MEASUREMENT AND ANALYSIS OF TRAFFIC IN A HYBRID SATELLITE-TERRESTRIAL NETWORK

by

Qing Shao
B.Eng., Beijing University of Posts and Telecommunications, China, 1993

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

In the
School
of
Engineering Science

© Qing Shao 2004

SIMON FRASER UNIVERSITY

April 2004

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without permission of the author.

APPROVAL

Name: **Qing Shao**

Degree: **Master of Applied Science**

Title of Thesis: **Measurement and Analysis of Traffic in a Hybrid Satellite-Terrestrial Network**

Examining Committee:

Chair: **Dr. Mirza Faisal Beg**
Assistant Professor, School of Engineering Science

Dr. Ljiljana Trajkovic
Senior Supervisor
Professor, School of Engineering Science

Dr. William A. Gruver
Supervisor
Professor, School of Engineering Science

Dr. R. H. Stephen Hardy
Examiner
Professor, School of Engineering Science

Date Approved: _____

ABSTRACT

Measurement and analysis of traffic traces are important for better understanding of network behaviour. In this research, we collected traffic traces from a hybrid satellite-terrestrial network operated by ChinaSat, a commercial satellite Internet service provider.

We performed traffic analysis on packet, connection, protocol, and application layers. We investigated self-similar and long-range dependent characteristics of the collected traffic traces by employing various estimators. A traffic model on the TCP connection level was proposed. The relative fitness of various distributions functions was investigated. We concluded that Weibull and lognormal distributions are ideal for modelling the TCP inter arrival times and the number of downloaded bytes, respectively.

We then used billing records data to obtain one-week traffic prediction using autoregressive integrated moving average model. We implemented and evaluated the predictability of the redundant wavelet transform and combined it with autoregressive model to perform short-term traffic prediction.

DEDICATION

To my family for supporting me throughout my graduate studies.

ACKNOWLEDGEMENTS

I would like to thank my senior supervisor, Dr. Ljiljana Trajkovic for her support, guidance, and knowledge and for giving me the opportunity to be a member of the Communication Networks Laboratory. I want to acknowledge all the members in the CNL for their help and for being both great friends and colleagues.

I would like to express my thanks to Dr. Hardy and Dr. Gruver for serving on my committee. Thanks for your precious time and effort to review my thesis and for your valuable comments.

My gratitude also goes to my old friend Dr. Zhiqiang Bi who introduced me to the analysis of DGX distribution.

I am also very grateful for the assistance of Yongxin Shi and Rui Huang, managers of the DirecPC system, ChinaSat, who helped me assemble the hardware used to collect the traffic traces. Without your help, this research would not be possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND ACRONYMS

ACF	AutoCorrelation Function
ARIMA	AutoRegressive Integrated Moving Average
DGX	Discrete Gaussian Exponential
DWT	Discrete Wavelet Transform
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
ISP	Internet Service Provider
LRD	Long-range Dependence
NMSE	Normalized Mean Squared Error
NOC	Network Operation Centre
PACF	Partial AutoCorrelation Function
POP3	Post Office Protocol 3
RFC	Request for Comments
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WSS	Wide Sense Stationary

CHAPTER 1 INTRODUCTION

1.1 Motivation and significance

Internet has continued to grow and change over the last decade. This evolution has been accompanied by the growth of traffic volume, the development of new protocols, and a variety of new Internet access technologies. Network traffic measurements are useful for network troubleshooting, workload characterization, and network performance evaluation. In order to detect the invariants in a rather dynamic traffic structure, measurement and analysis of genuine network traffic traces play an important and continuing role. In general, there are three steps in traffic analysis: traffic collection, analysis, and synthesis or prediction, as shown in Figure 1.1. Our research encompasses all three steps.

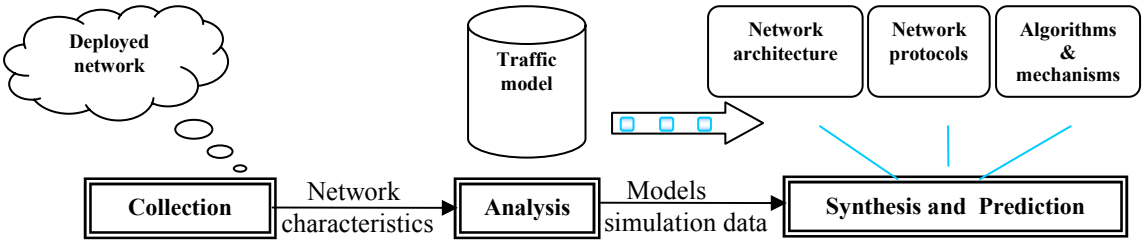


Figure 1.1 Methodology for traffic measurement and analysis.

Collection and characterization of the terrestrial Internet traffic have received considerable attention during the past decade [20], [31]. Numerous Web sites offer collected samples of Internet traffic traces [30]. To the contrary, few traffic traces have

been collected from wireless or satellite commercial sites. Models based on the past data may not accurately capture the current network behaviour. In addition, Internet traffic, which is constantly changing over time both in volume and in statistical properties, demands analysis that should be based on most recently collected data. In this research, we collected traffic from a hybrid satellite-terrestrial network that is operated by ChinaSat, a commercial satellite Internet service provider.

1.2 Objectives of this thesis

The objective of this thesis is to analyze and model the collected traffic traces, to characterize the underlying statistical processes and distributions, to implement a newly proposed wavelet approach to perform traffic prediction, and to investigate the predictability of this wavelet combined predictor under various predictor lengths.

1.3 Organization of the thesis

This thesis is organized as follows: In Chapter 2, we briefly describe the DirecPC satellite system and the traffic trace collection. Chapter 3 presents the statistical analysis and results. Chapter 4 describes the self-similarity study. Chapter 5 introduces the TCP-connection model. In Chapter 6, we describe fitting ARIMA model to the billing records data. We also describe implementation of the redundant wavelet model and performance evaluation of the long and short-term traffic prediction. Finally, we conclude with Chapter 7.

CHAPTER 2 TRAFFIC COLLECTION

In this Chapter, we first introduce the DirecPC system and then describe the traffic trace collection. We also clarify the data format.

2.1 Introduction to the DirecPC system

The advantage of satellite systems is that they broadcast information to large geographical regions without a need to solve the last mile access problem. In this thesis, we analyze traffic from a system called DirecPC. DirecPC is a satellite one-way, broadcast system offering three types of services to Intel x86 or Pentium™ based servers and workstations. The system is designed and manufactured by Hughes Network System company (HNS), which is the largest satellite communication vendor in the world. DirecPC system operates worldwide. Bell Express-Vu provides this service in Eastern Canada. DirecPC's three services are: Turbo Internet, Digital Package Delivery, and PC Multicast. In this thesis, we consider only the Turbo Internet service.

2.2 Turbo Internet system

The Turbo Internet system is designed to provide high speed Internet access for homes or enterprises. ChinaSat, the largest satellite communication service provider in China, operates DirecPC system to provide Internet access to over 200 Internet cafés across provinces. Each café offers on average 40 PCs to its customers surfing the Internet. As shown in Figure 2.1, a proxy server enables the PCs to simultaneously share the Internet access through a single connection account. This system utilizes two special techniques: IP spoofing and TCP splitting.

When a customer browses a website in the café, a request is sent through a terrestrial dial up modem to a local internet service provider (ISP). However, before the request leaves the customer's PC, the DirecPC software, which is installed on the proxy server, automatically attaches a "tunnelling header" (an electronic addressing mask) to the requested website so called unique resource locator (URL). This “tunnelling code” instructs the ISP to forward the URL request to the DirecPC Network Operations Centre (NOC) instead to the requested URL directly. Once the NOC receives the customer's request, the tunnelling header is removed and the request is forwarded by a high-speed link to the Internet. When the desired content is retrieved, the NOC sends the information to the DirecPC satellite that beams this information down to the DirecPC receiving system at the café’s network. This special routing technique is called IP spoofing.

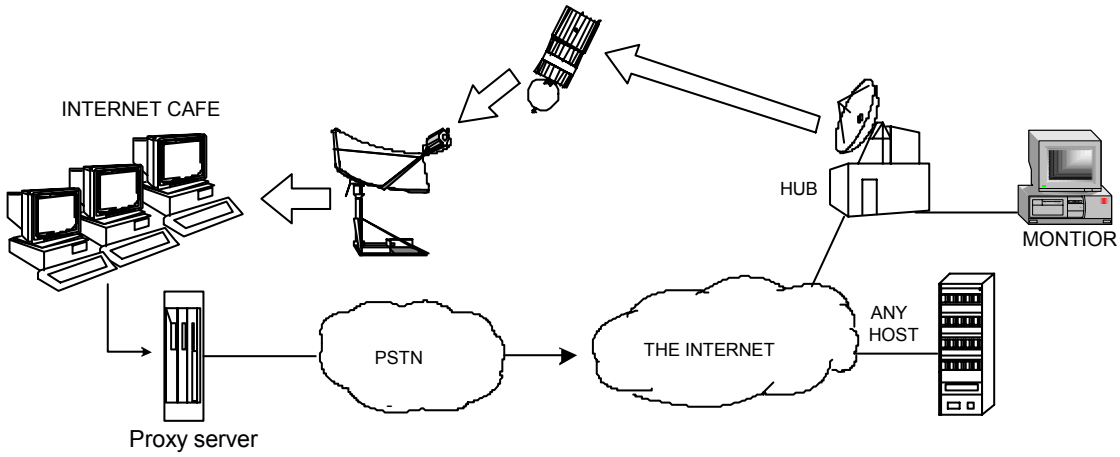


Figure 2.1 Elements of a hybrid satellite-terrestrial network with a monitoring location for traffic collection.

The link delay from the hub to the receiving station is ~ 250 ms because the satellite is located in a geostationary orbit. This delay causes a large delay-bandwidth product. In satellite systems employing TCP, throughput is related to the TCP window

size, which defines the number of packets that could be sent without being acknowledged. Hence, conventional TCP, with its relatively small window size, will exhibit poor performance in the case of networks with large delay-bandwidth product. The DirecPC system addresses this problem by employing a TCP splitting technique: the hub first acknowledges the downloaded packets from the Internet on behalf of the remote stations, and then it delivers these packets over the satellite links using a modified TCP with an enlarged window size [15].

The IP spoofing and TCP splitting techniques impose considerable memory requirements on edge routers. Whenever the hub sends downloaded data to the remote stations, a copy must be maintained in a retransmit buffer until acknowledgements from the remote stations are received. In this system, the available memory is shared equally among all connections.

ChinaSat operates this system at 10 Mbps information rate, the modulated carrier has ~ 12 MHz bandwidth, which is approximately one third of a satellite Ku band transponder. This capacity can be expanded as the bandwidth requirement increases. The router at the hub acts as an edge router between the satellite network and rest of the Internet. The connection between the hub and the Internet is set to 10 Mbps because the maximum throughput of the system is limited by the satellite carrier.

2.3 Traffic collection

There are many traffic collection programs available [14]. Many previous traffic traces were collected using the *tcpdump*. Several analysis tools for *tcpdump* traces exist. Hence, we chose *tcpdump* as the traffic collecting tool.

Both request and response information are re-routed through the satellite hub. Hence, the satellite hub is an appropriate location to collect traffic traces. The network topology and the measurement point (monitor) are shown in Figures 2.1 and 2.2.

Traffic traces were collected using a Linux PC equipped with a 100 Base-T Ethernet adaptor. The network access point for the trace logging was a port on the primary Cisco router at the network operation centre (NOC). It provided access to the inbound or outbound packets sent between hosts on the NOC LAN using the 10 Mbps Internet connection.

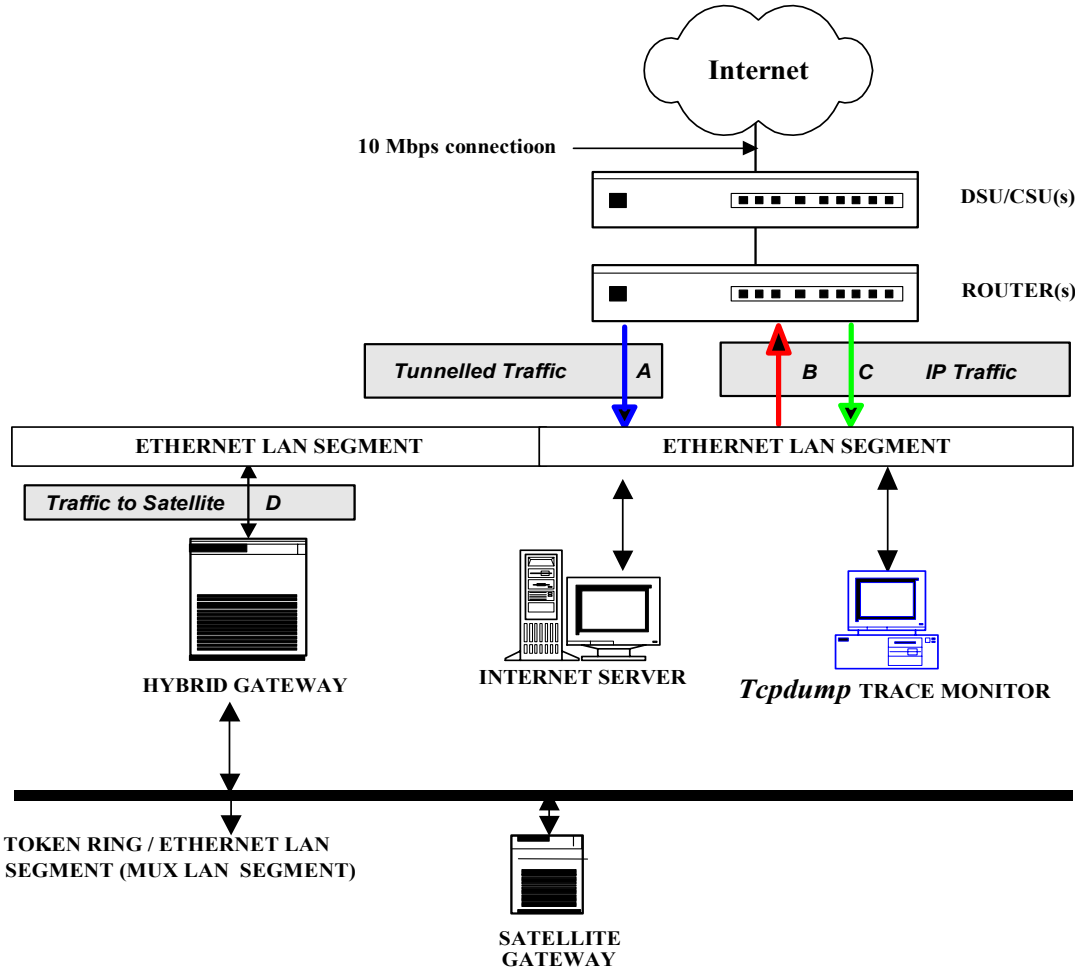


Figure 2.2 Traffic collection point and four types of traffic at the NOC.

We collected traffic traces A, B, and C as shown in Figure 2.2. The path between the hybrid gateway and the satellite gateway is a virtual LAN. We excluded traffic D from the port on the router. Each packet coming from a remote station is logged twice: It was first recorded as traffic A sent from Internet café to the hybrid gateway. It was then recorded again as traffic B passing from the hybrid gateway to the Internet. Traffic C is the downloaded traffic from Internet hosts. In this study, we ignored the tunnelled packets A because they are identical to packets B.

We continuously recorded traffic from 2002-12-14 11:30 AM to 2003-01-06 11:00 AM. 128 dump files contain ~ 63 Gbytes of the data.

2.4 Trace and billing records format

2.4.1 Trace format:

The general format of *tcpdump* data is:

timestamp src > dst: flags data-seqno ack window urgent options

The following is an excerpt from the *tcpdump* trace file:

11:30:39.256592 192.168.2.28.41728 > 61.172.254.140.41820: P 886861:886865(4) ack 4016227587 win 8192 (DF)

11:30:39.257144 192.168.2.30.41236 > 61.187.55.40.ftp-data: . ack 3944213984 win 0

11:30:39.262198 210.51.9.247.www > 192.168.1.242.40987: . 4208499036:4208500496(1460) ack 9307760 win 16427 (DF)

11:30:39.262200 sina35-13.sina.com.cn.7002 > 192.168.1.164.46208: P 3550870364:3550870404(40) ack 2771755 win 63672 (DF)

11:30:39.263719 61.188.177.114.7206 > 192.168.1.54.38329: . ack 718646 win 16652

11:30:39.265184 210.51.9.247.www > 192.168.1.242.40987: . 1460:2920(1460) ack 1 win 16427 (DF)

11:30:39.266887 192.168.1.164.43857 > 61.152.252.164.55901: . ack 3826781594 win 8192

Field name	Description
Timestamp	Indicates when *tcpdump* captured packet.
Src	The source destination IP address and port.
Dst	The destination IP address and port.
Flags	Some combination of S (SYN), F (FIN), P (PUSH), or R (RST) or a single "." (no flags).
Data-seqno	Describes the portion of sequence space covered by the data in this packet. Sequence numbers are of the form <first>: <last> (bytes). "This is interpreted as a packet containing octets for sequence numbers first up to but not including last, which is bytes of user data. For connection setup, the sequence numbers are absolute. After connection setup, the sequence numbers are relative to the sequence numbers established in connection setup" [29].
Ack	The sequence number of the next data expected in the opposite direction on this connection.
Window	The numbers of bytes of receive buffer space available on the opposite direction in this connection.
Urg	Indicates that there is "urgent" data in the packet.

Table 2.1 *Tcpdump* trace format

We only examined traffic B and traffic C, which are the packets sent between the hub and the Internet. Since the satellite hub strips the IP tunnelling header and re-sends the packet to the Internet, the source addresses of the re-sent requests are 192.168.*.* (which represents DirecPC terminal). The destination addresses is the requested websites addresses.

When the Internet hosts respond to these requests, the packets flow back to the hub. The IP addresses of those response packets are 192.168.*.*, while source IP addresses are the websites addresses.

