Fractal Network Traffic Analysis with Applications

A Dissertation

by

Jian Liu

In Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy in the
School of Electrical and Computer Engineering at
Georgia Institute of Technology

August, 2006

Fractal Network Traffic Analysis with Applications

Approved by:

Professor John A. Copeland, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor G. K. Chang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor James McClellan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Erik Verriest
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Mostafa H. Ammar
College of Computing
Georgia Institute of Technology

Date Approved: May 3, 2006

To my parents

ACKNOWLEDGEMENTS

This thesis is dedicated to my parents for their compassionate support and encouragement. Although we are thousands of miles apart, they are with me everyday. This thesis is also partially theirs. The work helps us understand each other better.

I am greatly indebt to my advisor Prof. John Copeland for his patience, his financial support, and the advice he gave his students personally, and in every group meeting for more than seven years.

I wish to thank my committee, Prof. Mostafa Ammar, Prof. G-K. Chang, Prof. James McClellan, and Prof. Erik Verriest. I thank Prof. Chris Hill, Prof. Gerd Mockenhaupt, and Prof. Brani Vidakovic for their instruction on solving math problems.

I also want to thank the engineers from Scientific Atlanta that I had coordinated with in my research projects on broadband networks in the past few years. They are Dr. Arturo Rodriguez, Dr. Bill Wall, Mark Schutte, Tom Heagy, and Ryan Pabis; also Dr. Tam-Amh Chu, Dr. Clements Leung, Prof. Jack Dennis of MIT, with whom I worked and learned about Gigabit network processor processing and designs in Acorn Networks; and also Mike Coffee, CEO of Commetrex, with whom I worked on media network processing.

Last but not least, I thank many of my colleagues and friends from the research groups of CSC, CSIP, the CS networking group, and the ISYE statistical group. I was fortunate to work with you and to learn from you. Particularly, I would like to mention Ms. Kathy Cheek, my administrator in CSC, who has helped me with tedious paper work and has done a wonderful job, and my friend Jonathan Atteberry, who helped me with my English, which I am still learning.

Finally, the completion of a thesis is not an end of anything but a beginning of everything. For years ahead, I believe there are many engineering and scientific problems to be solved using fractals, which motivates me to continue this work.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledgement of a data packet reception.
AR	Auto-regression.
BC	Binomial cascade process.
CM	Cable modem station.
DWT	Discrete wavelet transform.
DS	Downstream channels in cable systerms.
EB	Efficient bandwidth. It is a fractal measure on traffic bandwidth.
FARIMA	Fractional auto-regressive integrated moving average process.
FBM	Fractional Brownian motion process.
FGN	Fractional Gaussian noise process.
_H_-sssi	Hurst-parameterized self-similar process with stationary increments.
HE	Head-end system in cable networks.
HFC	Hybrid fibre coaxial cable.
H-T	Heavy-tailedness.
HTTP	Hypertext Transfer Protocol.
IFS	Iterated functional system.
IWM	Independent wavelet model for network traffic
LAN	Local area network.
LBL	Lawrence Berkeley Laboratory.
LRD	Long-range dependence. It is referred to the traffic scaling property in timescales above a second approximately.
M/G/∞	A queueing model for network traffic.
MAC	Medium access control layer in network protocol.
MA	Moving average.
MWM	Multifractal wavelet model for network traffic.
NACK	Acknowledgement of a packet loss for retransmission.
NS	Network simulator.
PDF	Probability density function.
QoS	Quality of services.
QPSK	Quadrature phase shift keying modulation.

RTT	Round Trip Time.
SCRA	Server-initiated collision resolution and avoidance scheme.
SRD	Short-range dependence. It is referred to the traffic scaling property in timescales below a second approximately.
TCP	Transmission Control Protocol.
UDP	User Datagram Protocol.
US	Upstream channel in cable networks.
WAN	Wide area network.
WMFA	Wavelet multifractal analysis.
WMM	Wavelet modulus maxima.

SUMMARY

Today, the Internet is growing exponentially, with traffic statistics that mathematically exhibit fractal characteristics: self-similarity and long-range dependence. With these properties, data traffic shows high peak-to-average bandwidth ratios and causes data networks inefficient. These problems make it difficult to predict, quantify, and control data traffic, in contrast to the traditional Poisson-distributed traffic in telephone networks. In this thesis, two analytical methods are used to study fractal network traffic. They are second-order self-similarity analysis and multifractal analysis. Using a number of experiments, the following results towards characterizing and quantifying the network traffic processes have been achieved:

First, self-similarity is an adaptability of traffic in networks. Many factors are involved in creating this characteristic. A new view of this self-similar traffic structure is provided. This view is an improvement over the theory used in most current literature, which assumes that the traffic self-similarity is solely based on the heavy-tailed file-size distribution.

Second, the scaling region for traffic self-similarity is divided into two timescale regimes: short-range dependence (SRD) and long-range dependence (LRD). Experimental results show that the network transmission delay (RTT time) separates the two scaling regions. This gives us a physical source of the periodicity in the observed traffic. Also, bandwidth, TCP window size, and packet size have impacts on SRD. The statistical heavy-tailedness (Pareto shape parameter) affects the structure of LRD. In addition, a formula to quantify traffic burstiness is derived from the self-similarity property.

Furthermore, studies of fractal traffic with multifractal analysis have given more interesting and applicable results. (1) At large timescales, increasing bandwidth does not improve throughput (or network performance). The two factors affecting traffic throughput are network delay and TCP window size. On the other hand, more simultaneous connections smooth traffic, which could result in an improvement of network efficiency. (2) At small timescales, traffic burstiness varies. In order to improve network efficiency, we need to control bandwidth, TCP window size, and network delay to reduce traffic burstiness. There are the tradeoffs from each other, but the effect is nonlinear. (3) In general, network traffic processes have a Hölder exponent α ranging between 0.7 and 1.3. Their statistics differ from Poisson processes.

To apply this prior knowledge from traffic analysis and to improve network efficiency, a notion of the *efficient bandwidth*, EB, is derived to represent the fractal concentration set. Above that bandwidth, traffic appears bursty and cannot be reduced by multiplexing. But, below it, traffic is congested. An important finding is that the relationship between the bandwidth and the transfer delay is nonlinear.

CHAPTER 1

INTRODUCTION

If the greatest invention in telecommunication in the late 19th century was the telephone, the Internet would be the comparable successor in the 20th century. Since its public début about two decades ago, the Internet has grown exponentially in the number of users, the number of computers, and the amount of network traffic! In this 21st century, what's confronting us is the challenge to manage and to process network data more efficiently.

In 1994, Leland and Willinger et al. [1] reported that network traffic exhibited self-similarity – a fractal concept in data analysis. Fractal traffic analysis and modeling have ever since been a popular research topic in network engineering. Mandelbrot[32] first coined the term "fractal" three decades ago and studied natural phenomena using a multifractal model. In 1988, Barnsley pointed out there are "fractals everywhere" and developed a popular fractal image compression algorithm called the "*iterated functional system"* (IFS) [35]. However, applications of fractal algorithms in data networking area are limited to traffic modeling [19]. Practical fractal algorithms in traffic measurement and management have yet to be well developed. There are several issues related to this: first, the fractal concept is a somewhat new idea and very non-intuitive, in contrast to our previous knowledge of traffic measuring; second, the causes of the fractal nature of network traffic are not well understood; and third, it is very complex to verify the ideas behind this fractal behavior because of the rapid change of network technologies which affect traffic characteristics.

1.1 Fractality

Essentially, fractals use a mathematical measure in the form of

$$\mu = \delta^{\alpha} . \qquad (1.1)$$

Here, δ is the unit, and α is an exponent referred to as the local singularity (to be defined in Ch.4). With $\alpha = 1$, μ equals to δ in the traditional form, and the measure μ of an object is simply a sum in units of δ. However, with α non-unity, μ varies at each point. These values of α characterize a fractal set in a functional distributional sense expressed in $f(\alpha)$. So, we can write

$$\mu(t) = \delta^{\alpha(t)}, \alpha \subset f(\alpha) . \qquad (1.2)$$

In addition, the measure μ also has scalability such that varying $\delta = 2^{-n}$ as $n \to \infty$ will not have an effect on $f(\alpha)$.

1.1.1 Heuristic from Sierpinski triangle

In fact, a fractal describes a self-organizing mechanism. It reproduces itself iteratively, but it allows certain randomness without destroying its mechanism, i.e., its original form. An example is shown in Figure 1-1. The sub-figure ***A*** is a standard Sierpinski triangle. For each iteration, a triangle can spin off into three half-scaled small triangles and the middle one thus formed is removed from further divisions. This process can be repeated infinitely in theory. Therefore, generally speaking, any small portion of a fractal set is infinite in the fractal space. However, for most of us, only a measurable object has some practical meaning. So, for a Sierpinski triangle, we can find the remaining size of the triangle at a scale *n* as

$$S[n] = N_n(\Delta) \times s_n(\Delta) = 3^n \times 2^{-2n-1}. \tag{1.3}$$

Δ denotes a small-scaled triangle. $N_n(\Delta)$ is the number of the small triangles remaining at a scale *n*, and $s_n(\Delta)$ is the size of one small triangle. A Sierpinski triangle has a *fractal dimension* d_f, which can be found by

$$\left.\begin{aligned} N_n(\Delta) &= 3^n \\ N_n(\Delta) &= \delta^{-d_f} \end{aligned}\right\} \xrightarrow{\delta=2^{-n}} d_f = \log_2 3 = 1.585. \tag{1.4}$$

So, we get

$$S[n] = 2^{-0.415n-1}. \tag{1.5}$$

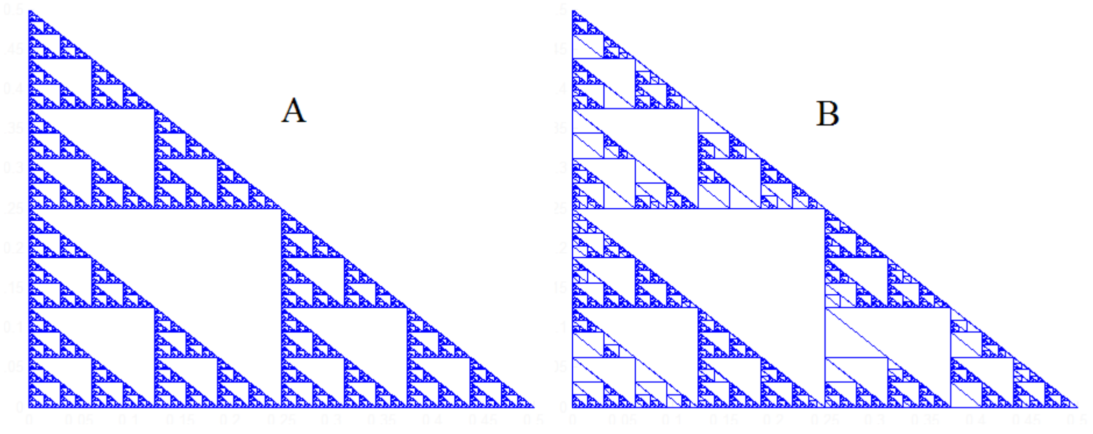


Figure 1-1 The Sierpinski triangles.

***A*. a normal Sierpinski triangle; *B*. a Sierpinski triangle with a stop probability 0.25 at each scale in iteration.**

The sub-figure ***B*** still looks like a Sierpinski triangle. However, if looking carefully you will find there are a number of missing triangles across many scales because a probability of stopping the triangles splitting is set. What is the size of this figure without counting the triangles in detail? In this example, the probability of stopping a spin-off is 0.25. Using (1.3) and (1.4), we find

$$d_f = 1.1699 \text{ and } S[n] = 2^{-0.8301n-1}. \tag{1.6}$$

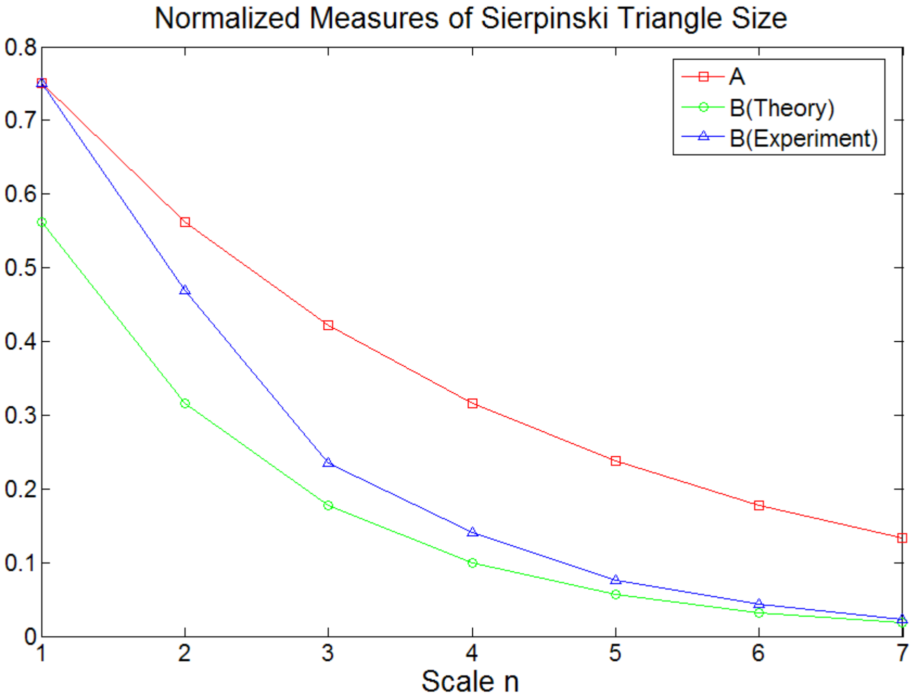


Figure 1-2 Normalized measures of Sierpinski triangle size for Figure 1-1 A and B

The Sierpinski triangles are empty sets as $n \rightarrow \infty$, but the sets have different characteristics. Figure 1-2 plots the equations of (1.5) and (1.6) for the triangles ***A*** and ***B***, respectively, and compares to a numerically-generated experimental plot. When scale n increases, the plot from the experiment approaches the plot of equation (1.6). All three plots will go to zero as the scale $n \rightarrow \infty$. A Sierpinski triangle with an infinite scale is empty even though it appears to have some complex structures. There are spaces inside. So, we can imagine that if a real system has a fractal behaviour with $\alpha < 1$, its efficiency will be affected.

1.1.2 Fractals in network traffic

In the case of network traffic, the “spaces” are unused bandwidth for a scaled interval of time (Figure 1-3). Similar to a Sierpinski triangle, traffic has different levels of “spin-offs”: at the packet level, transport layer protocols (TCP or UDP) dispatch packets in bursts; at the connection level, multiple objects (e.g., in a Web page) are correlated; and there are session-level traffic bursts as well. More detailed discussions of these mechanisms will be given in the later chapters.

