$T_{p,rs}$	the set of Web transactions between the proxy p and remote servers rs

Server traffic

To simplify the analysis we will split the server traffic into two sets, traffic that results from accesses by local clients, $T_{c,s}$, and by remote clients, $T_{rc,s}$. If we let T_s represent server traffic, then

$$T_s = T_{c,s} \cup T_{rc,s} \tag{2.2}$$

Equations 2.1 and 2.2 show that $T_s \subset T_{LAN}$ and we can extract T_s if we can classify the data for T_{LAN}.

Proxy traffic

The proxy(p) traffic also can be split into two sets, $T_{c,p}$ and $T_{p,rs}$, so the expression for the proxy traffic or T_p is

$$T_p = T_{c,p} \cup T_{p,rs} \tag{2.3}$$

Equations 2.1 and 2.3 show that $T_p \subset T_{LAN}$ and we can extract T_p if we can classify the data for T_{LAN}.

Client traffic

Client traffic is from all clients going to the local proxy, local servers, or remote servers (in case the clients are not using the proxy). This traffic does not contain accesses to the client local cache. We will call it T_c.

$$T_c = T_{c,s} \cup T_{c,p} \cup T_{c,rs} \tag{2.4}$$

Equations 2.1 and 2.4 show that $T_c \subset T_{LAN}$ and we can extract T_c if we can classify the data for T_{LAN}.

We can extract T_s, T_p, and T_c from our data easily. These are the main subsets of data that should be considered when characterizing the WWW interactions. Using our knowledge about the environment where the data was collected, we can classify T_c into that originating from single user machines (personal or used serially by several people), or multiuser machines. This classification is important for characterizing user behavior and for WWW traffic characterization. Our collection method proved to be very efficient since we used it reconstruct some of the missing data for our digital library server.

Chapter 3

WWW Proxy Traffic Characterization with Application to Caching

3.1 Introduction

The dynamics of Web traffic are not well understood. There are several differences between the Web and other types of network traffic. Those differences emerge from the protocol used and from Web users' behavior. With respect to the protocol, traffic is generated by clicking on hyper-links that are part of HTML pages and, as a result, usually a new HTML page or

an image is displayed. HTML pages contain formatted text and nicely organized graphics. Sometimes HTML pages lead to other types of media, such as video or audio. In contrast, traditional network traffic has formatted or unformatted text, and rarely uses graphics, video or audio. With respect to users, the lower level of expertise required to navigate with a Web browser has resulted in a large and diverse user population. Therefore, it is reasonable to assume that Web users behave differently from those who use other network resources. Change in user behavior over a period of time may influence Web dynamics too.

Characterizing World Wide Web proxy traffic helps identify parameters that affect caching, capacity planning and simulation studies. In this chapter we identify invariants that hold across a collection of ten traces representing traffic seen by caching-proxy servers. The traces were collected from governmental, industry, university, high school, and an online service provider environment, with request rates that range from a few accesses to millions of accesses per hour. We also show that the examined traffic is self-similar. We explore sources of Web self-similarity and we conclude that a strong source is the periodicity in the users' behavior. The tests revealed that there is a strong connection between access rate and hour of the day or week. We also report the hit rate and weighted hit rate obtained by running a trace driven simulation on the workloads to simulate a proxy with infinite cache. We note that accesses to unique servers and URLs are a small portion of the total.

Earlier versions of the work described in this chapter were reported in [23, 22]. The next

section relates this study to some of the relevant literature.

3.1.1 Related work

Most of the research that uses proxy traces aims to reduce network latency, enhance response time and conserve network bandwidth. Examples of such studies are simulations using server [28, 1, 29] and proxy traffic [17, 19, 30, 31]. In those studies the authors have shown that caching will reduce network traffic.

There have been several studies to characterize client workload [32, 33] and server workload [28]. There are, however, fewer studies to fully characterize proxy traffic. This is due to the difficulty of collecting proxy log files from different sources due to the sensitivity of such logs.

One of the earliest studies to characterize proxy traffic was done by Glassman [34]. Characterization included parameters such as document popularity, cache misses and cache hits and rate of change for Web pages. Sedayao [35] studied and characterized the distribution of several parameters. Gwertzman and Seltzer [36] have used several server and proxy traffic sources to characterize MIME types and the average life span of a Web document. A study that characterizes accesses of dial-in users with modem connections was done by Gribble and Brewer [37]. They have focused on inter-arrival time and discovered that the traffic is periodic on large time scales (hourly, daily and weekly). This is similar to what

we have noticed and reported earlier in [22].

Arlitt and Williamson [28] used six different log files to characterize accesses to WWW servers. From these logs the authors identified ten different invariants for Web server workloads. The invariants are important since they aim to represent universal truths for all Internet Web servers. The invariants in the study were used to identify two strategies for cache design and to determine the bounds on performance improvement due to each strategy. A study to test if those invariants hold true over time and location is necessary.

In [32] and [33] the data was collected from a group of clients accessing the Web. Cunha et al. [33] showed that many characteristics of the WWW can be modeled using power-law distributions. Crovella and Bestavros [32] showed that the Web traffic has characteristics that are consistent with self-similarity. They traced the reasons for Web self-similarity to the basic characteristics of information organization and retrieval.

Finally, Pitkow [25] comprehensively summarizes work done in the field of workload characterization.

3.1.2 Chapter outline

In this chapter we characterize the traffic seen by a caching-proxy by identifying a set of invariants that hold true between the examined workloads. We also test for the existence

of self-similarity in the traffic and the reasons for it. We examine caching-proxy traffic over a long period of time and see if the identified invariants change with time. Finally, we run a trace driven simulation for all workloads and report the hit rate and weighted hit rate for each workload.

3.2 Workloads used in the study

To generalize our results we picked 10 different workloads from our collection; the traces come from governmental, industry, university, high school and online service provider sources. The workloads were collected either using the the normal proxy logging methods, or by simulating a proxy log file using the developed theory in chapter 2 and the tools developed by our group. In the later case we collect the WWW LAN traffic; using Figure 2.1 and equations 2.1 and 2.3 we can extract the traffic that will use the proxy (if installed). Table 3.1 summarizes the workloads used in this study, showing dates of collection, numbers of accesses represented, and total bytes transferred.

3.3 Proxy Workload Invariants

In this section we identify invariants that hold true across the workloads studied. We follow closely the work done in [28] in establishing the invariants. However, since our workloads

Table 3.1: Workloads used in this study.

Workload	Duration	# accesses	Bytes transferred (MB)
DEC1	9/3/96	1304565	11,207
DEC2	9/19/96	1293147	10,889
BU(G)	11/29/94-2/27/95	52901	294
BU(U)	1/27/95-2/22/95	414350	1,201
Korea	9/02/95-9/26/95	1681963	21,942
VT-Lib	9/19/96-11/20/96	127853	589
VT-CS	1/1/96-11/18/96	570385	3,492
VT-Han	7/12/96-11/20/96	440345	2,578
AUB	10/21/96-10/22/96	19259	110
AOL	12/96(few minutes)	883082	6,018

are for a different class of HTTP traffic, namely traffic seen at a caching-proxy server, we expect that though our set of invariants will overlap with the set identified in [28], they will not be the same. Our workload invariants for proxy-server traffic are listed in Table 3.2. These invariants are discussed in detail below.

3.3.1 File sizes

Table 3.3 shows file size statistics for the workloads. For this analysis, we removed all files that have length zero, such as dynamically-generated or cgi-bin files. The median size in all workloads is very close to 2K; the minimum for the median is 1938 bytes and the maximum is 2658 bytes. This appears in Table 3.2 as an invariant. The mean file size ranges between 7K to 27K; this is consistent with the invariant for average server file size found in [28]. This is our second invariant and also appears in Table 3.2.

The value for third quantile is less than the average file size and the maximum file size is large when compared to the median and mean file sizes; this implies that the file size distribution is heavy tailed. The variability in file size is due to the different file types; video and audio file sizes are typically huge compared to text files but they are not accessed as frequently as other types. (This will be shown in the next section.)

In Figure 3.1 we show the size distribution for the tested workloads. Clearly the size distribution in all workloads is almost the same. In all workloads small size files are

Table 3.2: Invariants for Web Proxies

Number	Name	Description
1	Median file size	$\approx 2K$
2	Mean file size	less than 27KB
3	File types (accesses)	90%-98% of accessed files are of types Graphics, HTML, and cgi-map
4	File types (bytes)	Most bytes accessed are of type graphics
5	% of accesses to unique servers	less than 11%
6	% of servers referenced one time	less than 5%
7	Accesses concentration (servers)	25% are responsible for 80%-95%
8	Bytes concentration(servers)	25% are responsible for 90% of the bytes accessed
9	Success rate	88%-99%
10	Self-similarity	$0.59 \leq H \leq 0.94$

Table 3.3: File size statistics.

Workload	Max.(K)	Mean	Median	1st Qu.	3rd Qu.	S
DEC1	22,620	8,792	2,087	386	6,831	86,758
DEC2	31,690	8,598	1,939	320	6,582	88,312
BU(G)	16,710	27,480	2,259	766	7,028	336,495
BU(U)	14,160	16,240	2,652	1,046	6,876	145,939
Korea	56,220	14,640	2,222	506	7,197	152,902
VT-Lib	12,780	7,008	2,070	738	6,044	68,958
VT-CS	20,440	9,762	2,115	557	6,225	123,637
VT-Han	26,860	8,272	2,261	864	6,064	95,956
AUB	6,779	8,863	1,938	560	5,615	96,405
AOL	46,290	7,545	2,030	790	5,963	127,056

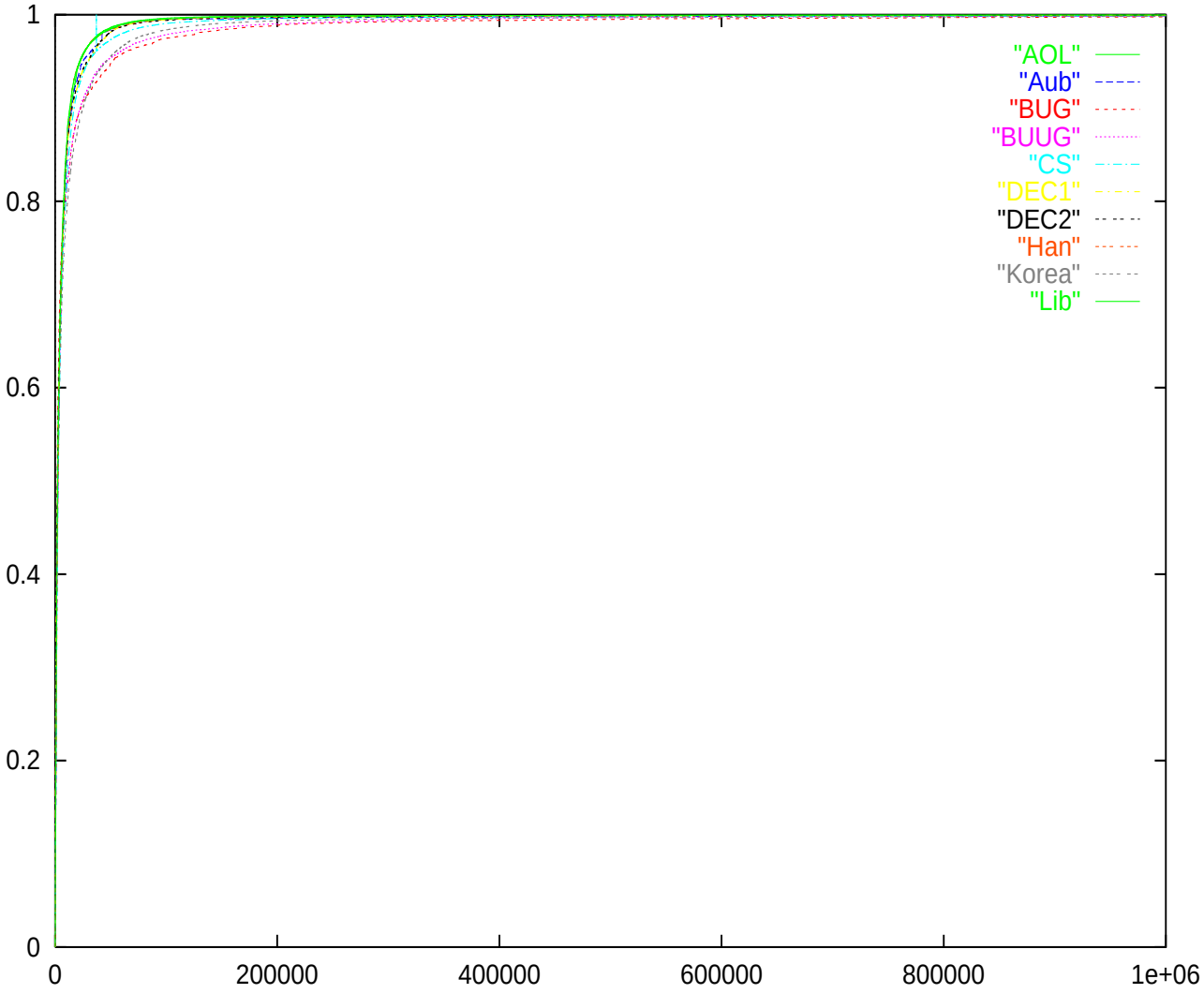


Figure 3.1: File size distribution for the tested workloads

accessed the most. In addition there is a small set of files which are very large in size and that are accessed a small number of times. The X axis in the figure was limited to the 1000000 bytes file size since if we try to plot larger files the details of the distribution in the figure will not show clearly.

In Table 3.4 we give a suitable statistical model of the file size for each workload. We used ExpertFit to derive the model in each case. The results for the Boston University workloads (BU(G) and BU(U)) were consistent with findings in [38]. In all workloads the best model suggested by ExpertFit was either a Weibull or a Lognormal distribution.

With the huge size of the tested samples, fitting the data to a statistical distribution can become very difficult and involves artistic as well as scientific judgment. Most of the goodness of fit tests rejected the suggested models, especially when we include the entire data set in the test. Sizes of the data sets tested in this case range between 10,000 and 60,000 points. For such large numbers of points we verified our models by testing the suggested distribution with sampled subsets from the original dataset or using the Q-Q plot. The Q-Q plot proved to be as effective as the other statistical tests. Another source of confidence was to compare our results with published results, when available. Modeling samples of data with such sizes is an open research topic; a tool to create random subsets of the original data set, fit the subsets into distributions, and then do goodness of fit testing will would be very useful. In a personal communication with the W3C Web characterization group, they

Table 3.4: Fitted statistical models

Workload	Model	Parameters
DEC1	Weibull	$\beta = 3,240,\ \alpha = 0.48$
DEC2	Weibull	$\beta = 3,822,\ \alpha = 0.48$
BU(G)	Lognormal	$\beta = 7.94,\ \alpha = 1.71$
BU(U)	Lognormal	$\beta = 7.97,\ \alpha = 1.57$
Korea	Lognormal	$\beta = 7.54,\ \alpha = 1.78$
VT-Lib	Lognormal	$\beta = 7.53,\ \alpha = 1.58$
VT-CS	Lognormal	$\beta = 7.37,\ \alpha = 1.89$
VT-Han	Lognormal	$\beta = 7.84,\ \alpha = 1.46$
AUB	Lognormal	$\beta = 7.88,\ \alpha = 1.44$
AOL	Lognormal	$\beta = 7.64,\ \alpha = 1.49$

mentioned that they are facing the same problem and they suggested another validation technique that uses regression to fit parts of the curves for the synthesized and the original data. In yet another case, when the data does not fit properly into any statistical model, we suggest using empirical distributions.

In Table 3.4 we list the parameters required to generate the synthesized data. The expressions for the suggested distributions can be found easily in the literature. One can use the listed parameters' values in the table to substitute for α and β in the expression for the suggested distribution to generate synthesized data sets for purpose of simulation and prediction.

3.3.2 File types

In Table 3.5 we show the percentage of accesses for each file type in each workload. Graphics files are the most accessed type in all workloads. However HTML and graphics account for less than 90% of the total accesses which is different from the servers invariants in [28]. By comparing the results from the different workloads we notice that BU(G), BU(U) and Korea follow the results reported in [28]. However, in the other workloads HTML and graphics represent less than 89% of the accesses. This is especially true in the DEC data which has a *no type* category. Based on Table 3.5, we can identify a third invariant, namely that in all workloads, graphics is the most accessed file type followed by HTML followed

by map/cgi.

The map/cgi percentage is lower in the BU workloads than in the other workloads. Since the BU workloads are the oldest set of log files (please see Table 3.1), this could mean that the percentage of *dynamic* documents is going up over time. Documents which are *dynamic* cannot be cached; hence if they have a higher percentage in the log file then we can predict that effectiveness of caching will decrease. In section 3.4 we will try to assess this hypothesis using the workload with the longest duration.

From Table 3.6 we uncover another invariant that is related to the previous one. File type graphics is responsible for most of the accessed bytes. This invariant might not hold for a long time, since video and audio files are becoming more popular.

By comparing Tables 3.5 and 3.6 we observe that types such as video, audio, compressed and Postscript/Portable Document Format (PS/PDF) constitute a very small percentage in number of accesses, however the percentage of transferred bytes for such types is significant. The files of type compressed include files that has the name extension *ZIP, GZIP, or Z.*

3.3.3 Concentration of references

In Table 3.7 we show the percentage of accesses to unique servers, which is less than 12% in all workloads, so approximately 90% of the accesses are to the same set of servers. This is

Table 3.5: Percentage of accesses for each type

Workload	graphics	HTML	CGI Map	audio	video	com-pressed	PS PDF	other	type type
DEC1	59.1	11.3	1.8	NA	NA	NA	NA	4.3	23.5
DEC2	58.5	10.9	1.6	NA	NA	NA	NA	4.5	24.5
BU(G)	82.8	12.8	2.1	0.1	0.0	0.2	0.1	1.8	NA
BU(U)	84.8	12.5	1.5	0.1	0.0	0.0	0.0	1.0	NA
Korea	66.8	21.7	5.7	0.2	0.2	0.4	0.0	4.9	NA
VT-Lib	67.1	15.0	12.7	0.0	0.0	0.0	0.0	5.0	NA
VT-CS	51.3	20.6	19.2	0.2	0.0	0.1	0.3	8.3	NA
VT-Han	69.5	14.8	8.7	0.1	0.0	0.0	0.0	6.7	NA
AUB	67.5	13.6	9.3	0.1	0.0	0.0	0.0	9.3	NA
AOL	73.9	14.2	8.2	0.0	0.0	0.0	0.0	3.5	NA

Table 3.6: Percentage of bytes transferred for each type

Workload	graphics	HTML	CGI Map	audio	video	com-pressed	PS PDF	other type	type type
DEC1	38.0	1.0	8.0	NA	NA	NA	NA	32.6	20.4
DEC2	40.1	1.0	7.5	NA	NA	NA	NA	30.8	20.5
BU(G)	21.4	2.3	10.0	28.7	3.5	6.6	10.3	17.1	NA
BU(U)	48.5	2.0	13.9	6.5	18.5	3.8	0.3	6.3	NA
Korea	48.8	2.7	7.7	2.1	12.5	14.4	2.6	9.1	NA
VT-Lib	64.1	1.5	13.3	1.7	11.6	1.5	1.2	5.1	NA
VT-CS	46.0	0.8	15.0	5.2	8.0	6.8	9.2	9.0	NA
VT-Han	53.9	1.1	9.3	7.5	13.7	2.8	0.6	11.0	NA
AUB	52.1	1.5	7.8	0.1	27.4	1.0	0.0	4.4	NA
AOL	59.9	7.9	11.2	1.5	5.0	2.2	0.4	11.95	NA

identified as an invariant and is listed in Table 3.2. We also can see that the percentage of servers which were accessed only once is very small, less than 5%; this is another invariant shown in Table 3.2. On the other hand, percentage of accesses to unique URLs, and percentage of URLs accessed one time, had close percentages in all workloads except for the DEC workloads which has a short duration and large number of clients, compared to the other workloads. By examining the logs carefully we found that many clients access a page once and stay inactive for the rest of the day. This keeps the number of unique pages and number of servers accessed one time very high.

The small percentages of accesses to unique servers and URLs, and servers and URLs accessed one time, suggests that the locality of reference is very high for all workloads and that caching should be very helpful. To check this assumption we use a simulation and check the hit rate and the weighted hit rate for the workloads. The results are described in section 3.5.

3.3.4 Distribution of accesses

In this section we examine the distribution for clients accesses and the distribution of accesses to servers and URLs. To do this we plot percentage of accesses vs. percentage of servers, URLS, and clients. Figure 3.2 shows that 25% of the servers get 80%-90% of the accesses while the other 10%-20% get the rest of the accesses. This is another invariant

Table 3.7: Concentration of references to URLs and servers.

Workload	%of accesses to unique servers	% of servers accessed one time	% of accesses to unique URLs	% of URLs accessed one time
DEC1	11.2	4.9	59.2	51.0
DEC2	11.3	4.9	60.1	51.8
BU(G)	1.3	0.2	1.8	0.5
BU(U)	0.5	0.0	0.9	0.2
Korea	1.3	0.3	2.8	1.2
VT-Lib	8.8	3.1	13.4	8.4
VT-CS	5.1	1.3	9.2	5.1
VT-Han	2.4	0.3	6.4	3.7
AUB	4.5	0.9	9.5	5.4
AOL	2.4	0.5	6.8	3.9

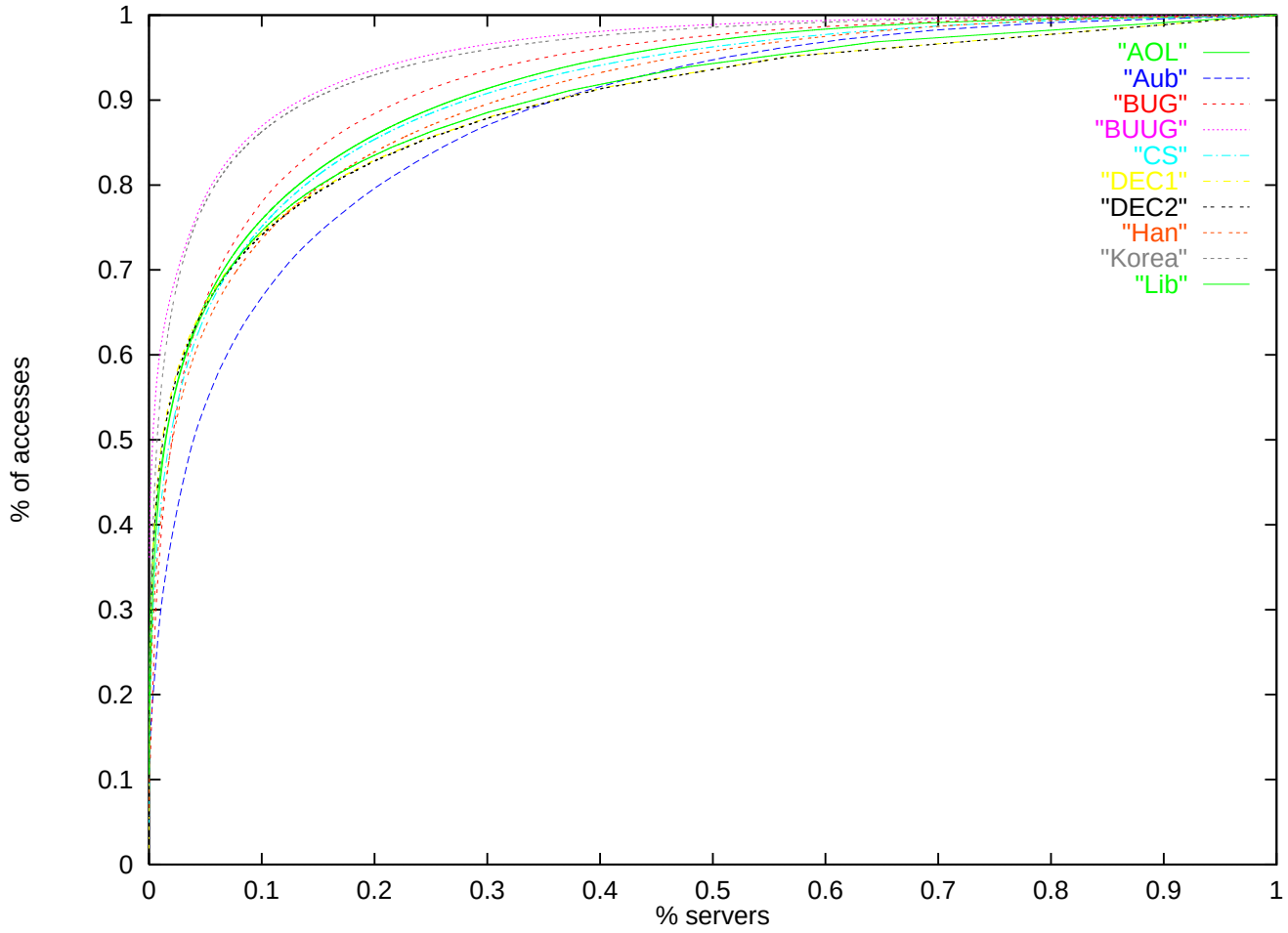


Figure 3.2: Servers concentration of accesses

that is listed in Table 3.2.

By examining Figure 3.3 we see that in all workloads, except the DEC workloads for the same reason explained in section 3.3.3, 85%-95% of the accesses go to 25% of the paths. The workloads which share this behavior have similar graphs and access distributions. The figure also shows that accesses to URLs has a heavy tailed distribution.

We examined client access distributions to see if we can find invariants across workloads. The percentage of clients versus percentage of accesses made by those clients is plotted in Figure 3.4. Except for the Boston workloads, in all other workloads 50% of the clients are responsible for 80%-95% of the accesses. This could be true because of the existence of multiuser machines associated with those other workloads where users can login and run multiple instances of the network browser. Two cases that represent the extreme points in the graph, Computer Science at Virginia Tech (VT-CS), and Boston University BU(G) are collected from similar environments, i.e., from computer science departments and graduate students. However in reality the behavior is completely different. The BU(G) curve is almost linear and approximately 50% of the clients are responsible for 50% of the accesses. On the other hand, 50% of the clients in the VT-CS workload are responsible for more than 95% of the accesses. One reason for the difference is that in VT-CS most of the accesses come from one multiuser machine. Other machines are accessed much less. In the BU(G) workload there are only five workstations, however in the VT-CS workload we have over

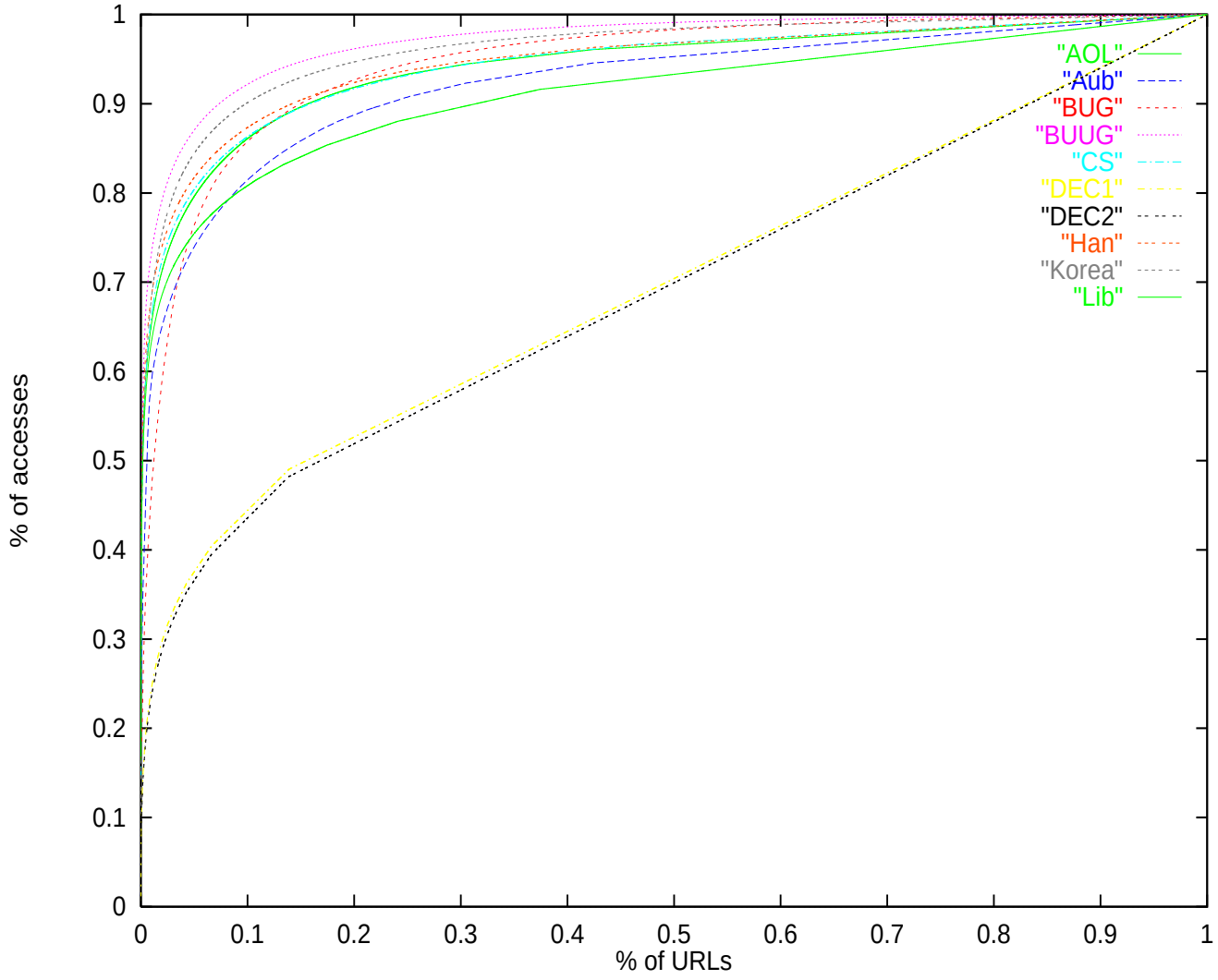


Figure 3.3: URLs concentration of accesses

thirty clients and most of them are multiuser machines.