The granularity of the time stamp is an important factor in traffic measurements. In our trace, timestamps have several milliseconds precision even though they seem to have the order of microsecond granularity because there are six decimal digits [24]. The following explains this discrepancy: the timestamp is generated by the Berkeley Packet Filter (BPF) driver from the Linux kernel. It is as accurate as the kernel's clock that has the granularity of microseconds. However, this is not the instance when the network interface of the monitor receives the packet from the network. There is a delay between the time when the packet arrives at the interface and the time when the kernel executes the "new packet arrives" session. Due to this difference, the granularity of the timestamps is limited to the order of several milliseconds [17]. To overcome this problem, specially designed network interface are developed to improve the accuracy of the timestamp. Patches are available to improve the system accuracy [18]. However, our monitoring machine was an online PC and we could not compile its Linux operating system without interfering with the system.

2.4.2 Format of the billing records:

In addition to the traffic trace, we also collected two months of billing records: from the 2002-11-01 to 2003-01-10. The billing system of the DirecPC system creates one file per hour, with a file name DDMMHHMM.BIL, where DD is the day of the month, MM is the month of the year, HH is the hour of the day, and MM is the starting

minute within the hour (usually 00). The records within the file are not sorted by the starting time but rather by the user name.

Field name	Field length	Description
RecLen	5	The length of the record including the new line character. Holds 00100.
RecTyp	3	It is fixed to 001 identifies this data as a Turbo Internet Call Record
SiteID	10	Identifies the subscriber by the unique nonchangable alphanumeric string.
Start	14	Starting time for this call record.
Stop	14	Stopping time for this call record.
Cmin	3	Number of active minutes.
Bill	1	Identifies whether the subscriber is using HNS 800 phone line or the subscriber's own Internet access provider: (1 = HNS phone line, 2 = subscriber's own Internet access provider).
CTxByt	10	The number of bytes transmitted over the space link during this call record. The subscriber is billed based on this field.
CRxByt	10	The number of bytes received from the subscriber.
CTxPkt	10	The number of IP packets transmitted over the space link during this call record.
CRxPkt	10	The number of IP packets received from the subscriber during this call record.

Table 2.2 Billing records format.

An example of the billing records:

00100 001 00045ACA01 20020601000003 20020601010003 059 2 0018256457 0004306379 0000058250 0000061516 0000000000 0000000000

00100 001 00045DBD01 20020601000003 20020601010003 060 2 0008485157 0004041736 0000062401 0000080688 0000000000 0000000000

00100 001 0004493C01 20020601000003 20020601010003 060 2 0016930887 0004538325 0000057763 0000063085 0000000000 0000000000

00100 001 00044DA701 20020601000003 20020601010003 060 2 0010063340 0001300979 0000020724 0000024944 0000000000 0000000000

Since we are interested in the aggregate traffic, the traffic load information is obtained from billing records by aggregating data from individual users. The individual data were obtained from columns: Start, Stop CTxByt, CRxByt, CTxPkt, and CrxPkt. This aggregation generates traffic load information with granularity of one hour. Statistical analysis in Chapter 3 and capacity planning in Chapter 6 are based on this data. In contrast, the *tcpdump* traces have much finer time resolution and contain additional packet header information, which permits a more detailed analysis in Chapters 3, 4, and 5.

CHAPTER 3 TRAFFIC CHARACTERIZATION

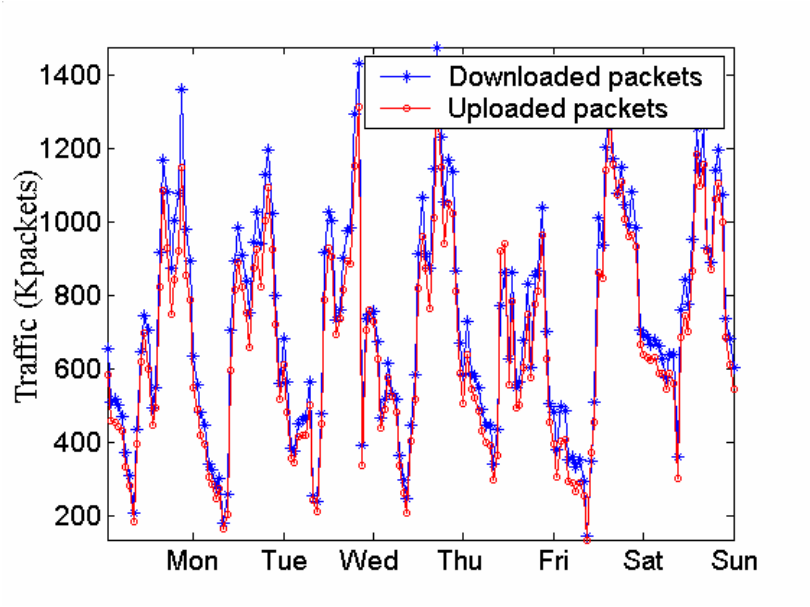
The objective of traffic statistical analysis and modelling is to describe the complex traffic in a concise way. To achieve this goal, models need to capture statistical properties that have important impacts and ignore those that do not. One can categorize traffic on many levels: packet, connection, protocol, and application. In Section 3.1, we analyze the statistical properties of the billing records. We describe the distribution of traffic in terms of the protocols and applications in Section 3.2, analyze the WWW traffic on the TCP connection level in Section 3.3, and characterize packet size in Section 3.4.

3.1 Analysis of billing records: traffic load

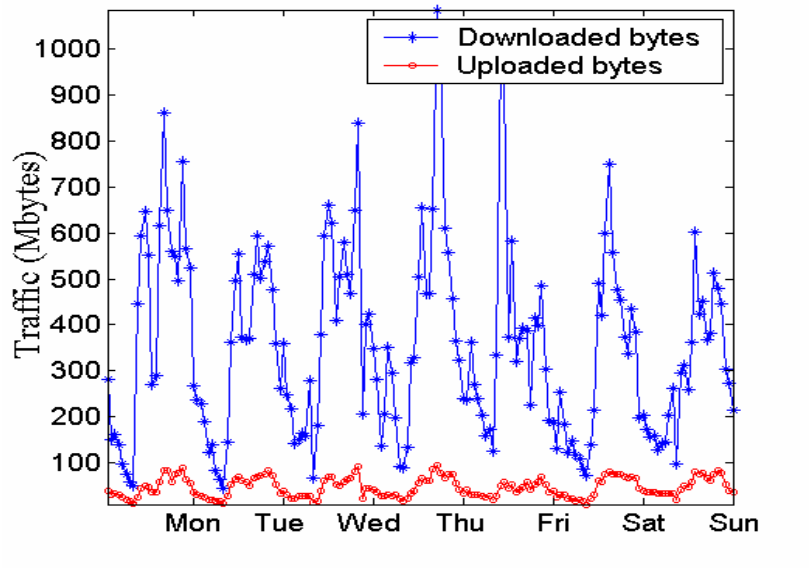
We used aggregated billing records to describe the traffic load in the system. We found that traffic exhibits a diurnal cycle. The number of data packets over time is shown in Figure 3.1 (a). The numbers of uploaded and downloaded packets are almost identical, with the number of downloaded packets being slightly higher. The gap may be explained by the small contribution of User Datagram Protocol (UDP) packets. When the traffic load is expressed in terms of bytes, as shown in Figure 3.2 (b), there is a visible difference between the upload and download directions. This difference indicates the asymmetric characteristic of the transmission.

Figure 3.2 illustrates the average traffic volume during a single day. The asymmetric pattern appears again. The network usage increases at 8:00 and reaches its maximum around 15:00-16:00. It gradually decreases during the night. This result

reflects the customers' behaviour in an Internet café. This observation may help the commercial operators manage the network capacity.

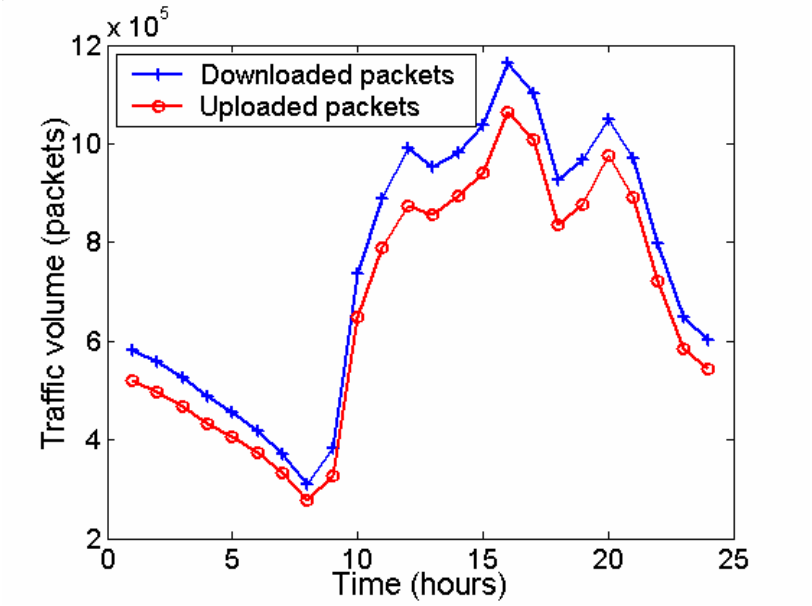


(a)

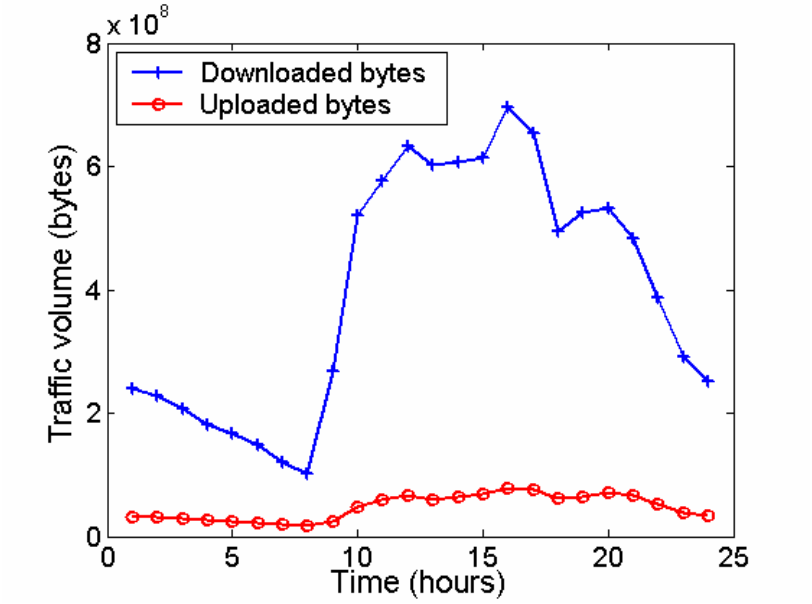


(b)

Figure 3.1 Weekly traffic volume measured in packets (a) and bytes (b). Traffic data was collected from 2002-12-09 to 2002-12-15.



(a)



(b)

Figure 3.2 Average traffic volume over a day measured in packets (a) and bytes (b). Traffic data was collected from 2002-12-09 to 2002-12-15.

3.2 Protocols and applications

All the packets in the collected traffic traces are IP packets. They were collected on the application and transport layers. Data are shown in Tables 3.1 and 3.2.

Protocol	Packet	% of packets	Bytes	% of bytes
TCP	36,737,165	84.32	11,231,147,530	94.50
UDP	6,202,673	14.24	601,157,016	5.06
ICMP	630,528	1.45	53,128,377	0.45
Total	**43,570,366**	**~100**	**11,885,432,923**	**~100**

Table 3.1 Protocols distribution of the collected IP traffic. Traffic data was collected on 2002-12-22.

Applications	Connections	% of connections	Bytes	% of bytes
WWW	304,243	90.06	10,203,271,455	75.79
FTP-data	636	0.19	1,440,393,636	10.7
IRC	2,324	0.69	120,873	0.008
SMTP	562	0.17	945,965	0.01
POP3	115	0.03	2,326,374	0.02
Telnet	70	0.02	280,286	0.002
Other	651	8.84	238,099,516	13.47
Total	**308,601**	**100**	**11,885,432,923**	**100**

Table 3.2 Application distribution of a traffic trace. Traffic data was collected on 2002-12-22.

We extract TCP packets that are used to establish TCP connections between the source and the destination. Those packets have either their synchronizing sequence number flags (SYN) or their finishing flags (FIN) set in the TCP header. Since a packet with flag RST means that the connection is terminated for some special reason, we ignore those special packets, which are rare in the traces. Packets with SYN or FIN flags are then classified according to the applications that generated them. The application

information is obtained from the port number. The contribution of major applications is shown in Table 3.2. Several well-known applications, such as POP3, are present, albeit in small proportions. Note that Web applications dominate the traffic. There are very few FTP-data connections, contributing with a large number of bytes. Category "Other" includes applications using a wide range of TCP and UDP port numbers. The most common port numbers in this category are 1755, 9065, 9013, and 9014. These non-standard ports are used by either new applications or unknown protocols. We suspect that they are most likely generated by online games that use unregistered fixed ports because playing computer games is very popular in Internet cafés. People rarely use POP3 and SMTP protocols to check and send their e-mail message from public machines. They now prefer to use the web-mail protocols.

3.3 WWW traffic on the TCP connection level

With the current growth of the WWW, Web proxy caching has been widely implemented in Web server. Collected Web data has been used to acquire the frequency-rank relation of client's requests. Most traces in the past [5], [8] were collected in a local or a campus settings, either from Web server logs or from Web proxies. Therefore, it is also of interest to examine the frequency-rank relation of requests for connections in traffic data collected from a geographically broad region. Our traffic trace includes end-users' requests in Internet cafés that are located across provinces in China. While past work mainly examined the frequency-rank relation of client's requests, we examine here frequency-rank relation of the number of connections by analyzing traffic distribution among the Web servers. This approach provides valuable information for traffic analysis

and traffic modelling on the connection level. Our work provides a unique empirical study of the frequency-rank relation of client's connections based on genuine traffic data.

We extracted the HTTP information from the traffic trace based on TCP connections. These traffic data usually exhibit rather skewed behaviour: few websites are very popular while most websites are seldom visited. Hence, common statistic parameters, such as mean, median, and variance, could not characterize these traffic data. We used Zipf distribution for their description because this distribution has been used in the past to describe skewed data [8]. Zipf's law states that the number of requests (frequency) is inversely proportional to its rank among the requests (the largest number of request corresponds to rank 1). The generalized Zipf distribution is defined as:

$$f_r \sim 1/r^{\theta}, \tag{3.1}$$

where fr is the number of requests and r is the rank of the website in terms of the number of requests. Its log-log plot is linear, with slope θ that is typically a constant less than 1. We examined various segments in the collected traffic trace, during various periods of a day or a week, and found that the frequency-rank relation of connections does not follow the Zipf distribution. As shown in Figure 3.3 (a), the Zipf distribution fits the mid range of the curve well while it exhibits top concavity [6]. This phenomenon appears frequently in multiple tests of sub-traces. We attribute this phenomenon to the practice that most browsers in the Internet cafés support HTTP/1.1 (rather than HTTP/1.0) because of the increased usage of Internet Explorer 6.0 and Netscape 7. An HTTP/1.1 connection can be kept open to transfer multiple Web pages from the same server, resulting in multiple requests per connection. Hence, the frequency-rank of client's connections will further deviate from linear relation as the number of persistent connections increases.

We found that collected traffic data follow the Discrete Gaussian Exponential (DGX) [6], as shown in Figure 3.3 (b). The DGX distribution is a discrete version of the continuous lognormal distribution:

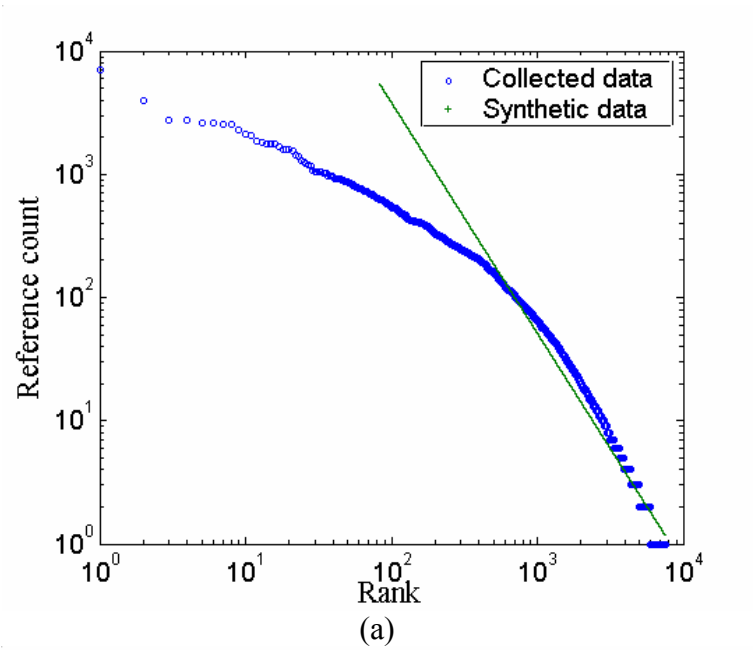
$$p(x=k)=\frac{A(\mu,\sigma)}{k}\exp[-\frac{(\ln k-\mu)^2}{2\sigma^2}],\quad \text{k = 1, 2, ...,} \tag{3.2}$$

where

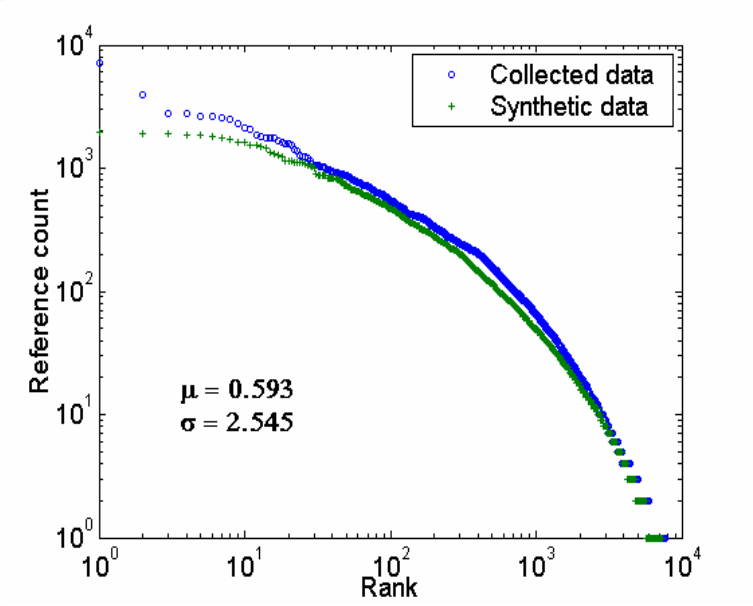
$$A(\mu,k)=\{\sum_{k=1}^{\infty}\frac{1}{k}[-\frac{(\ln k-\mu)^2}{2\sigma^2}]\}^{-1} \tag{3.3}$$

is a normalization constant that depends on parameters μ and σ. Parameters μ and σ are estimated using the maximum likelihood estimation.