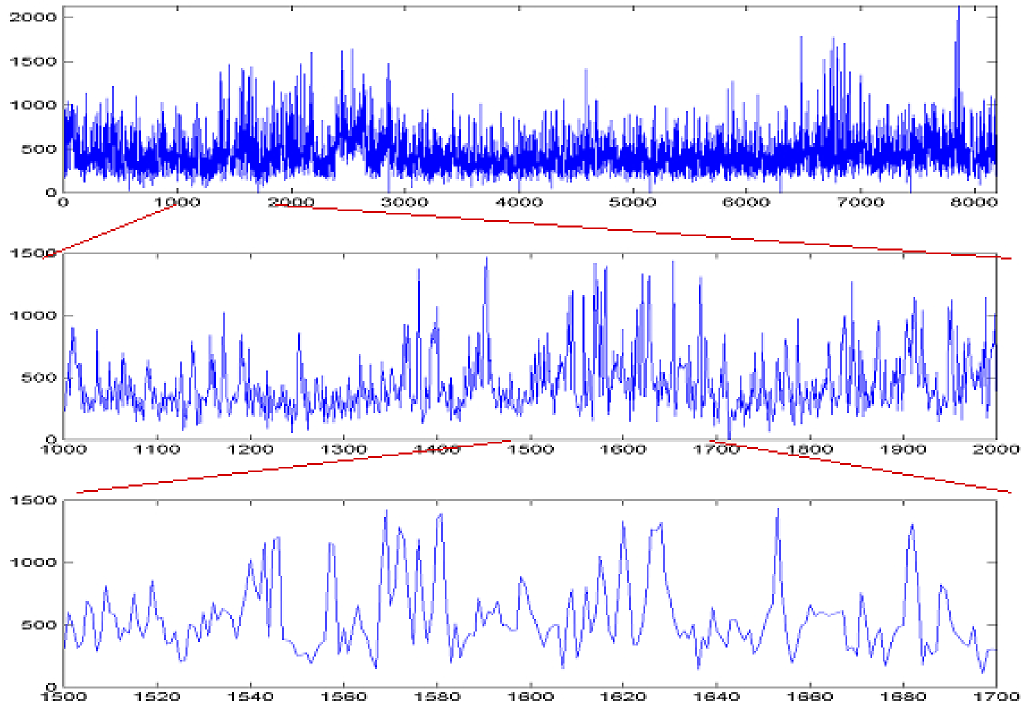


Figure 1-3 Fractal network traffic in several timescales

1.2 About this thesis

1.2.1 Problem statement and scope

This thesis uses fractal analysis to characterize increasingly bursty Internet traffic. It examines how the fractal nature of network traffic is developed due to the workings of network protocols and applications. The goal is to develop a better understanding of the fractal nature of network traffic, which in turn will lead to more efficiency and better quality of services on the Internet.

This document begins with a discussion of the fundamental theory of data network traffic. It introduces statistical second-order self-similarity analysis using wavelets and gives an overview of several popular traffic modeling approaches, e.g., M/G/∞, FARIMA, and the wavelet traffic models. Then, self-similarity analysis is applied to a series of NS-2 simulated datasets. The study improves the understanding of traffic self-similarity with several insightful observations.

Later in this thesis, the study focuses on multifractal analysis – a more fundamental and broader view of fractals compared to second-order self-similarity analysis. MFA is based on the concept of a local signal singularity characterized by the Hölder exponent α. The formulations of MFA are described. MFA is applied to both simulated traffic datasets and real traffic datasets from the Internet. The MFA study confirms the previous findings of the fractal behavior of network traffic. Based on the fractal property, a new concept of *efficient bandwidth* is developed and is applied to fractal traffic control to improve network efficiency.

1.2.2 Contribution of the thesis

This thesis:

- verifies that network traffic indeed is fractal. The self-similarity can be separated into two regions that have short-range dependence (SRD) and long-range dependence (LRD). The physical mechanisms that control the two regions are clearly identified and generalized by six factors. More specific physical evidences of fractal behavior of network traffic are described. I argue that the heavy-tailedness of file sizes, a current popular view of traffic self-similarity, is not the sole cause.
- applies multifractal analysis to study network traffic and characterizes network traffic with advanced multifractal spectra and multifractal measure. From multifractal analysis, it appears that the Hölder exponent is in the range of 0.7 ~ 1.3 in most network traffic traces.
- proposes a measurement of the *efficient bandwidth* to quantify a fractal traffic flow to reduce burstiness and demonstrates its usefulness for fractal traffic control to improve network efficiency and performance.

1.2.3 Organization of the thesis

This thesis consists of four main chapters (Chapters 2–5). Chapter 2 describes some basic network traffic theories: statistical second-order self-similarity. It also covers a few representative traffic modelling schemes. Interpretations of network traffic fractality are presented in both analytical networking contexts and experimental results. A further study of a real-traffic trace with comparison between Web traffic and KaZaa traffic is reported here to demonstrate that large file sizes do not necessarily lead to fractality but the self-organized and correlated traffic objects will.

Chapter 3 studies multifractal traffic analysis. The concept of multifractal is based on analyzing the signal local singularity or Hölder exponent, and it is formulated using both binomial multifractal model and wavelet transform. A number of datasets are tested using this analysis, including binomial cascades, Poisson processes, simulated traffic sets, and real Internet traffic sets.

Chapter 4 deals with applications of fractal traffic analysis. The concept of efficient bandwidth is developed, and the results from a NS-2 experiment are presented to demonstrate its effectiveness for TCP traffic. Additionally, a study of efficient transport of gaming traffic in UDP is presented.

Chapter 5 concludes the thesis with future work.

CHAPTER 2

FRACTALS WITH NETWORK TRAFFIC

This chapter studies the basic network traffic theory: statistical second-order self-similarity. Traffic self-similarity has a root from two fractal processes: Fractional Brownian Motion (FBM) and Fractional Gaussian Noise (FGN). The characteristics of FBM and FGN are represented by a *fractal spectrum* with a Hurst parameter, which can be easily analyzed using wavelet transform such as a Haar wavelet. Using self-similarity analysis, we can characterize fractal traffic in both the SRD and LRD regions. A series of NS-2 simulations are conducted to study the impacts on SRD and LRD using six parameters: bandwidth, delay, number of connections, packet sizes, TCP window sizes, and Pareto shape factor of connection sizes. Later, two traffic flows of Web and KaZaa in a real traffic trace are analyzed and compared on self-similarity. A traffic burst profile based on *burstiness* is derived to show the differences in the self-similarity between the Web and KaZaa traffic flows.

2.1 Background in network traffic

The essential property of fractals is scale-invariant, and this property is often described as self-similarity in the second order statistics. In the early 1990s, Leland and Willinger et al. [1] first reported the self-similar nature of network traffic. Before that study, traffic engineering had been successful in adopting a Poisson traffic model in traditional telephone networks. But, the new paradigm with fractal traffic has revolutionized traffic modeling. Among a number of network traffic models, fractional auto-regressive integrated moving average (FARIMA) and M/G/∞ are two famous examples. FARIMA captures both short-range and long-range dependence. It has been successful in video traffic modeling. However, for generic network traffic, FARIMA lacks connections with the network mechanisms, e.g., TCP and HTTP. It also has a high computation complexity, which makes it difficult to use in practice. On the other hand, the M/G/∞ model[11][12] is based on the aggregation of general ON/OFF processes. This model is directly related to the heavy-tailed property of traffic, e.g., network objects such as traffic connections and file sizes that are heavy-tailed. In M/G/∞, it is suggested that aggregating a large number of heavy-tailed processes results in long-range dependence. In the more recent studies, wavelet theory has been very popular in traffic analysis. A representation of traffic in wavelet domain requires only a few parameters across the scaling regions, which in turn leads to the wavelet traffic models including Independent Wavelet Model (IWM)[16][17] and Multifractal Wavelet Model (MWM) [19]. In this section, I will describe these traffic analysis theories and models in detail.

2.1.1 Basic fractal traffic theory

2.1.1.1 Self-similarity and long range dependence

The fundamental fractal traffic model is derived from Fractional Brownian Motion. FBM is a Gaussian process and can be used in general conditions. It has a homogenous parameter H. A FBM process B_H holds in the form of

$$B_H(st) = s^H B_H(t) \text{ for } s > 0 \,. \tag{2.1}$$

Here, $H \in (0,1)$ is the *Hurst parameter* – a structural parameter for all scales $s \to \infty$. In (2.1), the equality holds in a statistical distribution sense.

In addition, B_H is a zero-mean Gaussian process with the following two properties: (i) $B_H(0) = 0$; and (ii) the variance $\mathrm{E}\{B_H{}^2(t)\} = \sigma^2 \,|\, t \,|^{2H}$. As σ^2 is the energy of B_H, the variance is proportional to $|t|^{2\mathrm{H}}$. From (ii), we can also have

$$\mathrm{E}\{B_H(t) B_H(u)\} = \frac{\sigma^2}{2}(|\, t\,|^{2H} + |\, u\,|^{2H} - |\, t - u\,|^{2H}) \,. \tag{2.2}$$

Equation (2.2) shows that B_H is a non-stationary process. But the increment of FBM is a stationary process, which is known as Fractional Gaussian Noise. Let Y(t) denote a FGN process, and δ is the time interval. Then, we define

$$Y(t) := B_H(t) - B_H(t - \delta) \,. \tag{2.3}$$

Y(t) is a zero-mean Gaussian process, and its autocorrelation function satisfies

$$r_Y(k) = \frac{\sigma_Y{}^2}{2}((k + \delta)^{2H} - 2k^{2H} + (k - \delta)^{2H}) \ . \tag{2.4}$$

With (2.4), Y(t) is called *exactly second-order self-similarity*. It shows that Y(t) is second-order stationary because $r_Y(k)$ depends only on the interval k . For ½ < H < 1 and δ = 1, we obtain

$$r_Y(k) = \lim_{k \to \infty} H(2H - 1) k^{2H-2} \sigma_Y^2 \,. \tag{2.5}$$

From (2.5), we have $r_Y(k) > 0$. Thus, the autocorrelation is not summable for the lag $k \to \infty$. This property of Y(t) is called *long-range dependence*. It indicates that the autocorrelation has a decay rate slower than $1/k$.

2.1.1.2 Network traffic processes

In a general traffic model, a network traffic process is known as an *H*-parameterized self-similar process with stationary increments (*H-sssi*). Let $Z(t)$ denote an aggregated traffic process starting at $t = 0$. $Z(t)$ models a FBM process. So, we can write $X(t)$, the stationary increment process of $Z(t)$, as

$$X(t) = Z(t) - Z(t-1) = Y(t) + \mu_x . \tag{2.6}$$

By the definition (2.3), $Y(t)$ is a zero-mean FGN process. μ_x denotes the mean of $X(t)$. Equation (2.6) will be useful only in a statistical sense.

Let $Y^{(\mathrm{m})}$ represent the average of $Y(t)$ at an accumulation scale m, and we have

$$Y^{(m)}[n] = 2^{-m} \sum_{i=1}^{2^m} Y[2^m(n-1)+i] . \tag{2.7}$$

$Y^{(\mathrm{m})}$ is called *asymptotically second-order self-similar* if

$$\lim_{m\to\infty} r_{Y^{(m)}}(k) = \frac{\sigma_Y{}^2}{2}((k+1)^{2H} - 2k^{2H} + (k-1)^{2H}) . \tag{2.8}$$

From the properties of FGN, we know that $Y(0) = 0$ and $\mu_Y = 0$. Therefore, we can get

$$Y^{(m)} = 2^{-m(1-H)} Y \tag{2.9}$$

and

$$Var[Y^{(m)}] = 2^{-m(2-2H)} \sigma_Y^2 . \tag{2.10}$$

Correspondingly, for the increment traffic process X(t), it follows

$$X^{(m)}[n] = 2^{-m} \sum_{i=1}^{2^m} X[2^m(n-1)+i] \tag{2.11}$$

and

$$Var[X^{(m)}] = 2^{-m(2-2H)} \sigma_X^2 . \tag{2.12}$$

From (2.12), it is implied that there is an asymptotical scaling structure within the traffic model. As the accumulation level m increases, the variance will decrease asymptotically, and the decay in *log*-scale is a straight line with a slope determined by H.

2.1.2 Fractal traffic analysis

2.1.2.1 Discrete wavelet transform (DWT)

The wavelet transform is useful to analyze fractals. For fractal network traffic, discrete wavelet functions are used in the forms of (2.13).

$$\begin{cases} \psi_n^m(t) = \psi(2^{-m}t - n), \\ \phi_n^m(t) = \phi(2^{-m}t - n). \end{cases} \tag{2.13}$$

$\psi_n^m(t)$ is known as the mother wavelet, and $\phi_n^m(t)$ is the father wavelet. Using wavelet functions such as a Haar wavelet, a signal can be decomposed in the wavelet domain represented in (2.14).

$$x(t) = \sum_n a_n^M \phi_n^M(t) + \sum_{m=M}^{\infty} \sum_n d_n^m \psi_n^m(t) \tag{2.14}$$

with

$$\begin{cases} a_n^m = \int x(t)\phi_n^m(t)dt, \\ d_n^m = \int x(t)\psi_n^m(t)dt. \end{cases} \quad (2.15)$$

The decomposition comprises two parts: an approximation of the original signal at a coarse scale M and its complementary parts (orthogonal to the signal space) at the multiple finer scales. The approximation coefficients a_n^m are called the scaling coefficients, and the coefficients d_n^m are the wavelet coefficients.

2.1.2.2 Fractal spectrum

A self-similar process is also called $1/f$ noise. This is because its power spectrum has a power-law decay. The spectrum of FGN is shown as

$$\Gamma_f(\nu) \propto \xi|\nu|^{-(2H-1)} \text{ for } \nu \to 0. \quad (2.16)$$

The power spectrum (2.16) has two parameters (H, ξ): H is the Hurst parameter, capturing the scaling structure of the process; and ξ, independent of the Hurst parameter, represents the signal power at the finest scale.

A wavelet estimator of the Hurst parameter is given by Abry and Veitch [42]. Applying DWT (2.13), we can analyze this spectrum using the multi-scaled wavelet coefficients. As a wavelet function is an approximate band-pass filter, $\psi_n^m(t)$ has a time support of 2^{-m} and a frequency support of $2^m\nu_0$ at a scale m. Here, ν_0 is the center frequency corresponding to $\psi^0(t)$. So, we can derive

$$\begin{aligned} &\Gamma_f(2^m\nu) \\ &\approx E[\langle f,\psi_n^m\rangle^2] = \frac{1}{N_m}\sum_n^{N_m}|d_n^m|^2 \\ &= Var[d_n^m]. \end{aligned} \quad (2.17)$$

N_m is the number of wavelet coefficients at the scale *m*. Using (2.16) and (2.17), we will find

$$\log_2(Var[d^m]) = -(2H-1)m + \log_2(\xi C), \tag{2.18}$$

with

$$C = \int |\nu|^{-(2H-1)} |\Psi(\nu)|^2 d\nu . \tag{2.19}$$

For a fractal signal, (2.18) represents a linear relationship between the log-scaled signal energy and the scale *m*. Therefore, we can calculate *H* from (2.18). At two scales *j* and *k* (*j* < *k*), the wavelet estimator for the Hurst parameter *H* is found as

$$\hat{H} = \frac{1}{2(k-j)}\left(\log_2 \frac{Var[d^j]}{Var[d^k]}\right) + \frac{1}{2} . \tag{2.20}$$

The estimation of ξC can also be obtained from the linear regression of (2.18). Then, get $\xi = (\xi C)/C$. Detailed discussion can be found in [42][43].

2.1.3 Network traffic modeling

Traffic models are important to network performance analysis. They are required to capture the statistical characteristics of real traffic efficiently and accurately. The development of traffic modeling relies on the advances of traffic analysis. One of the traffic models is derived from an On-Off queuing model. In this model, "On" represents a busy data transmission period and "Off" represents silence with no data transmission. Statistically, if "On-Off" periods are heavy-tailed, aggregating a large number of these processes will result in long-range dependence[5]. On the other hand, self-similarity of a process is a property represented using its power spectrum. By exploiting the power spectrum, a number of traffic models[15][17][19][23] have been proposed since the late 1990s and early 2000s. In particular, the Haar wavelet has been used to analyze and to model network traffic in recent years[19].