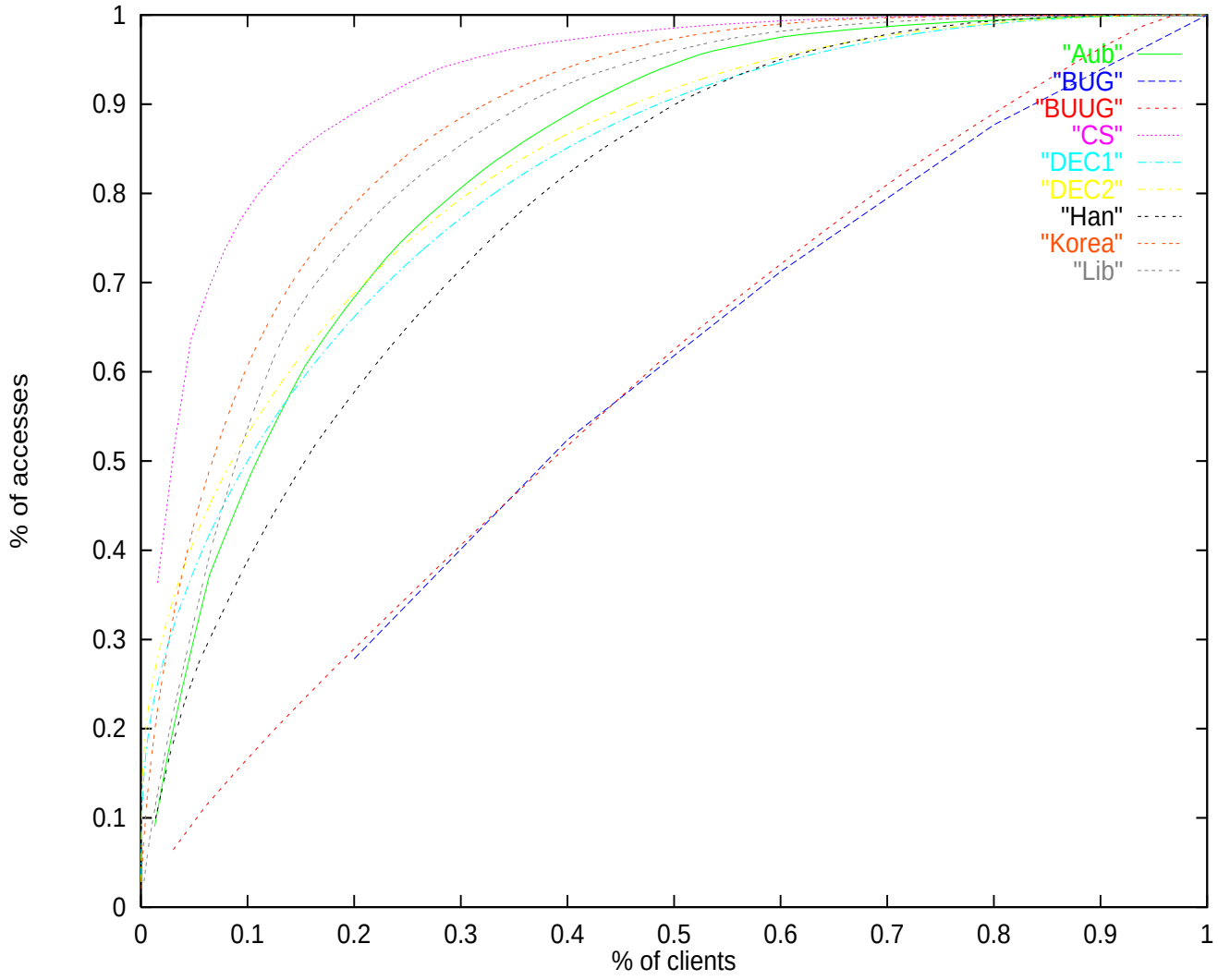


Figure 3.4: Clients concentration of accesses

Figure 3.5 shows that 90% of the bytes transferred come from 25% of the accessed servers. This is an invariant and is listed in Table 3.2.

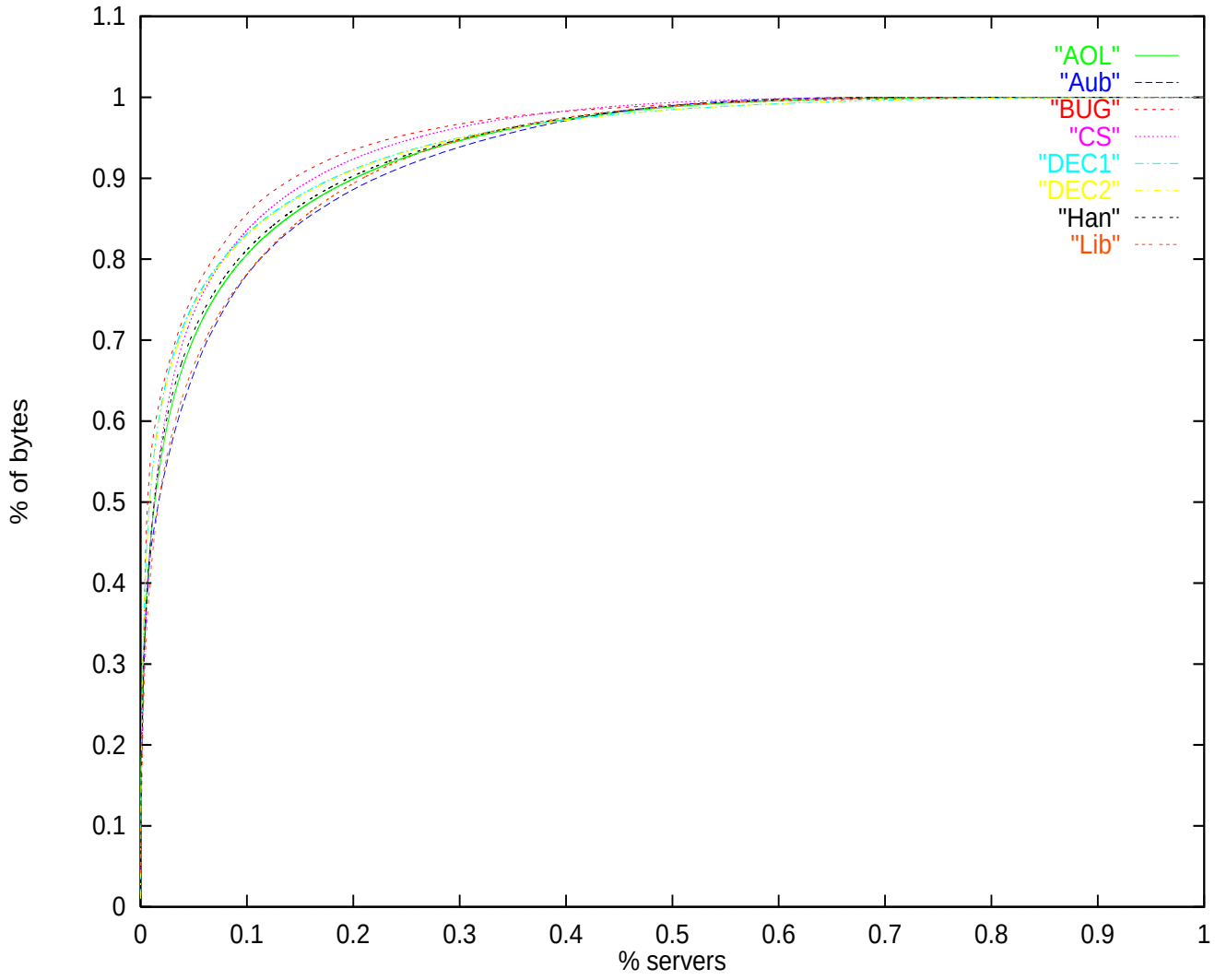


Figure 3.5: Servers concentration of bytes accessed

3.3.5 Rate of success and not modified files

Retrieving documents with no errors occurs with high percentage in all workloads, as can be seen in Table 3.8. We assume that a file is retrieved with no problems if it returns one of the following status codes in HTTP: 200 (*success*), 304 (*not modified*) or 204 (*OK but no contents*). In all workloads we notice this percentage is in the range 88%-99%. We include this invariant in Table 3.2. The 304 return code is of particular interest for us since it reflects the percentage of files that are retrieved from the local or proxy cache. We notice that it is higher in the DEC workload although the accesses to unique servers and unique URLs are very high and we will expect the accesses to the cache to be lower that the other workloads. The reason is that the DEC workload is only for one day and as a result the cache is fresh and all the accessed files are not flushed. On the other hand the other workloads are for longer times and some of the files which will be accessed more than one time might be deleted from the cache because they have expired. This shows that when we do caching and characterization studies we should consider collecting data for a sufficient period of time-at least several weeks.

Boston university and America on line workloads did not have the status code field in their log files. That is why we have "NA" for these workloads in Table 3.8.

The Status code 400, "bad request" appears in the digital proxy workload, however it is zero elsewhere. We could not find out the reason for t since the logs were encoded in a

way to preserve the anonymity of users, clients, servers, and URL's.

Table 3.8: Status codes and their percentages

Workload	200	204	302	304	400	401	404	500	other
DEC1	72.9	4.6	1.7	15.1	1.8	0.1	1.1	0.1	2.5
DEC2	71.1	4.8	1.8	16.4	2.0	0.1	1.1	0.2	2.3
BU(G)	NA	NA	NA	NA	NA	NA	NA	NA	NA
BU(U)	NA	NA	NA	NA	NA	NA	NA	NA	NA
Korea	84.4	0.0	1.0	5.6	0.0	0.1	1.7	6.0	1.0
VT-Lib	84.5	0.1	3.2	10.6	0.0	0.1	1.1	0.0	0.2
VT-CS	86.3	0.0	2.2	9.4	0.0	0.1	1.5	0.0	0.2
VT-Han	87.2	0.0	1.9	9.3	0.0	0.1	1.1	0.0	0.2
AUB	85.1	0.4	4.2	8.5	0.0	0.0	1.2	0.2	0.4
AOL	NA	NA	NA	NA	NA	NA	NA	NA	NA

3.3.6 Self-similarity

Self-similarity can be defined informally as: looking at the data with different time resolutions does not change the distribution. In other words, the data has similar properties to fractals where you can "zoom in" into the fractal and see the same structure.

This time independent structure of the data is shown in Figure 3.6. The figure shows the arrival times for a 17 day segment of a trace of all HTTP packets that have appeared on the Computer Science LAN. The x axis represents time and the y axis represents number of HTTP packets appearing on the network. These graphs can be used to test for the appearance of self-similarity in the data, however that does not prove that the data has this property.

Definitive tests for self-similarity in the collected data showed that these workloads are self-similar. Table 3.9 shows the estimated *Hurst* or H parameter [39] for tested workloads. We have generated two time series for each workload. The first one is *access rate* and the second is *bytes rate*. *Access rate* is defined as the number of accesses that a proxy gets per unit time. *Bytes rate* is defined as the number of bytes that a proxy sends per unit time. We used the R/S statistics and the normalized variance tests to estimate the H value [32]. For self-similar traffic H should be between 0.5 and 1. The closer H is to one, the stronger the self-similarity in the tested traffic. The Boston traffic has been proven previously to be self-similar [32]. Self-similarity is an invariant and it is listed in Table 3.2.

3.3.7 Invariants

Table 3.2 shows a list of the identified invariants across the workloads. Some of the identified invariants can be linked directly to caching and can be used to estimate if caching will help

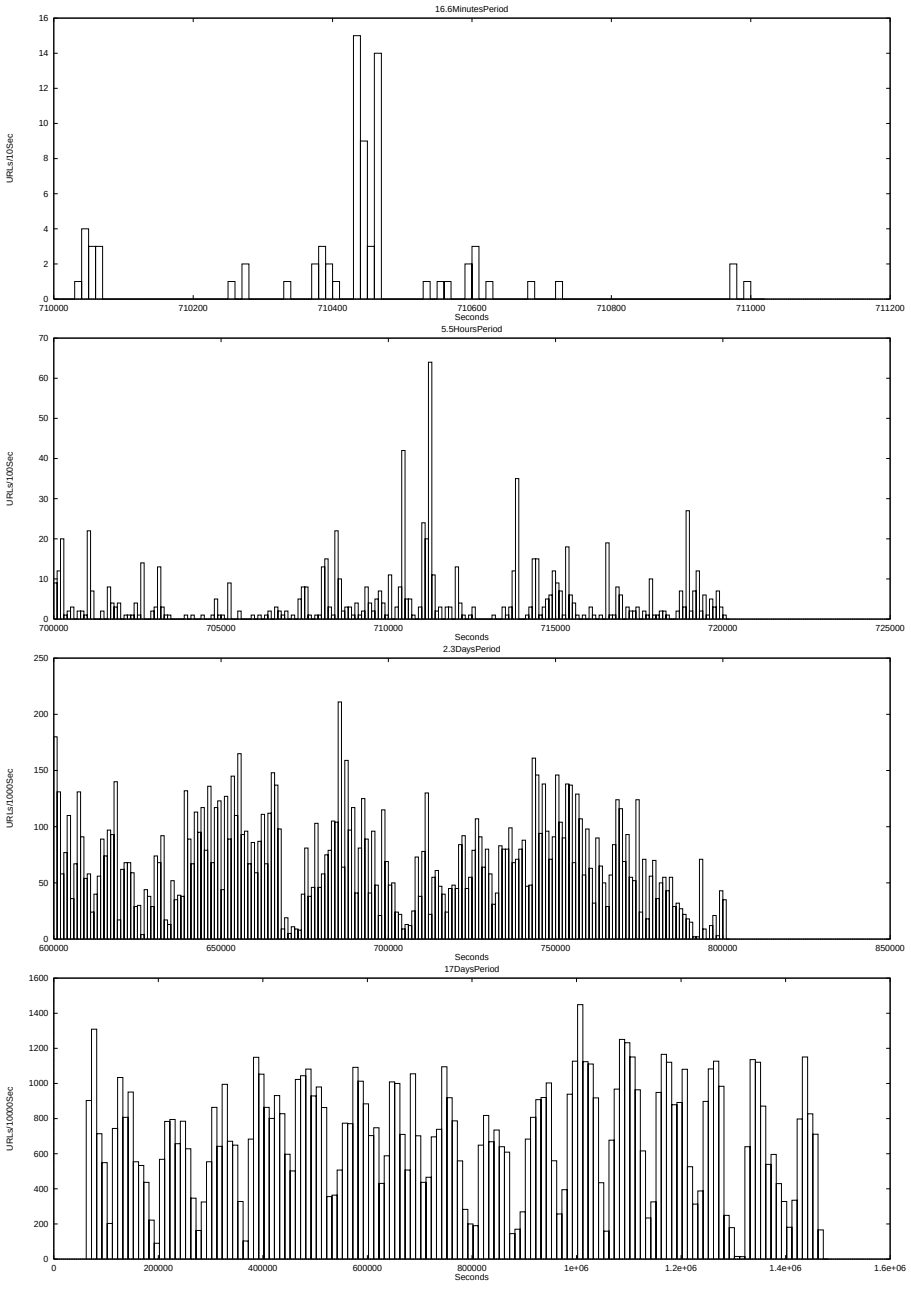


Figure 3.6: Visual test for similarity

Table 3.9: Values for the Hurst parameter

	Arrival rate		Bytes rate	
Workload	R/S	Var	R/S	Var
VT-Lib	0.93	0.93	0.81	0.86
VT-CS	0.86	0.82	0.80	0.70
VT-Han	0.79	0.75	0.59	0.65
AUB	0.94	0.85	0.82	0.81

or not. Invariants 5, 6, 7 and 8 should be good predictors for cache performance. The initial results reveal that we can model some of these distributions using Weibull distribution.

The previous discussion to identify invariants reveals two important questions that should be explored. First, do the identified invariants hold over time? Second, which of those invariants have more effect on caching? In the rest of this chapter we will attempt to answer these two questions.

3.4 Longitudinal study of invariants and self similarity.

In this section we test if the previously identified invariants hold for the VT-CS workload over time. To do this we split the workload into monthly log files and applied the same analysis that we used to identify the invariants across workloads. We used the VT-CS log file because it has the longest duration since it spans over 11 months. Although the conclusions drawn from such an analysis may not be generalized since we are using only one workload, still we can identify trends or changes that appear over time.

Table 3.10 shows the statistics for each month for the VT-CS data. We notice that accesses during the summer are lower than accesses during the regular academic year. The number of accesses is not large compared to other workloads since most of the students' accesses were to local servers and so were filtered out.

3.4.1 File size statistics

Table 3.11 shows file size statistics for each month. Some of the previously identified invariants are still true for the monthly analysis. Mean file size is still in the same range; in addition the median is also around 2K. However we notice that the mean file size seems to grow with time. To test if this is true or not we performed a linear regression with the

Table 3.10: Accesses and bytes transferred for each month

Month	Accesses	Bytes transferred (MB)
Jan	55,916	333
Feb	66,589	482
Mar	37,473	239
Apr	59,542	410
May	71,217	434
Jun	11,486	69
Jul	35,936	215
Aug	40,141	168
Sep	65,395	356
Oct	100,033	582
Nov	26,657	203

mean as the response variable and month as the dependent variable. The t statistic has a value of 3.05 and the slope is 391 which means that the model is a good predictor and that the mean file size is increasing over time. The increase appears to be due to the increase in the maximum file sizes available from the accessed Web servers, and the increase in the text, HTML, and image file sizes; we will show that this is true in the next section. This increase will continue as long as authors can create such pages, and users can access such files over the network with minimum problems. Table 3.11 reveals another trend which is the increase and decrease in the average file size; for example January and August show the lowest values. For a school setup this is normal since these are the two months when students are less active in browsing the Web. As a result we expect that students will access fewer files and that the sizes of files accessed will not vary greatly. That may be why the variance for these two months is relatively low compared to the other months.

3.4.2 File types

Table 3.12 shows percentage of accesses to each file type for every month. Another invariant from the previous section holds across the months, that is graphics and HTML account for the highest percentage of accesses. Linear regression, on the three file types graphics, HTML, and CGI/Map, revealed that all three percentages are changing over time. For the graphics file type the t statistic was -3.54 which means that the percentages of graphics

Table 3.11: File size statistics each month

Month	Max.	Mean	Median	1st Qu.	3rd Qu.	S
Jan	6,876,000	7,806	1,963	439	5,749	82,113
Feb	20,440,000	9,232	1,976	509	6060	139,921
Mar	3,964,000	8,707	2,060	431	6,196	73,368
Apr	14,580,000	9,328	2,015	468	5,812	108,827
May	12,620,000	9,163	1,884	407	5,927	104,259
Jun	2,836,000	11,940	2,608	849	7,214	74,181
Jul	16,810,000	10,910	2,250	768	6,468	150,187
Aug	5,759,000	8,115	2,176	758	6,283	67,139
Sep	17,350,000	10,680	2,250	687	6,405	172,139
Oct	13,820,000	11,070	2,371	780	6,773	127,460
Nov	16,950,000	13,890	2,544	838	6,922	182,887

files are decreasing. For the HTML file type the t statistic was 4.72; the percentage of HTML files is increasing. Finally, the CGI/Map has a t statistic of -4.96 which means that its percentage is decreasing. HTML and graphics are *static* documents, i.e., they are cacheable documents.

Table 3.12: Percentage of accesses for each file type

Month	graphics	HTML	CGI Map	audio	video	com-pressed	PS PDF	other
Jan	59.6	9.0	23.4	0.2	0.0	0.0	0.2	7.4
Feb	57.7	7.8	26.1	0.1	0.0	0.0	0.2	7.9
Mar	57.3	8.2	25.0	0.0	0.0	0.1	0.3	8.9
Apr	56.3	10.6	22.9	0.1	0.0	0.1	0.6	9.2
May	53.1	16.2	21.3	0.0	0.1	0.2	0.5	8.6
Jun	40.9	32.1	16.8	0.1	0.0	0.1	1.2	8.7
Jul	44.4	24.0	20.3	0.3	0.0	0.0	0.3	10.6
Aug	48.8	22.6	19.0	0.2	0.0	0.2	0.2	8.4
Sep	47.2	26.9	16.9	0.1	0.0	0.2	0.2	8.4
Oct	43.8	32.3	16.0	0.2	0.1	0.2	0.0	7.4
Nov	49.1	25.0	18.9	0.2	0.1	0.0	0.1	6.6

Table 3.13 shows percentage of bytes transferred for each type per month. Types such as

video, audio, PS/PDF and compressed appear with significant percentages due to their large sizes.

Table 3.13: Percentage of bytes transferred for each file type

Month	graphics	HTML	CGI Map	audio	video	com-pressed	PS PDF	other
Jan	52.5	0.4	16.8	0.8	13.3	2.5	8.7	5.1
Feb	57.5	0.3	20.1	5.7	5.8	4.1	7.4	9.5
Mar	48.7	0.4	18.9	5.2	1.4	1.0	11.8	7.9
Apr	40.0	0.1	13.6	3.5	6.4	3.5	21.9	10.8
May	48.0	1.1	14.8	1.4	11.4	3.9	8.5	10.9
Jun	40.5	0.1	11.1	4.4	2.0	6.3	19.9	15.7
Jul	38.1	0.6	12.5	20.9	0.8	3.9	14.4	8.8
Aug	54.4	0.1	14.0	1.7	7.2	5.0	7.1	6.4
Sep	45.3	2.5	15.6	2.3	7.2	17.9	6.1	4.6
Oct	46.6	1.3	12.2	10.1	7.6	10.0	2.8	11.0
Nov	39.9	0.9	12.3	6.2	21.2	5.9	3.2	10.5

3.4.3 Concentration of references

Table 3.14 shows that other invariants from the previous section, especially concentration of references (invariants 5, 6, 7 and 8), hold over time. A linear regression for each one of the variables with respect to month showed that the model is significant and that there is a positive relationship between each variable and the month. The t statistic is over 2.5 for all the variables. Hence, there is strong evidence that those percentages are increasing which means that locality of reference is decreasing and hence caching will be less effective.

3.4.4 Rate of success and not modified files

The identified invariant for success rate in the previous section holds true over the year for the VT-CS data. From Table 3.15 we notice that the percentage of the code 200 *(success)* is increasing. However the value of the status code 304 *(not modified)* is decreasing. This means that fewer files are being accessed from the cache. This is consistent with what we have noticed with respect to percentage of accesses to unique servers and URLs and with the increase in CGI/Map files.

Table 3.14: Concentration of references to URLs and servers per month

Month	% of accesses to unique servers	% of servers accessed once	% of accesses to unique URLs	% of URLs accessed once
Jan	4.4		8.9	4.2
Feb	4.3	0.7	8.5	4.0
Mar	5.2	0.8	10.9	5.6
Apr	4.1	0.7	8.6	4.4
May	4.6	0.9	8.4	4.3
Jun	6.9	1.3	10.8	5.5
Jul	6.2	1.5	10.0	5.6
Aug	5.8	1.5	9.6	5.3
Sep	5.7	1.3	9.6	5.5
Oct	4.9	1.1	8.7	4.8
Nov	9.2	2.8	13.1	7.5

Table 3.15: Status codes and their percentages

Month	200	204	302	304	400	401	404	500	other
Jan	81.7	0.0	3.1	12.6	0.0	0.2	1.8	1.1	0.5
Feb	81.5	0.0	2.4	13.8	0.0	0.1	1.8	0.1	0.3
Mar	82.4	0.0	2.2	12.4	0.0	0.2	1.8	0.4	0.6
Apr	82.6	0.0	2.0	13.6	0.0	0.1	1.4	0.1	0.3
May	84.6	0.0	2.2	11.1	0.0	0.1	1.5	0.1	0.3
Jun	90.1	0.0	1.7	5.9	0.0	0.1	1.9	0.1	0.3
Jul	88.2	0.0	1.9	8.4	0.0	0.0	1.0	0.1	0.4
Aug	84.6	0.0	2.3	10.8	0.0	0.1	1.5	0.0	0.6
Sep	90.0	0.0	2.1	6.0	0.0	0.1	1.3	0.0	0.4
Oct	92.9	0.0	1.9	3.4	0.0	0.1	1.4	0.0	0.3
Nov	90.5	0.0	2.2	4.6	0.0	0.1	2.7	0.0	0.3

Table 3.16: Theoretical maximum hit rate and weighted hit rate

	Final Size of Cache (MB)	Hit Rate (%)	Weighted Hit Rate (%)
DEC1	10,159.7	6.2	2.7
DEC2	9,881.9	5.5	2.7
Boston(G)	265.2	81.1	38.7
Boston(U)	507.9	85.5	46.7
Korea	10,078.3	63.1	51.2
VT-Lib	498.8	26.7	13.5
VT-CS	780.5	28.3	22.2
VT-Han	1,695.9	43.7	31.4
AUB	75.9	36.7	30.4
AOL	3,672.5	50.2	38.2

3.5 Invariants and Caching

Table 3.16 shows the infinite cache hit rate for each workload using simulation. We ran all the workloads through a trace driven simulation of a proxy server with infinite cache [19]. After examining Figures 3.2 and 3.3 we expect the DEC workloads to have the lowest hit rates due to their short duration and Table 3.16 confirms this result. Also by examining the same figures we expect that the BU(U), BU(G) and Korea workloads will have the highest hit rates since they have very high locality of reference. This is confirmed by the results in Table 3.16.

3.6 Autocorrelation and self-similarity

One of the identified invariants for the workloads from the previous section is self-similarity. We have shown that VT-CS data is self-similar. The value obtained for the Hurst parameter from the R/S plot was 0.86 and from the normalized variance plot was 0.82. For $0.5 \leq H \leq 1$ the correlation decays to zero slowly so that

$$\sum_{k=-\infty}^{\infty} \rho(k) = \infty$$

and the process has long memory [39]. As a result we expect that the access rate for the VT-CS data will have a strong correlation. To test for this we extracted accesses/minute, accesses/hour, accesses/day, and accesses/week. We then generated a time series and

plotted the autocorrelation function with different lags. Figure 3.7, which is the ACF plot for accesses/hour, shows that the correlation is strong and cyclic. The daily cycle is very clear in the figure; every 24 hours we get a peak.