We also examined the top ten most loaded websites. It was surprising that they are all registered under the Asia Pacific Network Information Centre (APNIC). They account for 60.23% of the entire traffic load. The heaviest loaded is a Chinese search engine website. This result indicates that traffic is non-uniformly distributed among the Internet hosts and that language and geographical factors are important for content delivery networks and caching proxy designs.



(a)

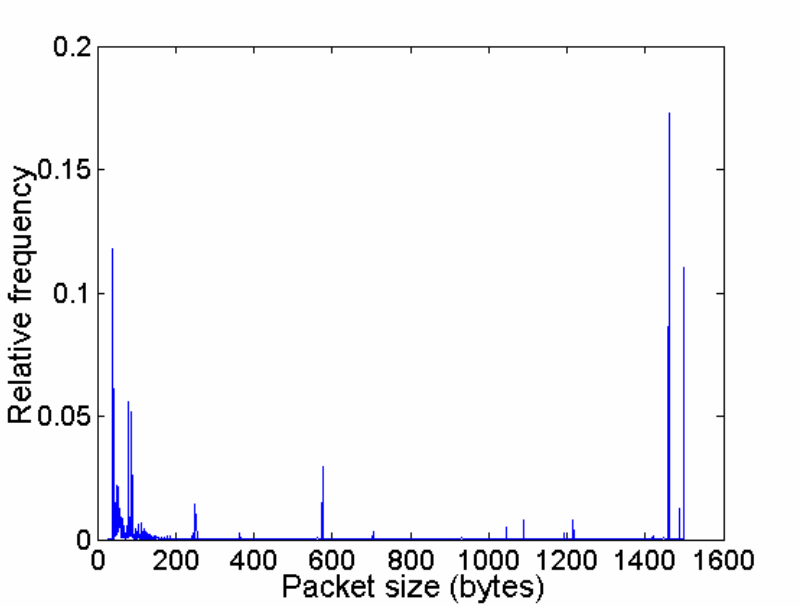


(b)

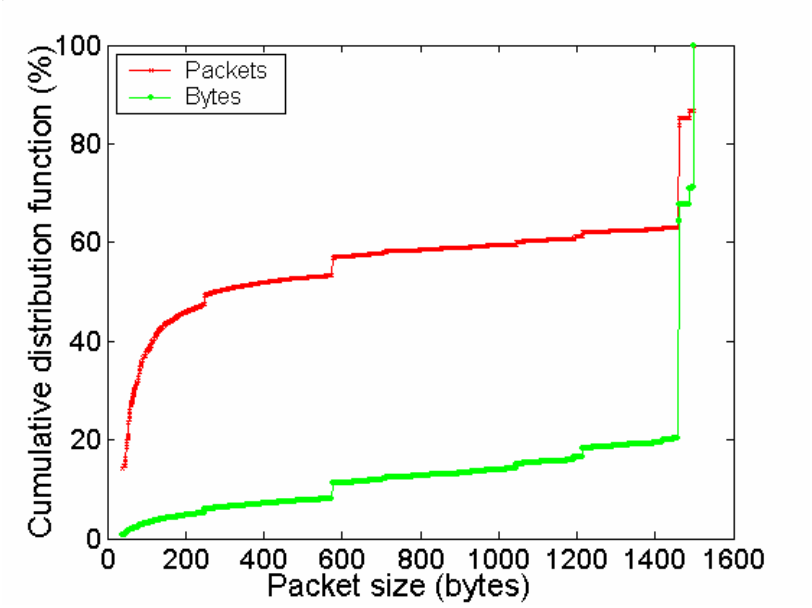
Figure 3.3 Frequency vs. rank of client's connections. Zipf distribution curve fit (a) and DGX curve fit (b). Traffic data was collected on 2002-12-22.

3.4 TCP packet sizes

Packet size statistic is shown in Figure 3.4. Packet size distribution is bimodal: there are numerous small acknowledgement packets and many large packets for bulk data file transfer type applications. There are very few in between. Ten most common packet-sizes are listed in descending order: 1,460, 40, 1,500, 80, 88, 576, 48, 55, 250, and 1,462 bytes. The maximum packet-size of 1,460 bytes is difficult to explain: a possible explanation may be a default buffer size in routers. Many 1,500-byte packets result from the limit imposed on maximum packet size in IP networks. Packets of 80 and 88 bytes are UDP packets. A large presence of 576-byte packets reflects TCP implementations without "path MTU discovery" using packets of 536 bytes (plus 40-byte acknowledgements) as the default Maximum Segment Size (MSS). This also explains the lack of the common 552-byte packets in the collected trace. The smallest packets, 40 bytes in length, are mainly TCP packets with ACK, SYN, FIN, or RST flags.



(a)



(b)

Figure 3.4 Relative frequency of packet sizes (a) and their cumulative distribution (b). Traffic data was collected from 2002-12-21 22:08 to 2002-12-23 3:28.

The cumulative distribution function showed in Figure 3.4 (b) shows that 50% of the packets are smaller than 300 bytes. They are mainly TCP acknowledgement packets and short HTTP click requests. More than 50% of the bytes are carried in 1,460-byte packets. This is due to the massive transfers of data in the down-stream path. In conclusion, most bytes are sent in large packets, and packet size distribution is bimodal. This concurs with results reported in previous studies [20].

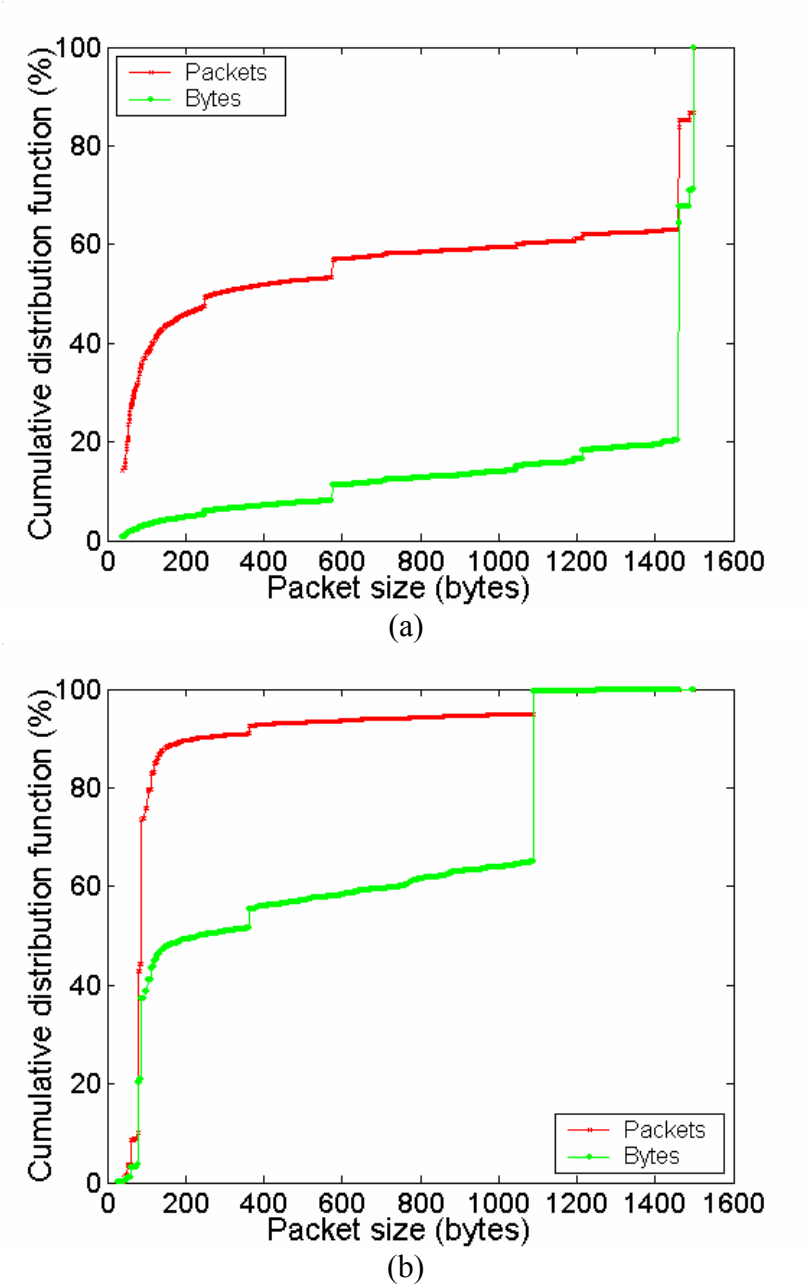


Figure 3.5 Cumulative distribution function of TCP packet sizes: downloaded traffic (a) uploaded traffic (b). Traffic data was collected from 2002-12-21 22:08 to 2002-12-23 3:28.

Figure 3.5 shows that most of the uploaded TCP packets are of small size, while most downloaded TCP packets are of large size. More than 85% of the uploaded TCP packets are 40 bytes long. They contain only header without any data. Most uploaded packets are

acknowledgements and simple requests. This is the cause of the asymmetric HTTP transmission pattern.

CHAPTER 4 ESTIMATION OF SELF-SIMILARITY AND LONG-RANGE DEPENDENCE

Self-similarity and long-range dependence have been well documented in traffic studies over the past decade [19]. They provide key concepts in analyzing traffic data. In this chapter, we first give an introduction to self-similarity and to five different estimators. We then address some practical considerations when using these estimators and present estimate result.

4.1 Stationary random processes

We are interested in the traffic volume measured in bytes, packets, or connections at time instance t. Traffic volume can be regarded as a discrete time stochastic process. For traffic modelling proposes, it is convenient to consider this process to be stationary. A random process $X(t)$ is said to be stationary if the statistical properties of the process do not vary over time. Strict stationary processes are difficult to find in the natural phenomena. Hence, researches generally investigate wide-sense stationarity (WSS), which require that the first and the second order statistical properties, namely the mean μ and variance σ^2, not to vary over time.

4.2 Self-similarity and long-range dependence

4.2.1 Introduction to self-similar processes

With the assumption that the processes we investigate are stationary, we are now ready to begin our discussion of self-similarity and long-range dependence. In literature, various definitions of self-similarity have been used. In this section, we follow closely the

definition of W. Willinger [19] because it is appropriate in the context of standard time series. First, we define the aggregate process $X^{(m)}$ of X at aggregation level m as

$$X^{(m)}(k) = \frac{1}{m} \sum_{i=(k-1)m+1}^{km} X(i) \quad \text{k=1, 2, ...} \tag{4.1}$$

The idea of aggregation is to equally divided X into blocks of length m and to use i to index the averaged blocks. "If X is the increment process of a self-similar process, that is $X(i) = Y(i+1) - Y(i)$, then for all integers m, the condition for self-similarity is

$$X = m^{1-H} X^{(m)}. \tag{4.2}$$

If a stationary sequence $X = \{X(i),\ i \geq 1\}$ satisfies equation (4.2) for all aggregation levels m, then it is called exactly self-similar. It is said to be asymptotically second-order self-similar if equation (4.2) holds as $m \to \infty$. Similarly, a covariance-stationary sequence is called exactly second-order self-similar if $m^{1-H} X^{(m)}$ has the same variance and autocorrelation as X" [2]. The autocorrelation of the second-order self-similar process can be described as:

$$r(k) \sim ck^{\beta} \quad k \to \infty, \tag{4.3}$$

where $c > 0$ and $\beta = 2 - 2H$. This implies that the autocorrelation function decays hyperbolically. It decays so slowly that its sum is not summable. This property is in contrast to traditional models, all of which have the property that the correlation structure of their aggregated process decays exponentially as k increases. Self-similarity indicates that the aggregated process $X^{(m)}$ is indistinguishable from X with respect to their first and second order properties. In the case of engineering applications, it has been shown that

the first and the second moments have physical meaning. Thus, the second-order self-similarity has been a dominant framework for modelling network traffic.

Figure 4.1 is derived from the collected traffic trace in order to observe the self-similar characteristics of the traffic. Each finer time scale observation is a zoom-in result of the first 10% samples of the previous coarser time scale. It indicates that the scaled counts of packet or byte arrivals look approximately the same regardless of the time scales.

4.2.2 Long-Range Dependence

One important property of the self-similar process is long-range dependence (LRD), which implies correlations across large time lags. When the autocorrelation function $r(k)$ satisfies:

$$\sum_{k=0}^{\infty} r(k) \to \infty , \tag{4.4}$$

we call the corresponding stationary process $X(t)$ long-range dependent. $X(t)$ is short-range dependent if the autocorrelation function is summable. Since autocorrelation is also used to define for second-order self-similarity, the term (exactly or asymptotically second-order) self-similarity and long-range dependence are sometimes used interchangeably because both definitions are essentially equivalent: they both describe the tail behaviour of the autocorrelation function. Nevertheless, these two terms are not equivalent. Self-similarity is usually used to describe the scaling behaviour of continuous or discrete process, while LRD focus on the tail behaviour of the autocorrelation of a stationary time series.

The attractive feature of the self-similar traffic models is that a single Hurst parameter can be used to describe the degree of self-similarity. For self-similar process with long-range dependence, $0.5 < H < 1$. As H approaches 1, the degree of both self-similarity and long-range dependence increases.

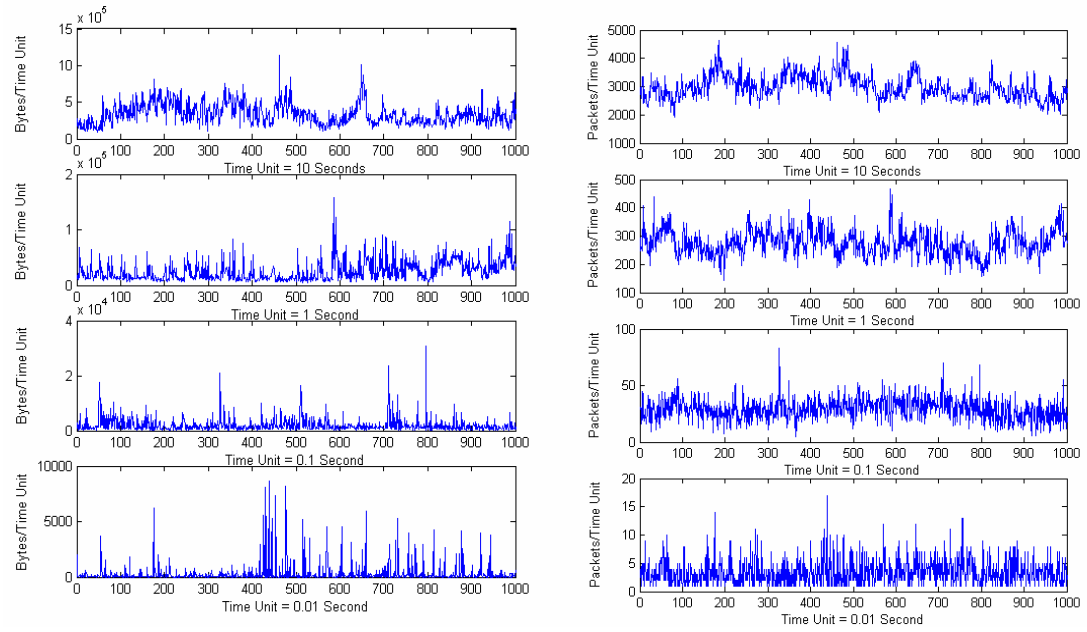


Figure 4.1 Self-similar characteristics of byte traffic (left) and packet traffic (right). The time scales from top to bottom are 10 seconds, 1second, 0.1 second, and 0.01 second respectively.

4.3 ON/OFF model and heavy-tailed distributions.

It is known that self-similar processes are generated by aggregating multiple ON/OFF sources. The distinctive features of those ON/OFF periods are heavy-tailed distributions with infinite variance [9]. An ON/OFF source model describes how the traffic fluctuates between a source/destination pair. During the ON state, the source continuously sends data at the constant data rate. During the OFF state, the source is silent. As shown in Figure 4.2, ON/OFF states can be represented by random process Xn and Yn, respectively.

Figure 4.2 ON/OFF source model.

M. E. Crovella and A. Bestavros [9] found that the Web and FTP file sizes follow "heavy-tailed" probability distribution, thus leading to the heavy-tailed distribution of transmission duration as well. Therefore, the self-similar traffic model has good physical explanations. Heavy-tailed distribution means the probability of a given variable decreases very slowly as the variable increase, as opposed to the exponential distribution where the rate of decay is much faster. The contribution of the tail part for such type of distributions cannot be ignored. This is the reason for calling them "heavy". We say that a random variable X follows a heavy-tail distribution if

$$P[X > x] \sim kx^{-\alpha} \qquad \text{as } x \to \infty,\ 0 < \alpha < 2. \tag{4.5}$$

A special feature of this distribution is that its variance is infinite. The variance shows the spread of data around the mean. For finite variance distribution, increasing the granuarility of the horizontal scale will make distribution appear smooth, clustered around the mean [13]. However, for infinite variance distribution, measuring the signal on different scales will not change the central tendency of the data. This property relates infinite variance to the self-similar behavior.