2.1.3.1 M/G/∞

M/G/∞ model[11][12][13] considers a queuing system, where arrivals are Poisson and are served by the infinite number of servers. The service times are i.i.d with general distributions. The system is characterized by counting the number of busy servers at a time *t*. Then, if the service time distribution is heavy-tailed, it converges to the long-range dependence at output. This model corresponds to the On-Off traffic model, in which the traffic sessions arrive by a Poisson distribution and the On-period of each session is heavy-tailed.

Heavy-tailedness (H-T)[13] *A distribution F is said to be heavy-tailed, if* $\overline{F}(x+y) \sim \overline{F}(x)$ *as* $x \to \infty$ *for any fixed y.*

The heavy-tailed distribution decays slower than exponential. A random variable *X* with a heavy-tailed distribution can be expressed as

$$\Pr\{X > x\} \sim cx^{-\alpha}, \text{ as } x \to \infty, \tag{2.21}$$

where α is a shape factor as $0 < \alpha < 2$, and the constant $c > 0$. For $0 < \alpha < 2$, the variance of *X* is infinite; for $0 < \alpha < 1$, the mean also becomes unbounded. In particular, Pareto is one of the most used heavy-tailed distributions. The probability density function of Pareto is given by

$$P(x) = \frac{\alpha b^{\alpha}}{x^{\alpha+1}}, \; 0 < \alpha < 2 \text{ and } x \geq b. \tag{2.22}$$

The mean of Pareto distribution is $\alpha b/(\alpha$-1). Besides Pareto, Weibull and lognormal distributions are also referred to as heavy-tailed distributions. However, they both have finite variances.

Furthermore, the relationship between the Pareto shape α in H-T and the Hurst parameter H in LRD is

$$H=(3-\alpha)/2\,, \mathit{for}\ 1<\alpha<2 \text{ and } 1/2<H<1. \tag{2.23}$$

2.1.3.2 FARIMA

FARIMA is a widely used model for stationary time series[14][15]. A FARIMA process, denoted as *f*ARIMA(*p,d,q*), is comprised of two basic polynomials: the auto-regression polynomial (AR) with degree p^{th} and the moving average polynomial (MA) with degree q^{th}. They are written as

$$\begin{cases}\alpha(B)=1-a_1B-a_2B^2\cdots-a_pB^p,\\ \beta(B)=1+b_1B+b_2B^2\cdots+b_qB^q.\end{cases} \tag{2.24}$$

Here, B is the backward shift operator, defined as $\mathrm{B}^jX(t)=X(t-j)$. The third part is a differentiation operator ∇ with $\nabla^d=(1-\mathrm{B})^d$. The differentiator is responsible for removing any seasonal trend from a data process X(t). Altogether, we have

$$\alpha(B)\nabla^dX(t)=\beta(B)z(t). \tag{2.25}$$

In (2.25), z(t) is noise, an i.i.d process. Note, ∇ is a differentiator on the left side but becomes an integrator on the right side. This is shown as

$$\Sigma=\frac{1}{1-B}=\sum_{i=0}^{\infty}B^i\ , \tag{2.26}$$

$$X(t)=\Sigma^dz(t). \tag{2.27}$$

For $0 < d < ½$, X(t) is long-range dependent. Both $\alpha(B)$ and $\beta(B)$ are used to characterize the SRD properties. For $d > ½$, the process is non-stationary. For a FGN process with autocorrelation function in (2.5), the relationship between the Hurst parameter H and d is $H = d + ½$.

Although FARIMA is flexible enough to model both LRD and SRD characteristics, it requires solving the polynomial coefficients. As the degrees of p and q increase, the computation complexity increases dramatically. This is one of the major drawbacks for FARIMA.

2.1.3.3 Wavelet traffic models

Wavelet traffic models are studied in [16][17][19][20]. By a wavelet transform, it is possible to decorrelate the dependent structure in fractal traffic. A key theory by Kapland and Kuo [38][39] states that the wavelet coefficients of fractal data are independent at each scale and those coefficients therefore can be modeled as zero mean Gaussian random variables. Because a Gaussian random variable only has two parameters, mean and variance, we can use the wavelet coefficients to capture the traffic LRD structure over the multiple scales. In data synthesis, Gaussian variables are generated by wavelet coefficient variances at those scales, and then data samples are calculated by an inverse wavelet transformation. By using the Haar wavelet, the computation for this procedure is simplified. The above method is often referred to as the independent wavelet model[17].

However, there is a problem with IWM. It produces unrealistic negative values for traffic due to using Gaussian random variables. Another approach proposed by Riedi, et al. [19] solved this problem. This model is referred to as the multifractal wavelet model.

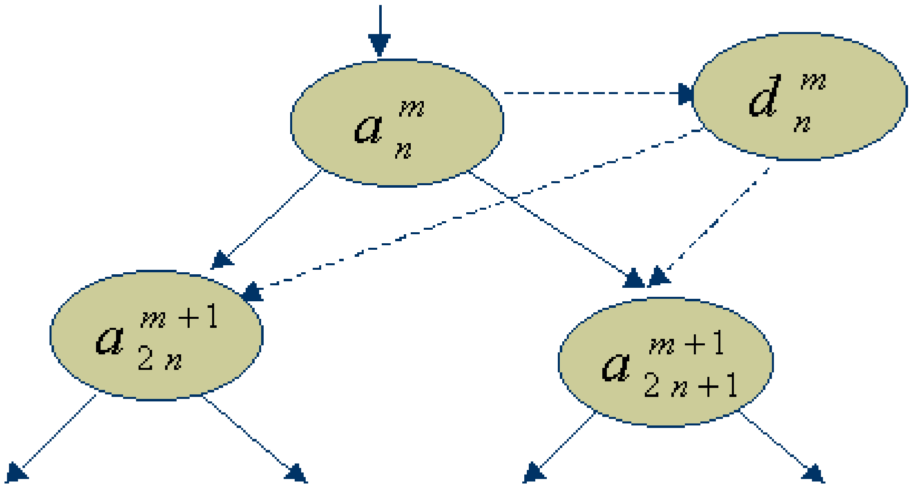


Figure 2-1 Illustration of MWM traffic model

The MWM model is illustrated in Figure 2-1. In a Haar wavelet transform, the scaling coefficients can be found by

$$\begin{cases} a_{2n}^{m+1} = \dfrac{1}{\sqrt{2}}(a_n^m + d_n^m), \\ a_{2n+1}^{m+1} = \dfrac{1}{\sqrt{2}}(a_n^m - d_n^m). \end{cases} \tag{2.28}$$

For a positive process, $a_n^m \geq 0$ and $a_n^m \geq |d_n^m|$ for any *m* and *n*. Let A_n^m be a random variable as multipliers and $A_n^m \in [-1,1]$. Then, set

$$d_n^m = A_n^m a_n^m . \tag{2.29}$$

Therefore, together with (2.28) and (2.29), obtain

$$\begin{cases} a_{2n}^{m+1} = \dfrac{1}{\sqrt{2}}(1 + A_n^m)a_n^m, \\ a_{2n+1}^{m+1} = \dfrac{1}{\sqrt{2}}(1 - A_n^m)a_n^m. \end{cases} \tag{2.30}$$

Using (2.30) and fitting the distribution of the random variable A_n^m, we can synthesize a traffic trace precisely. In Riedi's paper[19], two distributions for the random variable A_n^m were suggested: a symmetric Beta distribution and a point-mass distribution. In addition, it requires that A_n^m is identically distributed and symmetric in order to model a stationary process. From (2.30), the wavelet energy scaling property can be captured as

$$\begin{aligned}&\frac{Var[d_n^m]}{Var[d_n^{m+1}]}\\&=\frac{2\mathrm{E}[(A^m)^2]}{\mathrm{E}[(A^{m+1})^2](1+\mathrm{E}[(A^{m+1})^2])}\\&=2^{2H-1}.\end{aligned} \tag{2.31}$$

In order to match the energy scaling structure, we have to recursively solve (2.31) starting from the coarsest scale $m = 0$.

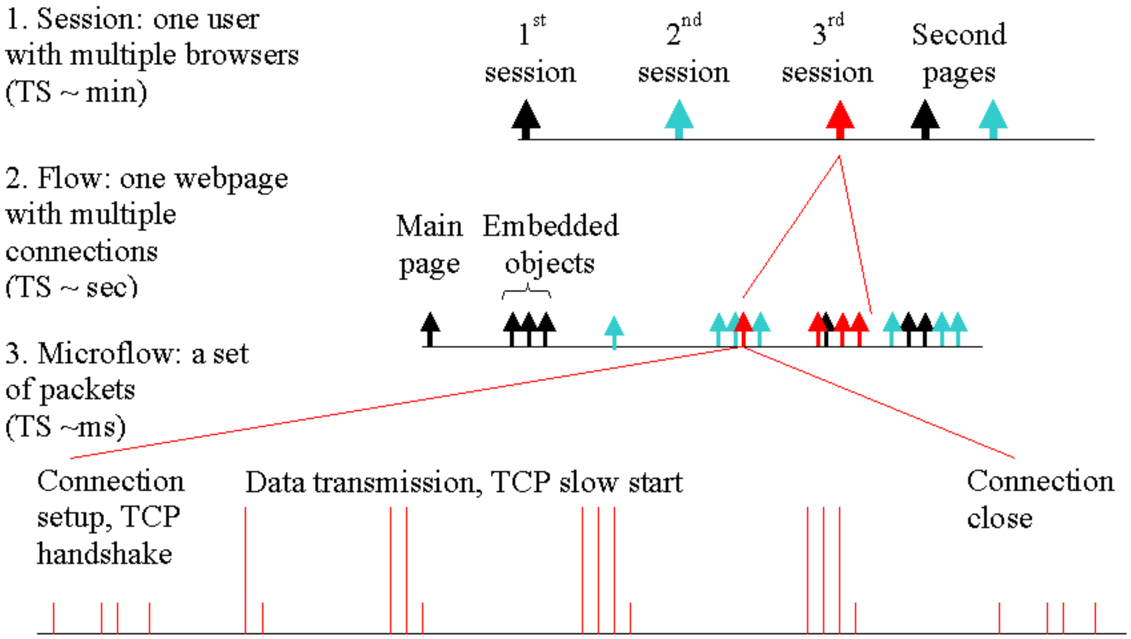
More importantly, the MWM algorithm can be identified with a binomial cascade and is close to the multifractal measure in a natural fractal structure. This provides some insightful ideas towards understanding the statistics of network traffic, which will be studied in much more detail in Chapter 3.

2.2 An application view on traffic self-similarity

In network engineering, finding the origin of traffic self-similarity and long-rang dependence is important to us. While the concepts of self-similarity and long-range dependence in traffic are counter-intuitive to most network engineers, we do know that their impacts are significant: network traffic is very bursty. This burstiness could cause network bottlenecks, packet loss, jitter, and low bandwidth efficiency. Traffic analysis and modeling generalize several mathematical methods, which improved the understanding of self-similar structure in network traffic. In this section, it gives a unified explanation of traffic self-similarity from an application structural point of view. The proposed explanation is based on Web traffic, which has dominated the Internet for a decade.

Previous studies [4][5][6] have shown that the heavy-tailedness of connection sizes, specifically the over abundance of large file sizes (corresponding to long active periods), is the main cause of LRD in network traffic. However, this explanation is incomplete. Some recent studies[21][22] have raised questions about it and suggested more complicated mechanisms behind both LRD and SRD. In particular, the heavy-tailedness is incapable of explaining the subtle scaling structure that appears for small timescales. In [7][8][9], Feldmann et al. described the network traffic structure as resulting from a data cascade. Traffic objects are regenerated from one protocol layer to another in a cascade fashion. In general, the mechanisms that govern the self-similarity can be categorized as due to two sets of rules: one that controls the scaling at the large scales (second ~ minutes) and one that controls the scaling at the small-scale range (ms ~ sec). These scaling regions can be easily observed from an energy-scaling diagram (see Figure 2-5). These mechanisms are described in detail as below.

2.2.1 Self-similarity at the large timescales



1. Session: one user with multiple browsers (TS ~ min)
2. Flow: one webpage with multiple connections (TS ~ sec)
3. Microflow: a set of packets (TS ~ms)
1st session
2nd session
3rd session
Second pages
Main page
Embedded objects
Connection setup, TCP handshake
Data transmission, TCP slow start
Connection close

Figure 2-2 Self-similarity in a Web traffic session

By definition, LRD resides in timescales starting from seconds. For the Web application, we may separate Web traffic into three levels in this time range (shown in Figure 2-2): session, flow and micro-flow. In a Web session, one user may open multiple Web browsers simultaneously, each of which initiates a group of connections at the flow level. Those connections transmit data for multiple objects in parallel. The size of each object varies independently depending on the contents either text or multimedia information, while the initial times of data transmission are closely correlated. Thus, a single click on a Web page will result in traffic bursts for some random period. This phenomenon is analogous to the ripples generated by throwing a stone into water. The periods of connections from this operation could last from a few hundred milliseconds to a few minutes.

The timescale range is related to the initial scales in LRD. LRD represents a trend of the signal energy scaling in a long run. In such a sense, why will heavy-tailed traffic elements cause LRD? LRD is a global estimation. Heavy-tailed traffic elements are those with very large values, but which appears with very low frequency, such as the downloading of a movie file. LRD indicates that the energy from these low-frequency elements will become more and more dominant as the timescales increase.

On the other hand, time and frequency are inversely related. At the small timescales, the signal energy is determined by the short-duration traffic activities while the long-duration traffic flows have much smaller effect. At large timescales, the short traffic activities become insignificant as they are averaged over long time periods. Based on this analysis, the self-similarity at the large timescales (i.e., LRD) is caused by a few long traffic flows whose average rates are proportional to the mean rate of the total traffic.

2.2.2 Self-similarity at the small timescales

On the other hand, SRD is found in the timescales below a few seconds, perhaps down to milliseconds. This is a time range below the connection level. Each individual connection is regarded as a micro-flow. Within micro-flows, due to network algorithms (such as TCP control mechanisms) packet bursts occur periodically. The packet bursts are statistically correlated across the timescale periods – bursts contain microbursts that are related both by sizes and time intervals. Therefore, a scaling structure exists.

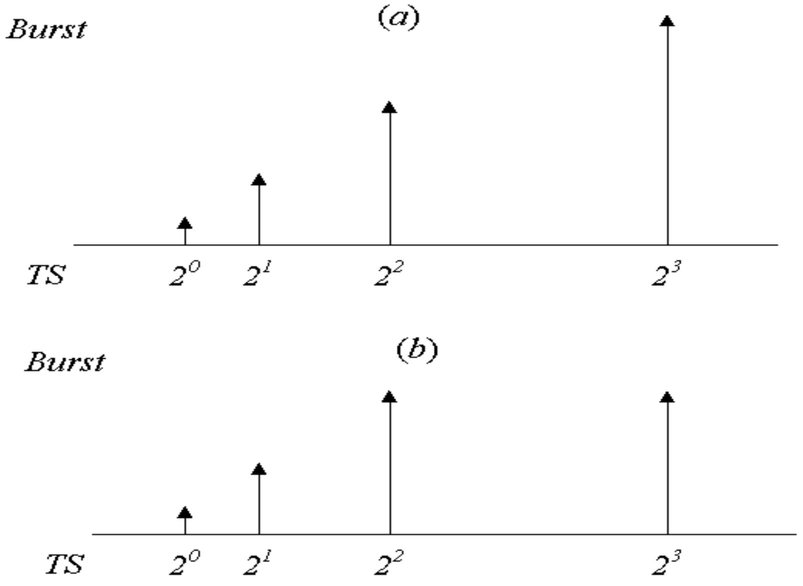


Figure 2-3 Bursts in scaling: (a) Scaling up (b) Scaling off

Figure 2-3 (a) illustrates an example of an up-scaling burst. As the scale increases from 2^0 to 2^3, the size of burst per time unit also increases, so the energy per time unit increases. An example of this phenomenon in network traffic is seen during the TCP connection start-up phase. The TCP sender's congestion window size increases exponentially with each ACK packet received. So, in one TCP connection, the number of transmitted packets are doubled in each RTT until a congestion occurs. Figure 2-3 (b) shows another case, in which the scaling structure stops at the timescale 2^3 because the burst size at 2^3 does not grow. The timescale where the scaling stops is referred to as the cut-off timescale and corresponds to a "knee" (a turnover) on the wavelet energy plot (i.e., the timescale *K* see Figure 2-5 in Section 2.3.1).