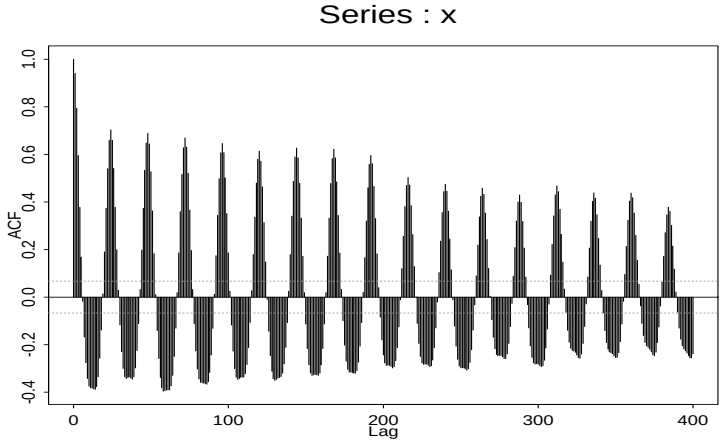


Figure 3.7: ACF for the VT-CS data by hour with Lag 100

In [32] the authors tried to identify sources of self-similarity in the Web traffic by showing that transmission times may be heavy-tailed, primarily due to the distribution of Web file sizes. The transmission times correspond to the *on* times and user think time corresponds to the *off* time. Multiplexing *on/off* times will generate a self-similar traffic. We think that there are other important sources for Web traffic self-similarity that the authors missed. Burstiness, which is a major characteristic of a self similar traffic, can appear as a result of protocol interactions and HTML document structure. For example, accessing one HTML page using HTTP 1.0 protocol will trigger a sequence of accesses to other embedded images, files or icons. This activity will appear as a burst followed by a low activity period. This

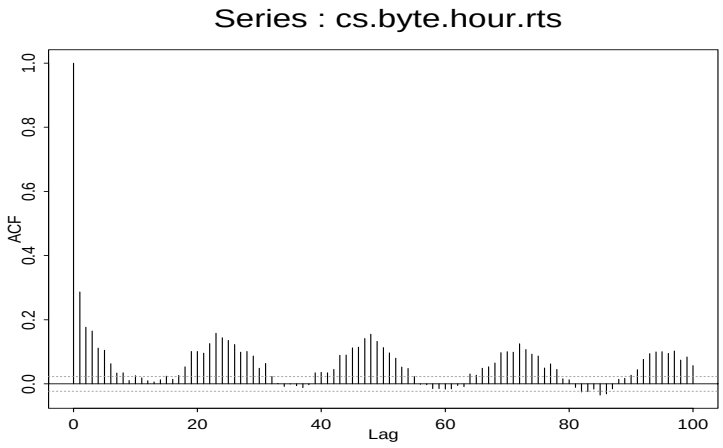


Figure 3.8: ACF plot for bytes/hour, VT-CS data

means that changes in the protocol and in the Web page design will affect Web traffic self-similarity contrary to what they have concluded in their paper.

Table 3.9 shows that self-similarity appeared in the tested data sets to be stronger with access rate than with bytes rate. As a result we think that another source of Web self-similarity is the repeated behavior of Web users since they do things in a repeated and correlated manner relative to minutes, hours, days and weeks. For example, in a school or a company environment, users will start accessing the Web slowly at the beginning of the day and in time more users will join the active ones. At lunch hour the number of accesses will go down and then build up again during the afternoon and the evening. Note that this phenomena is clear with a group of clients since most of the time they are in the same time zone. However servers get hits from all over the world and the cyclic or periodic behavior

will be weaker. This could be the reason why self-similarity was not identified as one of the invariants for the server workload in [28].

In this chapter we only try to show that there might be other sources for self-similarity and that it is dependent on users' behavior and schedule, protocol, and Web document structure in addition to the sources identified in [32]. Figure 3.7 shows that there is a significant correlation between hourly access rate. This leads us to hypothesize that a major source for self-similarity is the way users interact with Web browsers, the HTTP protocol, and the structure of Web documents. We also examined the ACF plot for accesses/minute, accesses/day and accesses/week. In all of them the correlation (ρ) was sinusoidal and different from zero. This means that the long range dependence is a behavior that exists for different time resolutions and the data is not completely random.

We test the effect of Web file sizes on the auto-correlation by plotting the auto-correlation function for bytes rate instead of arrival rate of accesses. We generated the time series bytes/hour for the same data set. Figure 3.8 shows that the correlation exists and it is periodic, however, its amplitude is less than the the amplitude noticed for request arrival rate. Self similarity also was found to be weaker in the bytes rate data (see Table 3.9). This implies that including file sizes might have a negative effect on self-similarity contrary to what was suggested by [32].

3.7 Conclusions

In this chapter we identified a set of invariants that hold true across different proxy workloads and one of the identified invariants was self-similarity. We also showed that some of the identified invariants hold over time for the VT-CS workload. One of our observations is that most of the accesses go to a small set of servers and URLs. In addition, in the majority of the workloads, a small set of clients are responsible for most of the accesses. For future work it is worth combining all workloads together and examining the distribution of accesses per server. Is there a universal set of popular servers across all workloads, and what are those servers? Does this set change over time?

We explored the sources for self-similarity in the VT-CS data and our conclusion is that a major source for self-similarity can be attributed to the periodic behavior of Web users, the protocol, and the structure of Web documents. The periodic behavior appeared at minute, hour, day and week intervals.

In the next chapter we will discuss how to model traffic that displays periodic behavior. We also will show that we can generate self-similar traffic using the suggested modeling technique.

Chapter 4

Modeling Proxy Web Traffic Using Fourier Analysis

4.1 Introduction

Proper and correct models of network and WWW traffic are important for simulation studies and for understanding the nature of interactions that appear over the Internet and the Web [40, 41, 42], as aids to planning network infrastructure. Recent studies to understand and model network traffic challenge the traditional Poisson distribution assumption [32, 43, 41, 42]. Leland and Wilson [44] show that simulations using real network traffic produce different results from simulations of synthetic traffic generated using traditional

approaches (that ignore long-range dependency in network traffic). An important feature of models created for Web traffic simulation studies is that they should have a period of several weeks or months in order to capture the cache behavior and predict user accesses for studies such as forward caching or pre-fetching. Forward caching is when documents are pushed to the places where it is anticipated that they will be accessed. Collecting traces that satisfy this requirement is a major challenge for technical and privacy reasons.

Web traffic, which is a major component of Internet traffic, displays a significant variance or burstiness over a wide range of time scales. This is an important distinguishing property of self-similar traffic. Web traffic also displays strong *auto-correlation* and *long-range dependence* as we showed in Chapter 3. The number of accesses per unit time at a certain time t is correlated with values at time $t + \tau$, where τ is an arbitrary time delay. As a result, modeling methods that assume independent and identically distributed data are not suitable for Web traffic data. New methods and approaches that capture the characteristics of Web traffic should be developed.

There are several efforts to define methods to generate self-similar traffic for network simulation purposes. Leland et al. [43] describe a methodology for generating bursty and hence self-similar traffic by multiplexing a large number of on/off sources with heavy tailed period lengths. Paxson [42] introduces "a fast Fourier transform method for synthesizing approximate self-similar sample paths for one type of self-similar process." However he

warns that "One must use caution in assuming that traffic sources are well modeled using self-similar processes." In Section 6 of the same article he questions "even if network traffic is long-range dependent, are self-similar models sufficient for capturing the long-range dependence?". By examining the Ethernet traffic collected from Bell Core we managed to remove the auto-correlation from the signal; after that we tested the random part for self-similar behavior and it was still self-similar. In other words self-similar models by itself might not be enough to capture the long-range dependency in the Web traffic especially if is constructed from a pure statistical on/off signals.

A daily cycle was observed by other researchers, see for example Gribble and Brewer [37]. To our knowledge, however, no one has succeeded in using this phenomena to model the traffic. Gribble et al. describes the daily cycle and argues that internet service providers should be ready to deal with this behavior. In this chapter we introduce a modeling algorithm that will help in forecasting future behavior by characterizing the deterministic part of the signal in addition to the random part. This chapter shows the importance of modeling the deterministic component by calculating the signal to noise ratio and showing that the deterministic component is almost 30% of the total signal; hence we cannot use pure statistical models for arrival rate. By using our earlier work to characterize other important parameters of the proxy traffic [22, 23] we are ready to generate synthetic proxy log files for simulation studies.

Barford and Crovella [38] use statistical techniques to model ON/OFF periods of arrival rates from clients at Boston University using data collected in 1994 and 1995. The work presented in this chapter is different from their paper in three ways. First, they model individual clients while we model the aggregate accesses of a group of clients to a caching proxy. Second, they use only one workload to create their models; however, we tested our approach on four different workloads and as well as on their logs. Third, they suggest an architecture of workload generators u their results and earlier results that include file size and file type distributions and other parameters. Our approach is actually to create synthetic proxy access log files for simulation studies. Using the results from this work and the results in chapter 3 and [22, 23] we have all necessary parameters to generate a synthetic proxy work l.

There are three main contributions in this chapter. First, we introduce a new approach to model traffic that has oscillatory or periodic autocorrelation and is self-similar. Using our suggested approach, we show how to generate synthetic traffic that displays similar burstiness characteristics. Our objective is not only to generate self-similar traffic, but also to identify and use traffic characteristics that can generate traffic that is long-range dependent and self-similar. The methodology is explained by applying it step by step to a specific proxy workload. Note, however, that the steps to model the deterministic part also have been tested successfully on other collected proxy workloads.

Second, by examining the auto-correlation plots for Web traffic, we uncover and explain the long-range dependency noticed in the literature [41, 22]. The auto-correlation function reveals periodic long-range dependency in the examined data. Tests over several workloads show that this periodicity is not arbitrary; it is common to all tested workloads and can be explained in terms of daily and weekly cyclic behavior of Web users. Although the daily and weekly cycles seem to be intuitive the strong and sinusoidal nature of the correlation makes the detailed analysis interesting and useful for modeling. In addition we demonstrate that these cycles are the source for Web traffic burstiness and hence they can be the source for Web self-similarity. In this chapter we go beyond identifying those cycles; we quantify and generate traffic based on user cyclic behavior.

Third, we show that the synthesized traffic has long-range dependent and self-similarity characteristics that are very close to the long-range dependent and self-similarity characteristics observed in the original data set. That is, we validate our analysis and modeling by showing our generated traffic is similar to the real traffic. This way we address the concern raised by Paxson in [42], section 6, which was quoted earlier.

We use proxy workloads because proxies are the most attractive solutions for scaling information distribution and copyright management [8]. We encourage caching using proxy servers because it captures locality of reference within the user community. Studies show that proxies have good potential for reducing network loads and thus are important for

building scalable architectures; see for example [17].

4.2 Background and discussion

4.2.1 Network traffic and long-range dependency

Long-range dependency in network traffic was noticed in the Ethernet measurement for a local area network (LAN) at Bell core and for Web proxy and server traffic [22]. Leland et al. [43] show how to model this data with a self-similar process. Beran [39] lists common features that distinguish a long-range dependent process, including both qualitative and quantitative features, and provides additional details on this subject.

The qualitative features include:

a) burstiness or long periods of high level activity are followed by periods of low level activity; and

b) the time series appears to be random when observed using a long period of time, however by examining shorter times one can see trends or cycles.

The quantitative characteristics of a long-range dependent series include:

a) the variance of the sample mean decays to zero at a rate slower than n^{-1}, where n is

the number of data points, and

b) the series correlations $\rho(t)$ decay to zero at rate that is proportional to $K^{-\alpha}$, where $0 < \alpha < 1$, and t is a large number.

4.2.2 Fourier analysis of time series

The main objective of this analysis is to study the structure embedded within the data in order to develop accurate models of it. The structure can be captured by representing the set of data in terms of sinusoidal functions. Sinusoidal functions are attractive because of their simple behavior under change of time scale.

Periodic and quasiperiodic data When a series can be expressed in terms of a sinusoidal function with one fundamental frequency ω and its harmonics, it is called periodic. However, if the data contains more than one incommensurate frequency and we need more than one periodic component to represent the signal, it is called quasiperiodic. The analysis done here shows that the data is quasiperiodic; however, for simplicity we always use the term periodic instead of quasiperiodic.

4.2.3 Web traffic characterization and self-similarity

The main objective of the study reported by Arlitt [45] and Arlitt and Williamson [28] was to identify workload invariants across a set of log files collected from different locations. The authors identified 10 invariants in all collected workloads. Self-similarity was not reported as an invariant across all workloads, however some of the tested workloads appeared to be self-similar.

Crovella et al. [32] used traces collected from Web clients to test for the existence of self-similarity in the client-based Web traffic and to explain the causes for this phenomenon. They conclude that traffic due to WWW transfers shows characteristics that are approximately consistent with self-similarity. They trace the causes of the existence of self-similarity to the basic characteristics of information organization and distribution. Accordingly, they claim that changes in the protocols or machine architecture will not affect self-similarity in Web traffic.

Abdulla et al. [22] identified the self-similarity property as an invariant for a set of proxy workloads. However, to our knowledge, no one have done a complete study to model proxy Web traffic in a way that the models can be used in simulation studies.

Table 4.1: Workloads used in this study

Workload	Collection date	Duration in days	Number of (K) accesses
Korea	9/2/95	24	1680
Library	1/14/97	62	241
CS	1/29/97	38	87
Engineering	7/12/96	131	440

4.3 Data sets used in this study

For this study we examine four log files collected from different communities. The files were analyzed using the scripts and programs listed in appendix B. The script extracts the times of access and then creates a time series of arrival rates per hour for each log file. The time series is further examined using Splus [46] and Matlab [47]. The workloads, except for Korea and Boston, by collecting the WWW LAN traffic and using Figure 2.1 and equations 2.1 and 2.3 we extract the proxy traffic.

Table 4.3 describes the workloads used in this study, showing dates of collection, duration, and number of accesses.

4.4 Visualizing Periodicity

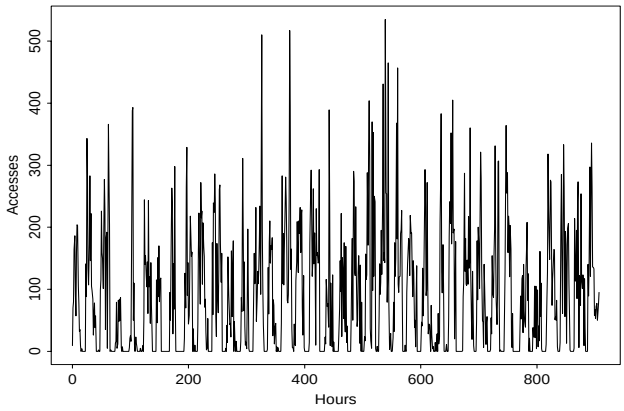
In this section we describe methods to identify what periodicities if any are present in the data. These tools can be used to test for periodic behavior on other data sets and with different time resolutions. We start by examining the time traces for the collected data.

4.4.1 Time traces

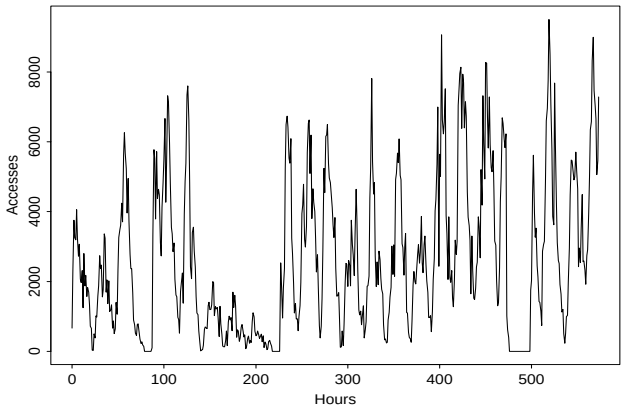
In Figure 4.1, we show the time traces for four samples of WWW access logs from our set of collected workloads. The graphs look different from each other and some of them suggest that the data is completely random. Figure 4.1(b) illustrates the shortest workload period: three weeks. The data in this workload seem to have a periodic signal. To explore this further we can zoom-in to each of the data sets.

In Figure 4.2 we show the first 200 hours from the previous samples for the CS department workload. By counting the main peaks in the figures, we note that there are 9 peaks; this was true for all of the tested workloads. This suggests that there is a periodic signal with a period approximately $T = 22.2$, which is very close to 24 hours or a day period. This value is roughly estimated by counting the number of peaks in the figure. We will see in section 5 how to extract the correct periods.

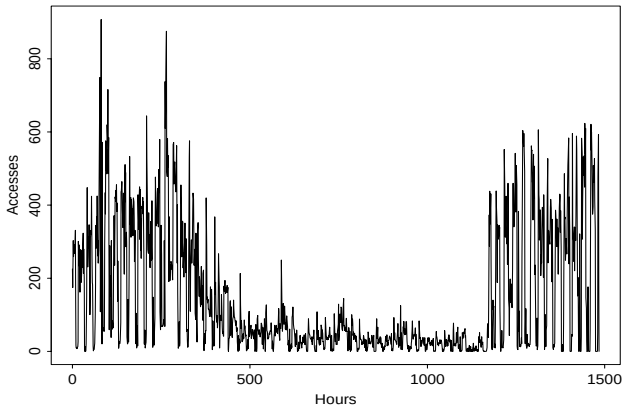
Although inspection of the time series visually can help identify strong periodic signals,



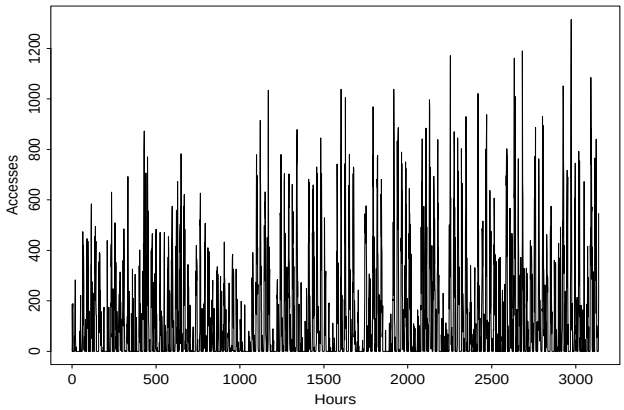
a- CS



b- Korea



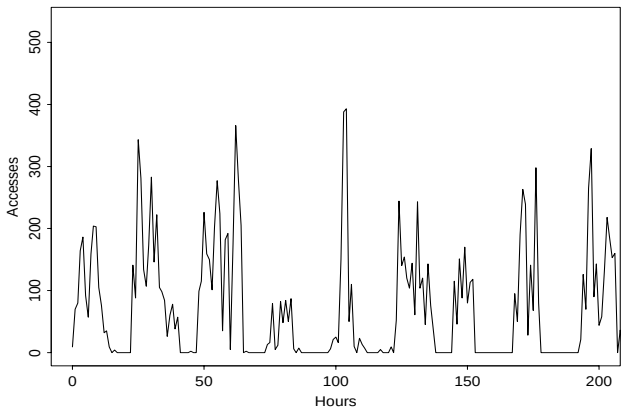
c- Library



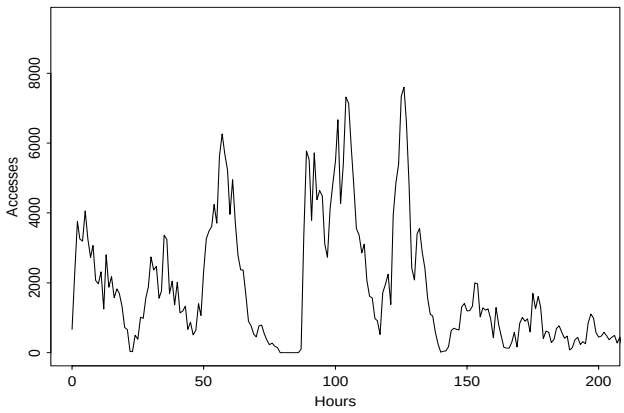
d- Engineering

Figure 4.1: Time traces for the collected data

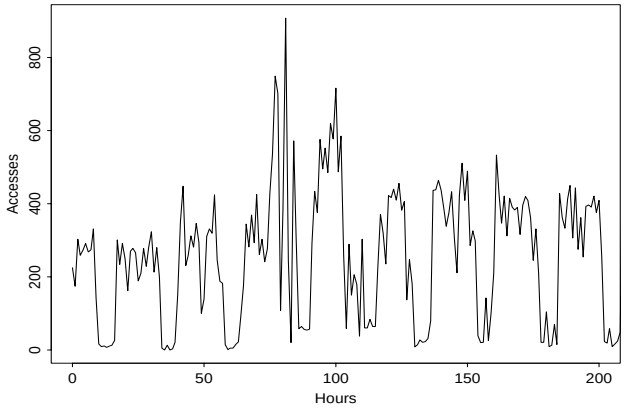
there might be some hidden periodicities that are not easily identified by such tests.



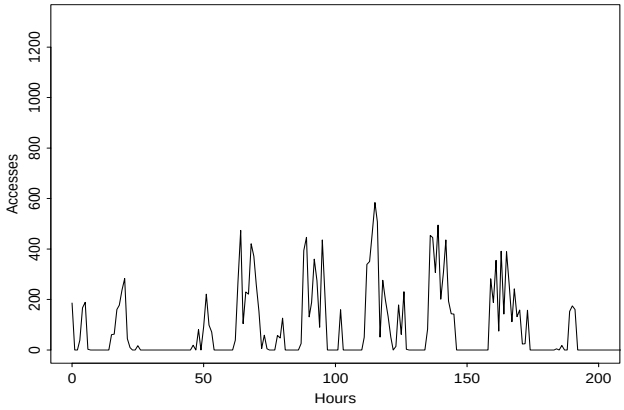
CS



b- Korea



c- Library



d- Engineering

Figure 4.2: First 200 hours of the CS time trace

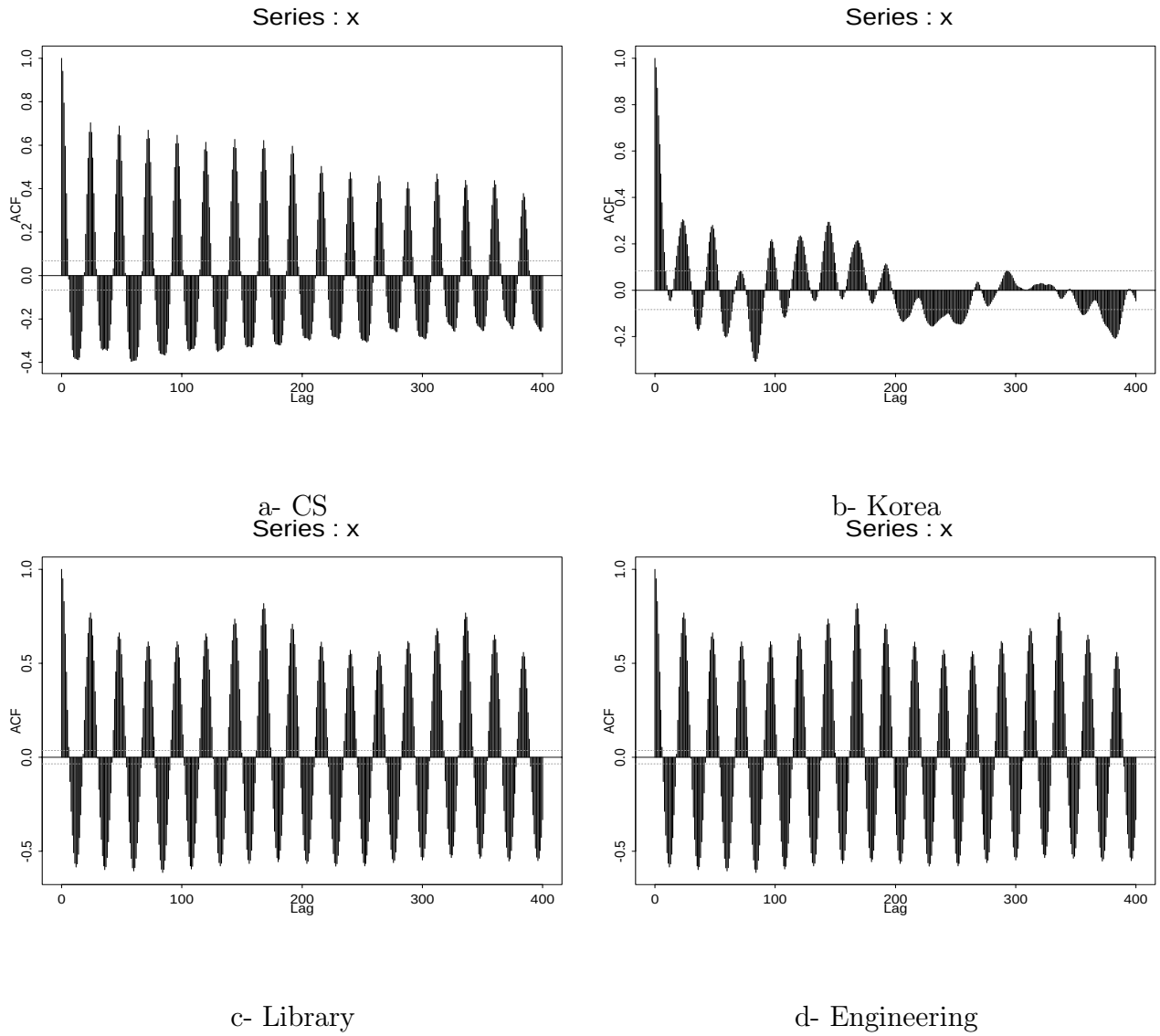
4.4.2 Auto-correlation function

The autocorrelation function $R_{xx}(\tau)$ is a measure of correlation between $x(t)$ and $x(t+\tau)$, where τ is a time delay [48, 49, 46]. For random signals, $R_{xx}(\tau)$ decays to zero as τ increases. However for a signal with a periodic component, $R_{xx}(\tau)$ does not decay to zero and is oscillatory. Using $R_{xx}(\tau)$ will give a less confusing representation of the data than examining the time trace, especially when there is a high-frequency component in the signal.

As we showed in the previous section, the data has a periodic component. However, we could not ascertain if more than one such component is present, and if so what are the frequencies involved.

By examining Figure 4.3 we clearly see that the number of accesses are strongly correlated and it is not clear whether the correlations decay to zero. Actually in some data (a, c, and d) the correlation gets weaker and then becomes stronger. By examining the figures carefully, we note the daily cycle mentioned previously. In addition a weekly cycle appears to be very strong in the data collected from the Engineering building and the Library but less strong in the case of the CS and Korea data. The reason for this is that the Korea data is for the shortest period and the CS data comes in second in terms of duration. When we have longer periods the weekly cycle appears stronger.

The autocorrelation function in Figure 4.3 was plotted after characterizing the mean using a moving average program. Plotting the ACF before taking out the mean will give a periodic correlation, however it will be weaker and may be on one side of the x axis especially if the data is not stationary.



a- CS b- Korea

c- Library d- Engineering

Figure 4.3: Auto-correlation functions

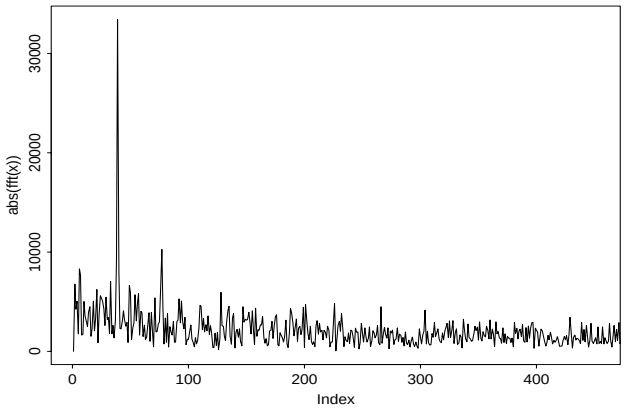
4.4.3 Fourier and spectral analysis

In Figure 4.4 we show the Fourier spectra for the four sets of data under examination. These plots were made after the data mean was extracted. (We show how to characterize the mean in the next section.) The plot shows half of the data points because the second half is a mirror image of the first half due to *aliasing* [50]. Again we clearly see the peaks that correspond to daily and weekly cycles. We now show how to find the frequencies associated with these peaks. Although the peaks in the Korea data are not isolated and clear as in the other data sets, they exist and can be considered a major component of the data. The reason that the peaks are not very well isolated is that the Korea data set was collected over three weeks only. A longer collection period will emphasize these peaks.