4.4 Estimating Hurst parameter in collected network data

Several methods are commonly used for estimating the long-range dependence in network traffic. To estimate the self-similarity in the collected traffic data, we employed

the following methods: Rescaled Range (R/S), Variance/Time Plot, Periodogram Whittle Estimator, and Wavelets analysis [1], [2].

4.4.1 R/S method

R/S method was first introduced by Hurst when he studied how to regularize the flow of Nile river [28]. This method can be described as follow: Let $X(k)$, where k= 1, 2, …, n be a set of n observations that have an expected value (sample mean) E[$X(k)$]. The scaled and adjusted range is given by:

$$\frac{R(n)}{S(n)} = \frac{\max(0, W_1, ...W_k) - \min(0, W_1, ..., W_k)}{S(n)}, \tag{4.5}$$

where S(n) is the standard deviation and for each k: 1, 2, …, n, and W_k is given by

$$W_k = (X_1 + X_2 + \ldots + X_k) - kE[\, X(k)\,], \quad k= 1, 2, \ldots, n. \tag{4.6}$$

The Hurst parameter has the following relation with *R(n)/S(n):*

$$R(n)/S(n) \sim cn^H, \tag{4.7}$$

where c is a constant value. By taking logarithms on both sides, we get:

$$\log(R(n)/S(n)) \sim \log c + H \log n. \tag{4.8}$$

Therefore, the slope of a fitted straight line to a plot log(R/S) vs. log (n) is the Hurst parameter, as shown in Figure 4.3(a).

4.4.2 Variance-Time method

Let $X(k)$ be a series of observations for k=1, …, n. This method utilizes a property of long-range dependent processes that the variance of the sample mean

converges to zero slower than the reciprocal of the sample size (1/m). It can be shown that

$$Var[X(m)] \approx cm^{2H-2}. \tag{4.9}$$

Therefore, the slope of a plot of *log(Var [X(m)])* vs. *log(m)* is $2H-2$. Hence, the Hurst parameter can be found as H= $1+slope/2$, as shown in Figure 4.3(b).

4.4.3 Periodogram method

This method considers the squared magnitude of the Fourier transform of the time series and computes the power spectral density as:

$$I_x(v) = \frac{1}{2\pi N}\left|\sum_{t=1}^{N} X(t)e^{-j2\pi vt}\right|^2, \tag{4.10}$$

where v is the frequency and N is the length of the time series. The periodogram of LRD series is proportional to v^{1-2H}. Therefore, Hurst parameter can be estimated by fitting a straight line to a log-log plot of the periodogram vs. frequency. The slope of the line is *1-2H*, as shown in Figure 4.3(c).

4.4.4 Whittle method

Whittle method is based on periodogram of the time series. The estimator is defined as the value that minimizes the likelihood function [2]. It estimates H and provides confidence intervals. It does not produce a graphical output.

4.4.5 Wavelet method

4.4.5.1 Wavelet theory

Wavelet transform operates in a similar manner as the Fourier Transform (FT). Instead of approximating the underlying signal using sine or cosine waves, wavelet

transforms use some special design functions that transform the data (via dilation and translation) into approximation and detail components. It does not only decompose the signal into different frequency or scale but it also easily relates frequency information to the time or position [11]. Wavelets, therefore, offer great flexibility to analyze random, chaotic, and noisy signals that are not well approximated by the FT. Though a set of wavelet filters, Wavelet transform can examine the signal on local as well as global features. Therefore, Wavelet transform is an ideal tool to examine self-similar processes.

A signal will be adaptively approximated by estimating shift and dilating parameters of orthogonal wavelets. The wavelets are defined as:

$$\psi_{a,b}(t) = |a|^{-\frac{1}{2}} \psi\left(\frac{t-b}{a}\right), \tag{4.11}$$

where $\psi_{a,b}(t)$ is the daughter wavelet, $\psi(t)$ is the mother wavelet, a is the (dilations) scale factor, and b is the (time location) shift factor.

It is convenient in practical applications, such as traffic modelling, that the parameters *a, b* corresponding to wavelets are sampled at powers of two (so called "dyadic") in the frequency-scale plane. A common definition of such discrete wavelets is

$$\psi_{j,k}(t) = 2^{-\frac{j}{2}} \psi(\frac{t-2^j k}{2^j}) \qquad j,k \in Z\,. \tag{4.12}$$

The discrete wavelet transform (DWT) maps the vector of n discrete signal $\boldsymbol{f}$ = *(f1, f2, ... fn)*' to a vector of n wavelet coefficients $\boldsymbol{w}$ = *(w1, w2, ...wn)*'. The DWT is mathematically equivalent to multiplication by an orthogonal matrix **W**:

$$\mathbf{w=Wf.} \tag{4.13}$$

In Chapter 6, we discuss a faster "pyramid" algorithm to calculate DWT.

4.4.5.2 Wavelet based estimator

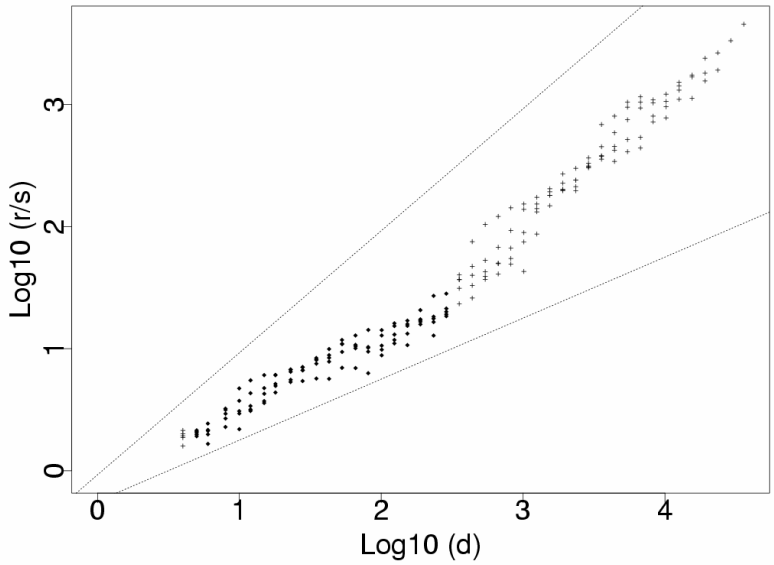
It is shown [8] that a spectral estimator of *X(n)* at frequency $2^{-j}v_0$ can be obtained by computing the time average of the wavelet coefficients $w_{j,k}$ at the scale j:

$$\hat{\Gamma}(2^{-j}v_0) = \frac{1}{n}\sum_{k=0}^{n_j-1} w_{j,k}^2 , \tag{4.14}$$

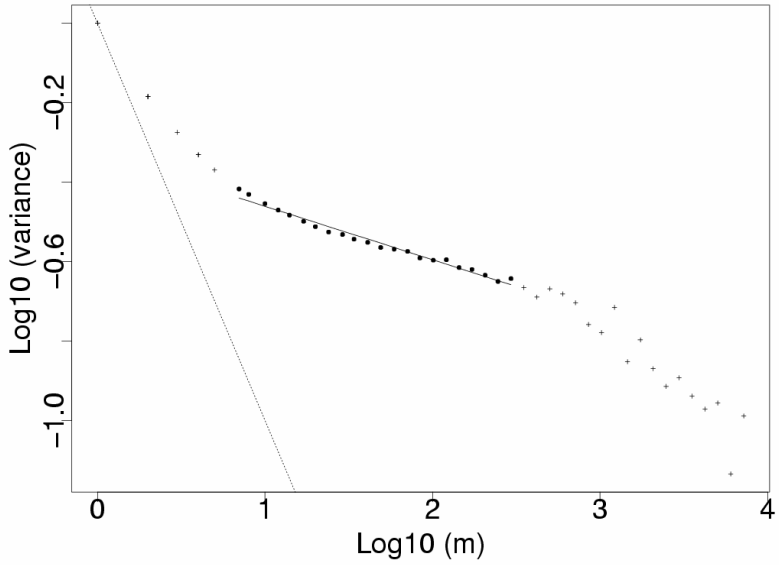
where n_j is the number of wavelet coefficients at scale j. The wavelets coefficients are shown to be quasi-decorrelated. Thus, the signal *X(n)* that is highly correlated in the time domain becomes quasi-decorrelated in the wavelet domain. This is one of main reasons why wavelet transforms are used for analyzing self-similar and LRD signals. When *X (n)* is a wide sense stationary process, it has been shown that the signal energy at different scale j has the following property [1]:

$$\log_2(E(w_j^2)) = (2H-1)j + c . \tag{4.15}$$

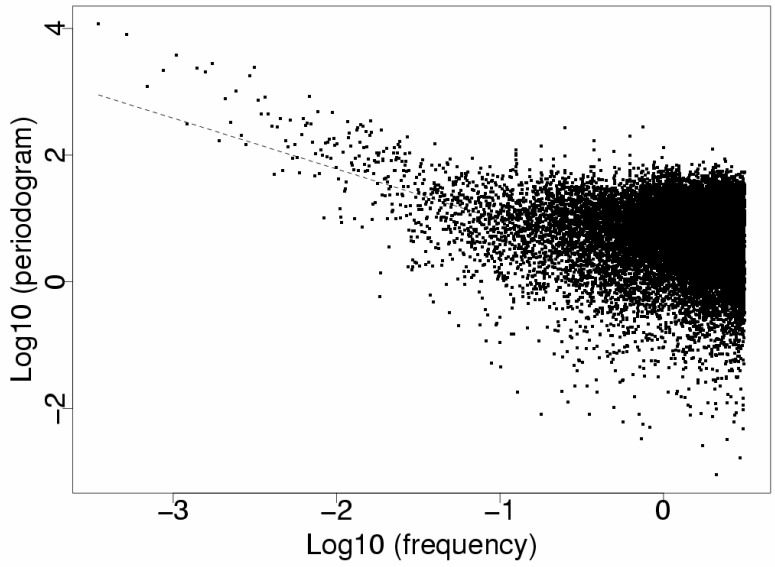
An estimation of H is given by fitting a straight line to a log-log plot of the energy vs. scale j. The slope of the line is approximately $2H-1$, as shown in Figure 4.3(d).



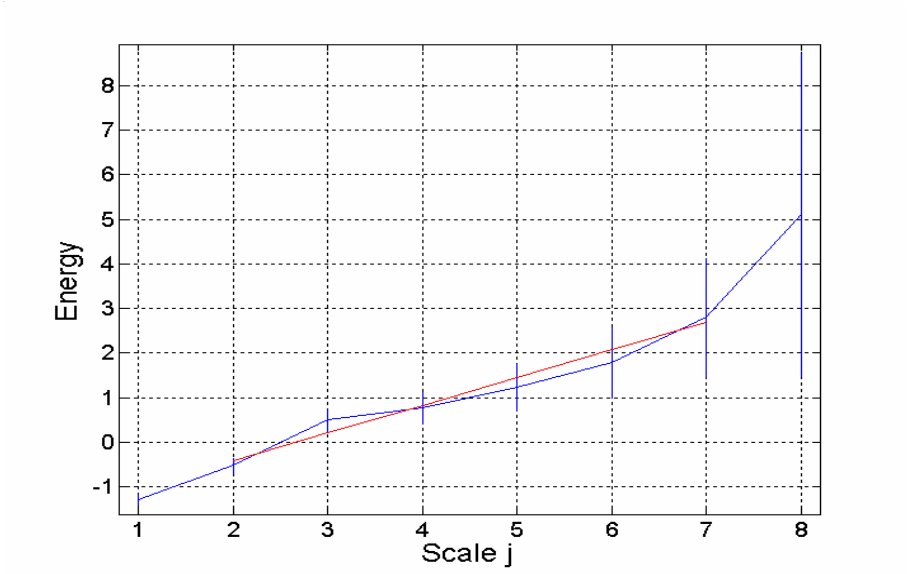
(a) R/S plot



(b) Variance-Time plot



(c) Periodogram plot



(d) Wavelet plot

Figure 4.3 Four graphical estimators of the Hurst parameters.

4.4.6 Estimation of self-similarity

4.4.6.1 Examining the stationarity

Self-similar traffic models are based on the assumption of wide sense stationarity. Hence, it is important to examine the stationarity before characterizing data using various estimators [35]. Note that traffic traces generated from synthetic Fractional Gaussian

Noise (FGN) processes [19] are stationary. In genuine traffic traces, there is a visible level shift and trend in the traffic load. This phenomenon can usually be observed in low-bandwidth networks, such as satellite networks, as shown in Figure 4.4(a). Testing stationarity is rather difficult [35]. An alternative approach is to decompose a trace into small sub-traces to eliminate the trend and the effect of level shifts [8]. In practice, this approach may result in short samples, which leads to unreliable estimates. Hence, a trade-off between finding stationary samples and the small size is advised.

In order to reduce the effect of non-stationarity, we first decomposed daily traffic trace into 24 non-overlapping sub-traces. Each sub-trace was then tested for the second order or wide sense stationarity. The sub-traces were first visually inspected to locate visible tendencies that imply non-stationarity. Each sub-trace was then divided into equal blocks. We examined the mean, variance, and Hurst parameter of each block by employing various statistical tests [32]. We found that one-hour traces may be treated as stationary, with the exception of several afternoon sub-traces. These afternoon sub-traces were further decomposed into smaller sub-traces, while keeping a sufficient number of samples.

4.4.6.2 Estimating the Hurst parameter

Figure 4.4 (b) shows the Hurst parameter estimated using various estimators. As expected, estimators produced different Hurst parameters. This is because the processes have different effects on the estimators [16]. Nevertheless, estimated Hurst parameters exhibit similar trend with traffic data. In our experiment, more weight was given to the wavelet estimator because it may help avoid the effects of non-stationarity [1]. To some degree, the wavelet curve also reflects the traffic fluctuation shown on Figure 4.4(a).

Nevertheless, we cannot conclude that the Hurst parameter increases as the traffic load increases, as observed elsewhere [19]. This also suggests that a single-parameter traffic characterization might not capture the complexity of the network traffic variability. It remains an open question how to test the stationarity and estimate self-similarity in genuine traffic traces.

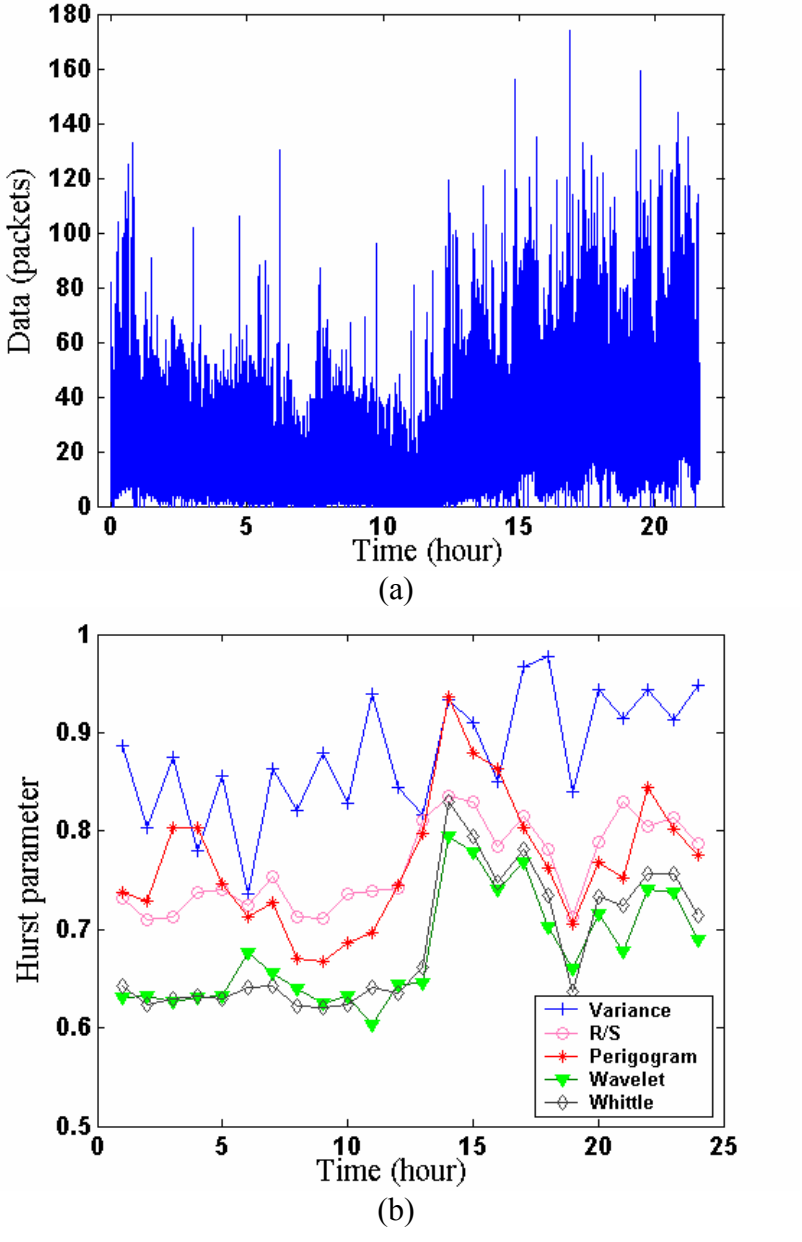


Figure 4.4 Non-stationarity in daily traffic (a) and variation of the Hurst parameter (b). Traffic data was collected on 2002-12-09.

CHAPTER 5 MODELLING OF TCP CONNECTION

In Chapter 4, we showed that the collected traffic traces are long-range dependent (second order self-similar) and that they can be modelled as the aggregation of Heavy-tailed distributions. Hence, we used distributions with heavy-tailed characteristics to create a traffic model.

5.1 Fitting four distributions to the collected traces:

TCP splitting technique described in Section 2.2 may improve the efficiency of a hybrid wired-wireless system. It imposes considerable memory requirements on the edge routers. Hence, it is important to model downloaded traffic on a TCP connection level in such hybrid systems. Collected traffic traces can be used to extract individual or aggregate TCP connections. The connection information is obtained from TCP headers (SYN and FIN flags). We modelled two important parameters: the number of downloaded bytes per TCP connection and TCP connection inter-arrival times. Suppose that the data are sent at a constant rate. We can relate downloaded data with ON periods. We consider the inter-arrival time as OFF periods since there is no data transferred during this period. We examined three distributions with heavy-tailed characteristic and used exponential distribution for comparison. Tables 5.1 and 5.2 [12] show the definitions of various distributions and the maximum likelihood estimators for their parameters, respectively.