The packet bursts on networks are similar to the above example, but they appear in a random fashion. SRD is referred to as packet level bursts, and can be further separated into three sub-levels.

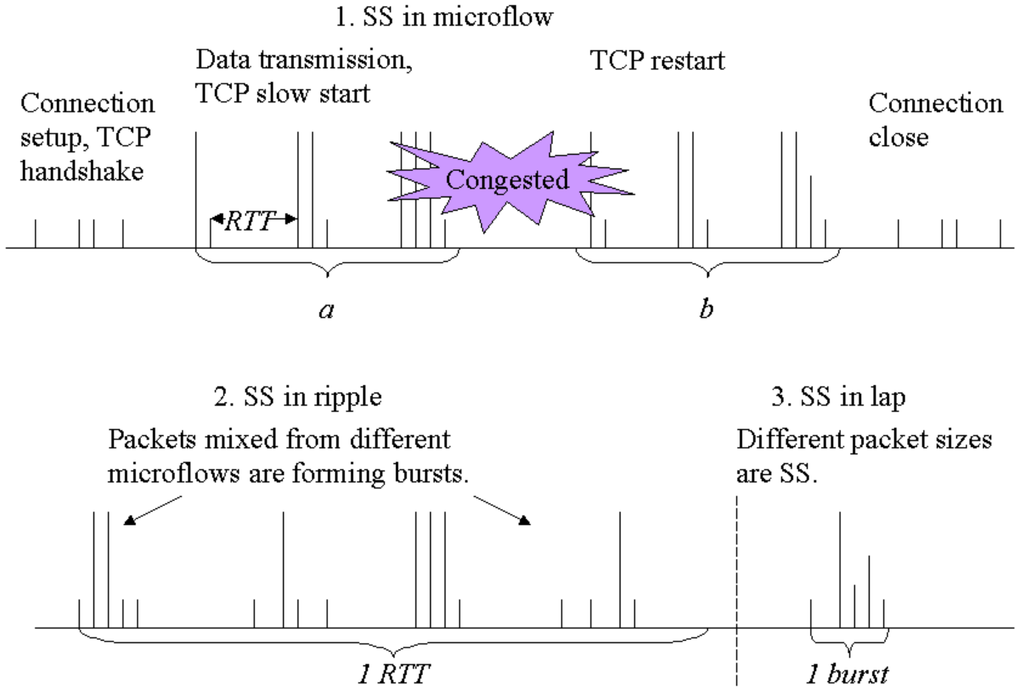

1. SS in microflow
Data transmission, TCP slow start
TCP restart
Connection setup, TCP handshake
Connection close
Congested
RTT
a
b
2. SS in ripple
Packets mixed from different microflows are forming bursts.
3. SS in lap
Different packet sizes are SS.
1 RTT
1 burst

Figure 2-4 Self-similarity at the packet level

Figure 2-4 illustrates the self-similarity for packet bursts at SRD. First, Figure 2-4.1 shows the correlation between TCP packet bursts. This is due to the mechanisms, such as TCP slow start, congestion, loss recovery and other similar transport-layer schemes that could cause the packet rate to alternate and send multiple back-to-back packets. For example, TCP increases the packet rate (the sender's congestion window size) in every RTT until the flow reaches the congestion point, and then TCP starts over (between the two parts *a* and *b* in Figure 2-4.1). This burst period is within the multiple RTTs, which determines the largest timescale (usually a few hundred milliseconds or seconds) in SRD. Secondly, within a RTT time range, multiple traffic flows share the network at the same time, and packet burst sizes vary for the different TCP flows. This diversity is illustrated in Figure 2-4.2. In this case, the TCP flows are synchronized, e.g., a fast TCP flow (with large packet bursts) shares the bandwidth with a slow TCP flow (with smaller packet bursts). Those packet bursts can divide a RTT period into few smaller intervals, so the energy scaling could occur within a RTT period (less than 100ms). This traffic level on self-similarity is referred to as “ripple”. Thirdly, in a single packet burst the packet size is a factor affecting energy scaling (shown in Figure 2-4.3). Packet sizes vary from the control packets (usually 40 bytes without payloads) to data packets (up to 1500 bytes on Ethernet). So, this variation occurs at the smallest timescale in the scaling range (usually within few milliseconds on Ethernet). This traffic level is called “lap”.

In the wide area network, it is possible that one packet burst can mix with packets from different TCP connections and different users. Packets from different users are independent from each other. So, they are generally uncorrelated and produce very limited SRD effect on the scaling structure. However, in a local network, a correlation exists because they share the same bottleneck to the Internet. The separate TCP flows from the same LAN are synchronized, so that it usually shows SRD. For this reasoning, we may assume that the aggregated TCP traffic flow from a single LAN is one flow, but one which will correlate with other such traffic flows at a common bottleneck going to an even larger network. This may depict a high-level picture of traffic self-similarity from a network topology point of view.

2.3 How does network shape traffic?

In this section, I use a semi-experimental method to demonstrate how the network itself impacts on the traffic self-similarity.

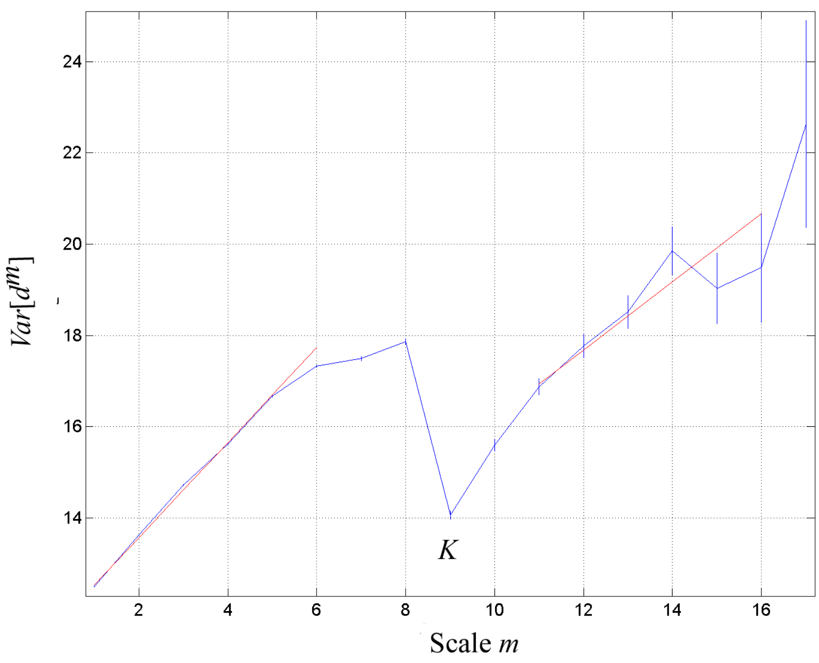


Figure 2-5 A log-scaled wavelet energy plot

2.3.1 The energy plot

Figure 2-5 is a typical log-scaled wavelet energy plot from analysis of a network traffic trace. This plot is calculated by the partial energy $Var[d_n^m]$ at each scale using (2.20) in Section 2.1.2.2. A linear section in this plot indicates an energy scaling. Clearly, there are two linear segments in Figure 2-5: the segment between scales 1-5 and the segment between scales 11-16. They are referred to as SRD and LRD, respectively. As mentioned earlier, another important property in this plot is the scale *K*, the "*knee*" for LRD.

2.3.2 A 5-parameter traffic profile

We can profile a traffic self-similarity based on the signal energy plot (Figure 2-5). Here, we need to assume that traffic has Gausianity. This assumption has been widely used previously [43] and is considered to be valid if traffic is an aggregation of many traffic flows from a large number of users, e.g., traffic data on backbone networks[10]. Using the energy plot, we can obtain a simple parametric traffic profile by five parameters: $(H, \xi)_{SRD,}$ K, $(H, \xi)_{LRD}$. Here, $(H, \xi)_{SRD}$ represent the scaling structure for SRD and $(H, \xi)_{LRD}$ for LRD. K is the knee scale.

2.3.3 The source traffic model

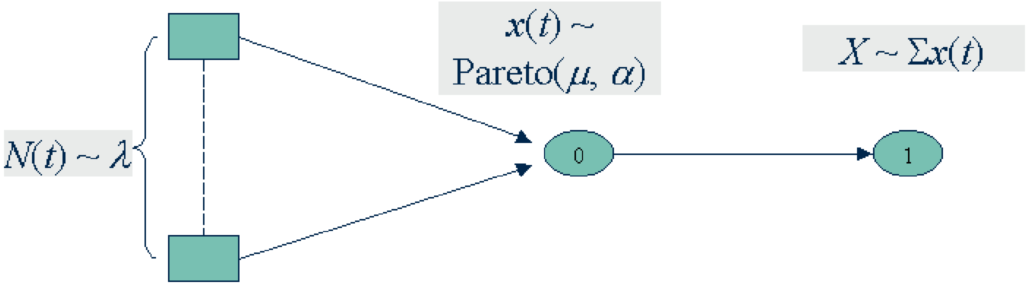


Figure 2-6 The source traffic model used in simulation

Figure 2-6 shows the source traffic model in the simulation. There are N TCP connections, with a Poisson arrival rate λ. The size of each connection follows a Pareto distribution with mean μ and shape α. Those traffic flows are aggregated at node 0 and go to node 1. Traffic is collected on the link between nodes 0 – 1.

2.3.4 Simulation and results

In the experiment, six variables are chosen. They are categorized into three groups:

- Variables on network: bandwidth and delay.

- Variables on traffic sources: the number of connections and the shape of Pareto distribution.
- Variables on TCP connections: the packet size and the TCP sender's maximum congestion window size.

The experiments are conducted using the NS-2 simulator, and the simulation time of each trial is one hour. The following observations are made based on the results shown in Figure 2-7 (a) – (f).

(a) Increasing the bandwidth does not have any impact on LRD, but will smooth the effect on SRD (Figure 2-7.a). Traffic flows originally are uncorrelated. However, since the link 0 – 1 is a bottleneck, TCP flows will be synchronized by TCP control mechanisms. This makes TCP flows correlated. The smaller the bandwidth the higher the correlation will be at the bottleneck. As discussed in Section 2.2, TCP mechanisms control traffic flows at SRD. Therefore, at some point, increasing the bandwidth will only affect SRD.

(b) Figure 2-7.b shows that the link transmission delay, i.e., 1/2 RTT, is vital in separating the scaling regions between SRD and LRD. Since the scaling change at *K* indicates the periodicity in the traffic correlation structure, RTT is the key factor that represents the period at this scaling range. Note, this timescale of *K* is approximately twice of the transmission delay, because the energy scaling stops at the ascending timescale by one. The delay does not change the scaling structure, but reduces TCP throughput as RTT increases. Because RTT separates the SRD and LRD regions, this implies that the transport protocols determine the scaling behaviors at the small timescales.

(c) Increasing the connection arrival rate raises the traffic loads, thus raises the energy level. But, it does not affect the scaling structure, i.e., the slope of the energy plot (Figure 2-7.c). This shows traffic independent among the connections.

(d) The Pareto shape factor of connection sizes determines the scaling structure at both SRD and LRD (Figure 2-7.d). As the factor α decreases to 1, the Hurst parameter H increases and is close to 1. Then, traffic is more correlated. The relationship is $H = (3 - \alpha)/2$ [6].

(e) Adjusting the TCP maximum congestion window sizes changes SRD but not LRD (Figure 2-7.e). This can be explained by packet-level burstiness on self-similarity (referred to the discussion in Section 2.2). It shows the difference of SRD with the same traffic load.

With a window size of 100 (packets), it has a SRD region across scales 1 – 8. For the packet size at 1000 bytes and the network bandwidth at 10Mb/s (in my simulation), this equals to a maximum packet burst period of approximate 80ms from one TCP connection. For a RTT of 300ms (corresponding to scale 8), it is approximate to a quarter of link capacity (80ms out of 300ms). This estimates the maximum packet burst period.

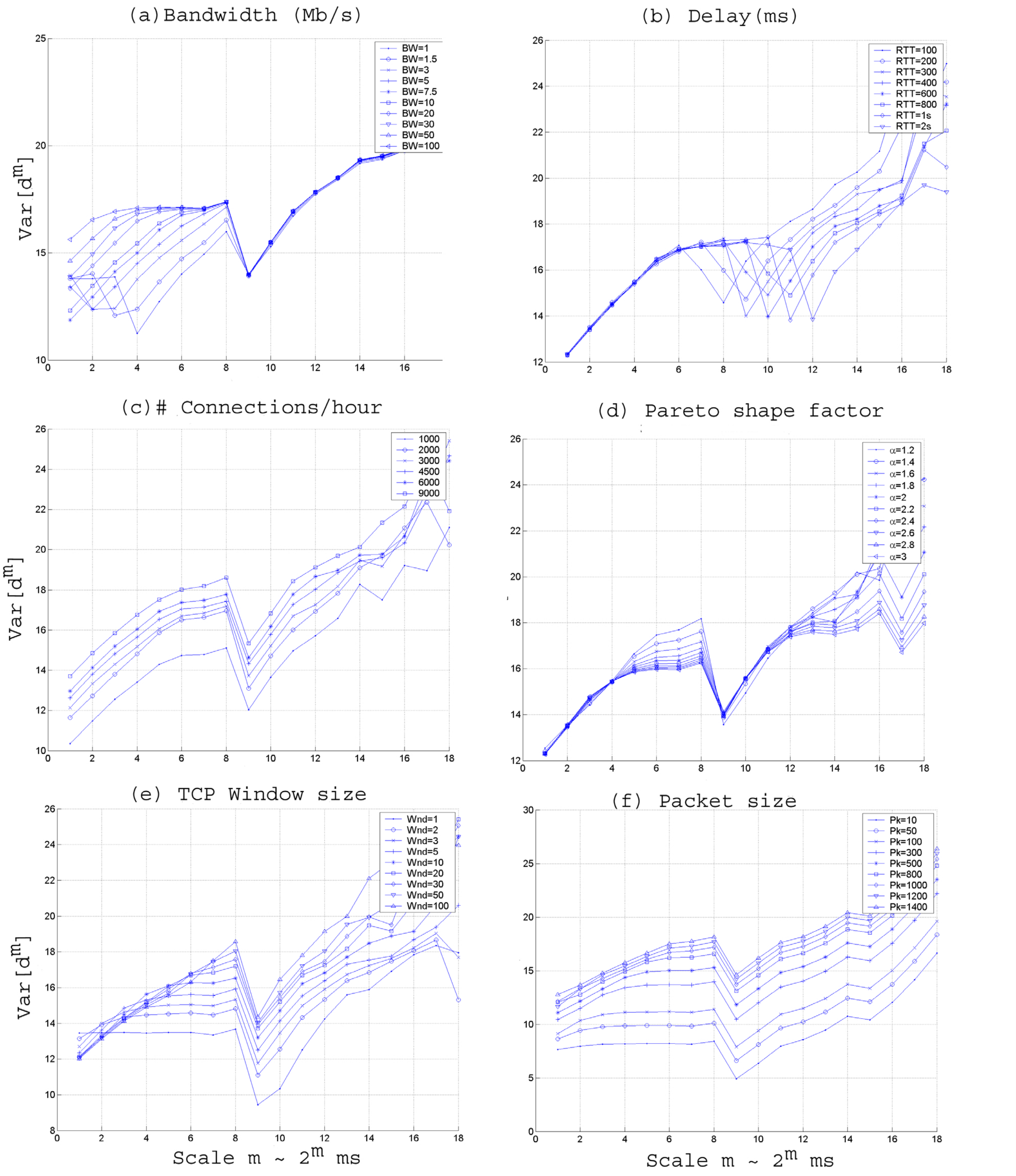


Figure 2-7 Log-scale wavelet energy plots: How does the network impact on self-similarity?