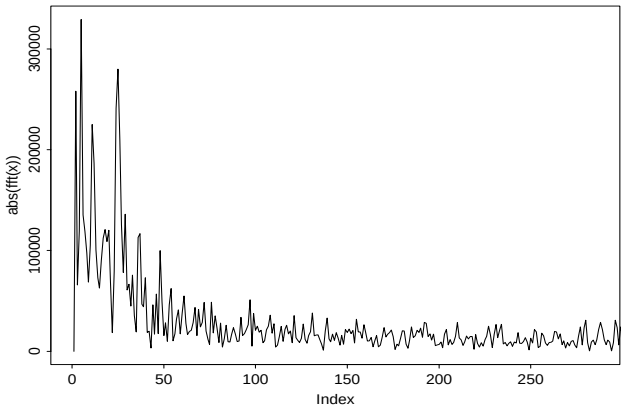
We can find the frequencies of these peaks by locating the index for each peak using the FFT command in Splus or any other mathematical software, dividing the index value by the number of points in the data set, and multiplying the result by 2π. To determine the period, we take the reciprocal of that number obtained.

Autoregressive spectrum

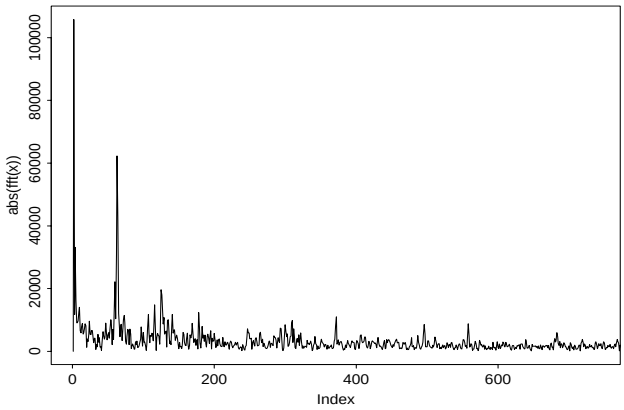
Another way to visualize periodicity in the data is to plot the autoregressive spectrum using the Yule-Walker algorithm. In Figure 4.5 we show the autoregressive spectra for the four data sets. Again the periodicity in the data is very clear in three of the data sets



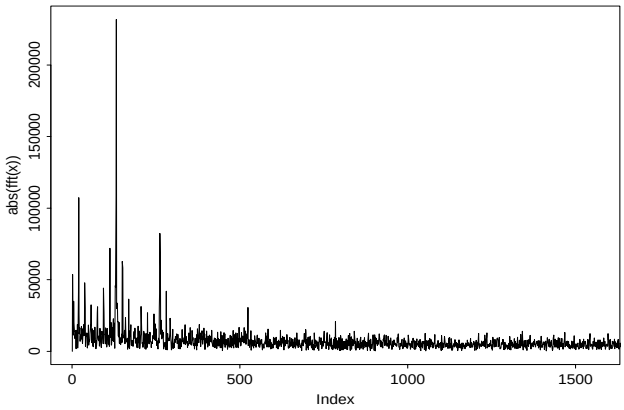
a- CS



b- Korea



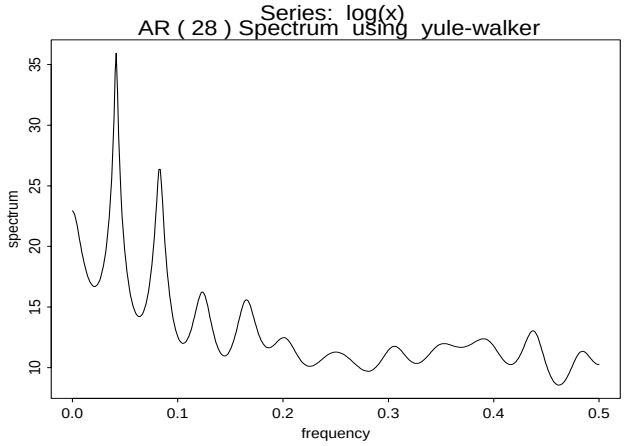
c- Library



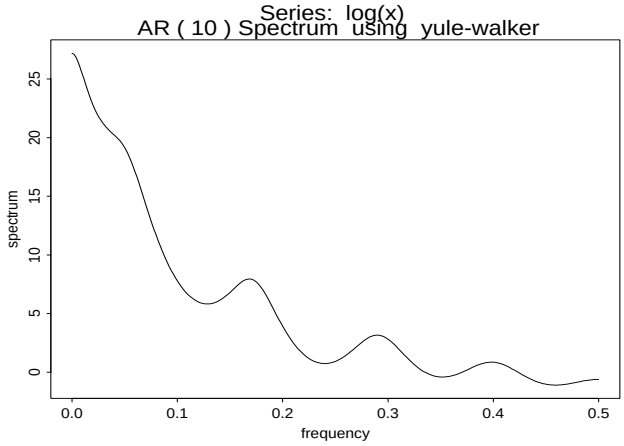
d- Engineering

Figure 4.4: The Fourier spectra

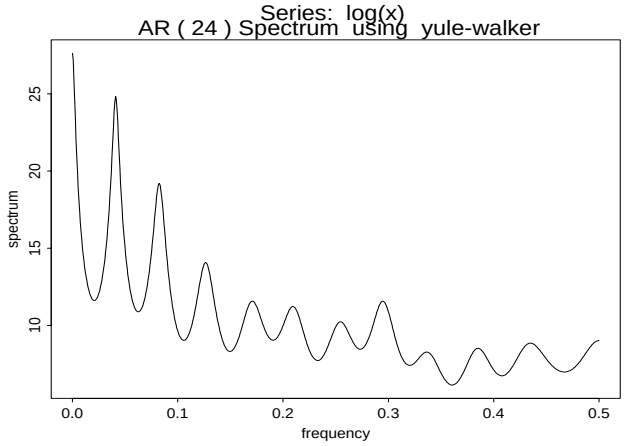
and it exists but is weaker in the case of the Korea workload. The peaks in these figures correspond to the major frequencies. From the figure we obtain the frequency $\omega = 2 * \pi/T$ and hence $T = 2 * \pi/\omega$.



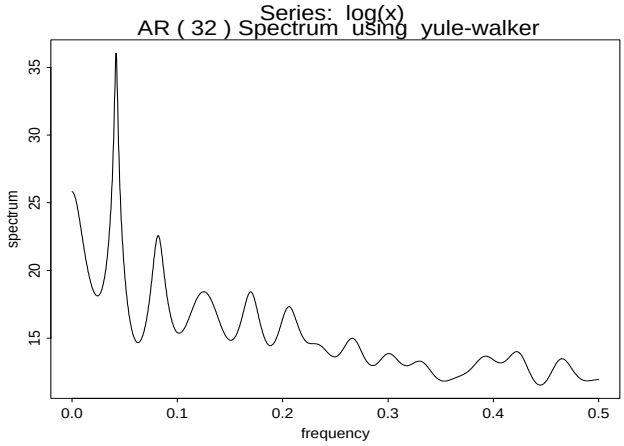
a- CS



b- Korea



c- Library



d- Engineering

Figure 4.5: The autoregressive spectrum

4.5 Modeling Algorithm

The tools described in the previous section can be used to test for the existence of periodic components in the data because identifying these components is important in the modeling process.

In Figure 4.6 we show the steps needed to develop a model for data that contains deterministic components and displays long-range dependency and oscillatory correlation. In some cases we might need to add tapering, or filtering, depending on the characteristics of the data under examination. The modeling steps are very easy and we will make the required software available for other interested researchers.

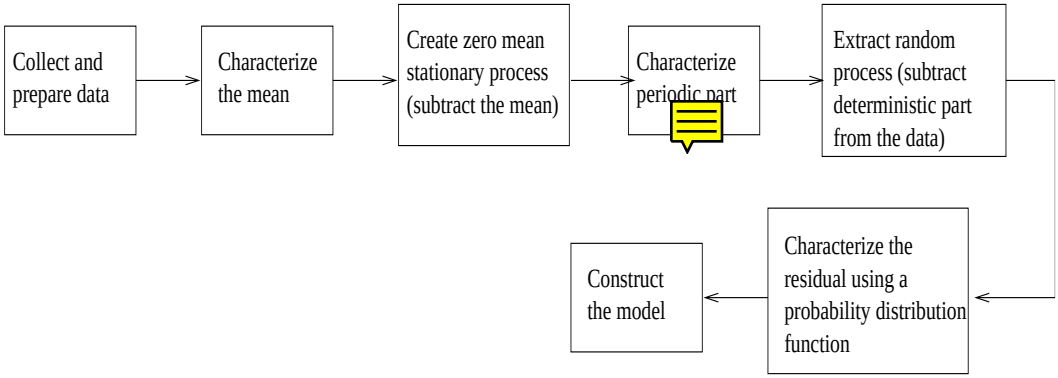


Figure 4.6: Steps to model the data

4.5.1 Characterizing the mean

The first step in the modeling process is to characterize the data mean. The mean might be a function of time in a linear or nonlinear fashion. In general, we can identify three main cases for the mean:

- constant mean,
- linear time-varying mean, and
- nonlinear time-varying mean.

For data with constant mean, we calculate the mean by summing all of the values and dividing the sum by the number of points. If the mean varies with time in a linear fashion, it can be characterized using linear regression. If the mean varies in a nonlinear fashion, then we can characterize it using a program that calculates the moving average over a specific window. For our data sets we used the second method. The main objective of this step is to create a zero-mean stationary process from the data set. Non-stationarity in our data can result from the change in the amplitude due to the change in number of users or can result from the main frequencies phase change. Our data showed constant phase difference for all major frequencies identified. That change was due to the change in the number of accesses. Hence, the effect of this change can be characterized by linear regression. The moving average should not be used here since it will remove some of the

targeted cyclic trends or changes in the original signal.

4.5.2 Periodic part

The data examined so far displays a strong and periodic correlation. Before building a complete model, we need to determine the amplitudes and periods of all of the periodic components. Finding all of the periods can be done by using Fourier analysis, as described in the previous section. Determining the amplitudes can be done using linear regression [50]. The details are given below.

Determining the frequencies

Frequencies, single or multiple, can be determined using Fourier spectra or autoregressive spectra. From the Fourier spectra we can use the index value for each peak to calculate the period T by dividing the number of data points by the index value. The frequency is easily obtained from the period. The autoregressive spectrum can give the frequency directly.

Determining the amplitudes

A data set that has a periodic signal and a constant mean can be expressed by the equation

$$x_t = \mu + R\cos(\omega t + \phi) + \epsilon$$

where μ is the mean, ω is the frequency of the signal, and ϵ is the residual or random part. This function is quadratic so we cannot use linear regression to find an R that minimizes ϵ. To solve this problem, we let $a = R\cos\phi$ and $b = -R\sin\phi$ [50]. Then, we can rewrite the previous equation as

$$x_t = \mu + a\cos\omega t + b\sin\omega t + \epsilon$$

After the previous transformation, the amplitude for each frequency can be obtained by linear regression over the data using the following generalized linear equation:

$$x(t) = \sum_i a_i \cos\omega_i t + b_i \sin\omega_i t$$

where the a_i and b_i are the required amplitudes (assuming that these amplitudes are stationary), and the ω_i are the frequencies associated with the peaks. This equation can be used to model data with multiple frequencies.

For the CS data, the peaks are at $\omega_1 = 1/24$, $\omega_2 = 1/12$, and $\omega_3 = 1/151$. For the day and half day cycles we can see that the numbers are exact for the corresponding periods. The week period is not exact (151) because of the length of the workload. In the longest workload the calculated weekly period from the FFT was 165.2 which is closer to the expected value of 168. This is due to the variation from week to week; when we have longer duration the effect of these variations is minimized. The first period corresponds to the daily cycle, the second corresponds to the periods of no activity at night, and the third corresponds to a cycle every 6.3 days which is approximately a week. By using linear

regression, we found the following expression to represent the periodic part of the data:

$$x_{per}(t) = 58.98\sin((2\pi t)/24) - 51.61\cos((2\pi t)/24) + 17.41\sin((2\pi t)/12) - 21.18\cos((2\pi t)/12) - 4.51\sin((2\pi t)/151) - 16.75\cos((2\pi t)/151) \quad (4.1)$$

In Figure 4.7 we compare the Fourier spectra for the generated data using equation 1 and the collected data. The figure indicates that the suggested model captures the main frequencies and their amplitudes.

4.5.3 Modeling the residual

The main purpose of characterizing the mean and the periodic part is to minimize the residual, which is the non-deterministic or random part of the collected data. Up to this point, we can stop and describe the data in terms of its deterministic component. The significance of characterizing the deterministic components comes from their implications. First, this characterization allows predicting the users' behavior in the future. Second, it provides a nice explanation for the repetitive users' behavior and the time dependent component of the Web data.

Before we perform the statistical characterization of the residual, we test it for the existence of major periods. This test will help identify if the residual still has any periodic components. We generate a data set using the suggested model in equation (1) and sub-

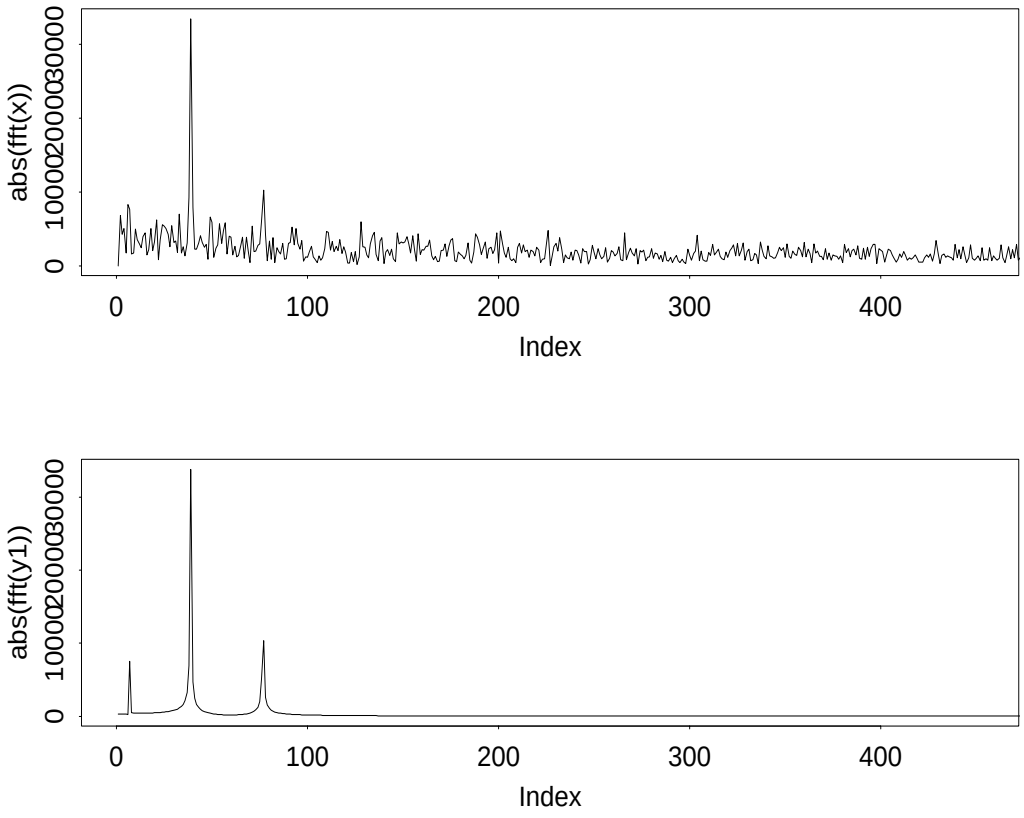


Figure 4.7: The FFT's of the original data and the generated data set using the periodic model

tract the generated data from the original collected data, yielding the residual ϵ. In Figure 4.8 we show the residual and the associated Fourier spectrum. It appears that the main frequencies have disappeared and that the residual is random.

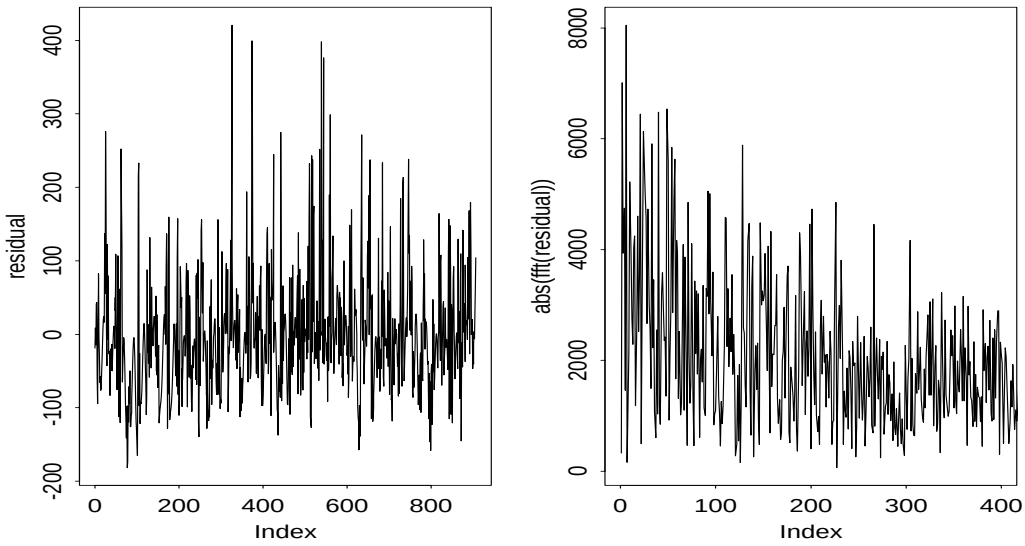


Figure 4.8: The time trace and the FFT of the residual

Autocorrelation and the residual part

One other test that can serve as a validation step and can explain the sources of the strong and periodic correlation in the Web traffic is the autocorrelation function. In Figure 4.9 we show the autocorrelation function of the residual. The horizontal band about the zero line represents the approximate 95% confidence limits for the null hypothesis that the autocorrelation has zero value. Comparing this figure with Figure 4.3(a), which contains the autocorrelation function for the data including the periodic component, we find that extracting the periodic part removed the periodic correlation in the data. This means that the main source for the correlation is the periodic behavior of the Web users.

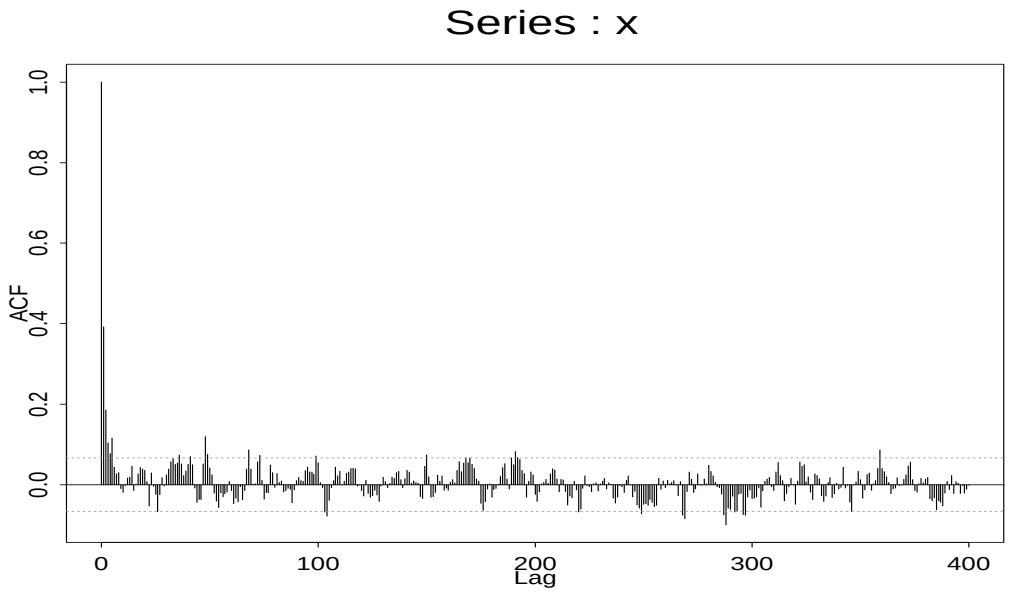


Figure 4.9: The autocorrelation function for the residual

In the rest of this section, we attempt to model the third component of the data, namely the random part using techniques described in [51].

Fitting the residual

We use quantile summaries and histogram plots to study the distribution for the residual. In Figure 4.10 we show the histogram and the normal quantile-quantile plots for the residual of the data under examination. The histogram shows that the distribution is asymmetric and the right-hand side of the distribution contains more values than the left-hand side. This implies that the distribution might be a long-tail distribution. The quantile-quantile

plot [52] confirms that the distribution is a long-tail distribution from the right-hand side.

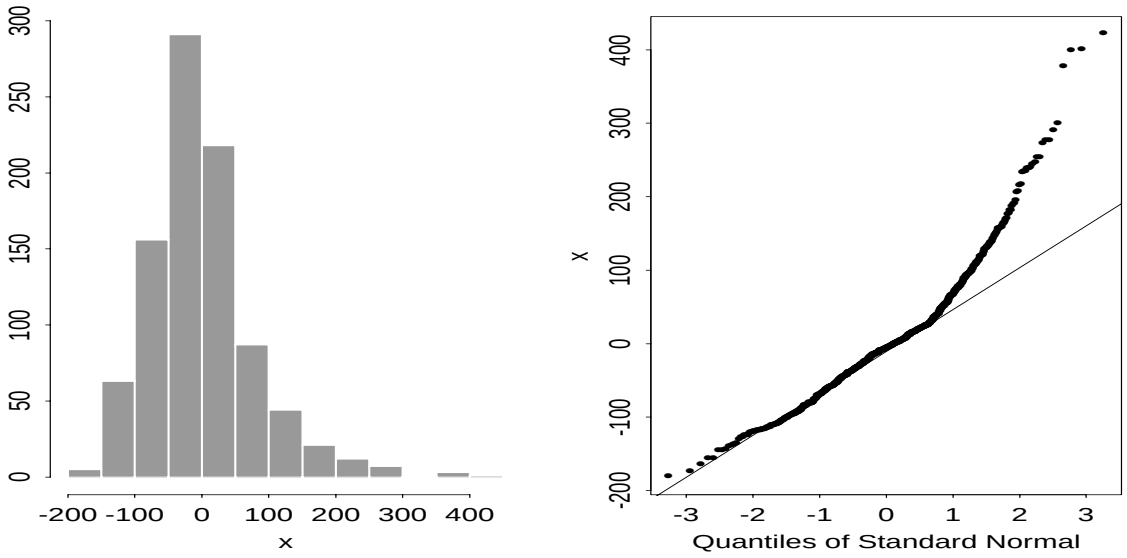


Figure 4.10: Histogram and Q-Q plot of the residual of the CS data

We use a Weibull distribution to model the residual. To generate a synthetic set of data, we used the UnifitII software [53]. The parameters for the distribution were estimated from the data. We followed the modeling process used in the previous chapter to validate the the model.

In Figure 4.11 we show the original data, its Fourier spectrum, the newly generated or synthetic data, and its Fourier spectrum. The main three frequencies and the general behavior of the data are captured in the model. The amplitudes of the negative peaks in the model are clearly higher than the ones in the original data. The reason for this is that

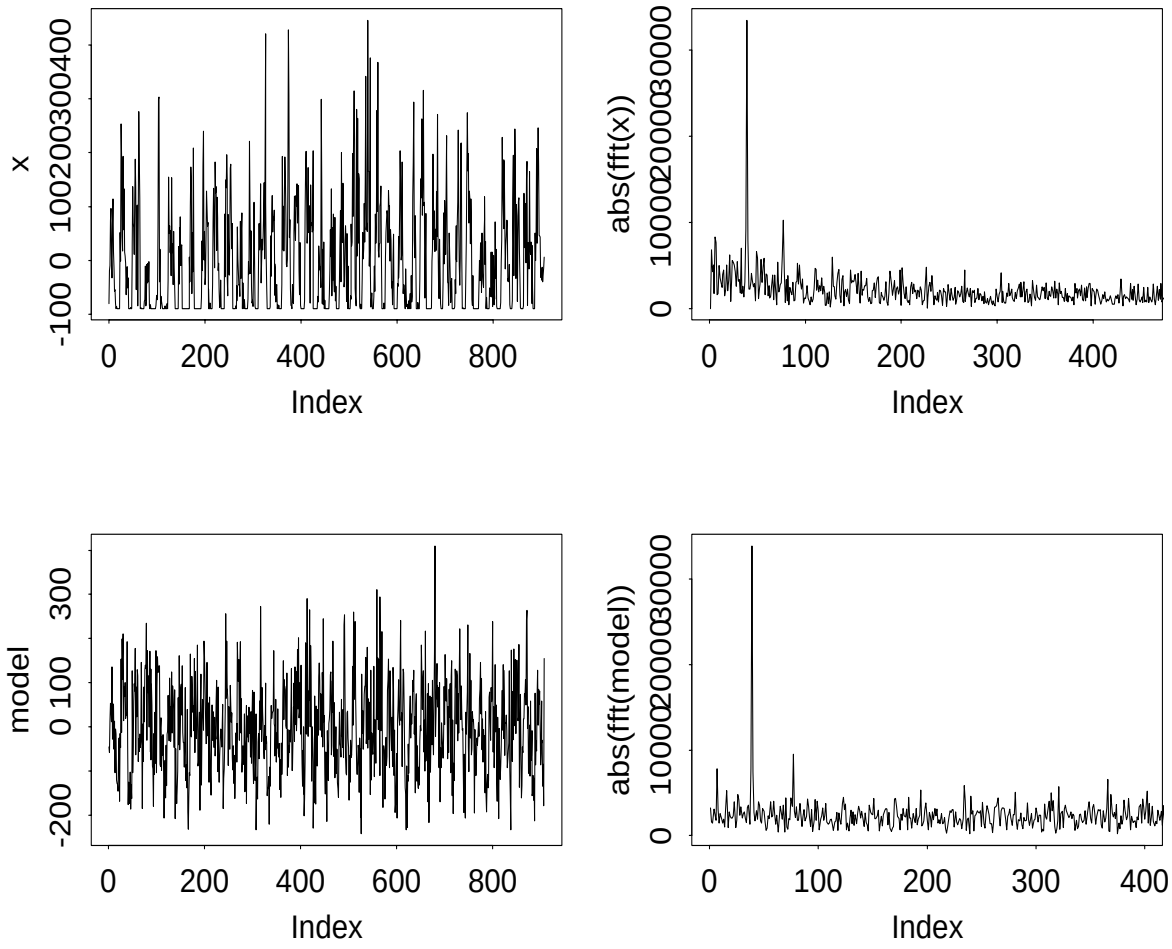


Figure 4.11: The original data, its FFT, and the synthetic data and its FFT

the periodic part is symmetric around the zero value while the original collected data does not have the symmetry property.

The final step in constructing the model is to get rid of the excess values from the negative peaks. A crude solution for this problem is to write a filter that will replace all negative values with zeroes. We apply this filter after we add the mean to the generated data.

4.5.4 Model validation

The time series of the synthesized data after applying the filter looks very similar to the time series of the original data. However, we will validate the model using other techniques such as the Q-Q plot rather than eyeballing the time series.

In Figure 4.12 we show the Q-Q plot for the original data versus the modeled data. The relation is almost linear which means that the two sets have very similar distributions.

In Figure 4.13 we show the autocorrelation function for the modeled data. Clearly the model displays a periodic autocorrelation and captures the long-range dependent behavior that was observed in Figure 4.3.

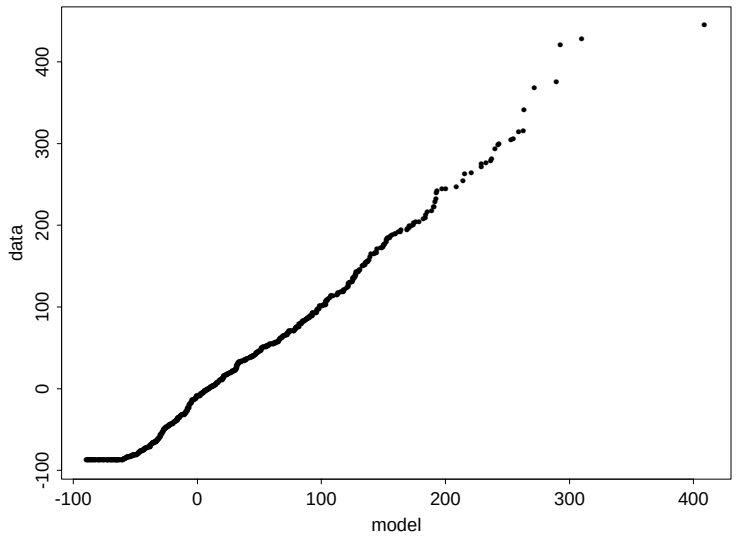


Figure 4.12: Q-Q plot of the model vs. original data

4.5.5 Self-similarity in the generated model

To measure if the synthesized data displays a self-similar behavior we use the R/S statistical test to obtain a value for the Hurst Parameter (H). We run the test on both the original and the synthesized data. The H value for both data sets was 0.6 and this means that they have very close self-similar characteristics. Thus by focusing on the network traffic characteristics we created a synthetic traffic that is self-similar and long-range dependent.