Distribution	Probability density	Cumulative probability
Exponential	$f(x) = \frac{1}{\rho} e^{-x/\rho}$	$F(x) = 1 - e^{-x/\rho}$
Weibull	$f(x) = \frac{1}{a}\left(-\frac{x}{a}\right)^{c-1} \mathrm{e}^{-(\mathrm{x/a})^{\mathrm{c}}}$	$F(x) = 1 - e^{-(x/a)^c}$
Pareto $(k > 0, a > 0; x \geq k)$	$f(x) = \frac{ak^a}{(x)^{k+1}}$	$F(x) = 1 - \left(\frac{k}{x}\right)^a$
Lognormal	$f(x) = \frac{1}{x\sqrt{2\pi\sigma}} e^{-[\log(x)-\xi]^2/2\sigma^2}$	No closed form

Table 5.1 Definition of probability distributions.

Probability distribution	Maximum likelihood estimator
Exponential	$\rho = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x}$
Weibull	$a = \left[\frac{1}{n}\sum_{i=1}^{n} x_i^c\right]^{\frac{1}{c}}, \quad c = \left[\left(\sum_{i=1}^{n} x_i^c \log x_i\right)\left(\sum_{i=1}^{n} x_i^c\right)^{-1} - \frac{1}{n}\sum_{i=1}^{n} \log x_i\right]^{-1}$
Pareto	$k = \min(X_i), \quad a = \frac{n-1}{\sum_{i=1}^{n} \log X_i - n \log k}$
Lognormal	$\xi = \frac{1}{n}\sum_{i=1}^{n} \log x_i = \overline{\log(x)}$ $\sigma = \left[\frac{1}{n}\sum_{i=1}^{n} ((\log x_i) - \xi)^2\right]^{1/2} = \sqrt{\mathrm{var}(\log(x))}$

Table 5.2 Maximum likelihood estimators for probability distributions.

After we estimated the parameters of each distribution, we use inverse function to generate the random variables that follow the heavy-tailed distribution. For example, lognormal inter-arrival times and downloaded data were generated by transforming normal samples $u \sim randn(n,1)$ via inverse function of lognormal distribution: $x = e^u$.

Figures 5.1 and 5.2 indicate that all distributions are good fits. Hence, a more rigorous statistical analysis is needed to select the best candidate.

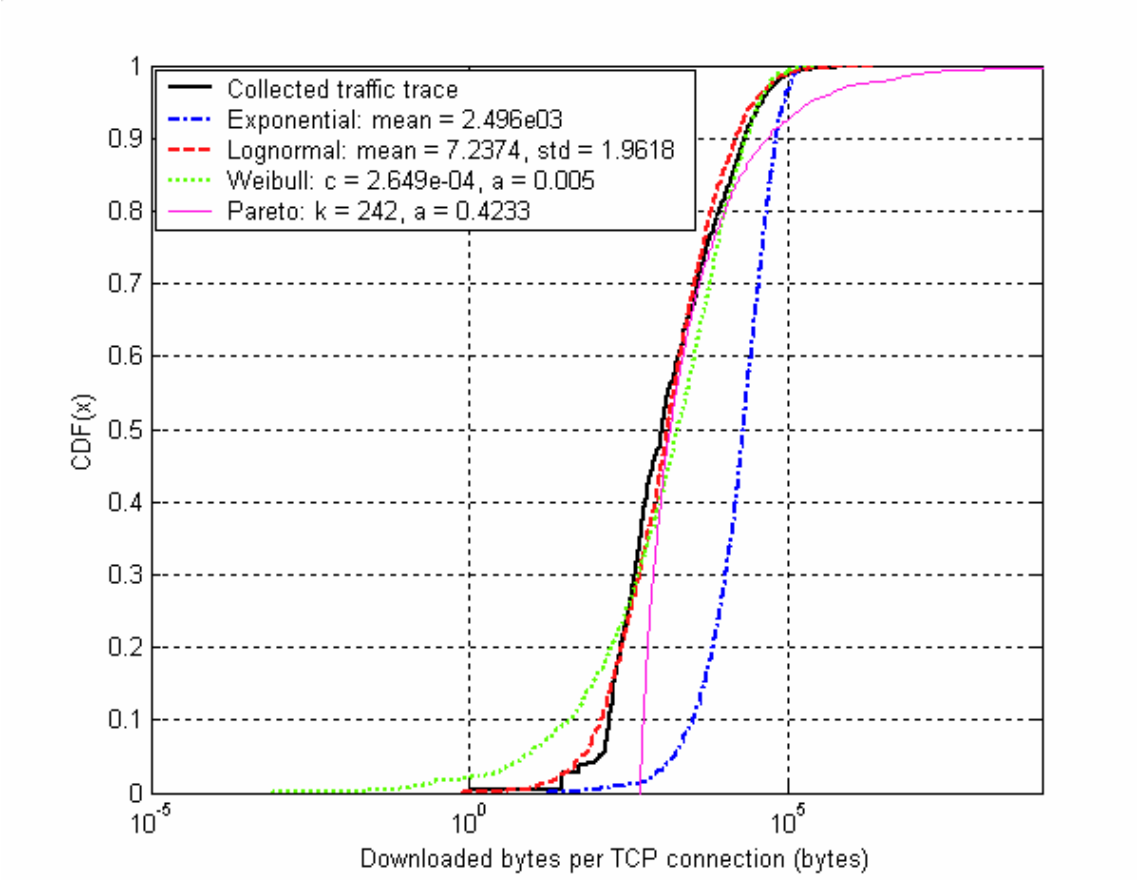


Figure 5.1 Downloaded data distribution fit vs. empirical cumulative distribution function.

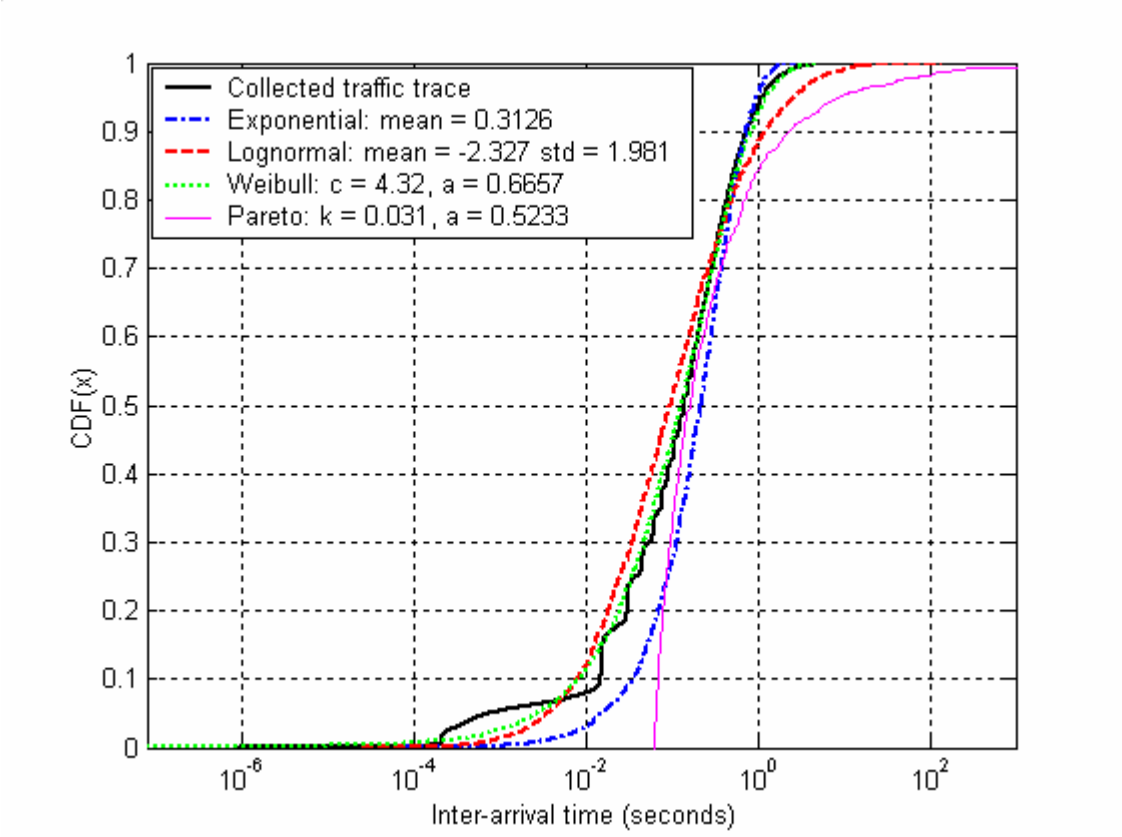


Figure 5.2 Inter-arrival time distribution fits vs. empirical cumulative distribution function.

5.2 Evaluation of different distributions

To evaluate and quantify the performances of various distributions, we employed goodness of fit tests or discrepancy measurements to evaluate the fitness of the models. In this research, we chose the Kolmogorov-Smirnov goodness of fit and χ^2 discrepancy tests.

5.2.1 Kolmogorov-Smirnov goodness of fit test

The goodness of fit test is usually described as the hypothesis test. It helps to decide whether to accept or reject a hypothesis that samples of a dataset follows a specific distribution [10]. The concept of significance level α is introduced to describe

the probability of rejecting a hypothesis when true. It can also be interpreted that the probability of accepting the hypothesis when true is $(1-\alpha)$. Usually, α is set to 5%. The Kolmogorov-Smirnov test compares the cumulative frequency of the observations with the cumulative frequency of the expected number generated by the specified distribution. The test uses the maximum difference over all values as a statistic. The value of this statistic is called P-value. If the P-value is greater than the significance level α, then the hypothesis can be accepted with $(1-\alpha)$ certainty.

A common problem when using goodness-of-fit tests is that results depend on the sample size [23]. Test will always reject the hypothesis when it is applied on large data sets, such as our traffic data. In this thesis, we apply the Kolmogorov-Smirnov test on randomly chosen subsets of the entire dataset [26]. Five series of Kolmogorov-Smirnov tests were applied to 200 randomly chosen samples from both downloaded data set and connection inter-arrival time data.

As shown in Table 5.3, all P-values of lognormal distribution are greater than 0.05 and are higher than the P-values for the other candidates. These results suggest that hypothesis that downloaded data per connection follows the lognormal distribution can be accepted with 0.95 certainty. Similarly, as shown in Table 5.4, we can accept the hypothesis that inter-arrival time follows the Weibull distribution with 0.95 certainty.

Model	1	2	3	4	5
Exponential	2.5×10^{-35}	6.1×10^{-47}	2.7×10^{-40}	5.1×10^{-48}	3.5×10^{-35}
Lognormal	0.16417	0.29098	0.36853	0.27441	0.20459
Weibull	5.5×10^{-3}	8.4×10^{-3}	3.32×10^{-3}	8.43×10^{-4}	1.54×10^{-3}

Table 5.3 P-value for downloaded data distribution.

Model	1	2	3	4	5
Exponential	6.2×10^{-5}	7.2×10^{-6}	2.5×10^{-7}	1.1×10^{-3}	3.7×10^{-5}
Lognormal	6.1×10^{-4}	6.2×10^{-3}	7.7×10^{-3}	4.8×10^{-3}	5.9×10^{-4}
Weibull	0.17523	0.73357	0.61562	0.75264	0.24156

Table 5.4 P-value for inter-arrival time distribution.

5.2.2 χ^2 discrepancy measure

χ^2 discrepancy is used to measure how close the real observations is related to a certain distribution [23]. It gives us a quantitative measure of the goodness of fit of a proposed model. First, n real observations are divided into N bins, and let Y_i to be observed samples falling into *i*th bin. Then, n samples that follow the Z distribution are also divided into N bins. Each bin has a probability p_i and it is proportion to the expected number falling into *i*th bin. The conclusion is made based on the magnitudes of the discrepancies between what is observed and what is expected. The ideal candidate should have the smallest discrepancy value. More precisely, let:

1. Y is the random variable we want to evaluate
2. Z is the candidate distribution
3. n is the number of observation of Y
4. N is the number of bins
5. p_i is the expected probability Z for the *i*th bin
6. Y_i's are the number of the observed samples falling into the *i*th bin.

The discrepancy measure is defined as

$$\chi^2 = \frac{X^2}{n}, \quad \text{where} \quad X^2 = \sum_{i=1}^{N} \frac{(Y_i - np_i)^2}{np_i}. \tag{5.2}$$

The results of the χ^2 discrepancy tests are shown in Table 5.3. The discrepancy of lognormal for downloaded data and the discrepancy of Weibull for inter-arrival time are minimum in their categories. These results concur with the results from Kolmogorov-Smirnov test.

Model	Exponential	Lognormal	Weibull
Downloaded byte	1,261,263	113,620	1,371,293
Inter-arrival	4,103,846	3,779,695	680,707

Table 5.5 Discrepany test for various distributions.

5.2.3 Conclusion

The Pareto distribution is unique among the four distributions. Historically, the Pareto distribution has been used to fit the upper tail of the trace rather than the entire body [23]. Therefore, we used Pareto distribution to fit part of the traffic data and did not use it to model the entire dataset. We selected to fit the Pareto distribution to the upper 90% of the data set.

From the results of Kolmogorov-Smirnov and χ^2 discrepancy tests, we can conclude that the Weibull distribution yields a suitable model for the inter-arrival times, while lognormal distribution is suitable to model the number of downloaded data per TCP connection. It is interesting to note that the exponential model underestimates both the small and the large values in both datasets.

CHAPTER 6 PREDICTING NETWORK TRAFFIC

Prediction plays a critical role in business planning. Traffic prediction is necessary for network managements such as resource allocation, buffer management, and congestion control. In this Chapter, we first use linear time series model to make long-term traffic prediction. We then use a wavelets based linear approach to make short-term prediction. We also evaluate the performance of these predictions.

6.1 Introduction

Time series modelling is one of the major prediction techniques used in practice. In time series prediction, historical data are collected and analyzed in order to develop a model that captures the underlying relationships among time series observations. The model is then used for forecasting to extrapolate the time series into the future. There are a wide variety of time series forecasting models. In general, they are divided into linear models and nonlinear models. In this thesis, we only consider the linear predictors.

The issue of predicting is a learning task. Assume that we are given a set of L past samples $x(n), x(n-1) \ldots, x(n-L)$ that are usually uniformly spaced in time. The requirement is to predict sample $x(n+k)$. $x(n+k)$ can be represented as:

$$x(n+k) = f(x(n-1), x(n-1), \cdots, x(n-L+1)), \tag{6.1}$$

where L is the prediction length, k is the prediction horizon, and f is the prediction function. In general, the design of this function involves two phrases: training and testing. In the training period, part of the time series $X(t)$ is used for training. In the testing

period, the remaining part of the series is used to evaluate the performance of the prediction.

In this thesis, we use the normalized mean squared error (NMSE) to measure the performance of a predictor. It is given by:

$$NMSE = \frac{1}{\sigma^2 N} \sum_{k=1}^{N} (x(k) - \bar{x}(k))^2 , \tag{6.2}$$

where $x(k)$ is the true value of the time series data, $\bar{x}(k)$ is the prediction, and σ^2 is the variance of the true sequence over the testing set. NMSE can be viewed as a noise to signal ratio. The smaller the NMSE; the better the predictor.

6.2 Using linear time series models to predict long-term traffic volume

Time series are usually modelled as stochastic processes. Linear models have good physical explanation and they are very easy to understand and implement. Classical linear techniques for time series analysis include moving average model (MA), autoregressive model (AR), and their mixture Autoregressive Integrated Moving Average (ARIMA) model. They have been predominant forecasting tools for more than 50 years. The ARIMA forecasting methodology is widely used in the financial markets to predict daily stock price and long-term exchange rate. Here, we are interested in overall traffic volume for capacity planning purposes. Hence, our historical data is obtained by aggregating billing records of individual users. The billing data recorded hourly customer's usage of the system. Hence, to obtain the overall hourly traffic we aggregated the individual usage. In this section, we first introduce AR, MA, and ARIMA models. We then describe how to use these models and give the results of the prediction.

6.2.1 Autoregressive process

Autoregressive process AR (p) was invented by Yule when he worker on prediction of the annual number of sunspots [7]. The AR model has become one of the most used linear time series models. Formally, AR (p) model can be written as:

$$X(t) = \phi_1 X(t-1) + \cdots + \phi_p X_{t-p}(t-p) + Z_t,$$

$$= \sum_{m=1}^{p} \phi_m X_{t-m} + Z_t \tag{6.3}$$

where X (t) is a linear combination of the past observations X (t-1), …, X (t-p). Z_t can be deterministic input or noise. Parameters ϕ_m can be estimated by the Yule-Walker equation [7].

6.2.2 Moving average process

The moving average process MA (q) can be written as follows:

$$X(t) = \mu + e(t) + \theta_1 e(t-1) + \theta_2 e(t-2) + \cdots + \theta_q e(t-q). \tag{6.4}$$

Equation 6.4 describes the MA model of order q, where μ is the mean of the MA (q) process and $e(t-q)$ represents the past disturbances. The disturbance terms represent noise and uncertainty. The idea behind this method is to eliminate the influence of shorter term fluctuations. Parameters θ_q can be estimated through the maxim likelihood estimation [7].

6.2.3 Autoregressive integrated moving average (ARIMA) model

Although the MA or AR models can capture the complex structure of the time series, the length of the model may be relatively long. In order to use parsimonious model

with relatively few parameters, we use combination of MA and AR models to construct autoregressive moving average (ARMA) model [36]:

$$X(t) = \mu + \phi_1 X(t-1) + \phi_2 X(t-2) + \cdots + \phi_p X(t-p) + e(t) + \cdots + \theta_q e(t-q). \qquad (6.5)$$

ARMA model is based on the assumption that the behaviour of the time series remains stationary. In practice, many non-stationary time series can be transformed into stationary series through differencing. The first order differencing data can be obtained by $y_t = x_t - x_{t-1}$. The autoregressive integrated moving average (ARIMA) model is developed to analyze the non-stationary time series data. If the first differenced data is stationary, ARMA models can be employed for further analysis.