On the other hand, at the TCP window size of one, there is no packet burst at the small timescales. SRD shows flat. For the same traffic load, traffic in this case is less bursty – packets are spread out under the scale 8.

(f) Like TCP window sizes, the larger packet sizes impact on SRD (Figure 2-7.f). The scaling structure appears to be more correlated as the packet size becomes larger, which results in the higher packet-level burstiness at SRD.

2.3.5 Traffic bursts with self-similarity

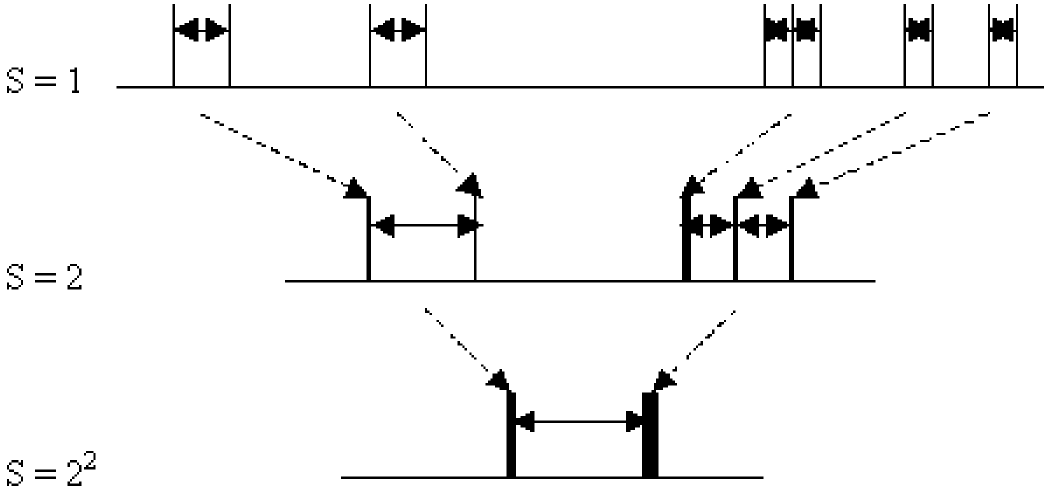


Figure 2-8 Self-similarity and bursts

The above experiments indicate that there is a relationship between self-similarity and traffic burstiness, which we reported in [49]. Illustrated in Figure 2-8, bursts are the discontinuous packet arrivals with time-intervals proportional to their scales. The greater the self-similarity ($H \rightarrow 1$), the smaller the intervals are, and the greater the bursts are. In other words, burstiness gives the sense of "*dense*".

2.3.5.1 A formulation of traffic burst

From self-similarity (2.1), we may estimate the traffic burst at a given scale. Considering, at the smallest scale $m = 2^0 = 1$, the average signal energy is expressed as

$$\varepsilon_X = E[(X)^2] = \sigma_0^2 + \mu_0^2 . \tag{2.32}$$

σ_0 is the standard deviation and μ_0 is the mean of X.

For the aggregated signal $X^{(k)}$ at scale k,

$$X^{(k)}[n] = 2^{-k} \sum_{i=1}^{2^k} X[2^k(n-1)+i] . \tag{2.33}$$

From self-similarity, we have

$$X^{(k)} = 2^{-k(1-H)} X \tag{2.34}$$

and

$$Var[X^{(k)}] = \sigma_k^2 = 2^{-k(2-2H)} \sigma_0^2 . \tag{2.35}$$

Correspondingly, at a scale k, the signal energy of $X^{(k)}$ is expressed as

$$\varepsilon_{X^{(k)}} = E[(X^{(k)})^2] = \sigma_k^2 + \mu_0^2 . \tag{2.36}$$

We assume that the signal energy at the scale k is concentrated on the “large” signal defined as *bursts* B_k. Therefore, $\varepsilon_X \approx B_0^2$, and $\varepsilon_{X^{(k)}} \approx B_k^2$. Conveniently, we obtain the formula

$$B_k = \sqrt{2^{k(2H-2)}(B_0^2 - \mu_0^2) + \mu_0^2} . \tag{2.37}$$

Equation (2.37) can be used to derive a traffic burst size B_k from B_0. We can estimate the burst size B_0 from the maximum packet size against the network bandwidth. For example, on a 10Base-T Ethernet, a 1500-byte packet at 10Mb/s determines the smallest time interval for the timescale is approximate to 1.2ms. If a timescale is chosen as 2ms, then we find B_0 = 6Mb/s. With an estimated Hurst parameter H at each scale, this formulation results in a traffic burst profile.

There are some limitations in this algorithm. First, the formulation requires an estimation of the Hurst parameter in every two scales, which may not give an accurate result. Second, the distribution of traffic changes from a small scale to a large scale. So, the algorithm is not accurate to calculate bursts at large scales. Given those issues, this is a simple method to estimate the traffic flow throughput at the connection level.

2.3.5.2 Analysis with real traces

A. Description of the trace

The trace in this experiment was collected at the gateway of a college network at 7:30-8:30PM on Friday Jan 25, 2002. The trace has one-hour duration consisting of 800k IP frames. About 95.3% of frames and 98.5% of bytes are TCP, and 4.2% of frames and 1.4% of bytes are UDP. Within TCP, 45% of frames and 63.2% of bytes are KaZaa traffic(port 1214), only 28.4% of frames and 27% of bytes are Web (HTTP port 80). So, KaZaa traffic is more than twice as much as Web traffic in bytes in this trace.

The network has a T1 connection to access the Internet and a 10base-T Ethernet LAN linked to local users. A 100Mb/s Ethernet router switches traffic at the central hub, at which the trace was collected. A Linux box running *TCPdump* performed the packet recording. The size of the network is several hundred users.

B. Self-similarity analysis on traffic

Figure 2-9 plots the wavelet energy for KaZaa, Web and combined TCP traffic. It is shown that Web traffic has a knee at the scale 8. The equivalent timescale is around

250ms, which is approximately twice the typical RTT that HTTP connections had in the trace. This agrees exactly with the result in the previous sections that the “knee” is controlled by RTT. In the SRD region (scales 1 – 5), KaZaa curve is close to the combined TCP trace, and in the LRD region (scales 8 – 18), the Web curve is close to the TCP trace. However, in the SRD region, the Web traffic has little impact, while the KaZaa traffic does not affect the LRD region.

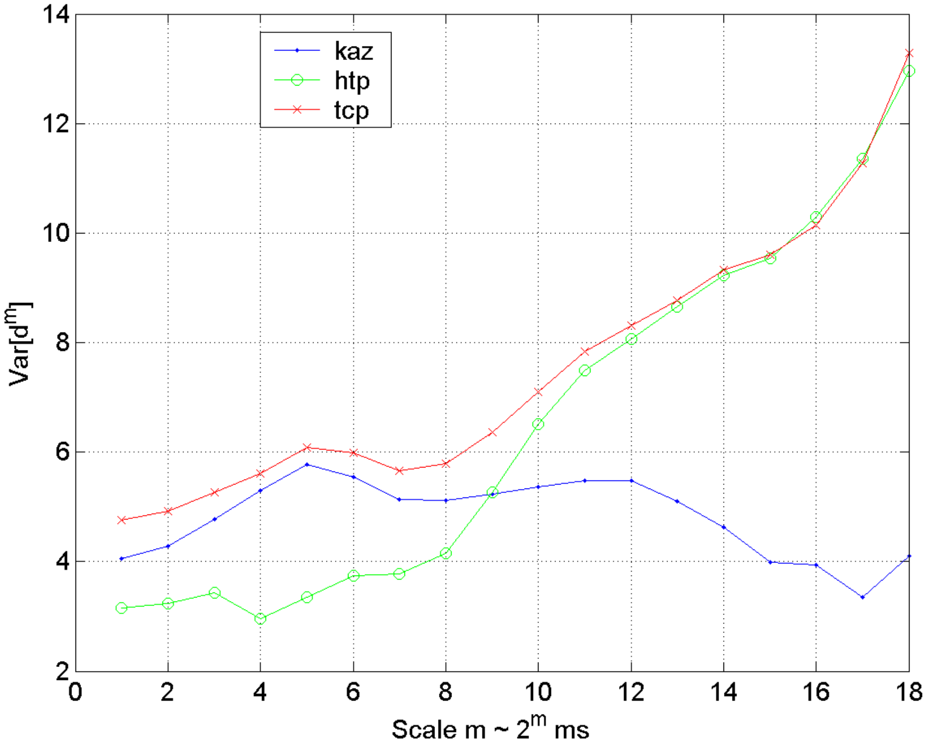


Figure 2-9 Comparisons of energy plots for KaZaa (kaz) and Web (htp) and the two-combined TCP traces

C. Traffic burst profiles

From a practical point of view, we may use traffic bursts in multiple timescales to classify the traffic processes. On a 10Mb/s Ethernet, the transmission time for a 1500-

byte packet is 1.2ms. The smallest timescale at 2.048ms is used to calculate the burst B_0. Applying (3.6), we can compute the burst sizes at ascending timescales. (Since it corresponds to a timescale, the traffic burst is the same as the burst bandwidth.)

Figure 2-10 plots the profile for the Web trace. At small timescales (m < 4), the bandwidth drops sharply. This corresponds to the packet-level bursts, showing the LAN behavior of traffic.

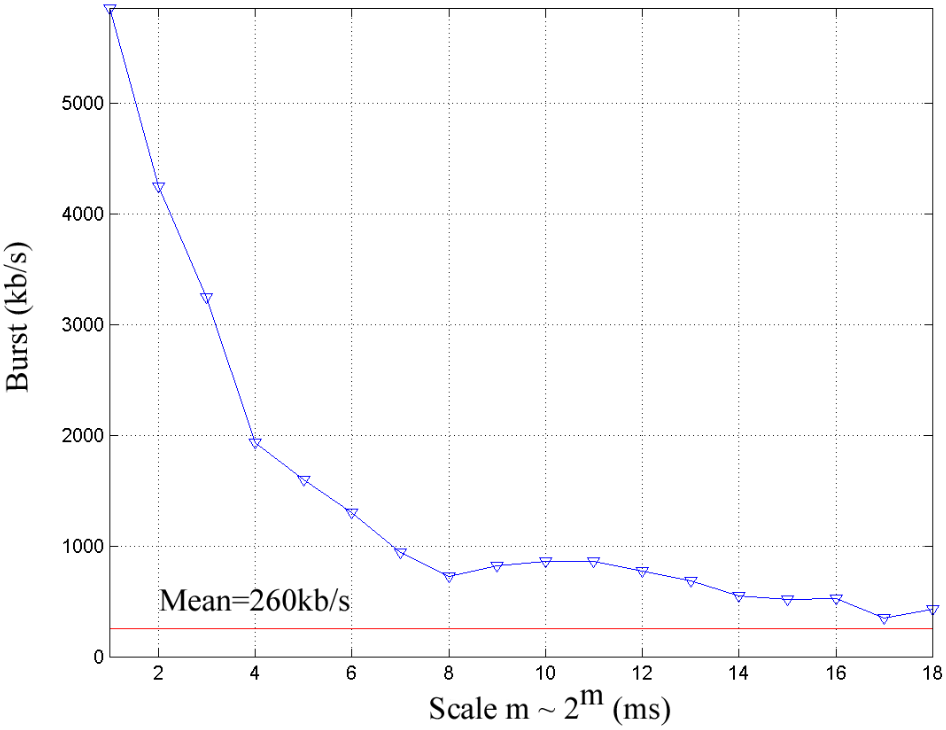


Figure 2-10 The Web traffic burst profile vs. timescales

The interesting region is the scale ranges m = 8 – 13 (250ms ~ 8s), where the burst bandwidth only fluctuates within 600 – 800kb/s. This is the range of the maximum throughput of the Web traffic flow at the common bottleneck of the T1 link access to the Internet. A scaling range can also be found in the energy plot (Figure 2-9), which is the initial linear region of LRD. It clearly indicates that the Web traffic is shaped by TCP mechanisms at these timescales. In other words, many TCP micro-flows of the Web traffic are multiplexed and share the T1 link at these timescales. Furthermore, the average throughput of the Web traffic flow is 260kb/s (below the burst bandwidth at the largest timescale). The plot shows the burstiness of Web traffic exists in all large scales, i.e., LRD.

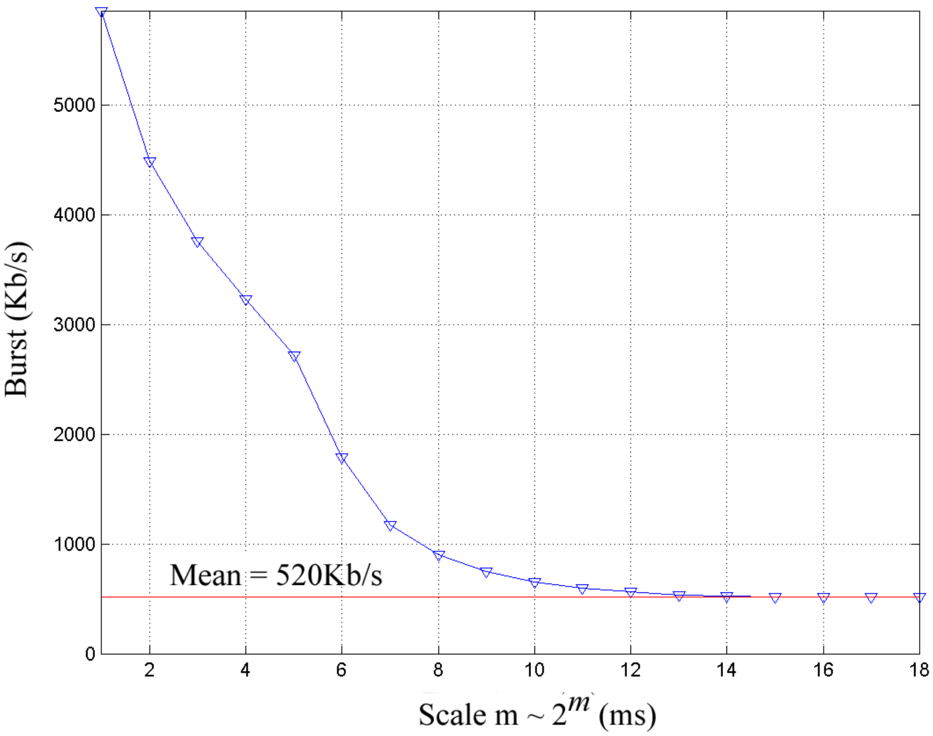


Figure 2-11 The KaZaa traffic burst profile vs. timescales

On the other hand, the traffic burst profile of the KaZaa trace displays a different picture (Figure 2-11). In the scale range m = 4 – 6, the traffic profile has a big "*belly*". But it has a very short "*tail*" after the scale m = 13, where the burst bandwidth becomes equal to the mean. So, this KaZaa trace has no LRD characteristics, but it is burstier at the packet level (the small timescales). These properties are also shown in the energy plot (Figure 2-9).