We have tested the modeling algorithm on the data sets that we have collected and we came up with the following general equation that can capture the changing mean, the periodic

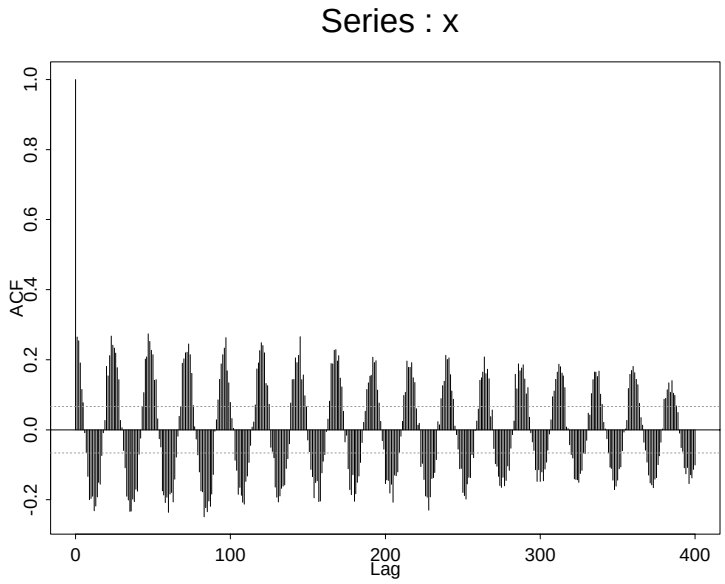


Figure 4.13: ACF for the synthesized data

part, and the random part:

$$X(t) = \mu(t) + \sum_i (a_i \cos \omega_i t + b_i \sin \omega_i t) + \epsilon$$

4.6 Measuring Periodicity in the Data

To find out the strength of our periodic signal with respect to the random part, we use a measure that is called *signal to noise ratio* (SNR) [50], defined as,

$$SNR = \frac{mean\ square\ value\ (MSV)\ of\ the\ signal}{mean\ square\ value\ (MSV)\ of\ the\ random\ part}$$

For the periodic signal

$$MSV = \frac{\sum(y_s - \overline{y_s})^2}{n-2}$$

Because $\overline{y_s}$ is zero (since we extracted the mean), the previous equation reduces to

$$MSV = \frac{\sum(y_s)^2}{n-2}$$

With a similar argument we can show that for the random signal

$$MSV = \frac{\sum(y_r)^2}{n-2}$$

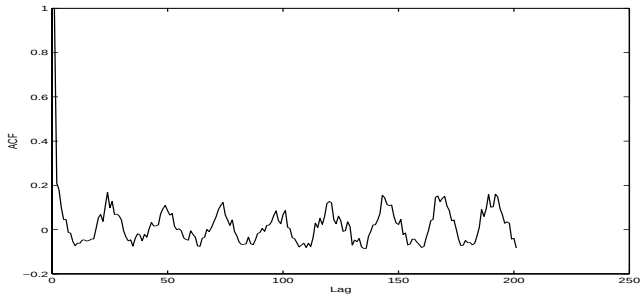
Finally we can write $SNR = \frac{\frac{\sum(y_s)^2}{n-2}}{\frac{\sum(y_r)^2}{n-2}}$, which reduces to

$$SNR = \frac{\sum(y_s)^2}{\sum(y_r)^2}$$

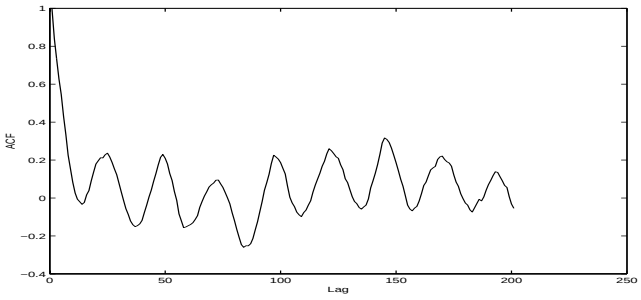
For the tested data we found that SNR=0.5639. This is a significant ratio, since the periodic signal is approximately 44% of the full signal.

4.7 Bytes rate

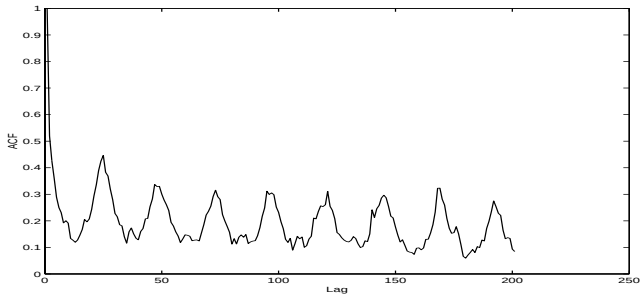
In this section we show that the bytes rate in the workloads listed in Table 4.3 in addition to Boston university workload have a periodic long range dependent behavior. In addition we will show in Appendix A that the ethernet data from Bell Core have a periodic long range



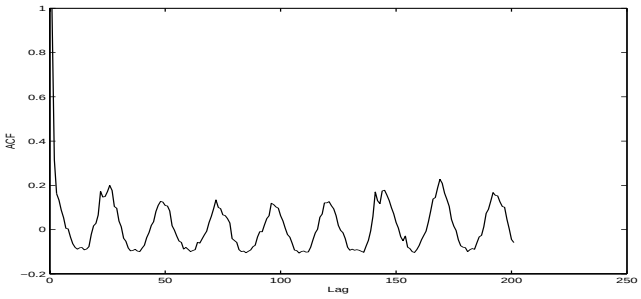
a- CS



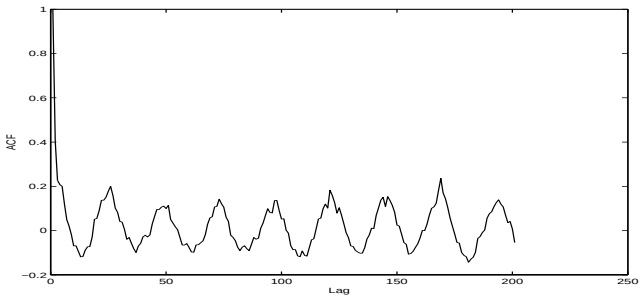
b- Korea



c-Library



d-Engineering



e- Boston

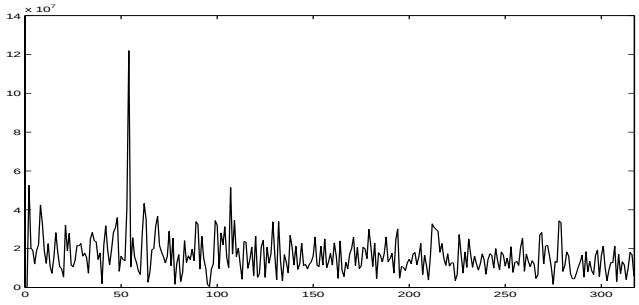
Figure 4.14: Auto-correlation functions for bytes rate

dependent behavior too. In Figure 4.14 we show the ACF for the four previous workloads in addition to the Boston university workload. The behavior is similar to the behavior in the arrival rate time series and we can use the suggested modeling algorithm to model this traffic. The amplitude, however, in Figure 4.14 is smaller than the one in the arrival rate; which means that the deterministic component is weaker than the one in the request arrival rate, Figure 4.3.

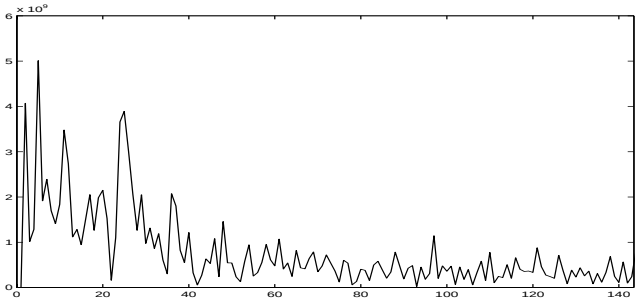
In Figure 4.15 we show the FFT for the bytes rate for the same workloads. We can see the same behavior noticed in Figure 4.4.

4.8 Implications and Future Work

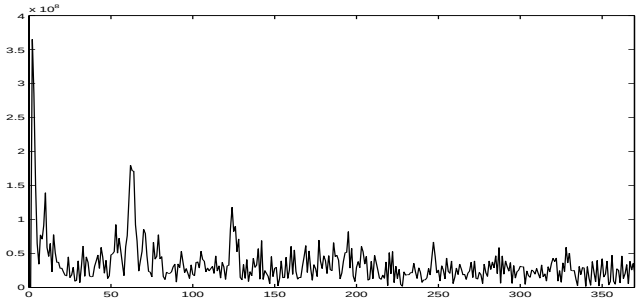
Accesses to Web proxies are highly correlated from day to day and from one week to another. A key source for this correlation is the daily and weekly schedule of Web users. Users who share the same time zone should have similar schedules and hence part of their accesses should reflect it. This result is important for commercial proxies since they will act similar to a local *broadcasting* station or local cable company that distributes information and other kinds of media. Proxies will be installed to support a group of users in a certain locality. Peak and minimum usage hours can be identified for upgrades, price commercials, and other time dependent jobs.



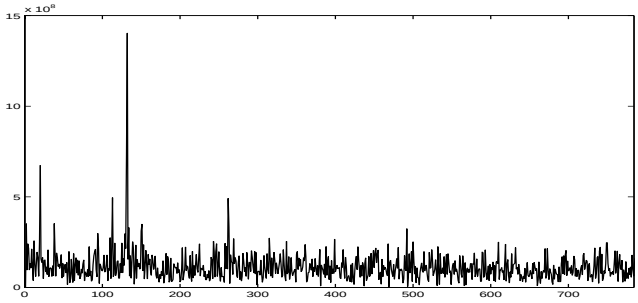
a- CS



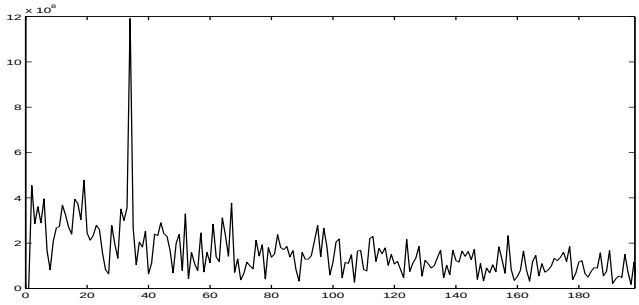
b- Korea



c-Library



d-Engineering



e- Boston

Figure 4.15: FFT for bytes rate

The fact that Web traffic has a significant deterministic and cyclic component is very important for internet service providers. This fact can be used for predicting peak hours and hence to schedule software upgrades or maintenance. It also can help in the efforts to smooth and distribute the traffic to the service provider proxy by advertising the low activity times and having lower pricing rates then to encourage and attract more users. Internet network protocol designers can use this information to support pre-fetching and filtering of data based on the daily and hourly level of activity and locality of reference of a group of clients connected to a certain proxy.

This result is also important when characterizing accesses to busy Web servers. The traffic to such servers can be split based on the time zones of the clients accessing the server. Then the characteristics of sub-groups can be modeled using the suggested approach.

The discovered periodic correlation indicates that queuing and statistical models are not the appropriate techniques to model such systems. Simulations need more accurate models that represent the characteristics of the Web traffic. We introduce a new modeling approach that captures the real characteristics of the Web traffic by using a combination of Fourier analysis and the traditional techniques of statistical analysis. The new approach characterizes the random and the deterministic parts of the data. The generated data displays long-range dependent and self-similar characteristics comparable to the ones observed in the original data. We demonstrate the modeling approach with an example and we validate each step

by comparing the generated data to the original.

We also have shown the same periodic behavior exists for bytes rate and hence we the suggested modeling algorithm works for it too.

Chapter 5

Digital Library for Computer Science Courses: Lessons Learned from Tracking Users' Accesses

5.1 Introduction

There are many good reasons to use the Web for deployment of educational material, including:

- The material is easily accessible from any place at any time, for example students can access it in the classroom, in the lab, and from home.
- Updating material is easier and students can access changes instantaneously.
- Online interactive demonstrations, exercises, quizzes, and other course ware resources can motivate and involve learners.
- Student/teacher and student/student communication is enhanced  g the new paradigm by:
  - posting the teaching assistant and instructor's email address on the course pages,
  - using discussion and conferencing tools, and
  - posting the teaching assistant and instructor's information such as office hours.
- The posted material can include links to other sources and the students can explore the new links easily.

It is important to know if the posted material is used by students and other users. It is hard, however, to judge the effect of this usage on the learning process. Each WWW server can be configured to log all accesses in a log file. Log files can give us valuable information about what is popular, and what is not. Unfortunately while such logs indicate "what" was accessed, "when", "where from", and sometimes "who" accessed it, they do not contain information on "why" it was accessed or "how much" benefit resulted. Hence, any question

that is related to the last two items cannot be answered easily. To answer such questions we have to look at the log files, come up with a questionare, interview users, and analyze the results of the interview guided by the users access history.

It is useful here to discuss two methods that are used to evaluate multimedia educational material. The two methods are *formative* and *summative* evaluation [54]. During the formative process, data is collected from users who are using the current system or material. Small numbers of students, sometimes working in pairs, are monitored and data is collected using surveys, video recordings, and/or logs. The results obtained from the analysis are given to the course ware designers to adopt recommendations or modify the presentation based on the conclusions obtained from analyzing the data. For Web course ware designers the main objective of this evaluation is to make sure that students will get the full benefit from deploying the course material over the Web.

In the summative evaluation, the success of the program is tested [54]. The process of summative evaluation is done over a long period of time and it should include analysis of all the material presented. The results should give us an idea about the long term benefits for education that will result from using the new technology. This phase can include controlled experiments with human subjects to measure how much they learn relative to the traditional environment.

In this chapter we analyze accesses to the Educational Infrastructure (EI) server as an example of a course ware digital library server. We compare the general statistical results with the results published in the literature. Especially, we try to assess the applicable invariants identified for Web servers in [28]. We define metrics derived from examining the log files to assess the value of using the EI digital library server. We then show how to apply these metrics to make useful conclusions about the effects of digital library servers.

5.2 Quantitative Analysis and Visualization Regarding EI

The Educational Infrastructure (EI) server is a DEC Alpha machine that hosts the online courses for the Computer Science Department. Key concepts of the EI project are to improve CS education by increasing interactivity and use of digital library [55, 11]. Currently the server hosts the home pages and class material for over 40 Computer Science courses. The courses cover undergraduate and graduate classes and different areas of the Computer Science field. The server has been used by faculty and students for three years. In addition, the server is being accessed by users from the Virginia Tech domain and by others from around the world.

Table 5.1: Number of accesses and bytes transfered from EI during the last three years

Year	Accesses	Bytes (MB)
1995	1,015,185	6,898
1996	2,247,780	28,225
1997	5,027,607	59,188

5.2.1 Accesses to the EI server

In Figure 5.1 we show the number of weekly accesses to EI for three years: 1995, 1996, and 1997. Weekly accesses to EI is almost doubling every year and the growth rate is very high. In Table 5.2.1 we show the number of accesses and number of bytes transfered from EI. The table confirms that the numbers are at least doubling every year. This growth is attributed to several effects. First, the number of courses has increased from a few courses to about 40 courses in the fall of 1997. Second, the increase in the number of students in the department. Third, faculty and new students are becoming more educated about the WWW and hence they do not mind using it as an educational tool. We noticed that accesses from outside the school also are increasing; the reason for this is that more people are using the Web and as a result more people know about our server.

In Figure 5.2 we compare accesses from remote clients to accesses from local clients. As can be seen the rate of access in both cases is growing. Accesses from local clients, however,

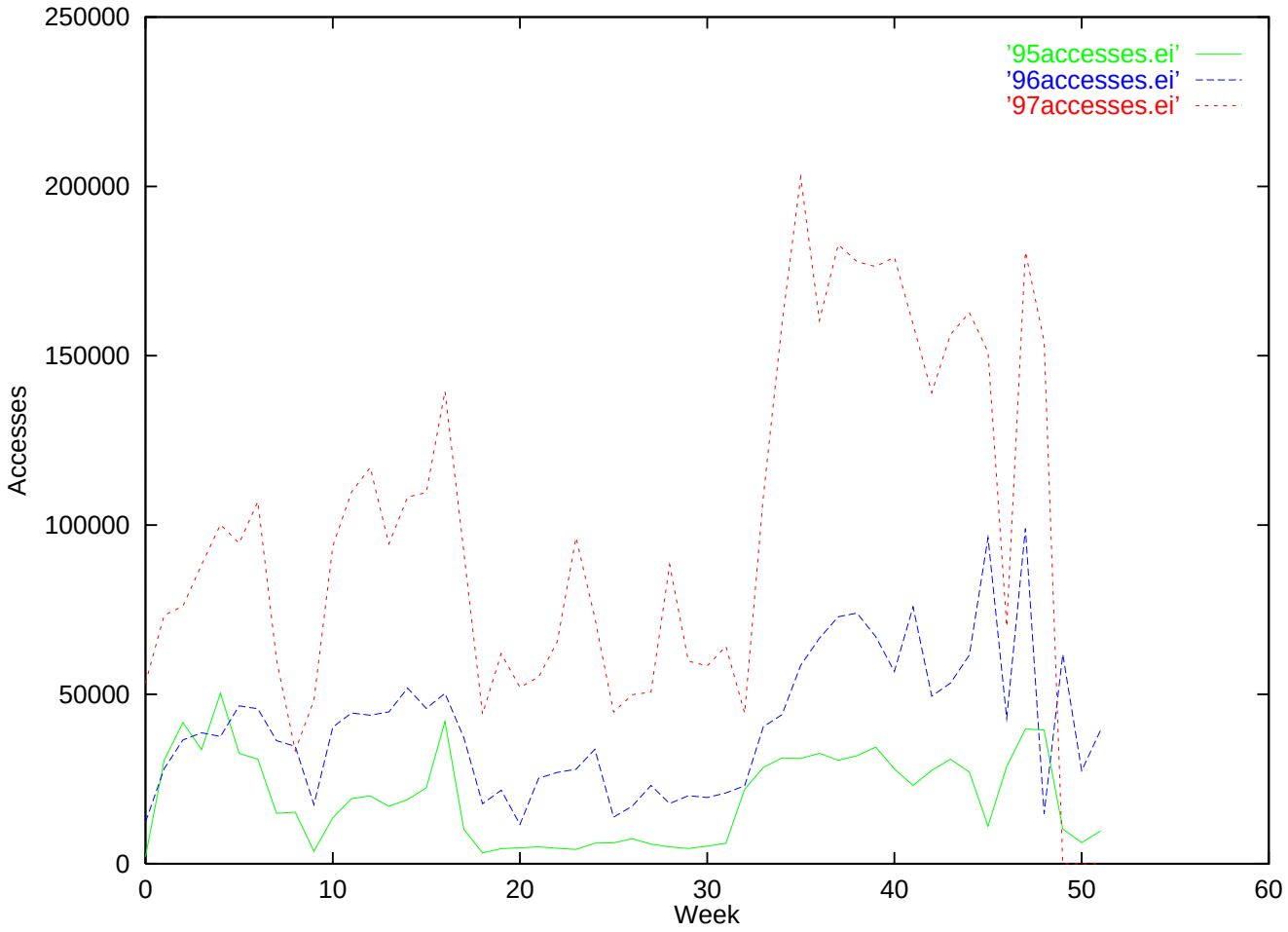


Figure 5.1: Weekly accesses to ei server

tend to be bursty compared to remote accesses. Burstiness in the local traffic is due to the school and students' schedules. For example, the graph between weeks 15 and 30 and weeks 70 and 85 are the periods of the summer when the accesses are low for a long time.

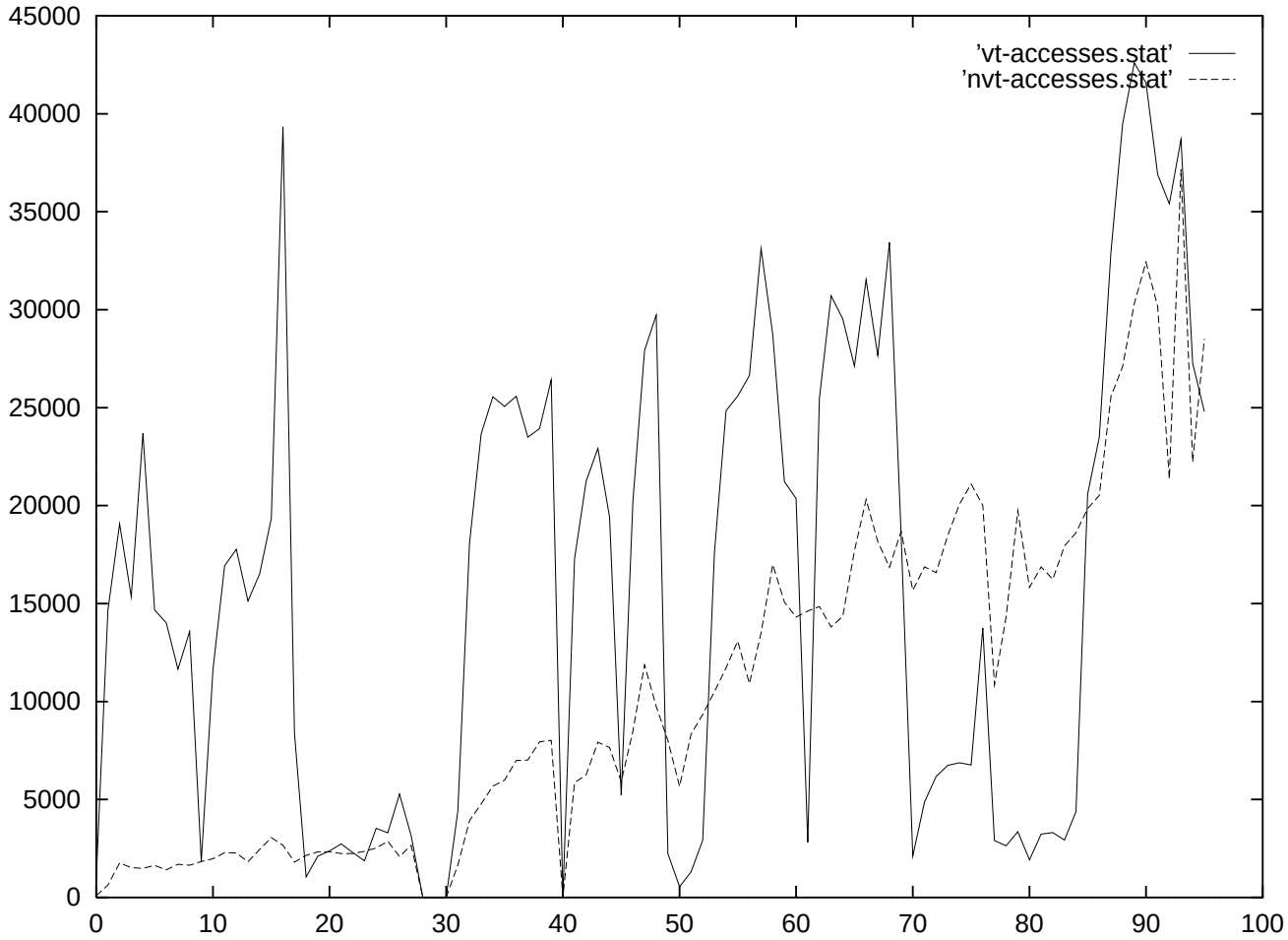


Figure 5.2: Accesses to EI server, remote clients vs local clients

Table 5.2: File type distribution on the EI server, remote clients vs. local clients

	Local clients		Remote clients	
File type	%Refs	%Bytes	%Refs	%Bytes
Graphics	45.95	40.94	44.93	57.97
Text	50.80	48.30	52.80	40.50
Script or Map	0.11	0.01	0.03	0.00
Audio	0.04	0.03	0.01	0.04
Video	0.04	6.68	0.01	0.50
Other	3.07	3.72	2.11	0.90

5.2.2 File Types

In Table 5.2 we show the percentages of references and number of bytes for different file types accessed from the EI server. The left two columns in the table show accesses from local clients. Local clients are clients in the Virginia Tech domain; this includes labs, offices, dorms, and apartments with ethernet or modem connections. With respect to accessed file types the behavior between the two groups is not really different. In both cases graphics and text (which includes HTML) are responsible for 95 percent of the references to the EI server. This is consistent with the identified invariant for accesses to proxies in [23] and servers in [28]. As expected, the numbers of references to audio and video files are very small. This is due to the low percentage of audio and video files in the server collection,

and because network performance does not encourage accessing this type of data.

We notice that remote clients don't access video files as much as local clients. We attribute this phenomena to network speed. If the speed was fast enough to view video files over the network, users will be encouraged to access them. The problem is less severe in the case of local clients.

Although the number of accesses to files of type video is negligible as can be seen in Table 5.2, still the number of bytes transferred as a result is high. Video and audio files should be measured in terms of both number of files accessed and bytes transfered since these files are huge in size. A file count will not give an accurate idea of the server and network load required for multimedia.

5.2.3 File Size

In Table 5.3 we show the file size statistics for each file type for local clients. In Table 5.4 we show the same information for remote clients. In both tables the mean and the median are different. The median is much smaller than the mean and the variance is large especially for the two main file types, graphics and text, which skews the file size distribution. The two groups also display differences especially in the mean and median of files accessed. The remote clients on average they access larger files.

Table 5.3: File size statistics for local clients

File type	Mean	Median	Var
Graphics	9353	1952	2.6312e+09
Text	10474	2287	4.1320e+09
CGI/Map	1758	960	1.8022e+07
Audio	97854	56003	4.4454e+09
Video	2.21e+06	2660335	4.1216e+12
Other	20111	2464	2.3169e+10

To better describe the data it is essential to consider the median and the mean since the file size distribution for each type is skewed. Measuring and characterizing the file sizes and distributions should continue since the distribution might change over time in accord with the change of technology. There is a study, however, that suggests that the number of dynamically generated documents does not increase in time [56] which is an accurate statement about our server, however, it was not true for the proxy traffic case which we demonstrated in chapter 3. Nevertheless, a change in technology might increase or decrease the popularity of certain file types.

Table 5.4: File size statistics for remote clients

File type	Mean	Median	Var
Graphics	19590	4622	1.3867e+09
Text	11976	3744	1.3659e+09
CGI/Map	1748	704	1.6814e+07
Audio	63832	49152	4.6792e+09
Video	1.3214e+06	208896	3.3113e+12
Other	9094	2304	3.5523e+09

5.2.4 Size distribution

In Figures 5.3 and 5.4 we show graphs for the cumulative distribution function of the file sizes accessed by local and remote clients. Statistical tests that we performed confirmed that a lognormal or Weibull distribution can be used to model file sizes for our digital library server. This is consistent with identified invariants for other servers [28].

5.2.5 Inter-arrival time

The inter-arrival time of accesses to the EI server was calculated from the log files. Also the cumulative distribution function for the inter-arrival time was calculated. Figure 5.5 shows a histogram plot for a sample of the inter-arrival time. We can see that most of the

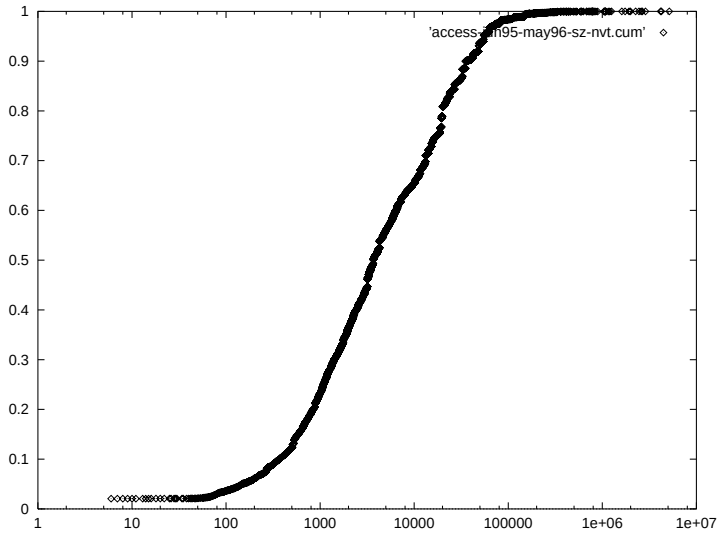


Figure 5.3: File size distribution for the EI server, remote clients

accesses are 50 seconds apart or less. The data contains lots of zeros which means that there were lots of consecutive accesses. This implies that the traffic is bursty and whenever there is an access then there is a great possibility that other accesses will follow. This is mainly attributed to the structure of Web documents, since a document normally contains several impeded images or icons that are accessed individually. This burstiness can be another source for Web traffic self-similarity. This implies that a change in the protocol design, such as using persistent connections, might decrease burstiness and hence decrease self-similarity. A change in the document structure, such as having fewer impeded images, might cause the same effect.