If the data has a seasonal component, the regular differencing cannot remove the seasonal effect. For example, for the hourly billing data collected over several weeks, data from a particular hour would not only related to the immediately preceding hours and but will also relate to the same hour in preceding week. To deal with such an effect, a seasonal form of the ARIMA model can be used. It is named ARIMA $(p,d,q)\times(P,D,Q)_s$ where s is the period of the seasonal pattern, and p, d, q are the orders of the autoregressive, differencing, and moving average, respectively. The P, D, Q are the orders of the seasonal autoregressive, seasonal differencing, and seasonal moving average, respectively. The time series data within a season is modelled with (p,d,q), while the seasonal effect of the data is modelled with (P,D,Q) based on data not within the season but *s* units apart.

6.2.4 Identification methodology for ARIMA model

6.2.4.1 Autocorrelation (ACF) and Partial autocorrelation (PACF)

ACF and PACF are used to determine the parameters of ARIMA model. The autocorrelation measures the dependencies between observation pairs, y(t) and y(t+k), separated by k time periods (lags). The sample autocorrelation coefficient r_k can be computed as:

$$r_t = \frac{\sum_{t=1}^{N-k}((x_t - \bar{x})(x_{t+k} - \bar{x}))}{\sum_{t=1}^{N}(x_t - \bar{x})}. \tag{6.6}$$

In addition to measuring the correlation between y(t) and y(t+k), partial autocorrelations consider the effect of intervening observation y(t+1) and y(t+k-1). PACF can be considered as the correlation between $y_t - E(y_t \mid y_{t-1}, \cdots, y_{t-k+1})$ and $y_{t-k} - E(y_{t-k} \mid y_{t-1}, \cdots, y_{t-k+1})$. PACF value ϕ_{kk} measures the correlation between y(t) and y(t+k) after removing the effects of y(t+1), y(t+2), …, y(t+k-1) as:

$$y(t+k) = \phi_{k1} y(t+k-1) + \phi_{k2} y(t+k-2) + \cdots + \phi_{kk} y(t) + e(t). \tag{6.7}$$

In summary, the ACF and PACF have different patterns with different underlying meanings. ACF measures the correlation between observation pairs while PACF measures their partial correlation.

6.2.4.2 Choice of the differencing order d

Most non-stationary data can be transformed to stationary by using differencing once. In case where the first difference is still non-stationary, we consider second difference on the first differenced data.

6.2.4.3 Choice of the order of p (autoregressive) and q (moving average)

After a time series has been transformed to a stationary one by differencing, the next step in fitting an ARIMA model is to determine the AR or MA terms. Box and Jerkin [7] introduced a method consisting of checking the ACF and PACF plots, shown in Table 6.1:

Process	Autocorrelation plot	Partial autocorrelation plot
AR(p)	Dominated by either damped exponentials or sine waves	The lag beyond which the value cuts off is order p
MA(q)	The lag beyond which the cuts off is the order q	Dominated by damped exponentials and sine waves
ARMA(p,q)	Exponential or sine wave decay after lag (q-p)	Exponential or sine wave decay after lag (p-q)

Table 6.1 Characteristics in ACFs and PACFs of stationary processes.

6.2.4.4 Choice of the seasonal order (P, D, Q)s

If the series has a strong and consistent cycle pattern, we need to consider a seasonal model. The seasonal period *s* can be observed from ACF plot or Fourier transform. Seasonally differencing is used to remove the seasonal effects. The seasonal differencing data can be obtained by finding $y_t = x_t - x_{t-s}$. The order *D* is usually set to one. Similarly to defining orders *p* and *q* in Table 6.1, *P* and *Q* values can be evaluated from ACF and PACF plots of the seasonally differenced series. The difference between estimating p, q and estimating P, Q, is that spikes in the plot should be examined *s* units apart.

6.2.5 Billing records: fitting ARIMA model

The first seven weeks of billing records were used to train the ARIMA model. The eighth week of data was used to test the performance of prediction data.

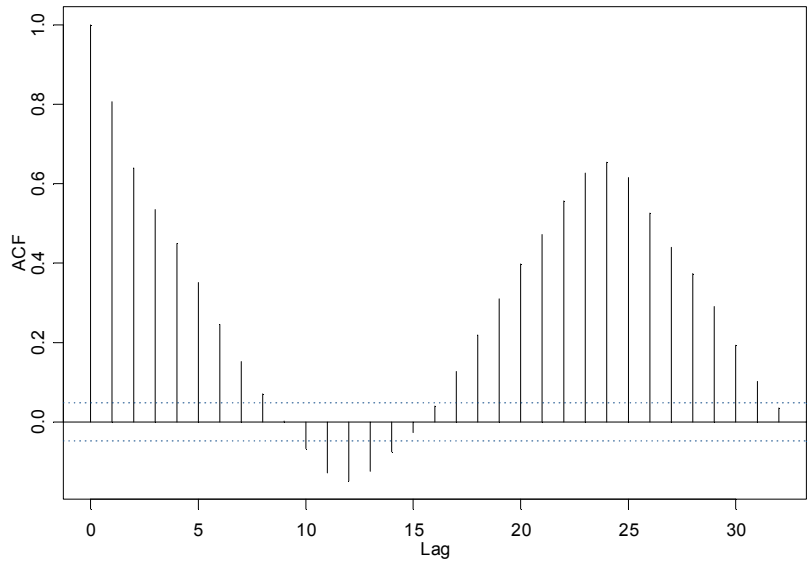


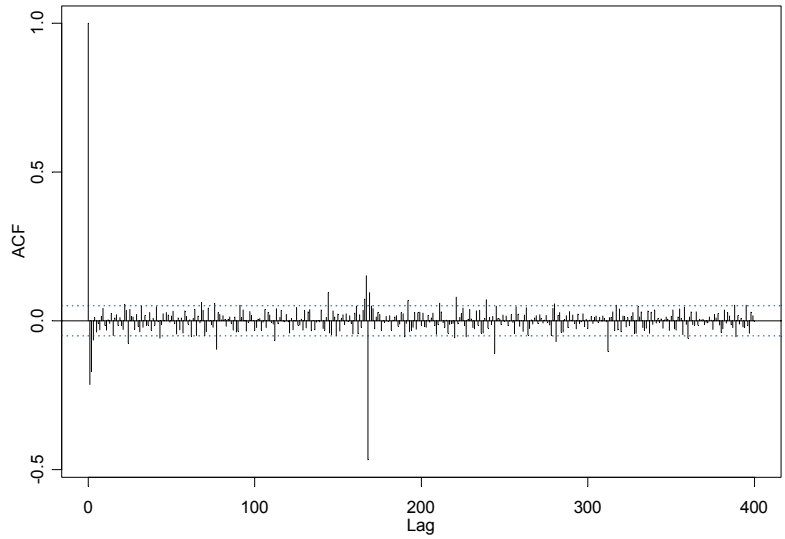
Figure 6.1 ACF of the billing records. The ACF coefficients do not decay but rather have a periodic pattern. The data was collected from 2002-11-01 to 2002-12-13.

6.2.5.1 Determine the regular differencing, seasonal differencing, and seasonal length

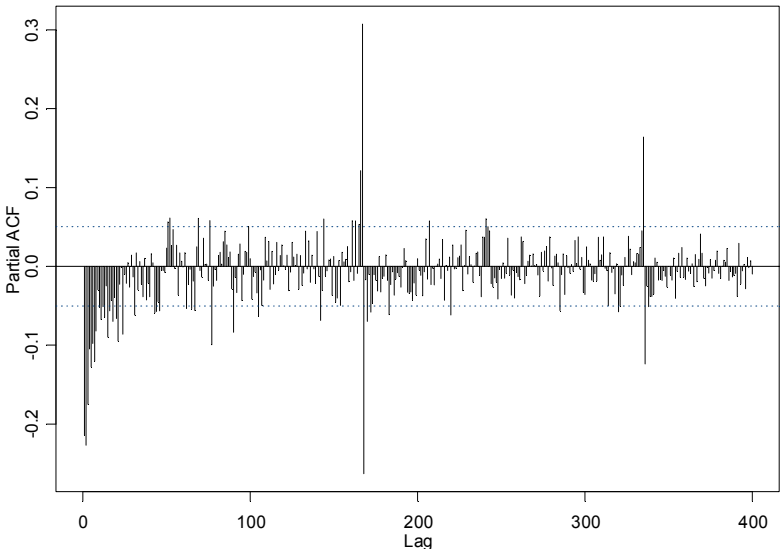
The difference order needs to be defined first. Figure 6.1 shows that there is no sign of a slow decay but rather a periodic pattern. Hence, we need to implement seasonal differencing (D=1). The ACF plot indicates daily cycles (24 hours) and weekly cycles (168 hours). Based on our experience and intuition the weekly cycle has stronger influence on traffic volume variations. Hence, we choose periodic length s = 168.

6.2.5.2 Finding pattern for the seasonal component

Figure 6.2 (a) shows that the ACF has a large spike at lag 168 below the x axis, while Figure 6.2 (b) shows that PACF decays to zero, alternating between lags 168 and 336. As a result, the seasonal components can be modelled as an MA (1) process, with P = 0 and Q = 1.



(a)



(b)

Figure 6.2 ACF (a) and PACF (b) of the seasonal differenced traffic. The data was collected from 2002-11-01 to 2002-12-13.

6.2.5.3 Finding pattern for the non-seasonal component

The procedure used in Sections 6.2.5.1 and 6.2.5.2 was applied to examine the patterns at the beginning of ACFs and PACFs within a season. ACF shown in Figure 6.2(a) has a exponential decay, while the PACF shown in Figure 6.2(b) displays a single

spike at lag 1 suggesting an AR (1) process for its non-seasonal components, with p = 1 and q = 0.

Finally, an ARIMA $(p,d,q)\times(P,D,Q)_s$ model was identified by combining results from the above three steps described in Sections 6.2.5.1 to 6.2.5.3. For the analyzed billing records, the model is $(1,0,0)\times(0,1,1)_{168}$.

6.2.6 Traffic prediction

After the order of the model has been determined, the parameters ϕ and θ in equation (6.5) can be obtained by using a maximum likelihood estimation. An one-step ahead forecast based on equation (6.5) is :

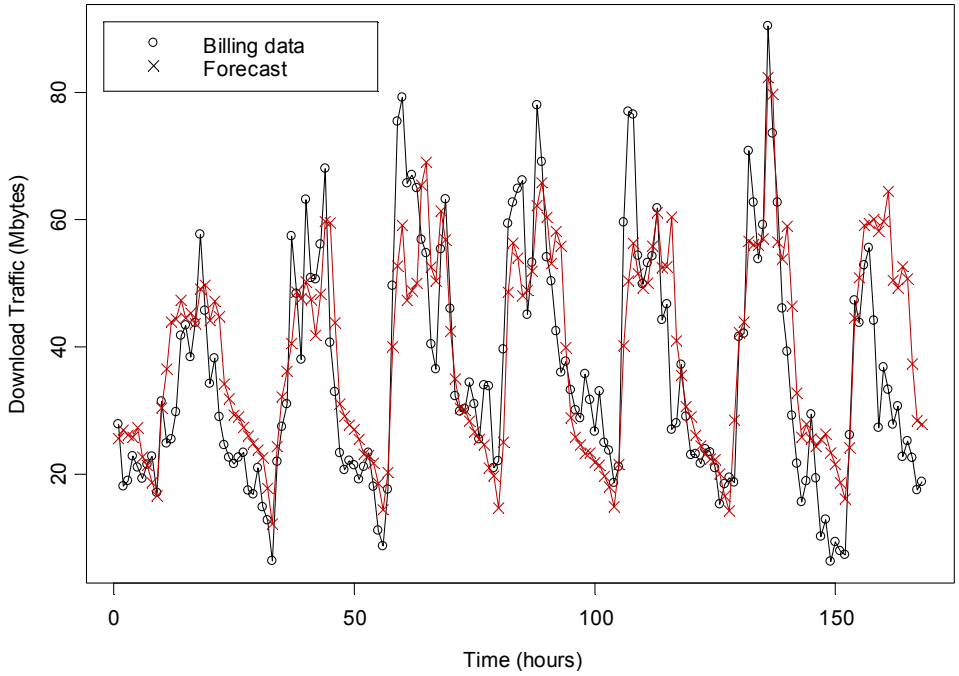
$$\bar{y}(t+1) = E[y(t+1) \mid y(t), y(t-1)\cdots]. \tag{6.8}$$

The n-step ahead predictions can be bootstrapped by incrementally stepping forward n time steps. Splus [25] maximum likelihood function and forecast function were used to obtain the parameter estimation and forecast [36], and to obtain the results shown in Table 6.2. The prediction performances for the uploaded and downloaded bytes are very different. As shown in Figure 3.1, the uploaded and downloaded traffic expressed in terms of packets are almost identical, which explains the similar prediction performance.

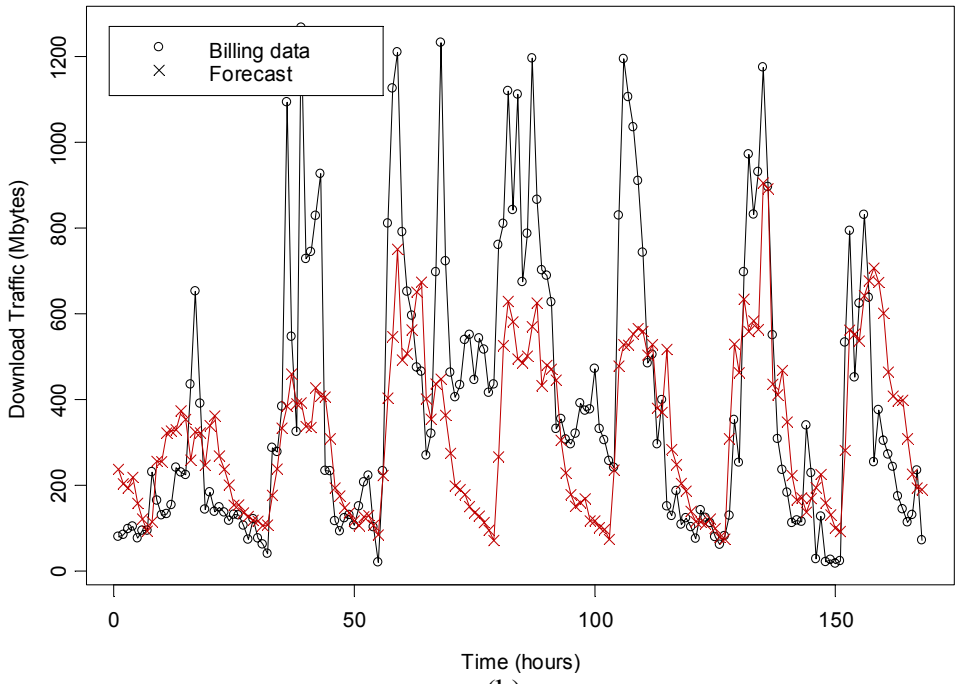
Since capacity planning does not use packets to describe the traffic bandwidth, we focus on analysis of traffic in terms of uploaded and downloaded bytes. Figure 6.3 shows that the predictor cannot capture the bursts of the downloaded byte traffic. This is because the traffic is highly asymmetric when examined in terms of bytes. The highly bursty downloaded traffic (in byte) makes prediction more difficult compared to the uploaded traffic (in byte).

Traffic type	Uploaded (Mbytes)	Downloaded (Mbytes)	Uploaded (packets)	Downloaded (packets)
NMSE	0.3653	0.5988	0.5268	0.5244

Table 6.2 Prediction performance for the different traffic. The data was collected from 2002-12-14 to 2002-12-20.



(a)



(b)

Figure 6.3 One week ahead prediction for uploaded bytes (a) and downloaded bytes (b). The data was collected from 2002-12-14 to 2002-12-20.

6.3 Using wavelet-linear method to predict short-term traffic volume

6.3.1 Introduction to short-term prediction

With the increase of multimedia application, a significant amount of reserved bandwidth is required in order to support real-time quality of service (QoS). Most problems are related to the signalling at per connection level. In early 1990, new mechanisms such as resource reservation protocols (RSVP) have been designed. They are designed for signalling requirements of an application to the network and setting up resource reservation along the path. Such types of signalling are usually initiated by the end system. To achieve the specified QoS, the edge router on the road will implicitly establish dedicated connections through the network. Similar concept and mechanism were also incorporated in multi protocol label switch (MPLS) networks.

In Chapter 5, we have shown that the TCP connection arrival process follows a heavy-tailed distribution. This implies that the connection arrival process is self-similar. The burst arrival process consumes large amount of network and central processing unit (CPU) resources necessary to execute the routing algorithm. As we described in Chapter 2.2, IP spoofing and TCP splitting techniques make the primary router of NOC an edge router, which has considerable memory requirement. If the traffic demand can be predicted ahead of the application, it will help the design of the primary edge router such as admission control, buffer management and the caching setup of the proxy server. Finally, it will help the system archive the desired QoS, reduce response time, and better balance the load. For this purpose, short-term prediction for TCP connection is highly valuable.

We employed a recently proposed wavelets related prediction method [3], [22] . The signal is first decomposed into different frequencies by employing redundant wavelet transform. We then apply linear prediction on each component and finally obtain the overall prediction by recombined those separate predictions. Intuitively, different components capture different characteristics of the signal. Finer scale or high frequency coefficients can be used to predict the short-term trends (near future), while coarser scale corresponds to low frequency coefficients can be used to predict the long-term trends. This approach utilizes the wavelets' ability to reveal features on the individual scales, which are hidden in the signal. Therefore, it is easier for predictor to predict periodic information on individual scales.

Unlike long-term prediction, short-term prediction usually requires fast algorithms because they are performed on a small time scale. In this thesis, we predict the traffic at 10 seconds resolution. As shown in Figure 6.4, the AR (autoregressive) model is selected as the linear prediction model because it is the simplest linear model.