Note that the KaZaa traffic has a higher throughput at 520Kb/s, twice as much as the Web trace (260Kb/s). By TCP mechanisms, KaZaa traffic microflows are also shaped at the T1 link. But, it is at a much larger timescale m = 12 (approximately 4s), because the average RTT in the KaZaa trace is around 2s.

This study raises the question of whether Heavy-tailedness is the sole cause of traffic self-similarity at LRD. In both size and throughput, the KaZaa traffic flows are much larger than the Web flows, but KaZaa clearly shows less LRD. So, what else could determine LRD in network traffic? From an application point of view, KaZaa has little correlation at the connection level while Web traffic does.

2.3.6 Summary

Self-similarity is a mechanism that affects network traffic actively. In this study, this mechanism is interpreted in great detail with respects to traffic structures at the different protocol layers and timescales. This view is an improvement over the popular theory that self-similarity of network traffic is only caused by the heavy-tailed distribution of file sizes.

The scaling properties of traffic can generally be divided into two regimes: the SRD region at small timescales and the LRD region at large timescales. Previously, it was unclear how these regions are separated and how traffic is controlled in each scaling region. In this report, it can be identified that the network transmission delay (related to RTT) separates the two scaling regions. This gives a physical source of the periodicity in traffic self-similarity. By analyzing the results from numerous simulations, I have found that:

(1) Bandwidth, TCP window size, and packet size have impacts on the SRD region.

(2) Heavy-tailedness (Pareto shape parameter) shapes the structure of LRD, but it is not the only factor, and the heavy-tailedness of traffic may not be caused solely by the distribution of file sizes.

(3) These network parameters (e.g., bandwidth, packet size, TCP window size and RTT time) reallocate data energy in the signal power spectrum. The effect of their mechanisms will be examined in more detail in Chapter 3 using multifractal analysis.

The property of burstiness on traffic self-similarity leads to the development of a traffic burst profile. The traffic burst profile is applied to a real traffic trace to profile KaZaa and Web traffic, and the results are consistent with the network mechanisms, e.g., they represent the packet-level bursts and the connection-level throughputs at a shared link.

CHAPTER 3

MULTIFRACTAL DATA ANALYSIS

Self-similarity analysis uses the fractal power spectrum (i.e. second-order statistics) and cannot reveal any detail at a particular point. Fractals are heterogenous rather than homogenous. They are characterized by pointwise singularities. In a deterministic fractal like a Sierpinski Triangle, we can "*zoom in*" the image and find similar structures inside the image. However, most natural fractal objects such as landscapes are random: there is not exactly same shape found in all scales. In engineering, we want to find a fractal measurement to analyze fractals with randomness. This measurement is called *Multifractal Measure*, originally introduced by Mandelbrot in 1972. Multifractal analysis has several formulations. Among them two representations have been widely used in the literature. One is its original derivation based on the binomial cascade process[34]. Another analyzes fractal data in the wavelet paradigm and uses wavelet coefficients to represent the multifractal functions[3][19][29]. Both methods describe the fractal structure using the distribution of Hölder exponents.

In this chapter, we will study these two methods and apply MFA to analyze network traffic.

3.1 Multifractal analysis

3.1.1 Local singularity with Hölder exponent

A fractal is characterized by its singularities. Singularity implies an irregular structure within a signal. In mathematics, the notion of *pointwise regularity* (also called Hölder exponent) is used to represent the singularity concept in fractals. Below is a brief introduction of signal singularity analysis based on the wavelet theory [28][29].

Pointwise regularity describes the function's differentiability at a particular point. A function f with m-differentiations in a neighborhood of v can be approximated by a Taylor series p_v with a polynomial of degree m:

$$p_v(t) = \sum_{k=0}^{m-1} \frac{1}{k!} f^{(k)}(v)(t-v)^k . \tag{3.1}$$

If f has a singularity at a point v, then f is non-differentiable at v and there exists the *Hölder exponent*[1] α characterizing the singular behavior as

$$\forall t \in \mathbb{R}, | f(t) - p_v(t) | \leq K \, | t - v |^{\alpha} \ . \tag{3.2}$$

Here, p_v is a polynomial of degree $m = \lfloor \alpha \rfloor$ as expressed in (3.1), and K and α are positive real numbers such that (3.2) holds. The Hölder exponent α_v is the supremum of all α in (3.2). If $0 < \alpha < 1$, the function *f*(*t*) is bounded but discontinuous at *v*.

Localize signal singularities using wavelet analysis

To localize the singularities within a signal, we apply a multiscaling process using a wavelet function. In the neighbourhood of a point v, the function f can be approximated as

[1] The pointwise regularity is also referred to as *Lipschitz-Hölder* exponent in the mathematic context.

$$f(t) = p_v(t) + \gamma_v(t) \tag{3.3}$$

with

$$|\gamma_v(t)| \leq K|t-v|^{\alpha}.$$

In (3.3), p_v is a polynomial of (3.1). So, $\gamma_v(t)$ is equivalent to the right hand side of (3.2).

To analyze the signal, we apply a wavelet function $\psi(t)$, which has n vanishing moments ($n > \alpha$). This means $\psi(t)$ has n continuous derivatives with fast decay. So, at a point v, we have

$$\begin{aligned} Wf(u,s) &= \int_{-\infty}^{+\infty} f(t) s^{-\frac{1}{2}} \psi(\frac{t-u}{s}) dt \\ &= \int_{-\infty}^{+\infty} p_v(t) s^{-\frac{1}{2}} \psi(\frac{t-u}{s}) dt + \int_{-\infty}^{+\infty} \gamma_v(t) s^{-\frac{1}{2}} \psi(\frac{t-u}{s}) dt \\ &= W\gamma_v(u,s). \end{aligned} \tag{3.4}$$

The stability of wavelet analysis requires that f has some uniform regularity. If f is uniformly singular with the Hölder exponent α over [a, b], then the wavelet transform is bounded as

$$|Wf(u,s)| \leq Ks^{\alpha+1/2} \text{ for } K > 0. \tag{3.5}$$

Therefore, we can apply the wavelet function $\psi_{u,s}$ to locate α over the range [a, b] [29] p.169.

In wavelet theory, this is analyzed asymptotically. Let $\psi_{u,s}$ have a compact support within $[-C, C]$. At the neighbourhood of a point v, the wavelet function has a cone of influence as $|u-v| \leq Cs$. In the cone of ψ, $|Wf(u,s)|$ has a local maximum, called the *modulus maximum*. From (3.5), we can find

$$\log_2 |Wf(u,s)| \leq \log_2 K + (\alpha + \frac{1}{2}) \log_2 s. \tag{3.6}$$

The Hölder exponent α is equivalent to the largest slope in (3.6) along the maxima line moving towards the point *v*. So, this analysis method that isolates signal singularities in a wavelet domain is referred to as *wavelet modulus maxima* (WMM) [29] p.176.

Wavelet modulus maxima (Mallat) A point (u_0, s_0) on the wavelet time-scale plane is called *modulus maximum* if it satisfies both $Wf(u_0, s_0) \neq 0$ and $\frac{\partial}{\partial u} Wf(u_0, s_0) = 0$.

The wavelet transform $Wf(u,s)$ can be interpreted as the derivative of *f* averaged in the neighbourhood of *u* with a kernel dilated by *s*. If a Gaussian function θ is used, the kernel θ has a fast decay and $\lim_{s \to 0} \frac{1}{\sqrt{s}} \bar{\theta}_s = K\delta[0]$. A Gaussian kernel can guarantee that a series of modulus maximum points converge from coarse scales to the finest scale at a singular point [29] p.178.

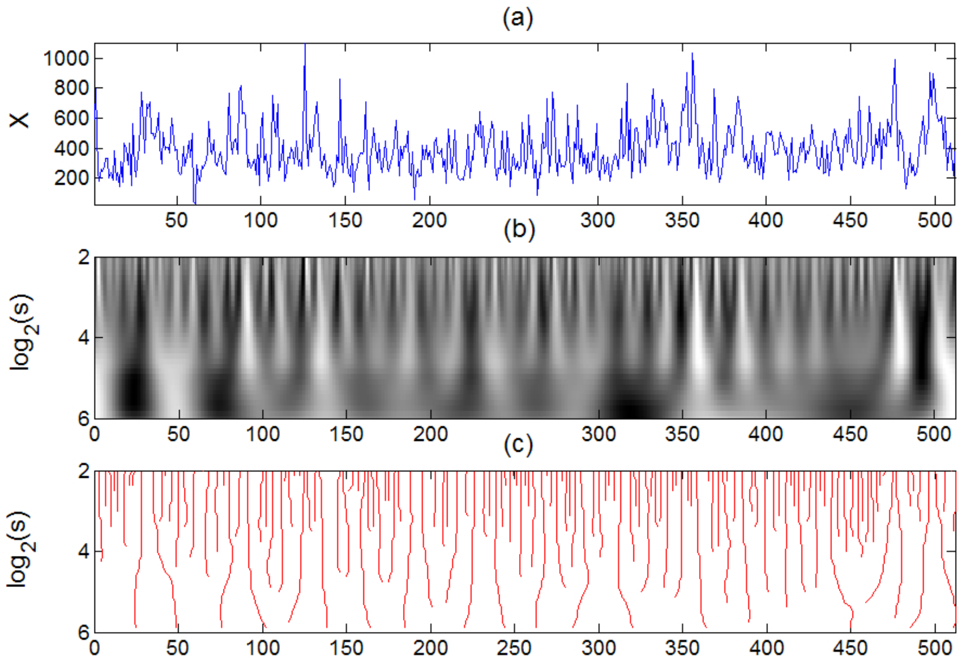


Figure 3-1 Wavelet modulus maxima of a network traffic trace

(a) A 512-point network traffic sample path. (b) Wavelet transformation of the signal with the first derivative of Gaussian function kernel. (c) Wavelet modulus maxima plots converge at the singular points. The figure is produced using *Wavelab 802* toolkit[46].

Figure 3-1 shows an example of wavelet modulus maxima analysis on a fractal signal. It demonstrates that a complex signal underlines some basic structure, which can be extracted by using wavelet modulus maxima analysis. The signal wavelet energy will follow the local maxima lines toward the singular points. For network traffic in Figure 3-1, we can see that the traffic signal is singular almost everywhere.

3.1.2 Multifractal

Wavelet theory can analyze various fractal signals. But it has a stability problem. The issue of choosing a wavelet function for a particular type of signals currently is an advanced research topic by itself. Network traffic generally has a Hurst parameter less than one. That means that the vanishing moment of a wavelet function does not help much isolating singularities of signals. To analyze a positive data process such as network traffic, we can use a simpler method, multifractal *binomial measure*, which was originally proposed by Mandelbrot. Several works have studied this subject [29][30][32][33][34].

3.1.2.1 Binomial cascade processes

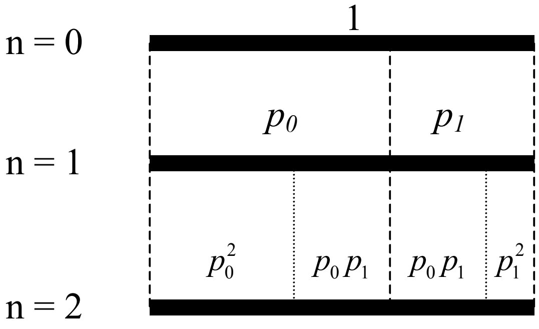


Figure 3-2 Illustration of binomial cascade

A binomial cascade (BC) [34] is a simple deterministic multifractal process. Shown in Figure 3-2, the process begins with a uniform mass of 1. Iteratively, each segment is subdivided into two parts proportional to p_0 and p_1. Here, set $p_0 + p_1 = 1$. At a stage n, the number of intervals is $N = 2^n$, which can be represented by a set $X_{\{n\}} = \{x_0, x_1, x_2, ..., x_{2^n-1}\}$. For example,

$$\begin{aligned}
x_0 &= \left[0, 2^{-n}\right], \\
x_1 &= \left[2^{-n}, 2^{-n+1}\right], \\
&\dots \\
x_i &= \left[2^{-n} i, 2^{-n}(i+1)\right], \; i = \; 1, 2, ..., 2^{\mathrm{n}} - 1.
\end{aligned}$$

Therefore, the mass of a subinterval x_i is $\mu_i = p_0^k p_1^{(n-k)},\ k \le n,$ and the binomial measure is denoted as $M(x_i) = \mu_i$. Let S_ξ denote a subset of subintervals with a measure $\mu_\xi = p_0^k p_1^{(n-k)},\ \xi = {}^k\!/_n$. The population of S_ξ is

$$N_n(\xi) = \frac{n!}{(\xi n)!\big((1-\xi)n\big)!}. \tag{3.7}$$

With $N_n(\xi)$, we can quantify a fractal set S_ξ with a particular value ξ, which leads to the formulation of fractal dimension and multifractal spectra.

3.1.2.2 Fractal dimension and scaling exponent $\tau(q)$

In a fractal set, the length of a subinterval is $\delta = 2^{-n}$. Therefore, we can calculate the measure $M_d(S_\xi)$ as $\delta \to 0$ and $n \to \infty$, and determine the fractal dimension $D(\xi)$ of the set S_ξ by the following:

$$M_d(S_\xi) = N_n(\xi)\delta^d \xrightarrow[\delta \to 0]{} \begin{cases} 0, & d > D(\xi), \\ \infty, & d < D(\xi). \end{cases} \tag{3.8}$$

Expression (3.8) implies that $M_d(S_\xi)$ will be countable as $\delta \to 0$ only for $d = D(\xi)$.

For a non-deterministic fractal, we have to measure the scaling function to calculate the fractal spectrum. A set *S* with *N* elements has a measure

$$M_d(q,\delta) = \sum_{i=1}^{N} \mu_i^q \delta^d = N(q,\delta)\delta^d \xrightarrow[\delta \to 0]{} \begin{cases} 0, & d > -\tau(q), \\ \infty, & d < -\tau(q). \end{cases} \tag{3.9}$$

The exponent $d = -\tau(q)$ determines that the measure converges as $\delta \to 0$. The moments of the random variable $\{\mu_I\}$ are scalable with $\delta^{\tau(q)}$. So, $N(q,\delta)$ can be expressed as

$$N(q,\delta) = \sum_{i=1}^{N} \mu_i^q \sim \delta^{\tau(q)}. \tag{3.10}$$

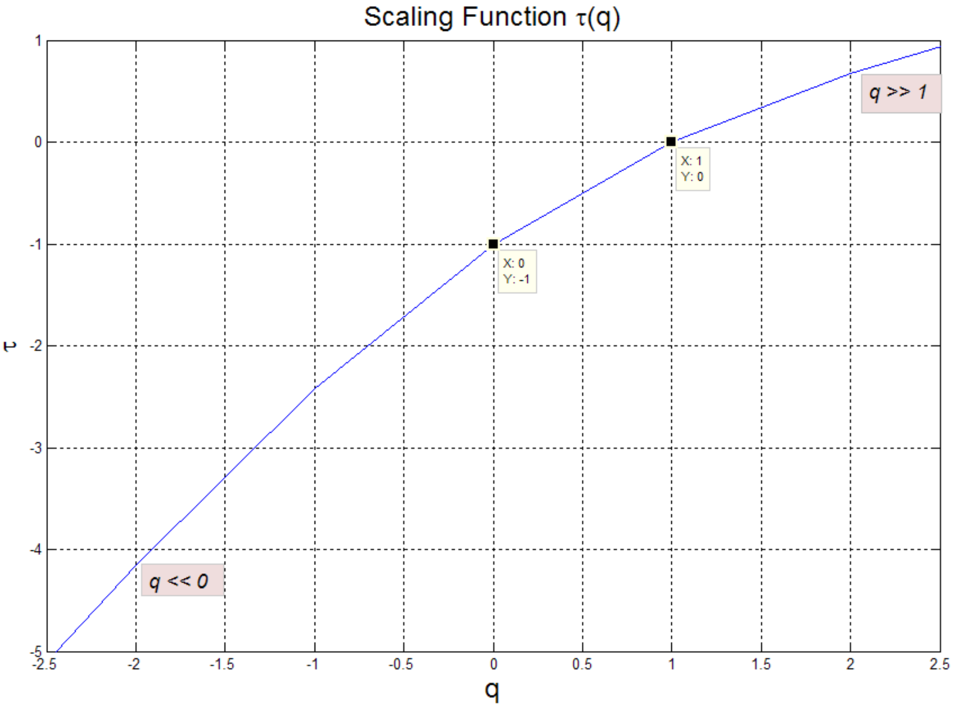


Figure 3-3 The scaling function of a binomial measure.