In Figure 5.6 we show the cumulative distribution function for inter-arrival time. The

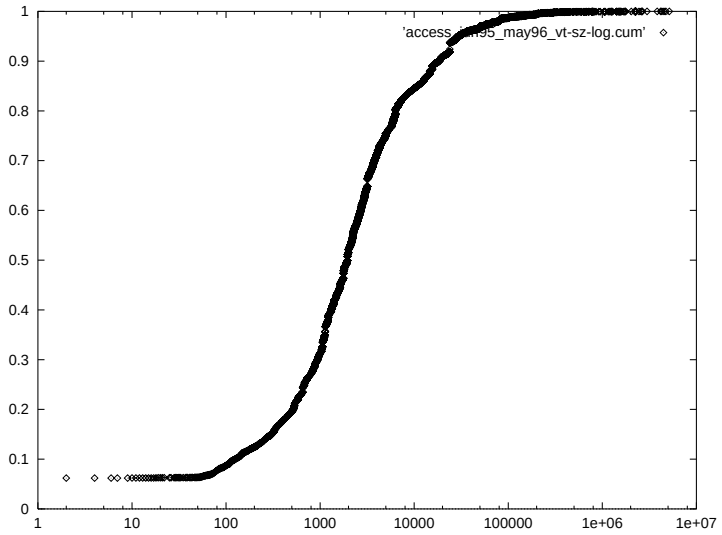


Figure 5.4: File size distribution for the EI server, local clients

graph confirms that the majority of the accesses appear within a short period of time. By examining the data we found out that [illegible]% of accesses appear 11.55 minutes or less apart.

5.3 Implications from Analyzing Students' and Teachers' Accesses

Studies of this type are very important for the following reasons:

- Educators, administrators, and sponsors want to know if and how digital libraries help with learning.

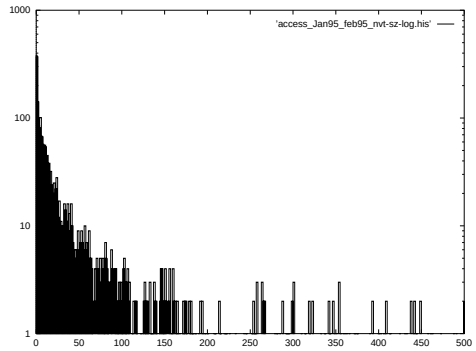


Figure 5.5: Inter-arrival time histogram shows that consecutive accesses are dominant

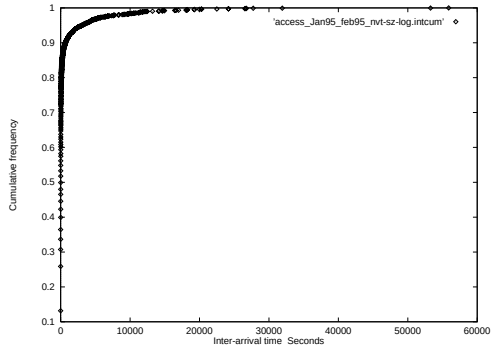


Figure 5.6: The Cumulative Distribution Function for Inter-arrival time

- Educators want tools to help them understand students' reactions to the various aspects of digital libraries, that can be used in conjunction with surveys and focus groups.

- Digital library researchers want validated methods to determine if digital libraries have beneficial effects, both as part of formative and summative evaluations.

As was discussed earlier with education-oriented digital libraries accessed over the WWW by a large numbers of users, it is very difficult to evaluate their effectiveness or to determine how to improve them over time.

Extensive logging of user accesses of the digital library of CS information and course ware on ei.cs.vt.edu started in January 1995. In addition, there has been logging of network traffic from various labs and buildings to understand how the use of that server compares with other WWW activities of students.

In Table 5.5 we show the results of the regression done to find out what significant factors will help in predicting number of accesses and transferred bytes from EI. To perform this regression we first extracted the accesses by faculty to the EI server, we also extracted accesses to each class pages. The level of how "paperless" a course is was calculated from the number of accesses that the faculty made to EI and faculty involvement in the project. The table shows that paperlessness, number of students attending the class, and the faculty are significant parameters in predicting accesses to EI. Given which courses lead to the most accesses, we sought to understand more about such accesses and their purpose, and how they relate to the course web.

Table 5.5: Regression showing relationship of number of faculty, students, and paperlessness to amount of use of ei.cs.vt.edu

	Coefficients	t Stat
Faculty	429.60	2.28
Number of Students	130.85	3.25
Paperless	11,377.69	8.60

5.4 Accesses to courses

Accesses to a course ware server will be affected greatly by the school schedule and offering of classes. As it was seen in Figure 5.2 accesses from local clients follow school calendar. In Table 5.6 we show number of accesses to the most popular course pages on our server. The table is divided according to the school calendar. We did that in order to see if we can find clues that the students access the course pages when it is offered.

From the table we notice that accesses to courses increase during the semester when it is offered. We can make a conclusion that the students are considering the server as a source for their readings. In a class such cs5604 during the fall of 95 there around around 25 students with 35,430 accesses; assuming that the bulk of the accesses were made by the students then during that semester each student averaged 1417 accesses per semester. The class material include links to other sources of information and those accesses might

have lead to other accesses that cannot be recorded in the server log file. In Fall of 96, however, accesses to the course pages went down sharply, compared to Fall of 95, this kind of behavior should be looked at to know the reasons for such change. This can happen because of number of students in the class, making alternative material sources available for students, or that group of students was not interested in using the Web. A survey about how do students feel about using the class material over the WWW will help in clarifying some of these numbers.

Another interesting thing to notice in the table is that some classes continue to be popular and they get lots of hits even during semesters when they are not offered. For example, cs4624 got 19,225 during the Summer of 96. This could mean that students are making use of the material after they finish the class, or maybe the class material is used by students as reference material. Accesses to the same class during the break are relatively higher than the other classes. By examining the log files we found out that these accesses were made mainly by the instructor during the authoring process in preparation for the spring semester.

Accesses to certain courses on EI generate lots of bytes. For example, 1,817 MB were transferred from EI in the Spring of 96 as a result of the accesses made to cs1704. The number of bytes transferred per access to each class differs between classes. The difference mainly comes from the types and sizes of files that the course pages contain.

Some course pages are more popular outside our domain for example History and sometimes the WWWbtb.

The NA in the tables mean that the course was not online yet.

5.5 Conclusions

The computer science digital library server characteristics not different from other server characteristics noticed in the literature. The server is utilized by the Computer Science students and faculty.

We found that paperlessness, number of students attending the class, and the faculty are significant parameters in predicting accesses to EI. Faculty access the server a lot before the start of the semester during the authoring and course preparation phase. Accesses after that are mainly to add, fix, and post information. The most popular pages accessed in the courses are the announcements and the grades (if available) page. The syllabus page is another popular page.This implies that the server is mainly used to enhance student/teacher communication.

Accesses to the server follow the school calendar closely. It also reflects the faculty and students schedule. For example beginning of August and January are active periods for faculty. Before the midterms and before finals accesses from students are very high. During

the spring break accesses are very low.

Table 5.6: Accesses to course pages

Course	Spring 95	Summer 95	Fall 95	Break	Spring 96	Summer 96	Fall 96
cs1206	NA	NA	57	6	20,568	719	2,936
cs1704	NA	NA	3	0	159,553	5,064	44,953
cs2304	10,337	655	14,894	69	5675	281	1048
cs2504	4,889	1,684	33,208	177	23,141	896	9,034
cs2604	35,430	5,740	24,796	86	22,358	1,501	7,586
cs3204	5,773	3,990	4,768	73	814	8,474	5,793
cs3604	23,875	1,104	20,836	198	16,915	3,525	18,847
cs4624	45,100	4,652	46,241	602	62,050	19,225	62,849
cs5604	9,189	380	37,031	422	5,667	2,138	8,291
History	2,711	108	5,506	53	1,693	991	3,161
Netinfo	45,100	4,652	46,241	602	62,050	19,225	62,894
WWWbtb	NA	NA	NA	NA	448	263	6,801

Table 5.7: Bytes transferred from course pages (in MB)

Course	Spring 95	Summer 95	Fall 95	Break	Spring 96	Summer 96	Fall 96
cs1206	NA	NA	0	0	136	10	20
cs1704	NA	NA	0	0	1,817	224	401
cs2304	18	1	30	0	10	0	3
cs2504	108	41	151	3	96	6	23
cs2604	67	10	43	0	17	2	19
cs3204	171	110	40	0	4	56	18
cs3604	79	3	75	1	57	17	65
cs4624	241	34	384	5	531	231	766
cs5604	34	2	140	1	26	12	39
History	45	3	127	1	44	14	66
Netinfo	241	34	38	5	531	231	766
WWWbtb	NA	NA	NA	NA	2	2	31

Chapter 6

Characterizing Users' Searches and Sessions

6.1 Introduction

In this chapter we analyze accesses from groups of clients to several Web Information Retrieval Systems (IRS), and in particular study their queries.

To help users locate information on the Web, special search tools and cataloging systems were developed. We call these tools Web Information Retrieval Systems (IRS). They have evolved rapidly since 1994, and have been used by millions who thereby had their first

experience with search engines. Borgman et al. [57] observed that "The end users who now dominate searching are using systems with exploratory interfaces, under less time pressure, and have less clear retrieval goals than do skilled search intermediaries."

As part of our effort to characterize WWW traffic so as to support modeling, planning and prediction, and to help scale up the Web, we seek to study retrieval-related traffic. In particular we study sessions, queries, and browsing activities using five log files from our collection; the log files come from diverse situations. We characterize accesses to different Web IRS to find out the most popular systems. Next, we study client systems to see how much Web traffic is due to multiuser systems and if such systems have similar access patterns to those of individual users. We then characterize queries by looking at their complexity with respect to numbers of terms and operators. Finally, we demonstrate that sessions with searching tend to consume less network bandwidth than other sessions. We do this by first defining an algorithm to extract client sessions. Second, we analyze the sessions and show that those with more search steps are responsible for fewer transferred bytes. We define user sessions in terms of "Browsing," "Searching," and "Next step" activities. Then we look at the most popular patterns in the identified sessions. Using regression, we discover a correlation between amount of searching in a session and the bandwidth requirement.

6.1.1 Clients, Users and Sessions

A WWW client can use an instance (or multiple instances) of a WWW browser. One instance of the browser can run on a single user machine such as a PC with Windows 95. In this case the client accesses are exactly the same as the user accesses. In some other cases multiple instances of the browser might be running on a multiuser machine. In such cases, accesses from this client might be due to several users running multiple instances of the browser.

Sessions are very hard to characterize [1]. In this paper we define sessions with respect to clients not users. We then use heuristics to define session boundaries, which serve as the basis for the details in section 6.

6.1.2 Finding Information Over the Web

Currently users can look for information over the Web using three methods:

Browsing: Users can follow links from page to page, reading or looking at what they find interesting. However, even those with a focused information need may get distracted during the browsing process and end up reading or looking at something completely irrelevant to their initial interest. On the other hand, browsing can be particularly efficient when users

are looking for information within a specific collection that is limited in size.

Using Web catalogues: Like with Yahoo, users with a general topic can apply a classification system provided to narrow their subject until they find what they are looking for. For example, if we are interested in HTML and want to read about it, we can take the following path to get to the required information: from the main directory in the Yahoo home page to Computers and Internet, World Wide Web, and finally HTML. Following these three steps will lead to several further choices about HTML including editing tools, standards, manuals, etc.

Using Web IRS: When looking for something more specific, users often turn to search engines such as Lycos or Infoseek. For example, we may want to read a certain article and know the author of the article but not know the journal or date. We can use the author's name and a key word from the article to locate it.

While the Web is used in other ways, these three methods cover a high volume of WWW traffic, and so are worth more careful characterization.

6.1.3 Related Work

In [58] the authors compare several WWW search engines by using five faculty members to search for certain subjects and then judge the relevance of the returned results. Evaluation of the search engines is done using "signal detection analysis." The authors provide two measures to evaluate search engines. The first one is the sensitivity of the search engine in finding useful information. The second one is how conservative or liberal the search engine is in determining which sites to include in the search results. The authors conclude that the current search systems exhibit poor performance.

Pollock [59] demonstrates that user background does not affect the search process or results. The author demonstrates this through an experiment that compares naive and non-naive user searches. The author concludes that there are misconceptions and problems with Web searching due to the design of available search tools.

Our approach is different in many ways from studies like those mentioned above. First, our data has been collected from a variety of user groups in natural environments, not experimental situations. Second, we do not evaluate the different search systems. We do study user queries to see if they utilize available functionalities. Third, we look at the usage of search engines and compare it to browsing. Fourth, we demonstrate a beneficial seeming correlation regarding searching by observing that sessions with more searching frequently have less browsing, so there are fewer bytes transferred over the network.

Table 6.1: Workloads used in this study

Workload	Period	Number of (K) accesses
Korea	9/2/95-9/26/95	1,682
Library (VT)	9/19/96-11/20/96	128
CS (VT)	10/9/96-11/10/96	92
AUB	10/21/96-10/22/96	19
AOL	11/96 (one day)	897

6.1.4 Workloads used in this study

We examine five log files picked from our collection and collected from different communities. Table 6.1 describes the workloads used in this study, showing dates of log file collection and number of accesses.

6.2 Workload Characterization

We define accesses to Web IRS to include all accesses to Web IRS such as Lycos, Infoseek and to cataloging systems such as Yahoo. For comparison, we count the number of accesses that contain a search command and compare it to the total number of accesses in the workload to all servers. However when we analyze queries we only look at accesses with an

Table 6.2: Accesses to Web IRS and their percentages

Workload	Accesses to Web IRS	% of total accesses
Korea	67756	4
Library (VT)	15408	12
CS (VT)	5780	6
AUB	3839	20
AOL	78026	9

explicit query command.

6.2.1 Characterizing Accesses to Web IRS

Table 6.2 compares accesses to search servers to the total number of accesses in each workload. We notice that in all workloads the percentage of accesses to Web IRS is no more than 20%. This is reasonable since the search activity is always followed by browsing activity to explore the returned URLs and how relevant they are to the search. However, we also notice that in the high school case the percentage is very high compared to the others; this is because the students were instructed to use the Web to find information. Interestingly the library workload was second highest, which reflects a key service of libraries, namely locating information. We call this the *task* effect; because the students were assigned a

certain task and library users often focus on search for information.

The Korea workload has the lowest percentage. One reason could be that the data was collected a year earlier than other workloads and Web users were not very well informed about Web IRS or because all search engines are located outside Korea, hence users avoid searching due to slow network connections. We expected that the CS workload ratio would be higher than other workloads in using Web IRS but local CS clients search the Web during only 6% of their accesses. This ratio is low compared to all other workloads except Korea. By examining the log file further we noticed that accesses from the CS clients are mostly to servers inside the department and those accesses are mostly repeated accesses to pages they are developing.

Table 6.3 lists the total number of clients in each workload and compares it with the number of clients who have accessed a Web IRS. We see that the ratio differs for different workloads. All clients in the high school have accessed Web IRS because they were supervised by teachers who knew about the available search tools and how to use them and who instructed the students regarding how to search.

Tables 6.2 and 6.3 show that Web users do use Web IRS to locate information. However the degree of computer knowledge does not appear to predict the percentage of accesses that are queries, for our workloads. This is consistent with the findings in [59]. However, the task effect is very clear in two cases, the library and the high school (AUB).

Workload	Clients (total)	Clients performing searches	% searches
Korea	2136	929	43
Library (VT)	348	203	58
CS (VT)	36	21	60
AUB	79	79	100

Table 6.3: Total number of clients and clients which accessed a search engine

Table 6.4 shows the distribution of accesses to different Web IRS. One thing that is common between all workloads except AOL is that Yahoo is the top used IRS. Why does Yahoo get so many accesses? It could be because it has combined Web catalogues and searching techniques, or because Yahoo was listed in the Netscape search page which is the page used by most of the users to access Web IRS. In a discussion with one of the users, she said that she uses Yahoo because the name is easier to remember and to type compared to, for example, Altavista. A usability study might compare all these Web IRS and find out why one is used more than the others.

Some of the Web IRS have 0 hits or 0 percentage hits; this does not mean that they are worse than the other tools. It could be that they were new and users had not discovered them yet. In addition some of these Web IRS are specialized in finding people not information; as a result we expect them to be accessed less frequently than the other IRS. Nevertheless, for our purposes using data related to the varied workloads and the Web IRS shown in

	Korea		Lib(VT)		CS(VT)		AUB		AOL	
Web IRS	Acc.	%	Acc.	%	Acc.	%	Acc.	%	Acc.	%
Altavista	0	0	846	5	1073	19	20	1	3370	4
AOL	193	0	1033	7	427	7	309	8	4235	5
ELibrary	0	0	170	1	0	0	0	0	1420	2
Excite	4	0	2381	15	477	8	186	5	6842	9
Four11	31	0	185	1	35	1	46	1	1370	2
Infoseek	17361	26	1178	8	563	10	479	12	1588	2
Lycos	2811	4	2650	17	983	17	431	11	2463	3
Magillan	179	0	2015	13	277	5	406	11	71	0
NLN	419	1	12	0	0	0	0	0	23	0
Opentext	312	0	29	0	0	0	0	0	48	0
Webcrawler	3653	5	155	1	97	2	473	12	46356	59
Yahoo	42792	63	4742	31	1834	32	1488	39	8028	10
100hot	0	0	0	0	14	0	0	0	60	0

Table 6.4: Distribution of accesses to currently available Web IRS

Table **??** should allow our results to be more generalizable than if a single workload or Web IRS was considered.

6.2.2 Characterizing Clients' Accesses

We examine the distribution of accesses per client and the cumulative accesses for the clients in each workload. In all workloads we notice that we have a small percentage of clients which are very active followed by less active clients. The number of accesses decays rapidly and shows an asymptotic behavior. By fitting the data to distributions using ExpertFit [60] we found out that they follow a "Weibull" distribution [61] with $\alpha = 0.44$ for the Korea workload, $\alpha = 0.51$ for the library workload, $\alpha = 0.37$ for the CS workload, and $\alpha = 0.48$ for the AUB workload. This indicates that a small percentage of the clients are responsible for most of the accesses. In the CS case most of the accesses came from a multiuser machine; that is why the shape of the curve is different from the other workloads.

Figure 6.1 shows the results of plotting the percentage of cumulative number of accesses for the clients vs. the percentage of clients. In all workloads used we see that 50% of the clients are responsible for more than 80% of the accesses. In the CS case 10%-20% of the clients are responsible for more than 80% because the students use a multiuser machine.

It is hard to make conclusions about specific users' behavior since our data is about client machines. However we can suggest possible explanations about general user behavior that

can be tested in the future. We can say that a small percentage of the clients access search tools on the Web and the rest of the accesses are due to browsing. Why would that occur? We speculate that users use the Web for repetitive known server accesses, or are not well trained to use Web IRS, or that they think of the Web as an extension to the TV where you browse channels until you find something interesting to watch. If our explanations are correct, one area of training might be to inform users about search tools and how to use them so more users can search effectively.

6.3 Queries

In this section we study queries submitted by users and also attempt to identify inefficiencies. We start by counting how many accesses to the Web IRS are queries and compare that with the total number of accesses to the Web IRS. Table 6.5 shows the number of accesses which contain explicit queries to Web IRS in each workload. The rest of the accesses to Web IRS may be to load the home page of the Web IRS, to read help pages about how to do the search, or to access inline images. The CS (VT) workload has the highest percentage of queries. A higher percentage in this table means that the users in this workload accessed the Web IRS more to submit queries and less to take other actions.

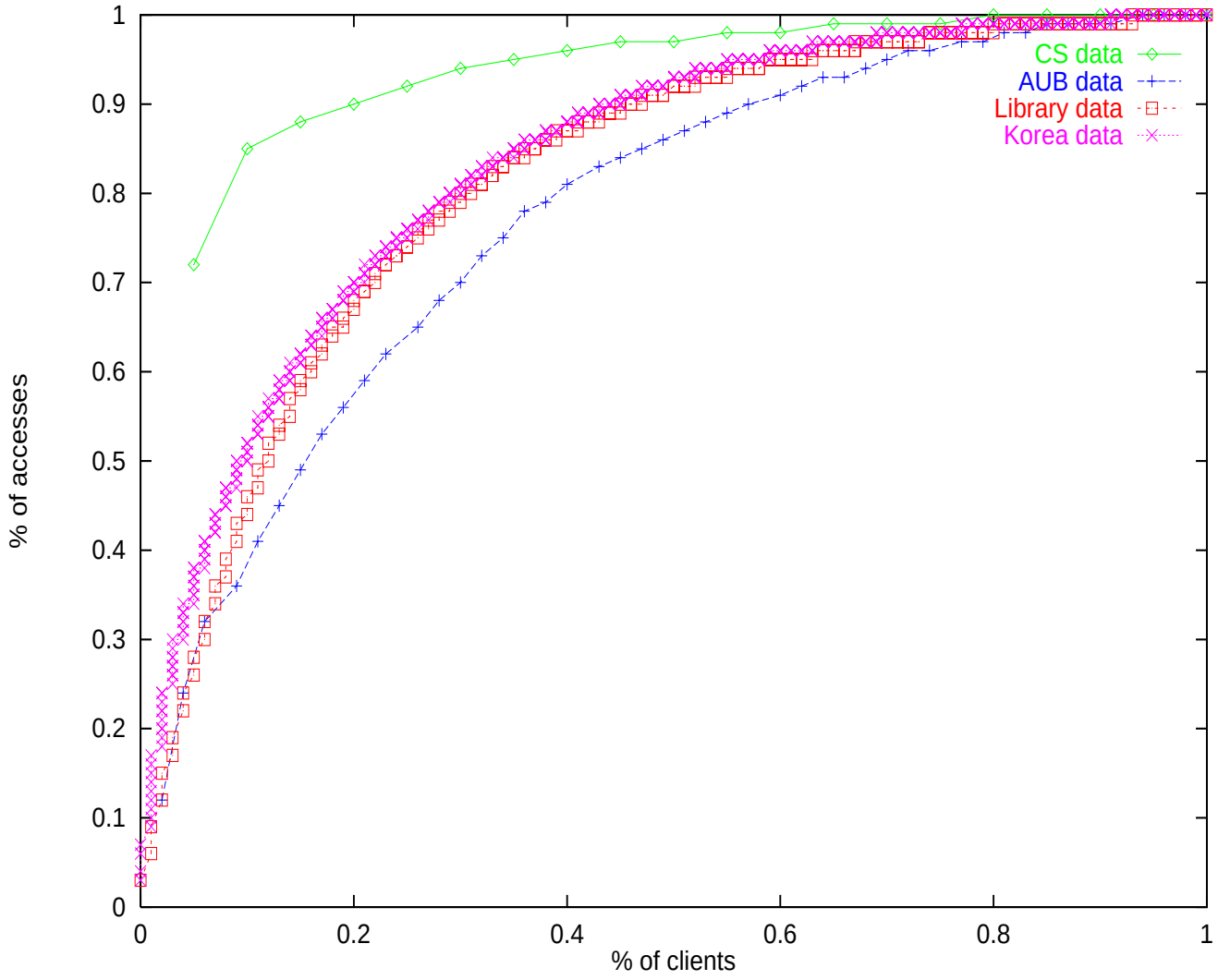


Figure 6.1: Percentage of accesses vs. percentage of clients

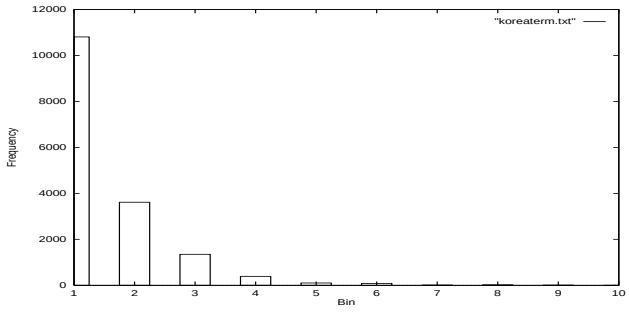
Table 6.5: Number and percentage of accesses that contain queries

Workload	Number of queries	% of accesses
Korea	16845	22
Library (VT)	2515	16
CS (VT)	1503	26
AUB	619	16
AOL	5342	7

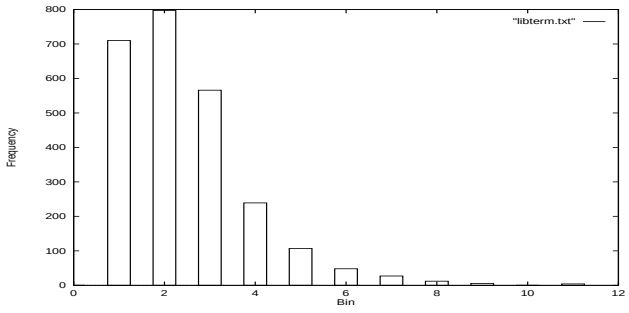
6.3.1 Query Complexity

The low percentage in the high school case shows that they have used Web IRS to browse through the built in indexes to browse for the relevant information. The high percentage in the Korea or CS cases shows that they accessed the search tools to submit a query. These conclusions were verified by examining the log files.

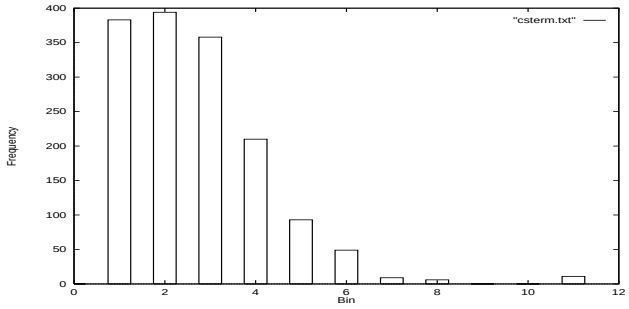
We measure query complexity by counting the number of *terms* and number of *operators* used in queries. We refer to each word in the query as a term; and each "and," "or," "not" or any proximity operator as an *operator*. Table 6.6 shows the distribution of number of operators/query for each workload. In each case most of the queries had no operators, as would be expected for one-term queries, or queries to a "natural language" Web IRS. (Note: This is the case for the Web IRS that we have examined; all have a ranking feature,



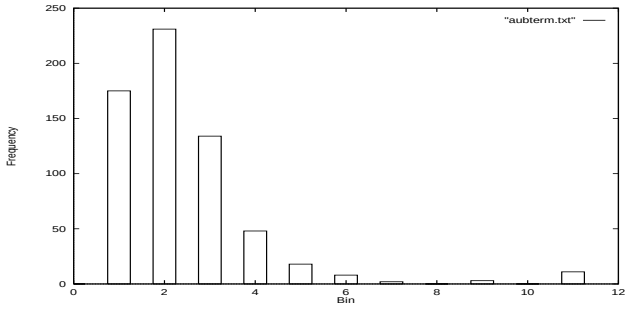
a- Korea



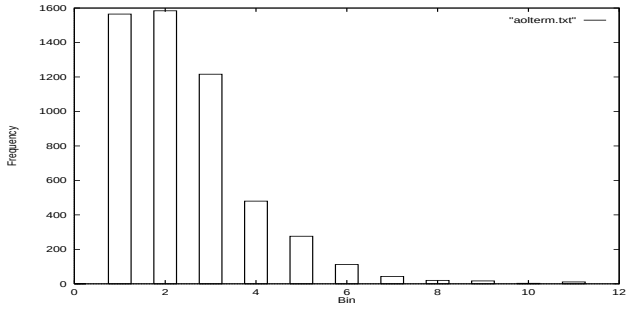
b- Library (VT)



c-CS (VT)



d-AUB



e- AOL

Figure 6.2: Number of terms/query

Table 6.6: Counts of number of operators/query

Workload	0 operators (%)	1 operator (%)	$\geq$ 2 operators (%)
Korea	16672 (98.94)	172 (1.04)	1 (0.01)
Library (VT)	2408 (95.74)	105 (4.17)	2 (0.08)
CS (VT)	1492 (99.27)	11 (0.73)	0 (0.00)
AUB	614 (99.19)	5 (0.81)	0 (0.00)
AOL	5050 (94.5)	282 (5.3)	10 (0.2)

or allow Boolean queries if special syntax rules are followed.)