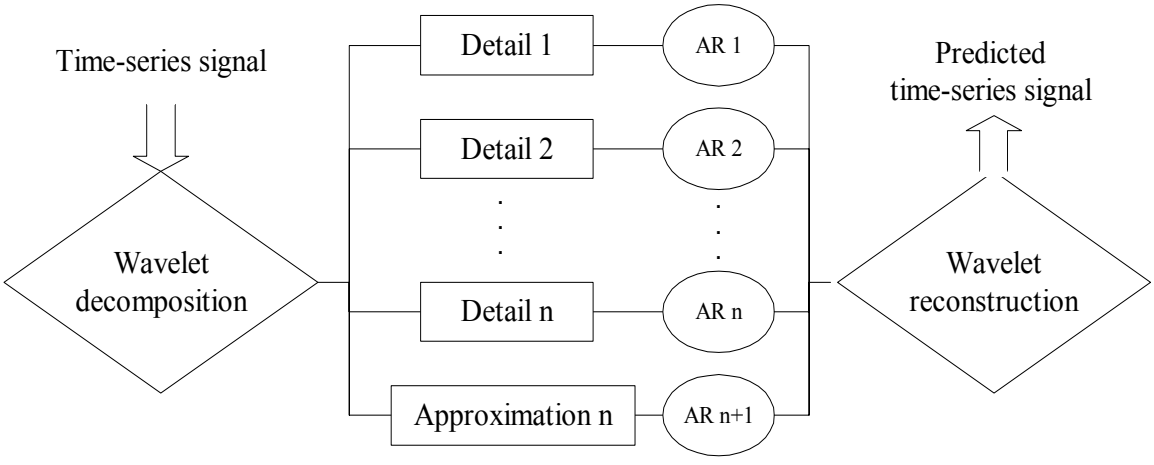


Figure 6.4 Combined wavelet-linear prediction method. AR linear predictors are applied to decomposed coefficients. The overall prediction is the aggregation of separate predictions.

6.3.2 Redundant wavelet

In network traffic modelling, the signal is discrete by default. Hence, Discrete Wavelet Transform (DWT) is usually used to analyze traffic data. As described in Section 4.4.5.1, calculating DWT expansions directly by matrix inversion is computationally intensive. In the mid-1980s, Mallat introduced his remarkably fast pyramid algorithm [11]. This algorithm is performed by passing the signal through a series of low pass and high pass filters, with discarding of one sample of every two, as shown in Figure 6.5 [27]. The DWT has many advantages in the area of signal compression. With DWT technique, a large proportion of the coefficients of the transform can be set to zero without loss of information.

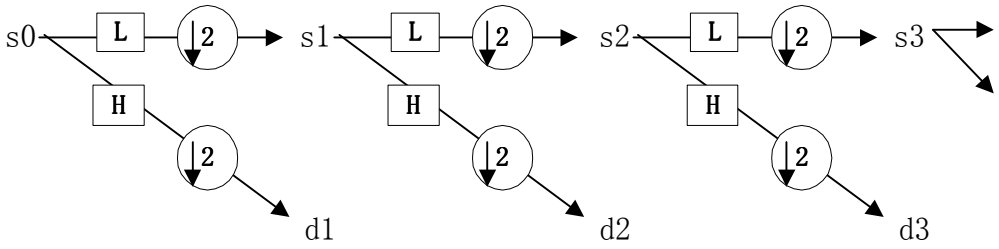


Figure 6.5 DWT pyramid algorithms. L indicates the low pass filter. H indicates the high pass filter. Filters are followed by a down-sampling operation.

When DWT is applied to multi-resolution time series analysis, it poses certain difficulties. First, it is somewhat difficult to graphically relate information at a given time instance at the different scales, as shown in Figure 6.6 (left). As we examine the decomposed signal from the finer to coarser scales, we become less certain about where the signal occurs in time because there are fewer coefficients. Second, DWT suffers from a lack of shift invariance [3]. By down sampling, the resulting wavelet coefficients become highly dependent on which coefficients remain. Therefore, a small change of the

input signal will finally lead to large changes for the coarser coefficients and possibly large changes in the reconstructed waveforms. At the expense of a greater storage requirement, this problem can be solved by employing a non-decimated or redundant wavelet transform.

The non-decimated (or redundant) wavelet is a non-orthogonal variant of the classical DWT. Using the decimated DWT, one can end up with n coefficients from n sampled signal. Unlike the classical DWT, which has fewer coefficients at coarser scales, each scale for the non-decimated DWT has n coefficients. Hence, there are (m+1)*n coefficients overall when redundant transformation is performed on m scales. Therefore, information at each resolution scale is directly related at each time point, as shown in Figure 6.6 (right).

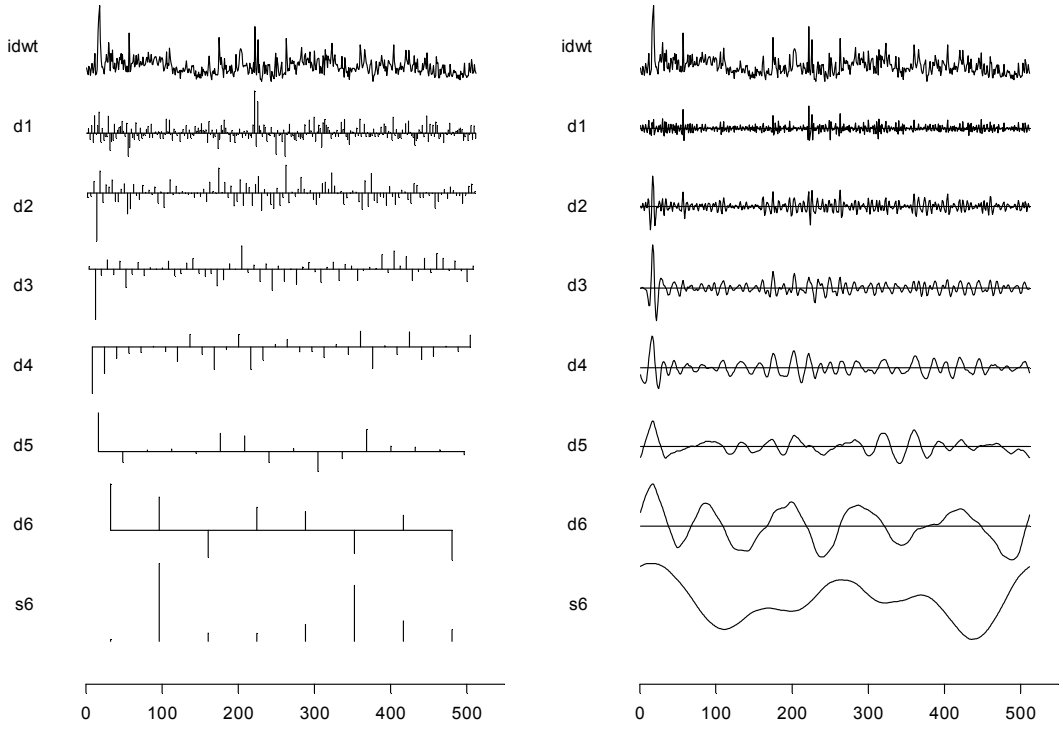


Figure 6.6 Comparison of decimated transforms (left) and non-decimated transforms (right). idwt: original signal, d1-d6: detail coefficients of five scales, s6: approximation coefficients at scale 6. TCP connection traffic data was collected between 15:00 and 19:10 on 2002-12-22.

6.3.3 Atrous Algorithm

An alternative non-decimated wavelet transform is given by the "atrous" ("black hole") algorithm [3], [34]. There are four steps for atrous transform:

Step 1: The finest scaling coefficients $s_0(k)$ are defined by $y(k)$, which is the the time series sample at time k:

$$s_0(k) = y(k). \tag{6.9}$$

Step 2: Calculate the other scaling coefficients as follow:

$$s_i(k) = \sum_{l=-\infty}^{l=\infty} h(l) s_{i-1}(k + 2^{i-1} l). \tag{6.10}$$

A diagram of atrous transform is shown in Figure 6.7 for three levels of wavelet decomposition. The distance between s_{i-1} used for computing $s_i(k)$ increases as the scale increases. This distance between samples of s_{i-1} that are used for computing $s_i(k)$ can be visualized as holes created between them, hence the name "atrous" algorithm comes. A B_3 spline, $(\frac{1}{16},\frac{1}{4},\frac{3}{8},\frac{1}{4},\frac{1}{16})$ is used for the low-pass filter h in equation (6.10). As shown in equation (6.9) and Figure 6.7 indicates that there is a boundary effect when calculating $s_1(N)$ for time series up to time N. $s_0(N+1)$ and $s_0(N+2)$ are needed in order to calculate the $s_1(N)$. Since we only have samples up to time N, those two samples need to be predicted. At this stage, we use a general boundary handling by employing the mirror treatment, where $y(N-k) = y(N+k)$ [17].

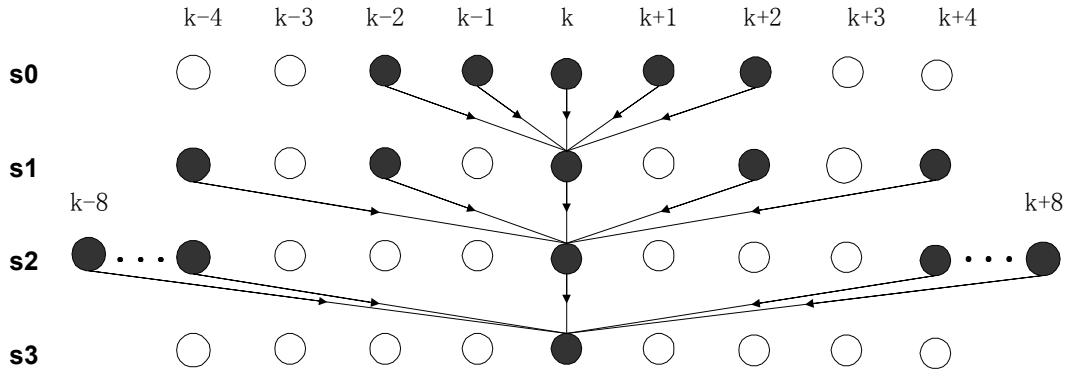


Figure 6.7 The scaling coefficients are calculated from the previous coefficients. The distance between previous coefficients increases as the scale increases.

Step 3: The wavelet coefficients that contain the detail are obtained as:

$$d_i(k) = s_{i-1}(k) - s_i(k). \tag{6.11}$$

Step 4: Continue the Steps 2 and 3 until *i* reaches the specified scale level.

The inverse transform is given by

$$y(k) = s_p(k) + \sum_{j=1}^{p} d_j(k) \quad \text{k=1, 2, ..., N.} \tag{6.12}$$

Figure 6.8 illustrates that the scale coefficients become smoother as scale increases. Low-pass filtering of a traffic trace also improves the predictability because it removes rapid fluctuations of the signal around mean value. Hence, we can expect that we will get better predictability at coarser scales.

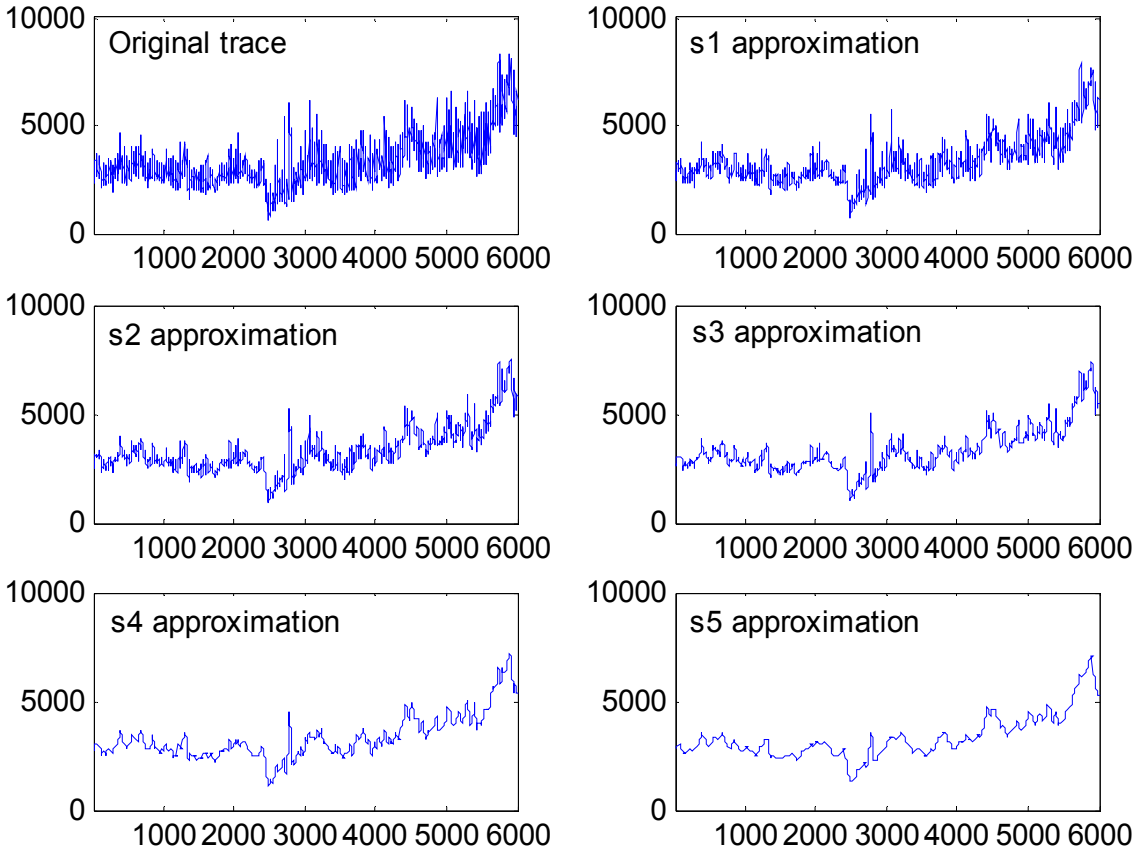


Figure 6.8 Inverse atrous transform. The approximation coefficients become smoother as the scale increases. s1-s5 are approximation coefficients from scale 1 to scale 5. The packet traffic data was collected from 12:00-13:40 on 2002-12-22.

6.4 Evaluate the prediction of short-term traffic

6.4.1 Performance of the prediction

Three-step ahead prediction for five scales is shown in Figure 6.9. AR predictor performs differently at the five scales. The sub-trace get more bursty from (a) to (e). Analysis in Section 6.3.2 indicates that the performance gradually improves on coarser scale. The original trace and its three-step ahead prediction are shown in Figure 6.10.

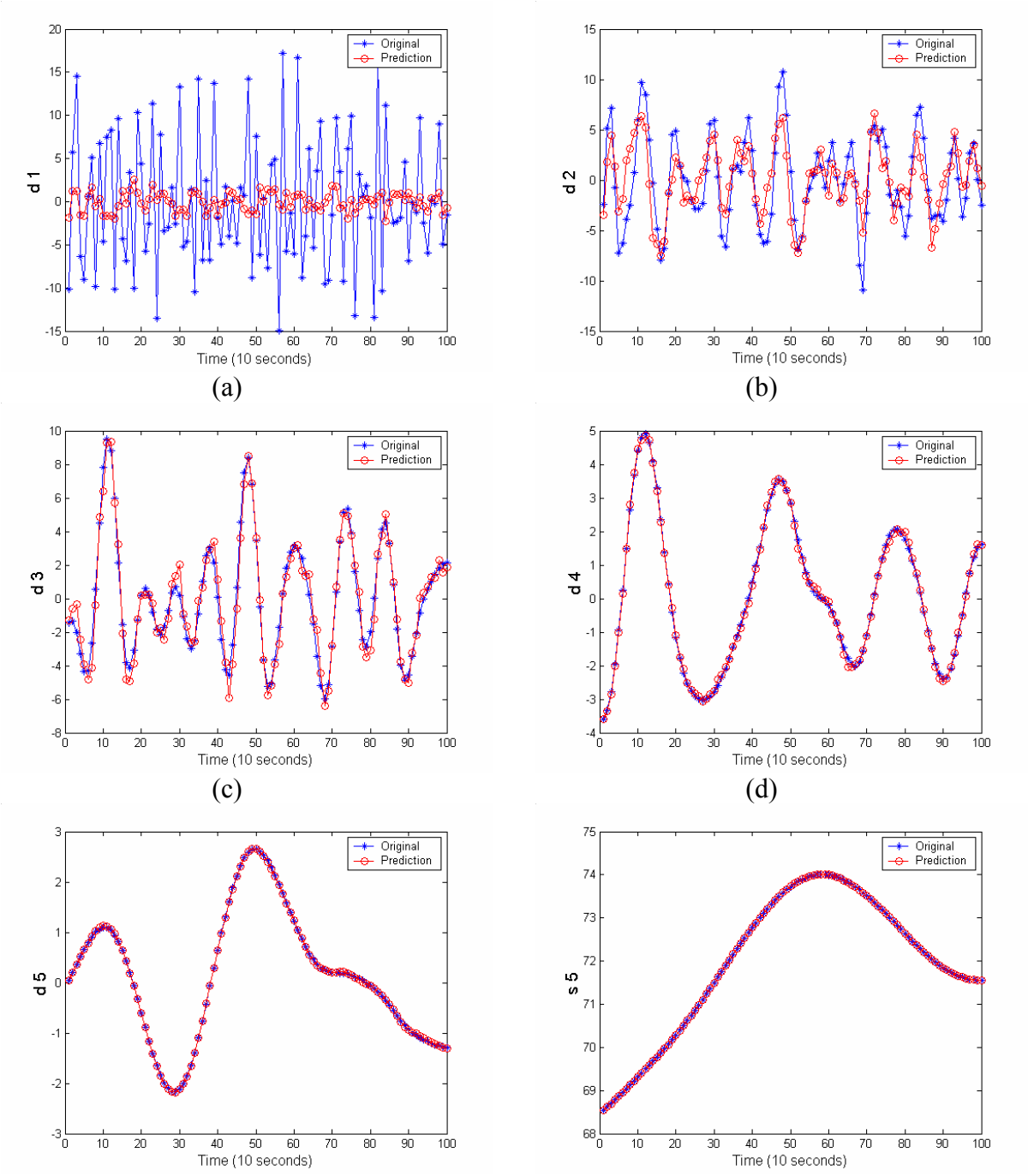


Figure 6.9 NMSE for coefficients on various scale. From the finest time scale (a) to the coarsest time scale (e), the performance of prediction greatly improves. TCP connection traffic data was collected during 12:00 to 12:17 on 2002-12-22 with the 10-second granularity.