$\tau(q)$ is ascending and has two special values at (0, -1) and (1, 0).

The exponent $\tau(q)$ is called *scaling function*, which weights the random variable $\{\mu_I\}$ by choosing a moment q and thus characterizes the fractal set. It can be expressed as

$$\tau(q) = \lim_{\delta \to 0} \frac{\ln N(q,\delta)}{\ln \delta} = \lim_{\delta \to 0} \frac{\ln\left(\sum_{i=1}^{N} \mu_i^q\right)}{\ln \delta}. \tag{3.11}$$

There are following special points on $\tau(q)$:

$$(i)\quad q=0,\ N(q,\delta)=N(\delta),\ \tau(0)=D;$$
$$(ii)\ \ q=1,\ N(q,\delta)=\sum_{i=1}^{N}\mu_i=1,\ \tau(1)=0.$$

When q varies in the range $(-\infty,+\infty)$, it selects different values of $\{\mu_I\}$. The small values of $\{\mu_I\}$ are captured by $q\to-\infty$, and the large $\{\mu_I\}$ are controlled by $q\to+\infty$. Correspondingly, there are two limits in $\tau(q)$. Let μ_- and μ_+ denote the minimum and the maximum of $\{\mu_I\}$, respectively. We should have

$$\left.\frac{d\tau(q)}{dq}\right|_{q\to-\infty}=-\lim_{\delta\to0}\frac{\ln\mu_-}{\ln\delta}=-\alpha_{\max}\,, \tag{3.12}$$

and

$$\left.\frac{d\tau(q)}{dq}\right|_{q\to+\infty}=-\lim_{\delta\to0}\frac{\ln\mu_+}{\ln\delta}=-\alpha_{\min}\,. \tag{3.13}$$

Also, at $q=1$, we have

$$\left.\frac{d\tau(q)}{dq}\right|_{q=1}=\lim_{\delta\to0}\frac{\sum_{i=1}^{N}\mu_i\ln\mu_i}{\ln\delta}=-\lim_{\delta\to0}\frac{\mathrm{H}(\delta)}{\ln\delta}\,. \tag{3.14}$$

$\mathrm{H}(\delta)$ is recognized as the *partition information entropy* at the resolution scale size δ. Let α_S be the fractal dimension of the set S_{α_S}, we have

$$\mathrm{H}(\delta)=-\alpha_S\ln\delta\,. \tag{3.15}$$

In particular, the binomial cascade process has

$$\tau(q)=-\frac{\ln(p_0^q+p_1^q)}{\ln 2}\,. \tag{3.16}$$

3.1.2.3 $f(\alpha)$ curve and multifractal spectrum

The scaling function $\tau(q)$ characterizes the multifractal set and leads to the formulation of the multifractal spectrum $f(\alpha)$.

$f(\alpha)$ on a multiplicative binomial process

From Section 3.1.2.1, (3.7) can be written approximately as [31] p.74

$$N_n(\xi) \simeq \frac{1}{\sqrt{2\pi\xi n(1-\xi)}} \exp\left\{-n\left(\xi \ln \xi + (1-\xi)\ln(1-\xi)\right)\right\}. \tag{3.17}$$

Then, we can define an exponent

$$f(\xi) = -\frac{\xi \ln \xi + (1-\xi)\ln(1-\xi)}{\ln 2}, \tag{3.18}$$

so that we have $M_d(S_\xi) \sim \delta^{-f(\xi)}\delta^d$. Therefore, the fractal dimension of set S_ξ is $d = f(\xi)$. $f(\xi)$ is a convex curve with a maximum $f(0.5) = 1$. The measure $M(x)$ is completely characterized by the union of fractal subsets S_ξ, each of which has its own fractal dimension.

Using the definition of Hölder exponent at a singularity, we can write

$$\mu_\xi = M(x(\xi)+\delta) - M(x(\xi)) = \delta^\alpha. \tag{3.19}$$

The measure $\mu_\xi = p_0^{n\xi} p_1^{n(1-\xi)}$ gives the relationship between the Hölder exponent and the fractal set index ξ as below

$$\alpha(\xi) = \frac{\ln \mu_\xi}{\ln \delta} = -\frac{\xi \ln p_0 + (1-\xi)\ln p_1}{\ln 2}. \tag{3.20}$$

Given p_0 and p_1 ($p_0 < p_1$), (3.20) is a linear mapping between α and ξ. For $0 \le \xi \le 1$, we have

$$\begin{cases} \alpha_{\min} = -\ln p_1 / \ln 2, \text{ as } \xi = 0 \text{ and } p_1 \ge \frac{1}{2}, \\ \alpha_{\max} = -\ln p_0 / \ln 2, \text{ as } \xi = 1 \text{ and } p_0 \le \frac{1}{2}. \end{cases} \tag{3.21}$$

So, $f(\xi)$ can be replaced by $f(\alpha)$.

In general, for a multifractal set S, suppose that there is a fractal subset S_α containing a Hölder exponent α. $N(q,\delta)$ is approximated as $\delta^{-f(\alpha)+\alpha q}$, which is dominated at $\alpha(q)$ as $\alpha(q) \times q - f(\alpha)$ reaches the maximum. The exponent is related to $\tau(q)$. If we know the scaling function $\tau(q)$, we may calculate $f(\alpha)$ as the following:

$$\alpha(q) = -\frac{d\tau(q)}{dq}, \tag{3.22}$$

then,

$$f(\alpha(q)) = q\alpha(q) - \tau(q). \tag{3.23}$$

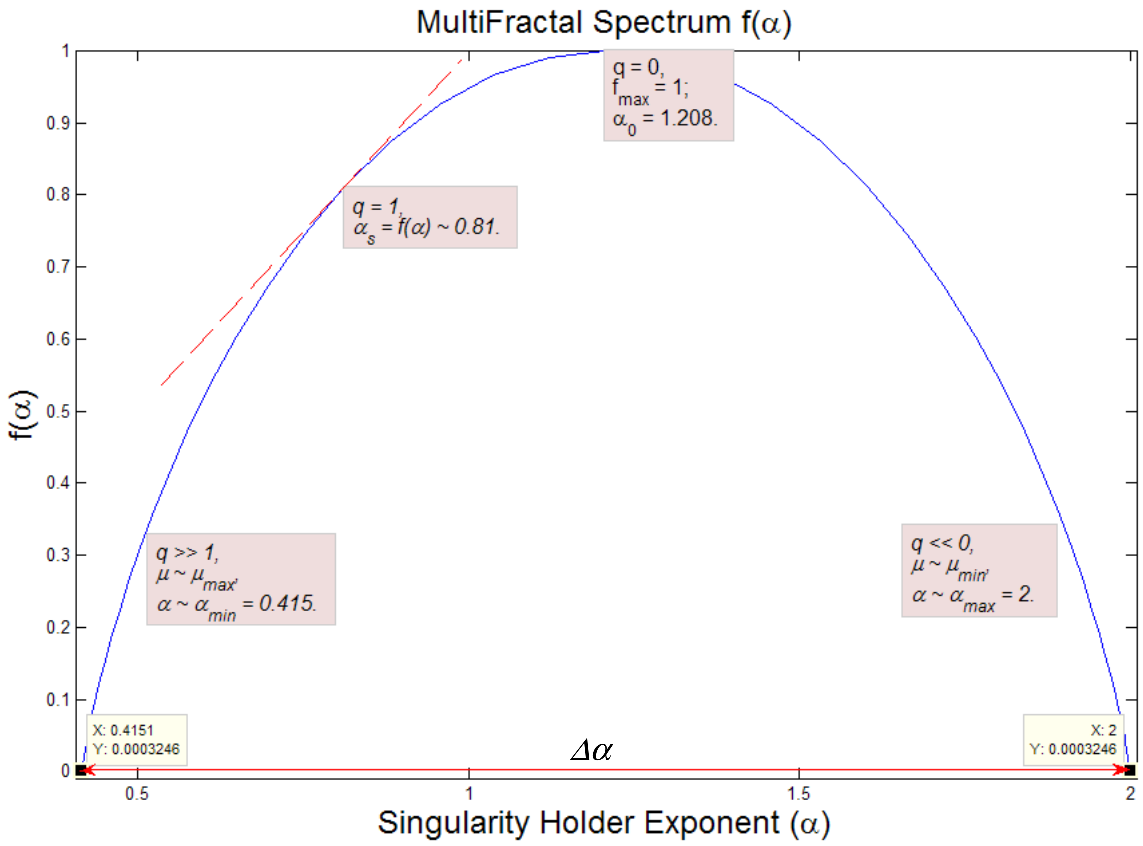


Figure 3-4 Multifractal spectrum f(a) for a binomial cascade process with p0 = 0.25.

Figure 3-4 is the multifractal spectrum for a binomial cascade with $p_0 = 0.25$. In the middle is α_0 – the center exponent corresponding to $f_{max} = 1$. The point at $\alpha_s = f(\alpha)$ is the partition entropy of the process. The range of multifractal spectrum is $\Delta\alpha$ between α_{min} = 0.4151 and α_{max} = 2, which are related to the multifractal measures μ_{max} as $q \rightarrow +\infty$ and μ_{min} as $q \rightarrow -\infty$, respectively. The parameter q is the moment for the multifractal measure $M_d(q,\delta)$.

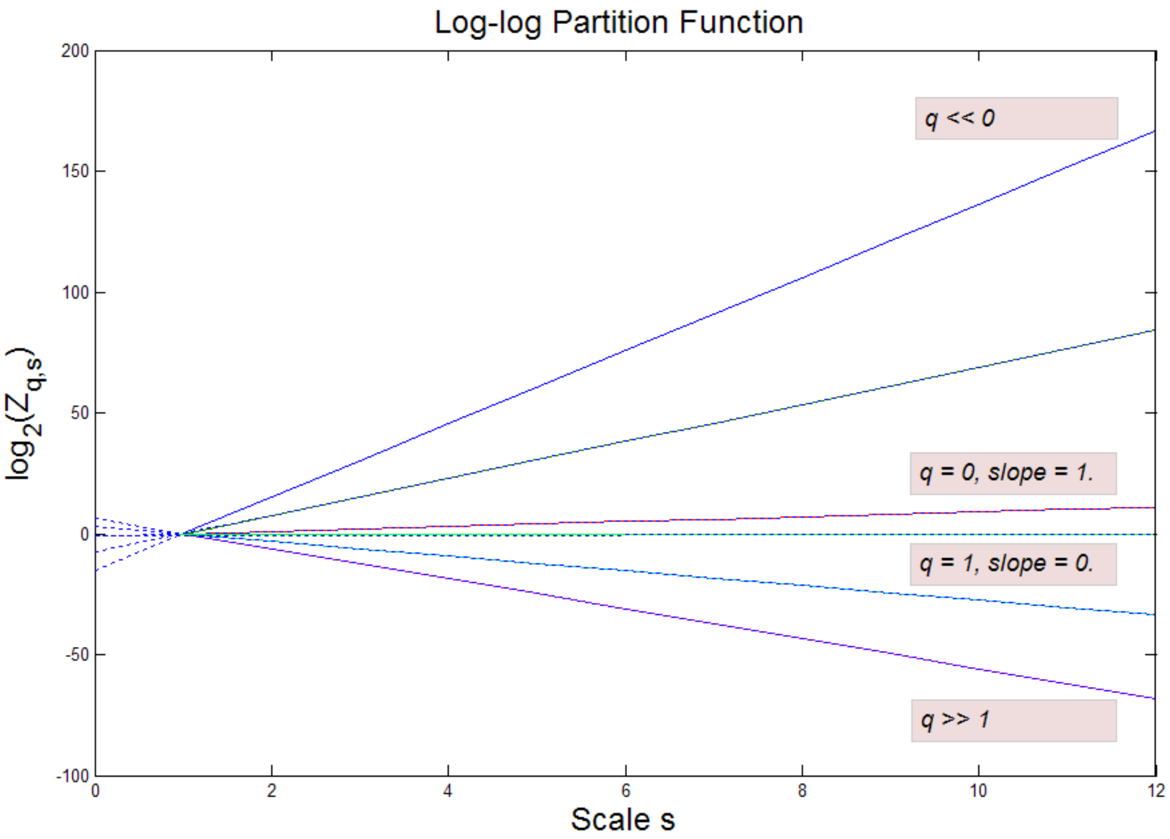


Figure 3-5 Plots of partition function at different q values.

3.1.2.4 Wavelet multifractal analysis (WMFA)

Multifractal spectrum can be calculated using a wavelet transform. In WMFA, we define a *Partition function* $Z(q,s)$, similar to the measure $M_d(q,\delta)$, as

$$Z(q,s)=\sum_{i}^{N}\left|Wf(u_i,s)\right|^q . \tag{3.24}$$

So, let $\tau(q)$ denote the scaling function that measures the asymptotic decay of $Z(q, s)$:

$\tau(q)=\lim\limits_{s\to 0}\dfrac{\ln Z(q,s)}{\ln s}$ and $Z(q,s)\sim s^{\tau(q)}$. With $\left|Wf(u,s)\right|\sim s^{\alpha+\frac{1}{2}}$, the scaling function is

$$\tau(q)=\inf_{\alpha}(q(\alpha+\tfrac{1}{2})-f(\alpha)) . \tag{3.25}$$

With an inverse of (3.25), the multifractal spectrum in WMFA is

$$f(\alpha) = \inf_q (q(\alpha + \frac{1}{2}) - \tau(q)) . \qquad (3.26)$$

3.1.2.5 Summary

In summary, a multifractal set *S* is a union of fractal subsets S_α with the pointwise Hölder exponent α. The multifractal set has a measure $M_d(q,\delta)$, which is controlled by the order of moments *q* and characterized by the scaling function $\tau(q)$. At a moment *q*, only proper values of μ_i are selected, and correspondingly $\alpha(q)$ is found. From $\alpha(q)$, we can calculate the multifractal spectrum $f(\alpha)$.

3.2 Multifractal data analysis

Multifractal analysis can classify different classes of processes. It is an advanced analysis method, but it is non-intuitive in real data analysis. In this section, MFA is tested using two known processes, binomial cascade and Poisson. Then, MFA of network traffic processes are compared with these cases.