Figure 6.3.1 shows the distribution of number of terms in the issued queries for each workload. From the figures we can see that users often use more than one term in their queries; in fact for all workloads except Korea (where perhaps search skills most need improvement) two terms per query is the most popular situation. When using more terms users may be trying to be specific to get more relevant results. In some queries the number of terms exceeded 5. In a ranked system this is beneficial in spite of the large number of partial matches if good terms are chosen, due to statistical correlation, since users can define a threshold when they will stop looking at retrieved documents, or can just examine the k best documents (e.g., for k=20).

By examining queries we noticed that the majority were simple natural language queries.

Clients rarely use the Boolean operators or the advanced features of the Web IRS.

6.4 Harvest, the server model

We collected data from the campus Web IRS server in the library at Virginia Tech accessible on the page www.vt.edu. This Harvest server is accessed by clients from inside and outside Virginia Tech. It represents a different model of data collection than all but our AOL workload since it is server based, not focused on the behavior of a group of clients as in the previous cases studied. The number of accesses to the server were 54118 out of which 1048 were invalid queries (1.94%); 12308 (22.74%) had 0 returned objects. The invalid queries were either due to spelling mistakes or misuse of operators. In addition Harvest users used stop words in 10.69% of the queries studied, indicating that they assumed it is a natural language system. However, Harvest is a Boolean system and does not rank returned objects, but handles a sequence of words by "anding" them.

Figure 6.3 and Table 6.7 show the distributions for number of terms and number of operators for users searching the Harvest server. The figures indicate that the distribution is essentially the same as the one noticed for other workloads. This may be true because the interface to Harvest is through a network browser so users may assume it is similar to other Web IRS.

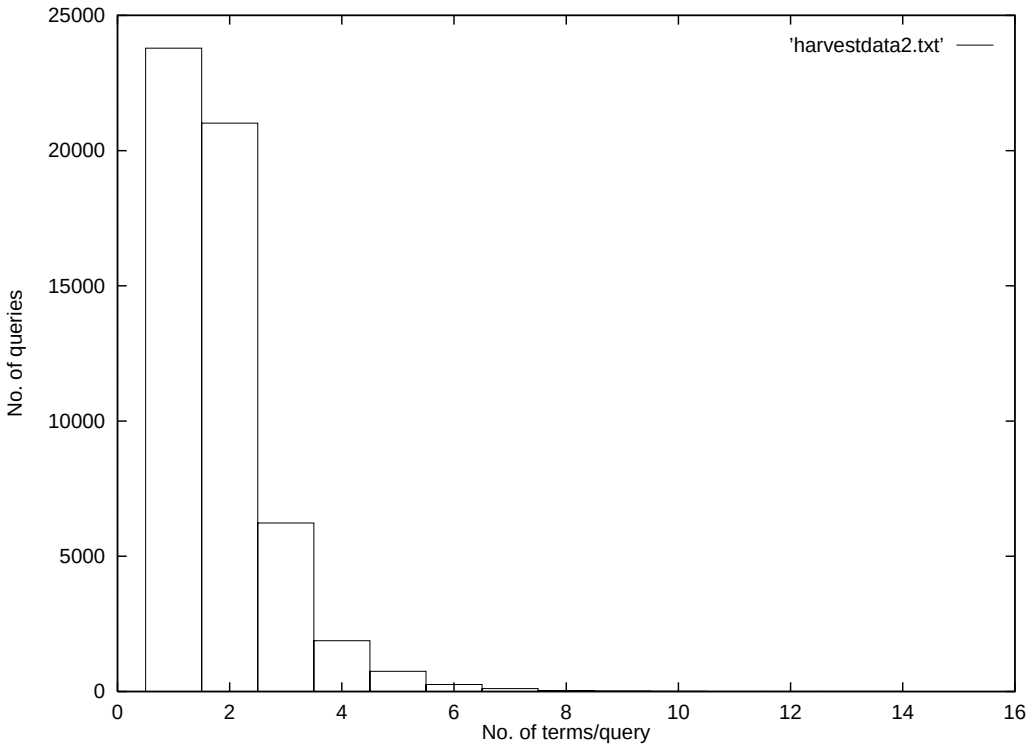


Figure 6.3: Number of terms/query, Harvest

Figure 6.4 shows the distribution for the number of retrieved objects from Harvest. We can see that most of the searches resulted in a small number of retrieved documents possibly because the collection searched is relatively small.

6.5 Characterizing Sessions

User and client sessions contain important information. The problem of extracting sessions is very hard and needs more research. In our study we used heuristics to identify sessions

Table 6.7: Number of operators/query, Harvest

Workload	0 operators (%)	1 operator (%)	$\geq$ 2 operators (%)
Harvest	52700 (97.38)	1224 (2.26)	194 (0.36)

and then studied the correlation of number of searches and the total number of bytes transferred. We also classified the behavior of clients in terms of three steps: "browse" or "b", "search" or "s", and "next set of results" or "n". Here "search" or "s" is defined to describe accesses with an explicit query; "next" or "n" is for accesses with "n" in the query field; this is what the client will send as a result of the user clicking "next" set of results. Any other action is considered to be a "Browse" or "b".

The first step in processing was to extract the interactions for each client from the log file that contains all accesses generated from that client. The next step was to identify session boundaries in each client's log file using the algorithm of Figure 6.5. By examining the log files we selected 300 seconds as a safe threshold to indicate a session break. We scanned the client logs generating a sequence of interactions for each identified session. For example a certain client might have 10 sessions in its log file. Each session is represented as a sequence of "b", "s", and "n" codes. The most frequent common sequences, which appear in all workloads, in effect yield a partial grammar of client sessions, empirically based. See Table 6.9.

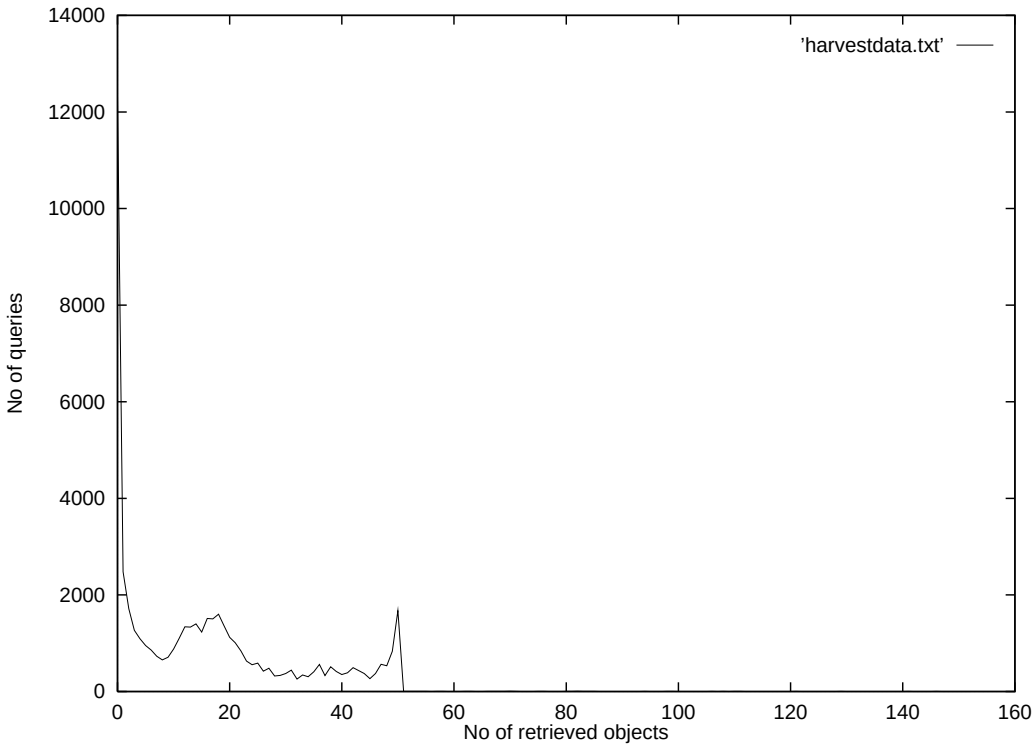


Figure 6.4: Number of queries vs. Number of retrieved objects

We explore the correlation between searching steps and the number of bytes transferred. We defined $RSNB = (s+n)/b$, which is the ratio of search and next steps over the browsing steps. We conjectured that if we have more search-related steps then the number of bytes transferred for the session should be less. We used regression and correlation to determine the relation between the above ratio as well as counts of b, s, and n codes, and the number of bytes transferred.

The correlation analysis for four of the workloads (Korea, AUB, CS (VT), Library(VT)) showed that there is a negative correlation between the ratio $(s+n)/b$ when compared with

```
 If (access is from same client)
    {
     if (accessed path is related to previous accessed path)
{This is the same session}
     else {
   if (Time between accesses< 300 seconds)
       {This is the same session}
   else { different session}
  }
     }
 else {different session}
```

Figure 6.5: Algorithm to define a client session

Table 6.8: Regression results for bytes transferred and b, s, n

	Korea		Lib(VT)		CS(VT)		AUB	
Variable	Coeff.	T ratio	Coeff.	T ratio	Coeff.	T ratio	Coeff.	T ratio
RSNB	903312	-5.49	1077943	-4.71	1556183	-5.38	4794527	-1.95
b	10929	38.46	3170	17.12	5483	19.35	9639	14.09
s	-2037	-0.13	128678	12.41	36788	1.83	-103431	-4.78
n	21519	2.05	46705	3.06	-47947	-1.68	-1084	-0.03

the number of bytes transferred. Table 6.8, in the first row, shows the regression results for number of bytes transferred as a function of the ratio RSNB. The regression results for all four workloads are highly significant. This negative correlation suggests that a higher searching and browsing ratio corresponds to fewer total bytes transferred.

The regression analysis for number of bytes transferred and the b, s, n counts are shown in Table 6.8 in the last three rows. The coefficients of the variable b are significant in all workloads. The coefficients of the variable s are only significant in the library and AUB workloads. The coefficients of the variable n are only significant in the Korea and library workloads. It appears that b is the most important variable that determines the total bytes transferred.

To simplify the analysis of access patterns, we replace a sequence of b's with "B". Table 6.9 shows the top 10 popular access patterns for the workloads. The most popular access pattern in all sessions is the sequence with b steps only and it is represented by B in the table. This pattern occurs in over 85% of all sessions which suggests that users are browsing most of the time. This is consistent with our conclusions in section 4.2. This high percentage suggests how much bandwidth we might save if we enable users to search more and browse less when they look for specific information, such as by having more effective retrieval methods. It is interesting to note that the first three access patterns not only are common between all workloads but also that they have very close percentages. The second popular access pattern is "BsB", which implies that the client has done one search after browsing and ended the session with browsing. Access pattern "BnB" is interesting because we had an n appearing without a search. By examining the log file we found out that this can occur if the client uses the catalogue in some Web IRS to get to a certain category, then presses "next screen" to get more hits.

6.6 Conclusions and Future Work

In this study we found that users do use the Web IRS to locate and find information. We also found that a small percentage of the clients are responsible for most of the accesses to the Web IRS. In particular less than 50% of the clients are responsible for more than

Access Pattern	Korea (%)	Library (VT)(%)	CS (VT)(%)	AUB(%)
B	85.25	93.66	89.78	85.25
BsB	5.63	3.86	4.02	5.63
BsBsB	2.00	0.62	1.72	2.00
BsBsBsB	1.15	0.20	0.70	1.15
BsBnB	0.51	0.05	0.14	0.51
BnB	0.46	0.93	0.27	0.46
BsBsBsBsB	0.35	0.07	0.41	0.35
BsBnBnB	0.30	0.01	0.00	0.00
BnBnB	0.15	0.08	0.15	0.15
BnBnBnB	0.15	0.03	0.00	0.00

Table 6.9: Percentage of the most frequently repeated string patterns in all workloads.

80% of the accesses. Clients rarely use the special additional features available through the Web IRS interface. It appears that users prefer to issue queries with one term and without Boolean operators.

By looking at the access patterns in the sessions we found that the most frequent repeated pattern is browsing only. Finally, using regression and correlation analysis we found that when a session involves more searching, then less network and server bandwidth is used.

Future work should identify why users rarely invoke advanced Web IRS features during searches. More work should be done to characterize queries to Web IRS servers. Ultimately, through more log analysis, supplemented with user interviews and usability studies, better models of how users of the WWW find and work with information should enhance its scalability.

We are conducting a more comprehensive study to analyze queries. We will look for common user errors such as minor spelling mistakes or misuse of Boolean operators. We think that Web IRS should help correct such mistakes for users. Another study could look into the session characterization problem in more detail.

Chapter 7

Network Traffic is not the Sum of its Components

7.1 Introduction

In this chapter we define scenarios that can be used to test the WWW and distributed information systems' scalability with emphasis on the educational systems. We do that by first defining a scenario, second we define the objective of developing scenarios, and finally we discuss several scenarios that will show how to utilize our findings to build scalable web architectures for the future.

7.1.1 Scenarios

A scenario is a representative example or instance of WWW usage. The scenario can focus on clients, servers, users, network, or any other potential WWW object. To make it useful, a scenario has to have well defined output parameters that need to be studied and input parameters that can be changed during the analysis. For example we might want to study the number of clients that can be supported by a network connection given that those clients have certain working habits. The objective of the scenario can be to understand the system better, predict future behavior, and assist with capacity planning among other things. In our case we will use scenarios to show effects of caching on the WWW scalability.

We will start our analysis by raising some questions that we would like to examine.

7.2 A Simple WWW Model

In this section, we raise some fundamental questions about the ability of the Web to scale for future demands. Then we introduce a simple model of the WWW to help answer these questions. We will discuss the results obtained from the analysis of the model and come up with recommendations to achieve scalability. Finally we will list potential future research problems.

7.2.1 Questions We Want to Examine

We examine the following questions:

1. For a fixed number of clients in an enterprise, what is the bottleneck in client access of WWW documents? What can be done to alleviate the bottleneck?

2. What happens when we start using video and audio heavily; can the scaled model withstand the new requirements imposed by the new media types?

3. For a fixed document request rate by a client to a popular Web server, how many Web clients can one server support? What can be done to support a larger number of number of clients?

7.2 The Model

Our model is illustrated in Figure 7.1 and uses the following parameters:

n : total number of clients sharing one internet connection; in Figurer 7.1, n is the clients inside the box labeled enterprise.

pv : percentage of transferred bytes that is video data.

pa : percentage of transferred bytes that is audio data.

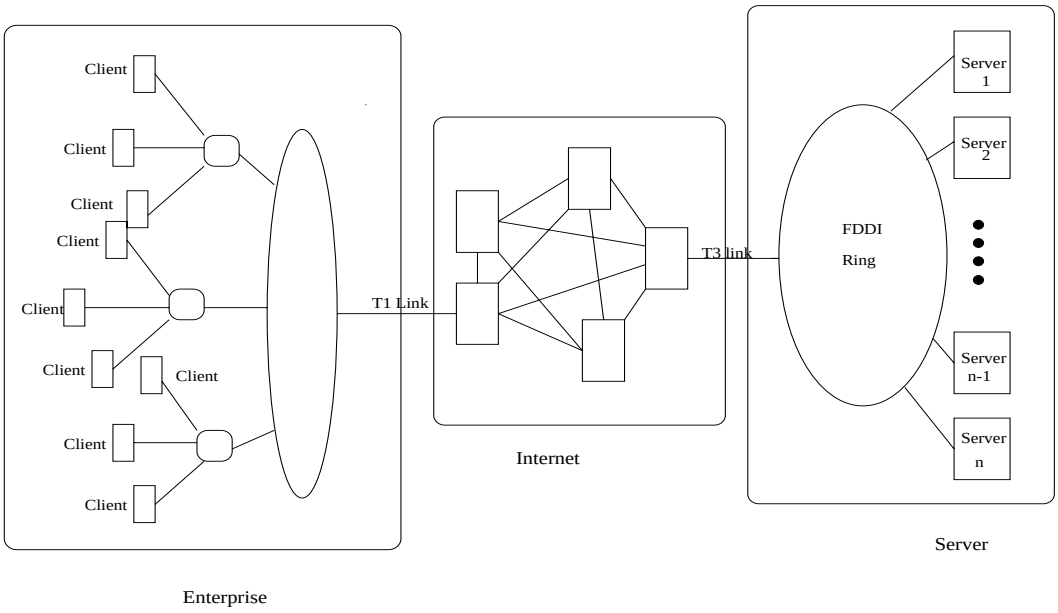


Figure 7.1: WWW Model

pr : percentage of transferred bytes that are not video or audio (e.g., jpeg, gif, txt, html); $pr = 1 - pv - pa$.

U : utilization of a client (or fraction of time that a client is busy [i.e., a user is using a Web client]).

R : requests/sec generated by a client when busy.

v_s : video file transfer rate needed to achieve real time playback.

a_s : audio file transfer rate needed to achieve real time playback.

r_s : average transfer rate for the rest of the files that makes accessing the Web acceptable for the rest of the users.

c_m : data compression ratio.

h_r : proxy cache hit rate [62].

c_r : client cache hit rate.

C_t : traffic generated from one client.

I_c : speed of the enterprise connection to the Internet.

The number of clients that can be supported in a school with a T1 link can be easily computed by the following formula:

$$n = \frac{I_c - (\text{outgoing traffic})}{C_t}. \tag{7.1}$$

For simplicity we will assume that the outgoing traffic equals zero; later we will change this assumption and see the effect of the outgoing traffic. The previous equation becomes

$$n = \frac{I_c}{C_t}. \tag{7.2}$$

We need to device a formula for C_t; it can be shown easily that C_t can be computed by the following expression,

$$C_t = U * R * [(pv * v_s) + (pa * a_s) + (pr * r_s) * c_m]. \tag{7.3}$$

7.2.3 Measurements and Model Parameter Justification

Let us solve a hypothetical scenario, based on the numbers justified below. Then we vary these numbers and study the effect on scalability. We call this a scenario because the parameter values we use are representative of values encountered in certain measurements, but are not necessarily representative of any real enterprise or organization. Table 7.1 lists the parameters and their values.

We choose the values for pv, pa, and ps based on the results obtained from log files we collected and reported on [19]. The value of U was calculated from the log files of a PC machine that is used by several students in a research lab by dividing the number of hours the machine was used on average by 24. The value for r was obtained by tracing 10 different users' sessions and averaging them. In each session we counted the number of GETs and POSTs issued by a client; this number was then divided by the total session time to get the average requests/sec from the client.

The value for v_s was obtained by assuming that we want to play real time video over the network; we used QuickTime format with 160X120 resolution at 10 frames/sec accompanied with 8 bit 22kHz mono sound. The v_s value was computed by dividing the size of the QuickTime movie by its duration. We did this for five different movies and then we took the average of the obtained numbers. Since we are looking for the bytes transfer rate that is required to deliver live video then we can average the file sizes. The transfer rate

to playback a QuickTime movie with 160X120 resolution at 10 frames/sec in real time is much less than the rate reached when downloading a movie to play it back, given that the network is not heavily loaded. We study the effect of playing real time video since we expect it to be used heavily in the future. Also this transfer rate will provide a lower bound on retrieving video files since this is the slowest rate that is needed to view QuickTime video files with resolution 160X120 at 10 frames/sec.

The value for a_s was obtained in a similar way to v_s but we used Real Audio files. Real Audio files require very low transfer rate 8796 b/sec.

The value for r_s was chosen by noticing that most Web users are happy when they get a transfer rate of 5 KB/sec.

The value for the proxy cache hit rate was measured from our log files, ranging between 30%-60% [62]. Similar numbers have been reported in the literature [14]. In this chapter we use the value 40% for the network cache hit rate. The value for the client hit rate *cr* also was measured by examining the caches on several clients in one of our labs, and the measurement did not look at the files that have been reused by a single client; instead we measured the percentage of the shared files in clients' disk caches between several Netscape clients in an under graduate lab over a month period. The measured value was 17%.

7.3 Model Analysis

7.3.1 Scenarios 1 and 2, Enterprise Side

Scenario 1 deals with network scalability. The scenario will answer the following types of questions: How many clients can we support in an enterprise (a university campus, Virginia Tech, as an example)? How many clients can we support if we add caches? How many clients can we support if we use data compression? At what point is increasing network bandwidth the only way to increase the number of clients? Will the increase of media quality (e.g. better sound, better video) decrease the number of supported clients? Is this also true if the percentage of using video and audio increases?

To answer these questions we will look at the part labeled "Enterprise" of Figure 7.1, which represents a simplified model of the campus network topology. Before we start the analysis we will make the following assumptions: First, the load on the Internet backbone is light compared to its bandwidth; this assumption eliminates the internet as a potential bottleneck. Second, there is an infinite number of servers that can be accessed and the load is distributed between the servers; this assumption eliminates the server as a bottleneck. Later we will see how the server can be a bottleneck and how we can adjust for that. Third, we will use a compression ratio of 1:2, or $c_m = 0.5$ for file types other than audio and video since the latter two types are normally compressed. Fourth, we will ignore the

Table 7.1: Factors and their values for the base case.

Factors	pv	pa	pr	U	R	vs	as	rs
Values	4%	18%	78%	.125	.124	468262	8796	40960

LAN connections inside the enterprise for two reasons. First, the LAN bandwidth ranges from 10MB/sec to 155MB/sec or beyond (Ethernet to ATM) which is higher than the connection to the internet (normally T1 with 1.5MB/sec speed). Second, the number of clients sharing a LAN segment is very small compared to the total number of clients.

We start the scenarios by computing the number of clients that can be supported for a base case; the values for the factors used to compute this base case are shown in Table 7.1.

From equations 2 and 3, the number of clients that can be supported in the base case is n =1942 clients. This number is reasonable for a medium size enterprise such as a medium size school. Is this a reasonable number for the same enterprise 5 years from now? First let us see how we can use caching, compression, and the increase of network bandwidth to increase the number of supported clients.

Table 7.2: Factor level combination for one-at-a-time approach; underlined items indicate what is varied.

Case Number	Client Caching	Proxy Caching	Compression	Network Connection T1/T3
0	No	No	No	T1
1	Yes	No	No	T1
2	Yes	Yes	No	T1
3	Yes	Yes	Yes	T1
4	Yes	Yes	Yes	T3

Scenario 1

Table 7.2 lists the factors and their combinations starting with the base case 0. To examine the effect of a factor, we will change it while keeping the other factors fixed and compute the number of clients that can be supported. This is a classic one-at-a-time experimental design approach. We will assume for simplicity that pa, U, R, and r_s will not change. We also will drop pr from the table since its value depends on pv and pa.

Table 7.3 shows how caching can increase the number of clients from 1942 in the base case to 2339 when we use client caching and to 3899 when we use a combination of client and proxy caching. Compression shows good results, increasing the number of clients to 4707 if

Table 7.3: Number of clients that can be supported for each combination.

Case	Number of Clients	% change over case 0
0	1942	-
1	2339	20%
2	3899	66%
3	4707	21%
4	233932	2900%

we manage to compress the text data into half of its size we get 21% increase in the number of clients. The most impressive result is achieved when we use the 45 MB Bytes per second since we are getting 29 times the number of previously supported clients. Note further that a combination of the previous techniques will allow supporting a huge number of clients. These numbers, however, are obtained by assuming all the mix of data types does not lead to an increase in the number of bytes. The next scenario will show what happens when we increase data quality (i.e., have richer multimedia content) and utilization.

Table 7.4: Factor level combination for one-at-a-time approach; underlined items indicate what is varied

Case	*pv*	*vs*	*as*	*U*
0	4%	160X120X10	Real Audio	.125
1	40%	160X120X10	Real Audio	.125
2	40%	320X240X10	Real Audio	.125
3	40%	640X480X30	Real Audio	.125
4	40%	640X480X30	10XReal Audio	.125
5	40%	640X480X30	10XReal Audio	.36

Scenario 2

In scenario 2 we will start from the results achieved in scenario 1. Assuming that we have the situation of case 4 in Tables 7.2, how does the number of supportable clients change by changing the factors shown in Table 7.4?

Table 7.5 list the cases and the resulting numbers of clients. The percentage of change in this table reflects the percentage of decrease in the number of supported clients relative to the case on the line above.

Table 7.5 shows that the effect of increasing the percentage of video data from 4% to 40% reduces the number of supported clients by 294%. The biggest effect was due to the

Table 7.5: Number of clients that can be supported and the percentage of change, relative to previous case, due to the corresponding factor combination in Table 7.4

Case	Number of Clients	% change
0	141214	-
1	59322	294%
2	15919	272%
3	676	2254%
4	674	0.3%
5	234	63%

video quality increase to 640X480, 30 frames/sec, when the number of clients decreased by 2254%. The increase in audio quality did not make a big difference for two reasons, first we assumed that the percentage of audio files will remain 18%. Second, we only used multiples of the rate of the current Real Audio play back rate which requires little bandwidth. There is no need to assume that we will need much higher audio rates since with the current technology we can achieve good sound quality without sacrificing lots of bandwidth. Case 5 is worth looking at, because in the future we expect that the demand for digital video will be very high and the expected video quality will be either near the assumed value in the previous calculations or even better. This implies that in an enterprise such as Virginia Tech we will only be able to support 234 clients. We expect that the school will have more simultaneous active clients. This means that even with client caching, proxy caching, and compression we have a serious problem. In this case we might turn to the school Internet connection and instead of using a T3 with 45MB/sec, use an ATM connection which will increase the number of clients , but only to 668 client. 668 is still not many, so we might think of using GB/sec network connections.

So far we have assumed that the enterprise has only one type of traffic, that is traffic from accesses by the clients. What if the enterprise has servers with data on them? For simplicity and to get an idea of the effect of such traffic let us assume that the outgoing traffic is the same as the incoming traffic. Then we have to divide the number of clients in half in Tables 7.3 and 7.5, which means that we will reach a bottleneck faster.

7.3.2 Scenario 3, Server Side

From the server side (see Figure 7.1) we are interested in the number of clients that a certain server can support. The following scenario will help us answer questions of the following type: how many clients can a contemporary workstation support at the same time? Can we use server caching to support more clients? When do we need to increase the server network bandwidth?

Contemporary workstations can get to a state where they cannot keep up with the request rate r_r or cannot fork the number of processes required to satisfy the arrived requests. This can happen to servers that host popular data, or servers that are used as network directories, for example SunSite or Yahoo. To alleviate this problem caches at the server side should be used. If the server is replicated m times, the request rate will drop on each server to r_r/m and the required number of processes to be forked to r_r/m. The designers of the network architecture can pick a workstation that can support a request rate$= r_r$ and connect more than one workstation with a high speed network such as an FDDI ring as shown in Figure 7.1. This solution was adopted by NCSA to scale up their popular server [12].

7.3.3 Scenario 3

Scenario 2 deals with server scalability; the factors pv, pa, pr, v_s, a_s, and r_s are defined the same way as they were defined for the previous scenario except that these values represent the traffic going out from the server. Let R_s represent the rate of requests arriving at the server. This number might be very high compared to the client request rate since it represents the aggregation of requests by all clients. For a Web dedicated workstation, such as the SGI WebFORCE, the benchmarks show that it will sustain up to 96 requests/sec [63]. Note that there 86,400 seconds per day, so this type of workstation could support 8.3 million requests per day if they are uniformly distributed, and 345 K requests per hour (if one considers peak times most important).

We will assume that the server will not do anything more than retrieving and sending the required documents, and the server will not support querying or searching the data stored on the server since this will add extra load. We will try to eliminate the bottlenecks one after another and see how we can achieve the needed bandwidth to deliver video of 640x480 resolution at 30 frames/sec.

We will assume that all the requests that arrive at the server will result in a document transfer, in other words there is no proxy or client caching. We will include compression as a factor, and will assume that the compression ratio cm is 0.5 for data types other than video or audio data types.