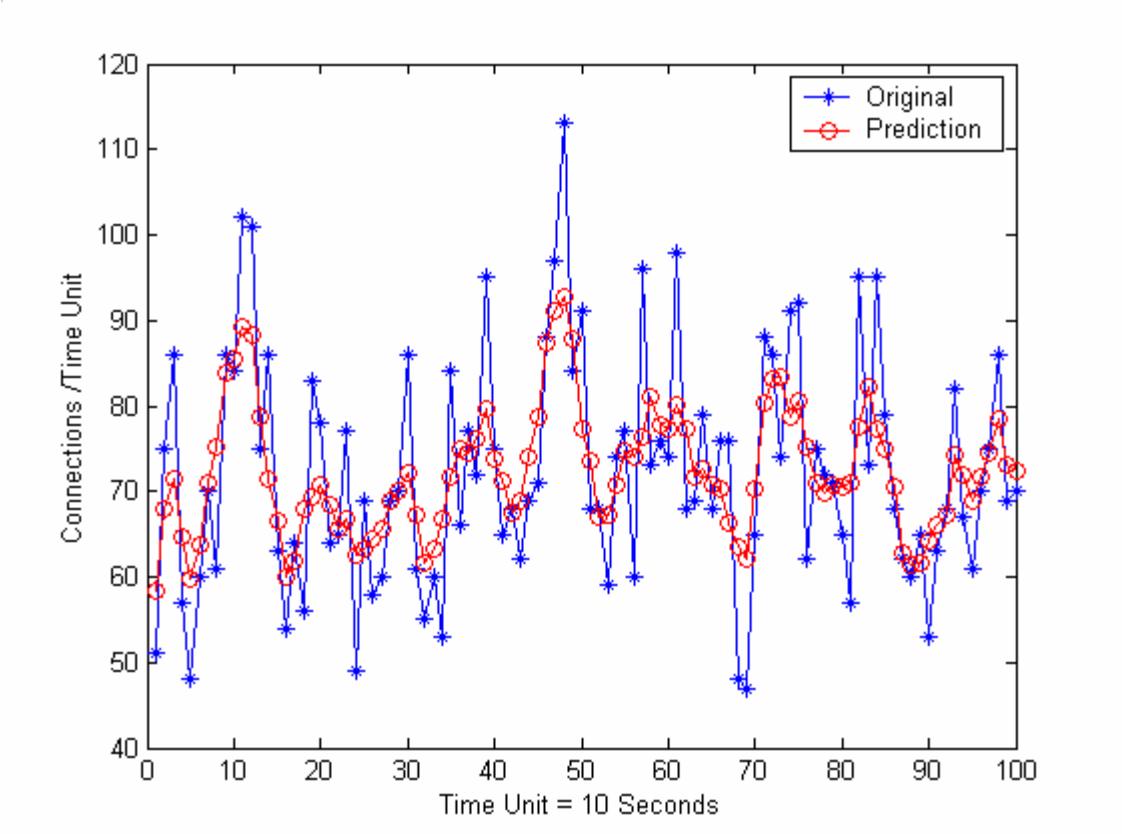


Figure 6.10 Prediction using AR+atrous method. TCP connection traffic was collected during 12:00 to 12:17 on 2002-12-22 with the 10-seconds granularity.

Table 6.3 shows NMSE for various scales. The combined method performs much better than the direct AR method.

Predictor	D1	d2	d3	d4	d5	s5	signal
AR+atrous	0.9854	0.3711	0.0380	0.0019	1.43e-004	5.3e-006	0.4261
AR	-	-	-	-	-	-	1.2654

Table 6.3 NMSE for different scales obtained by combined method and AR predictor.

As shown Figure 6.12, different scale coefficients capture different underlying characteristics of the data; coarser scale coefficients capture long-dependent characteristic from data. Figure 6.6 indicates that computing the last element of $s_1(k)$ in

equation (6.10) requires $s_0(k-2)$, $s_2(k)$ requires $s_1(k-2^2)$, and $s_3(k)$ requires knowledge of $s_2(k-2^3)$. The coarser scale coefficients are calculated, the larger number of previous finer scale coefficients are required. For example, at time k, predicting one-step ahead value *x(k+1)* requires predicting from $s_5(k+1)$ to $d_1(k+1)$ and finally combining them. If the prediction length is 15, $s_5(k-15)$, $s_5(k-14)$, … and $s_5(k-1)$ need to be included in order to predict $s_5(k+1)$. $s_5(k-15)$ will depend on $s_4(k-15-2^5)$, while $s_4(k-15-2^5)$ will further depend on $s_3(k-15-2^5-2^4)$. Thus $s_5(k-15)$ will depend on $s_0(k-15-2^5-2^4-2^3-2^2-2^1)=s_0(k-77)=x(k-77)$. This explains why coarser coefficients can capture longer dependence from the data. Therefore, B_3 spline atrous transform with AR prediction length of 15 implies that it utilizes the information up to 77 previous samples. To the contrary, direct implementation of AR of length 15 only includes information from the past 15 observations. This is the reason why the combined method has better performance.

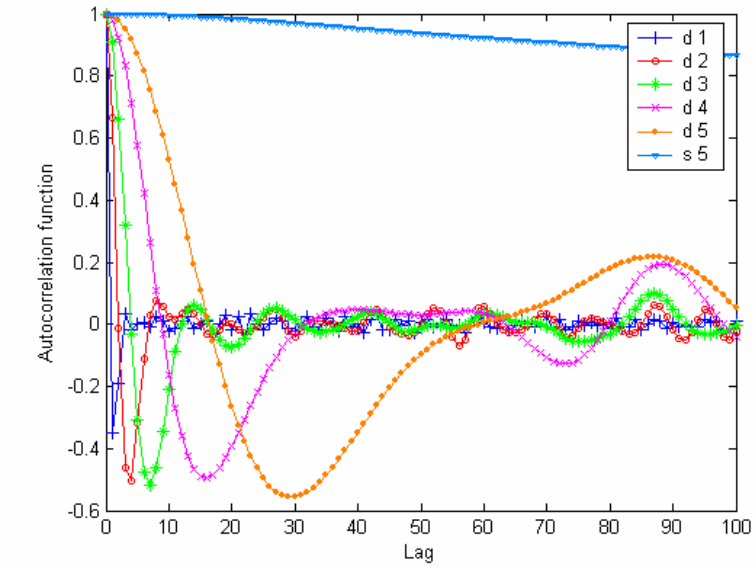


Figure 6.11 ACF of the different scale coefficients.

6.4.2 Performance of the predictor versus the length of the predictors

Although the combined AR+atrous method can improve the predictability, its cost is additional storage. The longer the predictor length is, the more storage is needed. Therefore, it is important to investigate the influence of prediction length. We employed linear AR predictor with and without atrous transform and calculated the NMSE for various predictor lengths.

The performance of the two predictors with varying lengths of the predictor is shown in Figure 6.12. The predictability does not improve significantly after certain predictor length. In the collected traffic trace, the ideal predictor length is approximate 15. This implies that including more historical traffic data does not necessarily improve the performance of NMSE. To improve the predictability, only a certain amount of past data is required to be stored. This result also concurs with previously reported performance of the linear AR predictor for self-similar traffic [21].

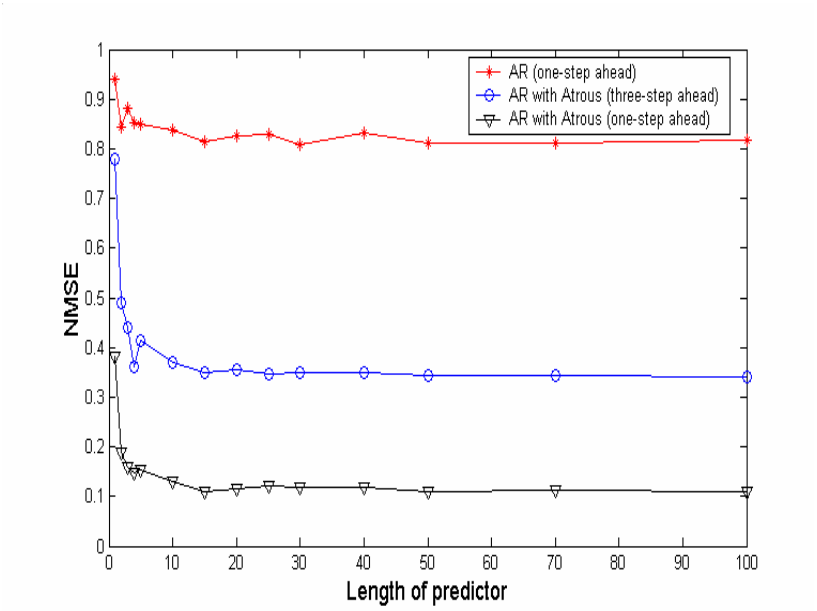


Figure 6.12 Performance of the predictors versus the length of the predictor.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

7.1 Conclusion

In this thesis, we followed three steps in traffic measurement and analysis: traffic collection, traffic characterization and modelling, and traffic prediction.

7.1.1 Traffic collection

Collecting traffic is an important part of traffic engineering. Choosing suitable collection tools and finding the suitable collecting location are essential. Traffic collection procedure is time consuming. The data requires large storage and need powerful database server to be processed. The granularity of the trace is critical factor for further analysis. In this thesis, we describe the traffic collection from a DirecPC system. One month of traffic trace was collected by *tcpdump* software from the primary router at the network operation center, and two month of billing records were collected from billing system.

7.1.2 Traffic characterization and modelling

We first analyzed the billing records. The traffic data show a diurnal circle and a highly asymmetric traffic transmission pattern. The collected traffic traces were then characterized based on application, protocol, connection, and packet levels. The Web application and TCP protocols dominate the traffic. We further investigated the frequency-rank relation of client's connections in the collected traffic trace. We found that this relation can be modelled by the DGX distribution. We also explained why the known Zipf law failed. This result indicates that traffic is non-uniformly distributed among the hosts. We concluded that the language, geographic, and commercial elements

greatly influence the distribution of traffic data. Our analysis shows that packet size distribution is bimodal and. most bytes are sent in large packets.

We estimated the Hurst parameter using various estimators and under different link utilizations. We also addressed the issue of non-stationarity in investigations of the self-similar behaviour of traffic traces. Because the DirecPC system utilizes the TCP splitting technique, we modelled the traffic on the TCP connection level. After evaluating the fitness of the models, we concluded that Weibull and lognormal distributions are ideal for modelling the TCP inter-arrival times and the number of downloaded bytes, respectively.

7.1.3 Traffic prediction

In this thesis, we also performed long-term and short-term traffic prediction through different approaches. We used ARIMA model to predict long-term traffic. The result shows that downloaded traffic have more difficult to predict than to predict uploaded traffic because downloaded traffic have larger volume and is more bursty. By combining wavelet with AR model, we performed short-term traffic prediction for the TCP connection data. This approach greatly improved the prediction performance. Finally, we investigated the predictability of this approach and concluded that the prediction performance does not improve beyond certain prediction length.

7.2 Future work

One possible future direction of this work is to simulate the hybrid satellite-terrestrial network, use the derived TCP-connection model as input trace, and compare

the throughput and queue size with the collected traffic trace as input. It would be of interest to investigate the effects of various parameters on the system performance.

In this thesis, only wavelet-linear predictor was implemented for short-term traffic prediction. However, the linear models may be overly simplified for some traffic data. It would be ideal to use nonlinear models to describe more complex observations. The neural network is a promising approach to model and predict the nonlinear and non-stationary data. The idea of neural network is to mimic how human brains process the information. The brain is a multi layer structure that works as a parallel computer. The neuron receives information through a number of inputs nodes. Those inputs are then multiplied by the weights that are adjusted through the data training. The outputs of neural networks can model very complex system. The parallel and hierarchical characteristics of neural networks make them ideal for implementation using VLSI technology. Therefore, we can expect that neural network related approaches may be implemented for real-time application such as signal processing and short term prediction.

Researchers are combing wavelet transform and neural network approaches to perform prediction on time series data. Wavelet packet [33] and time delay recurrent network [4] are promising approaches in this direction.

REFERENCE LIST

[1] P. Abry and D. Veitch, "Wavelet analysis of long-range dependent traffic," *IEEE Transactions on Information Theory*, vol. 44, no. 1, pp. 2-15, January 1998.

[2] R. J. Adler, R. E. Feldman and M. S. Taqqu(editors), *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*. Boston, MA: Birkhauser, 1998, pp. 27-52, pp. 177-217.

[3] A. Aussem and F. Murtagh, "Web traffic demand forecasting using wavelet-based multiscale decomposition," *International Journal of Intelligent Systems*, vol. 16, no. 2, pp. 215-236, 2001.

[4] A. Aussem and F. Murtagh, "Combining neural network forecasts on wavelet-transformed time series," *Connection Science- special issue on Combing Neural Nets*, vol. 9, no. 9, pp. 113-121, 1997.

[5] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in Web client access patterns: characteristics and caching implications" *World Wide Web, Special Issue on Characterization and Performance Evaluation*, vol. 2, no. 1-2, Blacksburg, VA, pp. 15-28, January 1999.

[6] Z. Bi, C. Faloutsos, and F. Korn, "The "DGX" distribution for mining massive, skewed data," in *Proc. of 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, August 2001, pp. 17-26.

[7] G. Box, and G. Jenkins, *Time Series Analysis: Forecasting and control* 2rd ed. Oakland, CA: Holden-day, 1976, pp. 208-329.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proc. of IEEE INFOCOM*, March 1999, pp. 126-134.

[9] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM Trans. on Networking*, vol. 5, no. 6, pp. 835-846, December 1997.

[10] R. B. D'Agostino and M. A. Stephens(editors), *Goodness-of-Fit Ttechniques*. New York NY: Marcel Dekker, 1986. pp. 63-93, pp. 97-145, pp. 421-457.

[11] I. Daubechies, *Ten lectures on Wavelets*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999, pp. 104-144.

[12] A. Feldmann, "Characteristics of TCP connection arrivals," in *Self-Similar Network Traffic and Performance Evaluation*, K. Park and W. Willinger(editors). New York: Wiley, 2000, pp. 367-399.

[13] T. B. Fowler, "A short tutorial on fractals and Internet traffic," *The Telecommunications Review,* vol. 10, pp. 1-15, 1999.

[14] Free packet sniffer software [Online]. Available:

http://netsecurity.about.com/cs/hackertools/a/aafreepacsniff.htm

[15] V. Jacobson and R. Braden "TCP extensions for long-delay paths," Request for Comments: 1072, 1998.

[16] T. Karagiannis, M. Faloutsos, and R.H. Riedi "Long-range dependence: now you see it, now you don't!," in *Proc. GLOBECOM '02*, Taipei, Taiwan, November 2002, pp. 2165–2169.

[17] A. Kato, J. Mural, S. Katsuno and T. Asami. "An Internet traffic data repository: the architecture and the design policy," in *Proc. 9th Annual Conference of the Internet Society*, San Jose, CA, June, 1999.

[18] Kernel traffic [Online]. Available:

http://www.kerneltraffic.org/kernel-traffic/kt19990304_8.html

[19] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, no. 1, pp. 1-15, February 1994.

[20] S. McCreary, "Trends in wide area IP traffic patterns," in *Proc. of 13th ITC Specialist Seminar on Measurement and Modelling of IP Traffic*, Monterey, California, September 2000, pp. 1-11.

[21] S. Ostring and H. Sirisena, "The influence of long-range dependence on traffic prediction," in *Proc. IEEE Int. Conf. Communication,* June 2001, vol. 4, pp. 1000-1005.

[22] D. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-term forecasting of Internet backbone traffic: observations and initial models," in *Proc. IEEE INFOCOM 2003*, San Francisco, CA, April 2003, pp. 1178-1188.

[23] V. Paxson, "Empirical derived analytic models of wide-area TCP connections." *IEEE/ACM Trans. Networking*, vol. 2, no. 4, pp. 316-336, August 1994.

[24] V. Paxson and S. Floyd, "Wide-area traffic: the failure of poisson modelling," *IEEE/ACM Transactions on Networking,* vol. 3, no. 3, June 1995, pp. 226-244.

[25] S-PLUS [Online]. Available:

http://www.insightful.com/products/splus/default.asp.

[26] M. A. Stephens, private communications.

[27] G. Strang and T. Nguyen, *Wavelets and Filter Banks*. Wellesley MA: Wellesley-Cambridge Press, 1997, pp. 189-193.

[28] TCPDUMP [Online]. Available: http://www.tcpdump.org/tcpdump_man.html

[29] V. Teverovsky, Estimation methodologies [Online]. Available:

http://math.bu.edu/people/murad/methods/

[30] The Internet Traffic Archive [Online]. Available: http://ita.ee.lbl.gov/

[31] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network Magazine*, vol. 11, no. 6, pp. 10-23, November 1997.

[32] D. Veitch and P. Abry, "A statistical test for the time constancy of scaling exponents," *IEEE Trans. on Signal Processing*, vol. 49, no. 10, pp. 2325-2334, October 2001.

[33] L. Wang, K. K. Teo, and Z. Lin. "*Predicting time series with Wavelet packet neural networks*," in *Proc. International Joint Conference on Neural Networks*, Washington, DC, July 2001, vol. 3, pp. 1593-1597.

[34] P. Yu, A. Goldberg, and Z. Bi, "Time series forecasting using wavelets with predictor-corrector boundary treatment," in *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, August 2001.

[35] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. "On the constancy of Internet path properties," in *Proc. of ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, November 2001, pp. 197-211.

[36] E. Zivot and J. Wang, *Modelling Financial Time Series with S-Plus.* Seattle, WA: Insightful Corporation, 2003, pp. 57-103.