We will start by assuming that the server can support 96 requests/sec with no errors [63]. For this scenario we will use two cases from the previous scenarios as base cases. The first base case is the same base case for scenario 1, where we concluded that 1942 simultaneous clients can be supported. This case represents the currently measured values for the previous parameters. The second base case will use the values shown in the last raw of Table 7.4 since this represents our upper bound on video resolution, audio quality, usage, and utilization. Using equation 7.3 for the first base case we get:

$$C_t = .125 * .124 * [(.04 * 468262) + (.18 * 8796) + (.78 * 40960)] * .5 = 405 \text{ b/sec per client.}$$

and for the second base case we get:

$$C_t = .36 * .124 * [(.4 * 7492176) + (.18 * 87960) + (.42 * 40960)] * .5 = 67628 \text{ b/sec per client.}$$

Starting from 96 clients accessing the workstation at the same time, Tables 7.6 and 7.7 show what happens when we increase the number of clients in multiples of 10.

Table **??** shows that with the current media mix and demand of digital material (base case 0 from Table 7.4), with a T3 link the server can support up to 96000 clients. The bottleneck in this configuration is the server; scaling it up requires 1000 server side caches. The price of such a caching system will be beyond the reach of most enterprises. This result shows that the server hardware will be a major future bottleneck. To solve this problem we should consider ideas such as data migration or replicating the server in other places

Table 7.6: The bottlenecks from increasing the number of clients by multiples of 10 for the first base case of Table 7.4

Clients	Server Connection T1/T3/FDDI	Transfer Rate MB/sec	Number of Server Caches	Bottleneck
96	T1	.037	1	T1
960	T1	.37	1	server
9600	T1	3.7	10	T1 and server
96000	T3	37	100	server

where we anticipate lots of access activity. This solution will allow distributing the cost of the server replicas to other enterprises.

Table 7.7 shows that if servers are to support high quality real time video in the future (base case number 5 in Table 7.4) they have to have very high speed networks (greater than 1GB/sec) and they should implement server caching.

The last base case in Table 7.4 can generate lots of problems in the future since it will require lots processing and I/O speed to deliver different streams with adequate QoS. There is lots of research on how to design disks and layout media streams for fast media delivery [].

Table 7.7: The bottlenecks from increasing the number of clients by multiples of 10 for the first base case of Table 7.4

Clients	Server Connection T1/T3/FDDI	transfer Rate MB/sec	Number of server Caches	Bottleneck
96	T1	6.2	1	T1
960	T3	61.9	1	T3 and server
9600	FDDI	619	10	FDDI and server
96000	1GB/sec	6190	100	network and server

7.4 Conclusions and Future Research

To achieve scalability we should use a combination of:

- client caching,
- proxy caching,
- server caching,
- data compression and delta encoding,
- HTTP protocol modifications to support some of these ideas,
- and increasing the network bandwidth.

The demand for high network bandwidth will increase and this will trigger more Web usage again introducing more demand for bandwidth. To be able to transfer real time video with high quality we need GB/sec network connections especially to connect popular servers. Clients can live with 10MB/sec or 1MB/sec depending on how many clients will share the network connection. Thus Virginia Tech's move to switched ethernet for office connections seem adequate for typical requirements.

The previous scenarios show that scalability of the Web is achievable; however it is not an easy or cheap process. Future research should focus on constructing better and more detailed models. Modeling is an important concept in designing huge and expensive systems and it is of great help in designing infrastructure for the Web.

Traffic modeling is an important research area. WWW traffic is hard to characterize because the Web has so many user communities that differ in their use of the Web. We have shown in chapters 3, 5, and 6 that we can find commonalities in the behavior and we have utilized that to recommend caching based on the spatial and temporal locality of reference. We think, however, that it is important to characterize the differences in the usage behavior and find out methods to utilize the differences to recommend better architectures.

Efforts to log and monitor Web traffic should continue, since the WWW is changing rapidly and the assumptions that might work for today might not be valid tomorrow. It is very

important to identify the factors that will shape the Web in the future, observe the rate of Web growth, observe users' behavior and see how behavior changes with the new technology.

Chapter 8

Conclusions and Future Work

There were four main objectives behind the work described in this dissertation

- identify and give a taxonomy of sources of Web traffic needed for caching studies,
- characterize Web traffic to proxy servers and test for web traffic self-similarity and long range dependency,
- characterize search sessions and accesses to Web IRS, and a course ware digital library server, and
- define scenarios to be used for forecasting, in the test for the new HTTP-NG protocol, and performance evaluation.

All these objectives have been achieved. Currently, we have a huge collection of proxy and web traffic that represent a diverse user population. The logs occupy over 40 GB of storage in compressed format. The collection process involved working with tools such as tcpdump and snoop; writing programs in perl and C to filter and encode the data. Several problems have risen during the data collection process, for example, to handle the user anonymity traffic collected was processed and each client address was replaced by unique identifier. All the processed files are mapped to a single data-base to keep clients mapping consistent between different files.

Setting up and maintaining the machines to collect the data was another obstacle. Data files grow in size very fast and has to be transferred to the main store as fast as possible. This led us to come up with a more automated logging process [**?**] where we use scripts to move the files from the collection machines to the main store every week. Maintaining the hardware running was another problem that required immediate attention. One reason for this strict continuous data collection is to maintain a set of log files that can be analyzed for historical changes, such as changes of file types or sizes over time.

Using the taxonomy in chapter 2 we split the traffic collected from the LAN into different classes to be analyzed. In some cases we managed to extract a missing week of the EI server log file from the network log files and using equations 2.1 and 2.2.

In chapter 3 we identified a set of invariants that hold true across different proxy workloads and one of the identified invariants was self- similarity. We also showed that some of the identified invariants hold for the VT-CS workload overtime. One of our observations is that most of the accesses go to a small set of servers and URLs. In addition, in the majority of the workloads, a small set of clients are responsible for most of the accesses.

We explored the sources for self-similarity in the VT-CS data and our conclusion is that a major source for self-similarity can be attributed to the periodic behavior of Web users, the protocol, and the structure of Web documents in addition to the distribution of file sizes. The periodic behavior appeared at minute, hour, day and week intervals. Accesses to Web proxies are highly correlated from day to day and from one week to another. A key source for this correlation is the daily and weekly schedule of Web users. Users who share the same time zone should have similar schedules and hence part of their accesses should reflect it. This result is important for commercial proxies since they will act similar to a local *broadcasting* station or local cable company that distributes information and other kinds of media. Proxies will be installed to support a group of users in a certain locality. Peak and minimum usage hours can be identified for upgrades, price commercials, and other time dependent jobs.

The results of this characterization were used by Motorola and the W3C who described our proxy characterization as a comprehensive study for proxies.

The fact that Web traffic has a significant deterministic and cyclic component is very important for internet service providers. This fact can help in the efforts to smooth and distribute the traffic to the service provider proxy by advertising the low activity times and having lower pricing rates then to encourage and attract more users. Internet network protocol designers can use this information to support pre-fetching and filtering of data based on the daily and hourly level of activity and locality of reference of a group of clients connected to a certain proxy. We can use this discovered temporal and spatial locality in the Web traffic to support a distribution model that can be used to push data to caches where it is anticipated that these files will be accessed.

Some of those ideas will be utilized to build a smart caching systems that will benefit from the previous identified characteristics to distribute and validate cache documents before it is accessed.

In a study to characterize WWW latency we found that a latency displays a periodic behavior and its peaks coincides with the peaks of request rate to proxies from our server. This suggests that the latency is due to the network not servers. This provides an insight to enhance the formula suggested by our group to use a replacement algorithm based on network delays.

The discovered periodic correlation indicates that queuing and statistical models are not the appropriate techniques to model such systems. Simulations need more accurate models that

represent the characteristics of the Web traffic. We introduce a new modeling approach that captures the real characteristics of the Web traffic by using a combination of Fourier analysis and the traditional techniques of statistical analysis. The new approach characterizes the random and the deterministic parts of the data. The generated data displays long-range dependent and self-similar characteristics comparable to the ones observed in the original data. We demonstrate the modeling approach with an example and we validate each step by comparing the generated data to the original.

In Appendix A we show that the periodic behavior exists for the ethernet traffic, however, the complete modeling is beyond the scope of the work described here.

As part of characterizing specific WWW activities we provided a characterization to the our course ware digital library server and accesses to Web IRS. In addition of the full characterization described in chapter 5 for the digital library server, we found that paperlessness, number of students attending the class, and the faculty are significant parameters in predicting accesses to EI. Faculty access the server a lot before the start of the semester during the authoring and course preparation phase. Accesses after that are mainly to add, fix, and post information.

With respect to characterization accesses to Web IRS we found that a small percentage of the clients are responsible for most of the accesses to the Web IRS. In particular less than 50% of the clients are responsible for more than 80% of the accesses. Clients rarely use

the special additional features available through the Web IRS interface. It appears that users prefer to issue queries with one term and without Boolean operators. By looking at the access patterns in the sessions we found that the most frequent repeated pattern is browsing only. Using regression and correlation analysis we found that when a session involves more searching, then less network and server bandwidth is used.

The demand for high network bandwidth will increase and this will trigger more Web usage again introducing more demand for bandwidth. To be able to transfer real time video with high quality we need GB/sec network connections especially to connect popular servers. Clients can live with 10MB/sec or 1MB/sec depending on how many clients will share the network connection. Thus Virginia Tech's move to switched ethernet for office connections seem adequate for typical requirements.

The work in this dissertation show that scalability of the Web is achievable; however it is not an easy or cheap process. Future research should focus on constructing better and more detailed models. Modeling is an important concept in designing huge and expensive systems and it is of great help in designing infrastructure for the Web.

8.1 Future Work

Traffic modeling is an important research area. WWW traffic is hard to characterize because the Web has so many user communities that differ in their use of the Web. We have shown in chapters 3, 5, and 6 that we can find commonalities in the behavior and we have utilized that to recommend caching based on the spatial and temporal locality of reference. We think, however, that it is important to characterize the differences in the usage behavior and find out methods to utilize the differences to recommend better architectures.

Efforts to log and monitor Web traffic should continue, since the WWW is changing rapidly and the assumptions that might work for today might not be valid tomorrow. It is very important to identify the factors that will shape the Web in the future, observe the rate of Web growth, observe users' behavior and see how behavior changes with the new technology.

This study shows understanding the dynamics of the current architecture gives insight to some of the problems that we are currently facing. Previously the belive was that servers are the main bottleneck in WWW architecture, however, our results imply that the network is the major source of the delay. More studies to characterize the exact point where the delay appears on the internet is important. Initial hypothesis thinks that the source of delay comes from the connection provided by the ISP.

Bibliography

[1] H. Braun and K. Claffy. Web traffic characterization: An assessment of the impact of caching documents from NCSA's Web server. In *Proc. 2nd Int. WWW Conference*, Chicago, October 1994.

[2] R. Wooster and M. Abrams. Proxy Caching that Estimates Page Load Delays. In *6th International World-wide Web Conference*, Santa Clara, California, April 1997.

[3] V. N. Padmanabhan and J. C. Mogul. Improving HTTP latency. In *Second World-wide Web Conference '94: Mosaic and the Web*, 1994. http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/mogul/HTTPLatency.html.

[4] S. E. Spero. Analysis of HTTP performance problems, July 1994. http://elanor.oit.unc.edu/http-prob.html.

[5] J. Gettys. HTTP problem statement. <URL: http://www.w3.org/Protocols/HTTP-NG/951005Problem.html>, 1995. W3C.

[6] B. Janssen. PARC HTTP-NG project. <URL: http://www.parc.xerox.com/istl/projects/http-ng/>, 1997. Xerox.

[7] H. F. Nielsen and J. Gettys. HTTP-NG - the next generation. <URL: http://www.w3.org/Protocols/HTTP-NG/Activity.html>, 1997. W3C.

[8] T. Winograd. The proxy is where it's at. Technical Report CS-TN-97-51, Computer Sci. Dept., Stanford Univ., Stanford Univ., CA., February 1997.

[9] B. Liu. World Wide Web latency and proxy caching. Master's thesis, Computer Sci. Dept., Virginia Tech, 1998. Thesis in Progress.

[10] H. Maurer. *Hyper-G now Hyperwave, the Next Generation Web Solution.* Addison Wesley, 2725 Sand Hall Road, Menlo Park, CA 94025, 1st edition, 1996.

[11] G. Abdulla, W. S. Heagy, and E. A. Fox. Quantitative analysis and visualization regarding interactive learning with a digital library in computer science. In *Poster in Proc. of 2nd ACM International Conference on Digital Libraries*, Philadelphia, PA, July 1997. ACM.

[12] T. T. Kwan, R. E. McGrath, and D. A. Reed. User access patterns to NCSA's World-Wide Web server. Technical Report UIUCDCS-R-95-1934, Dept. of Comp. Sci., Univ. of IL, February 1995.

[13] A. Luotonen and K. Altis. World-Wide Web proxies. *Computer Networks and ISDN Systems*, 27(2), 1994. <URL: http://www1.cern.ch/PapersWWW94/luotonen.ps>.

[14] R. Malpani, J. Lorch, and D. Berger. Making World Wide Web caching servers cooperate. In *4th International World-wide Web Conference*, pages 107–117, Boston, December 1995.

[15] G. Abdulla, B. Liu, R. saad, and E. A. Fox. Characterizing world wide web queries. Technical Report TR-97-04, Computer Sci. Dept., Virginia Tech, March 1997.

[16] H. F. Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, a. Wium Lie, and C. Lilley. Network performance effects of http/1.1, css1, and png. In *Proc. SIGCOMM97*, Cannes, French Riviera, FRANCE, September 1997. ACM.

[17] M. Abrams, C. R. Standridge, G. Abdulla, S. Williams, and E. A. Fox. Caching proxies: Limitations and potentials. In *4th International World-wide Web Conference*, pages 119–133, Boston, December 1995. <URL: http://ei.cs.vt.edu/~succeed-/WWW4/WWW4.html>.

[18] S. Williams. HTTP: Delta-Encoding Notes, January 1997. <URL: http:-//ei.cs.vt.edu/ williams/DIFF/prelim.html>.

[19] S. Williams, M. Abrams, C. R. Standridge, G. Abdulla, and E. A. Fox. Removal Policies in Network Caches for World-Wide Web Documents. In *ACM SIGCOMM'96*

Conference, pages 293–305, Stanford University, California, August 1996. <URL: http://ei.cs.vt.edu/~succeed/96sigcomm/96sigcomm.html>.

[20] J. C. Mogul, F. Douglis, A. Feldman, and B. Krishnamurthy. Potential benefits of delta encoding and data compression for HTTP. In *pSIGCOMM*, Cannes, France, September 1997.

[21] A. V. Hoff and J. Payne. Generic Diff Format Specification. <URL: http://www.w3.org/TR/NOTE-gdiff-19970901>, 1997. W3C.

[22] G. Abdulla, E. A. Fox, M. Abrams, and Stephen Williams. WWW Proxy Traffic Characterization with Application to Caching. Technical Report TR-97-03, Computer Sci. Dept., Virginia Tech, March 1997.

[23] G. Abdulla, E. A. Fox, and M. Abrams. Shared User Behavior on the World Wide Web. In *Proc. of WebNet97*, Toronto, Canada, November 1997. AACE.

[24] G. Abdulla, A. H. Nayfeh, and E. A. Fox. Modeling Correlated Proxy Web Traffic Using Fourier Analysis. Technical Report TR-97-19, Computer Sci. Dept., Virginia Tech, November 1997.

[25] J. Pitkow. Summary of WWW characterization group, 1998. Unpublished Document.

[26] W. Richard Stevens. *TCP/IP Illustrated, Volume I: The Protocols*. Addison Wesley, 2725 Sand Hall Road, Menlo Park, CA 94025, 1994.

[27] M. Abrams and Stephen Williams. Complementing surveying and demographics with automated network monitoring. *World Wide Web*, 1(3):101–119, July 1996.

[28] M. F. Arlitt and C. L. Williamson. Web server workload characterization: The search for invariants. In *Proc. SIGMETRICS*, pages 126–137, Philadelphia, PA, April 1996. ACM.

[29] T. Kwan, R. McGrath, and D. Reed. NCSA's World Wide Web server: Design and performance. *IEEE Computer*, 28(11):68–74, November 1995.

[30] N. Smith. What can archives offer the World-wide Web. In *Proc. 1st World-wide Web Conference*, Geneva, May 1994. <URL: http://www.hensa.ac.uk/www94>.

[31] D. O'Callaghan. A central caching proxy server for WWW users at the University of Melbourne. In *First Australian World-wide Web Conference*, University of Melbourne, Australia, 1995.

[32] M. E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic evidence and possible causes. In *Proc. SIGMETRICS*, pages 160–169, Philadelphia, PA, May 1996. ACM.

[33] C. R. Cunha, A. Bestavros, and M. E. Crovella. Characteristics of WWW client-based traces. Technical Report BT-CS-95-010, Dept. of Comp. Sci., Boston Univ., Boston, MA 02215, July 1995.

[34] S. Glassman. A caching relay for the World-wide Web. *Computer Networks and ISDN Systems*, 27(2):165–173, 1994. <URL: http://www1.cern.ch/PapersWWW94/steveg.ps>.

[35] J. Sedayao. Mosaic will kill my network. In *Second World-wide Web Conference '94: Mosaic and the Web*, Chicago, IL, October 1994.

[36] J. Gwertzman and M. Seltzer. World Wide Web cache consistency. In *Proc. of the 1996 Usenix Technical Conference*, Boston, MA, 1996.

[37] S. D. Gribble and E. A. Brewer. System design issues for internet middleware services: Deduction from a large client trace. In *Submitted to the 1997 Usenix Symposium on Internet Technologies and System*, Monterey, California, 1997.

[38] P. Barford and M. Crovella. An Architecture for a WWW Workload Generator. In *Proc. SIGMETRICS*, 1998. to appear.

[39] J. Beran. *Statistics for Long-memory Process*. Chapman and Hall, New York, 1994.

[40] Peter B. Danzig and Sugih Jamin. tcplib: A library of tcp intenetwork traffic characteristics. Technical Report USC-CS-91-495, Computer Science Department, University of Southern California, 1991.

[41] V. Paxson and S. Floyd. Wide-area traffic: The failure of Poisson modeling. In *Proc. SIGCOMM94*, pages 257–268, London, August 1994. ACM.

[42] V. Paxson. Fast, approximate synthesis of fractional gaussian noise for generating self-similar network traffic. In *CCR, submitted for publication*, 1997.

[43] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, February 1994.

[44] W. E. Leland and D. V. Wilson. High time-resolution measurement and analysis of LAN traffic: Implications for LAN interconnection. In *Proc. of IEEE Infocomm '91*, pages 1360–1366, Bar Harbour, FL, 1991. IEEE.

[45] M. F. Arlitt. A performance study of internet web servers. Master's thesis, University of Saskatchewan, Saskatoon, Saskatchewan, May 1996.

[46] StatSci. *S-PLUS Users's Manual, Version 3.2*. MathSoft Inc., Seattle, Washington, December 1997.

[47] Mathworks. *Matlab Users's Manual, Version 5.1*. Mathworks Inc., Natick, MA 01760, 1997.

[48] A. H. Nayfeh and B. Balachandran. *Applied Nonlinear Dynamics*. John Wiley, New York, 1995.

[49] George E. P. Box and Gwilym M. Jenkins Gregory C. Reinsel. *Time Series Analysis*. Prentice-Hall, London, 1994.

[50] P. Bloomfield. *Fourier Analysis of Time Series: an Introduction*. John Wiley, New York, 1976.

[51] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, New York, 2nd edition, 1991.

[52] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley, New York, 1991.

[53] S. G. Vincent and A. M. Law. UniFit II: Total support for simulation input modeling. In *Proc. Winter Simulation Conference*, pages 371–376, Arlington, VA, December 1992.

[54] Open University and University of Hull. Evaluation methods and procedures for studying learners' use of media. http://www-iet.open.ac.uk/iet/PLUM/plum.html, 1996.

[55] E. A. Fox and N. D. Barnette. Improving education through a computer science digital library with three types of WWW servers. In *Second World-wide Web Conference '94: Mosaic and the Web*, 1994. <URL: http://ei.cs.vt.edu/papers/WWW94.html>.

[56] S. manley and M. Seltzer. Web facts and fantasy. In *Proceedings of the 1997 USENIX Symposium on Internet Technologies and Systems*, Monterey, Canada, December 1997.

[57] Christine L. Borgman, Sandra G. Hirsh, and John Hiller. Rethinking online monitoring methods for information retrieval systems. *Journal of the American Society for Information Science*, 47(7):568–583, July 1996.

[58] Carsten Schlichting and Erik Nilsen. Signal detection analysis of WWW search engines. In *Designing for the Web Empirical Studies*. Microsoft, October 1996. <URL: http://www.microsoft.com/usability/webconf.htm >.

[59] Annabel Pollock and Andrew Hockley. What's wrong with internet searching. In *Designing for the Web Empirical Studies*. Microsoft, October 1996. <URL: http://www.microsoft.com/usability/webconf.htm >.

[60] Averill M. Law and Stephen Vincent. *ExpertFit User's Guide*. Averill M. Law and Associates, Tucson, AZ 85717, August 1995.

[61] P. L. Meyer. *Introductory Probability and Statistical Applications*. Addison Wesley, 2725 Sand Hall Road, Menlo Park, CA 94025, 2nd edition, 1970.

[62] M. Abrams, S. Williams, G. Abdulla, S. Patel, R. Ribler, and E. A. Fox. Multimedia traffic analysis using Chitra95. In *Proc. ACM Multimedia '95*, pages 267–276, San Francisco, November 1995. ACM.

[63] M. Blakeley. Webstone performance analysis: Sun netra i20. <URL:http://www.sgi.com/Products/WebFORCE/WebStone/sun-ss20/sun-ss20.html>, December 1995. SGI.

[64] T. S. Rao. Analysis of nonlinear time series (and chaos) by bispectral methods. In *Workshop on Nonlinear Modeling and Forecasting*, volume Studies in the Science of Complexity, pages 199–226, Santa Fe, NM, September 1990. Santa Fe Institute.

Appendix A

Ethernet Traffic

Figure A.1 is a graph of the auto-correlation function for the Bell Core ethernet data after taking the log of the data. By examining the figure we notice that it also has a periodic auto-correlation although it has a different time scale (milliseconds). In the WWW traffic we attribute The periodic signal to the periodic behavior of WWW users work habits or schedules. Often a user accesses a page that might have several embedded images, or pages, pause for a while then follow a link from the current page or start a new access. Users access the Web most of the time according to the normal live schedule, for example they access it during the day and they stop during the night (there are exceptions but, however they do not alter the general behavior of Web users).

For the ethernet traffic the periodic signal in Figure A.1 cannot be attributed easily the

user behavior since they appear order of milliseconds. They might result from a machine or network cycles. These periods, however, of on/off behavior seem to be deterministic and cyclic. As a result we argue that modeling such behavior should be done with a different method other than the suggested statistical on/off periods used in the literature. We think that the on/off statistical model cannot produce the deterministic and periodic signal.

We used a moving average program to model the a moving average of the data; We then subtracted it from the original data. We plotted time trace and the ACF for the data after the subtracting the mean. The results are shown in Figures A.2 and A.3.

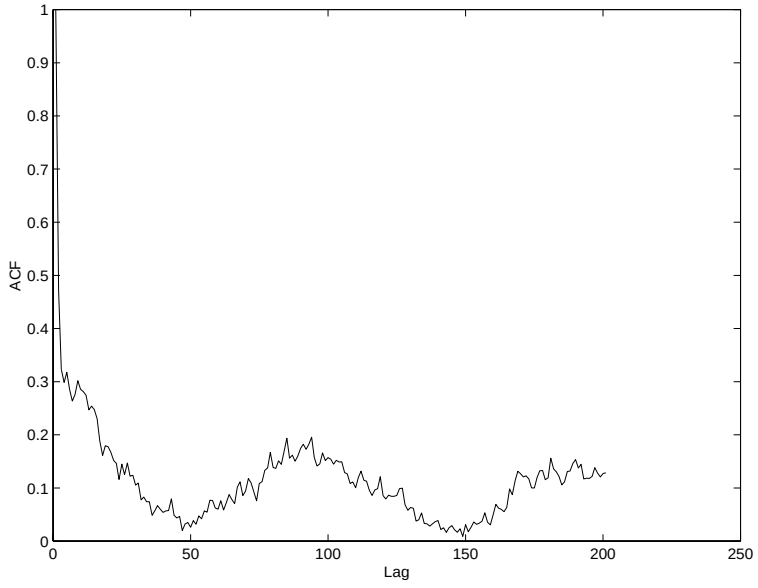


Figure A.1: ACF for the ethernet traffic collected from Bell core

The time trace clearly shows that still there are bursty periods, hence, the data might be

self-similar. To confirm this we computed the Hurst parameter, using the R/S test, for the resulted data. The test confirmed that the data has a Hurst parameter value of 0.7 which is greater than 0.5 and this implies that the traffic is self-similar. Figure A.3, however, shows that the data is not long range dependent. The aut-correlation dies and and goes to zero and there are no cycles.

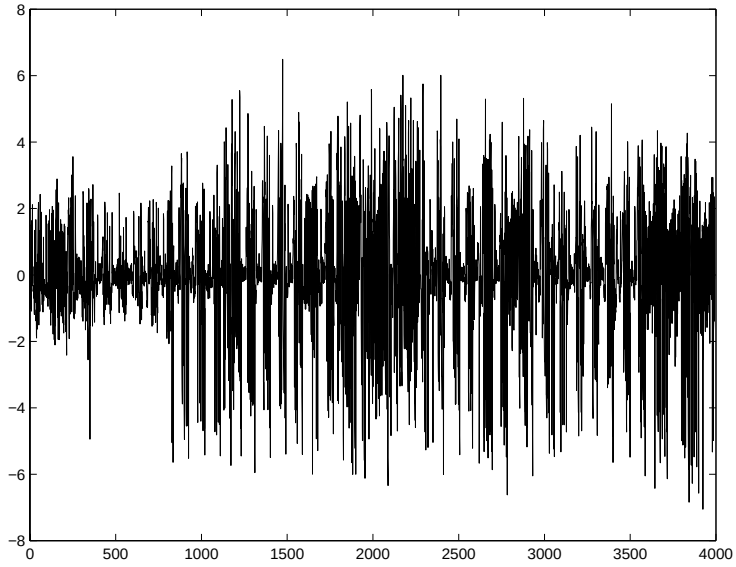


Figure A.2: ACF for the ethernet traffic collected from Bell core

From this test we conclude that a self-similar traffic can be long range dependent or independent. Modeling self similar data with on/off statistical models will not capture the long range dependency and the cyclic behavior of the traffic and hence the model will not represent the real characteristics of the data.

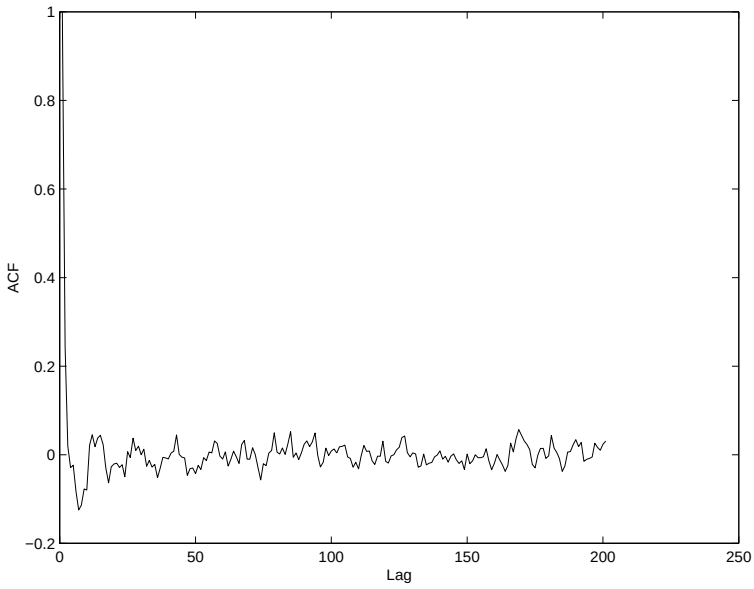


Figure A.3: ACF for the ethernet traffic collected from Bell core

Alternative methods based on our approach in chapter 4 should be developed to model such data. For example the work by Rao described in [64] provides extended techniques to model data with similar characteristics.

Vita

Abdulla was born on a sunny day under the clouds :)...