3.2.1 Multifractal spectra of general data processes

3.2.1.1 Binomial cascade processes (BC)

Described in Section 3.1.2.1, a binomial cascade process has a binomial measure $\mu_i = p_0^k p_1^{(n-k)},\ k \le n.$ BC is a deterministic fractal process. Shown in Figure 3-6, there are three traces for p = 0.25, 0.35, and 0.45 respectively. We see that the trace 1 (p = 0.25) is more vibrant than the trace 3 (p = 0.45). Figure 3-7 shows the histograms of the three traces with a $\log_2$ scale. It illustrates that their log-scaled distributions are symmetrical.

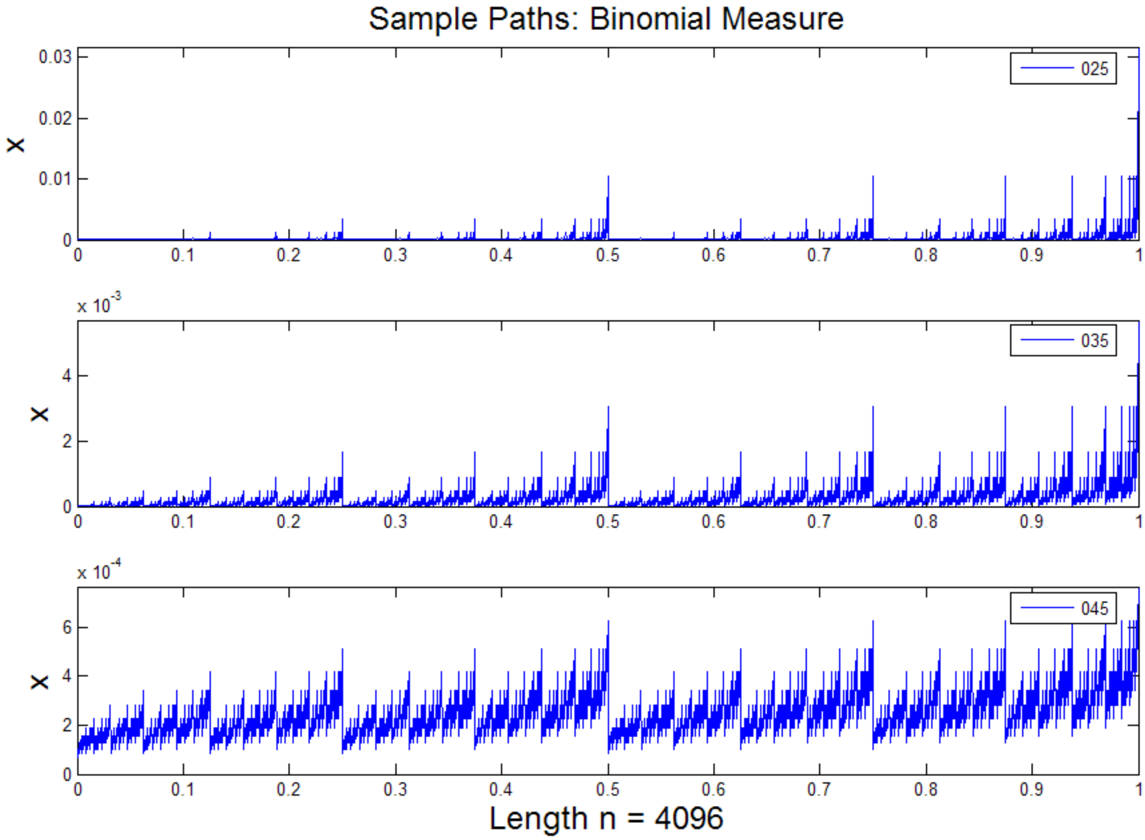


Figure 3-6 Sample paths of three binomial cascade processes: p = 0.25, 0.35, and 0.45, respectively.

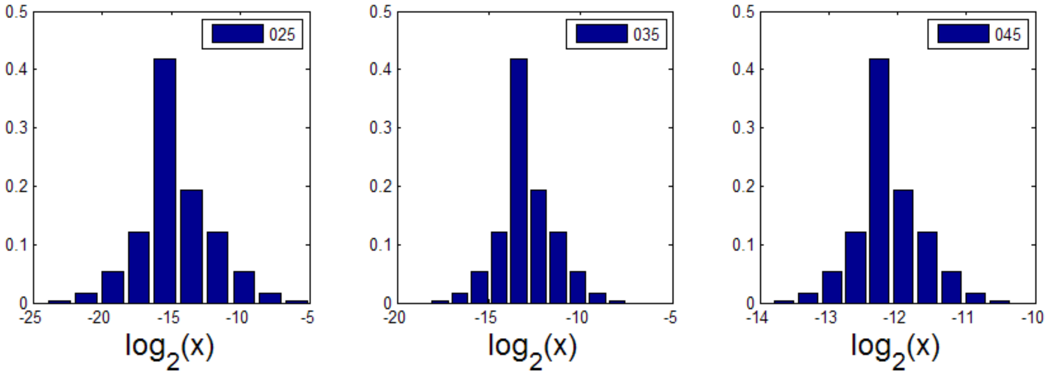


Figure 3-7 Histograms in log-scale for the three binomial cascades.

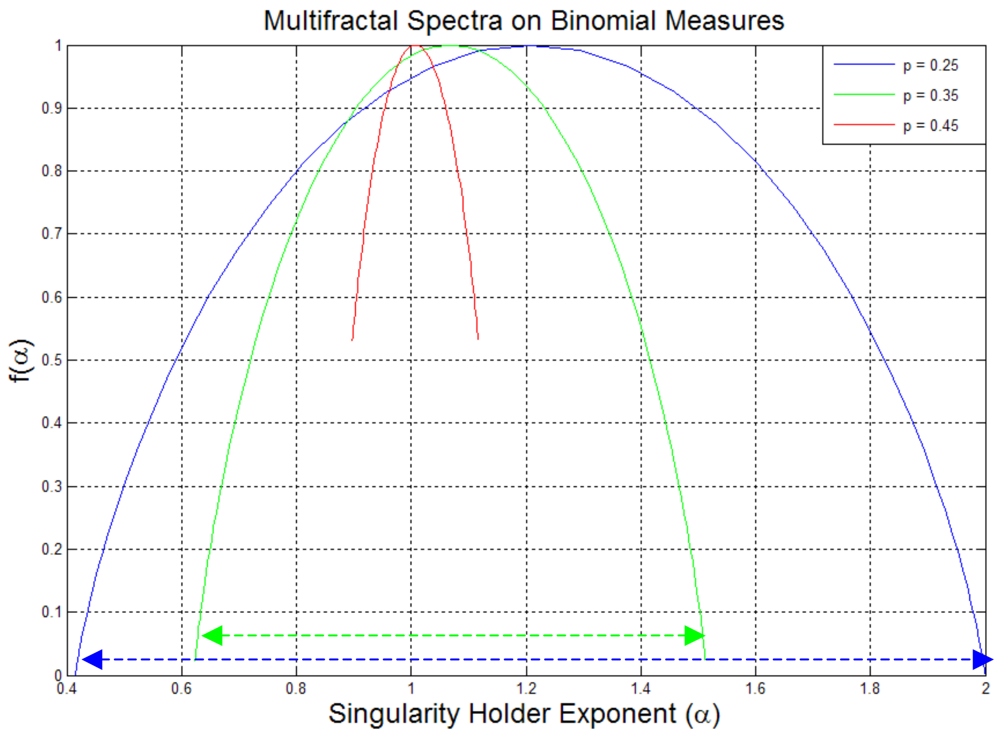


Figure 3-8 Multifractal spectra of three binomial cascade processes.

The multifractal spectra of these BC processes are plotted in Figure 3-8. We can find the following properties:

(1) The width of $f(\alpha)$ is related to the variations of data values. A large span of $f(\alpha)$ reflects large data changes in the processes.

(2) When p increases, $f(\alpha)$ moves to the left, and α_0 increases. This shift indicates the changes of the data structure.

(3) $f(\alpha)$ is symmetrical and centered at α_0. This is because the log-scaled histograms are also symmetrical in Figure 3-7.

3.2.1.2 Poisson processes

Figure 3-9 plots three Poisson processes with mean = 1, 2, and 5, respectively. Figure 3-10 illustrates their log-scaled histograms.

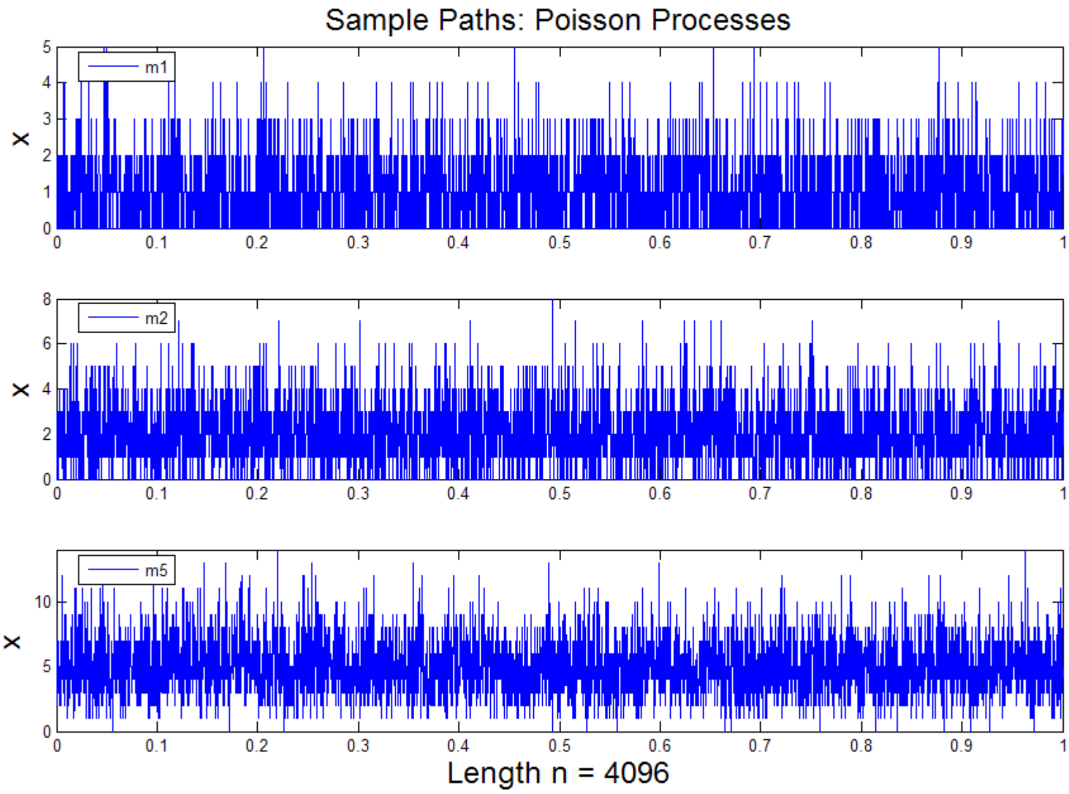


Figure 3-9 Sample paths of two Poisson processes, mean = 1, 2, 5.

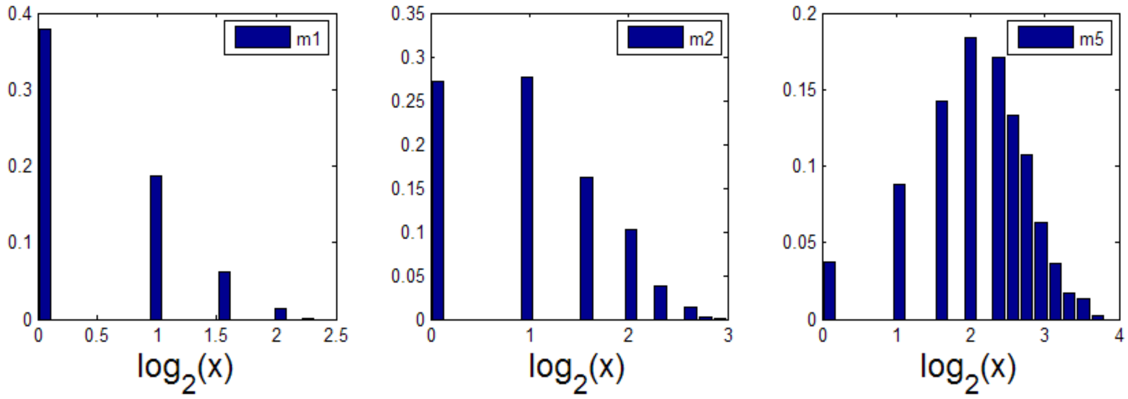


Figure 3-10 Histograms with log-scales of three Poisson processes with mean 1, 2, and 5.

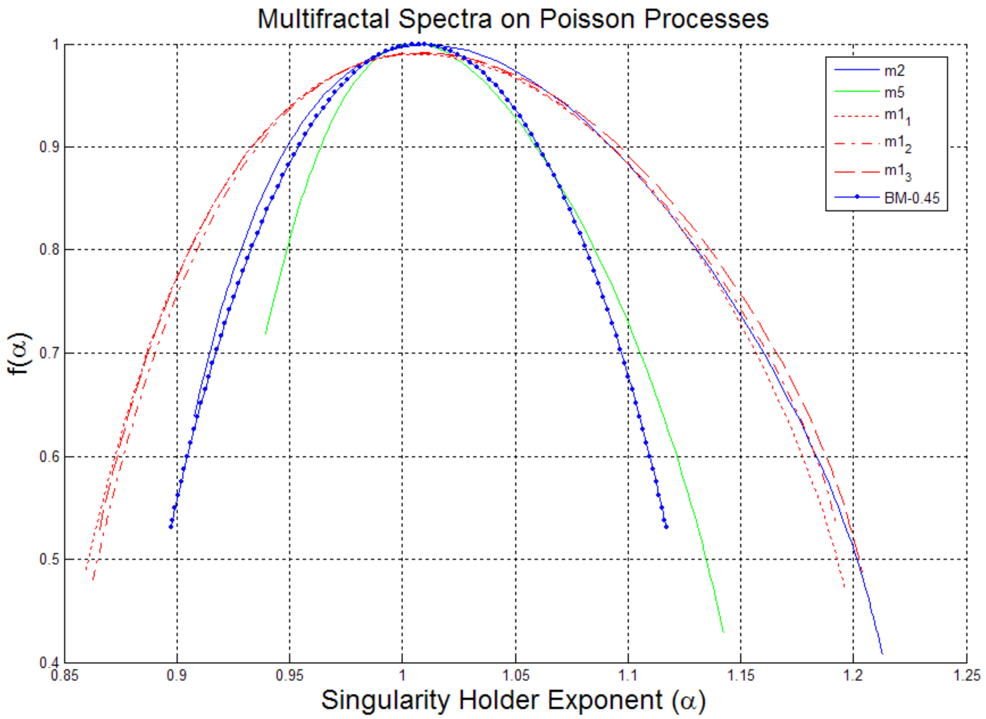


Figure 3-11 Multifractal spectra of Poisson processes.

In this test, Poisson random processes are compared with a binomial cascade process (p = 0.45). Figure 3-9 plots three Poisson processes with the same mean = 1, and two other Poisson processes with mean = 2 and 5, respectively. Although Poisson processes are not fractal, they exhibit scaling in some limited ranges. We can observe the following features from Figure 3-10, Figure 3-11, and Figure 3-12:

(1) ***Asymmetrical***: Unlike BC, the multifractal spectra of Poisson processes are asymmetrical. This can be explained by their asymmetrical log-scaled distributions shown in Figure 3-10. In a Poisson process, there are a few large values and many small values close to the mean. So, it appears in MFA that the left-side curve is short and the right side is extended.

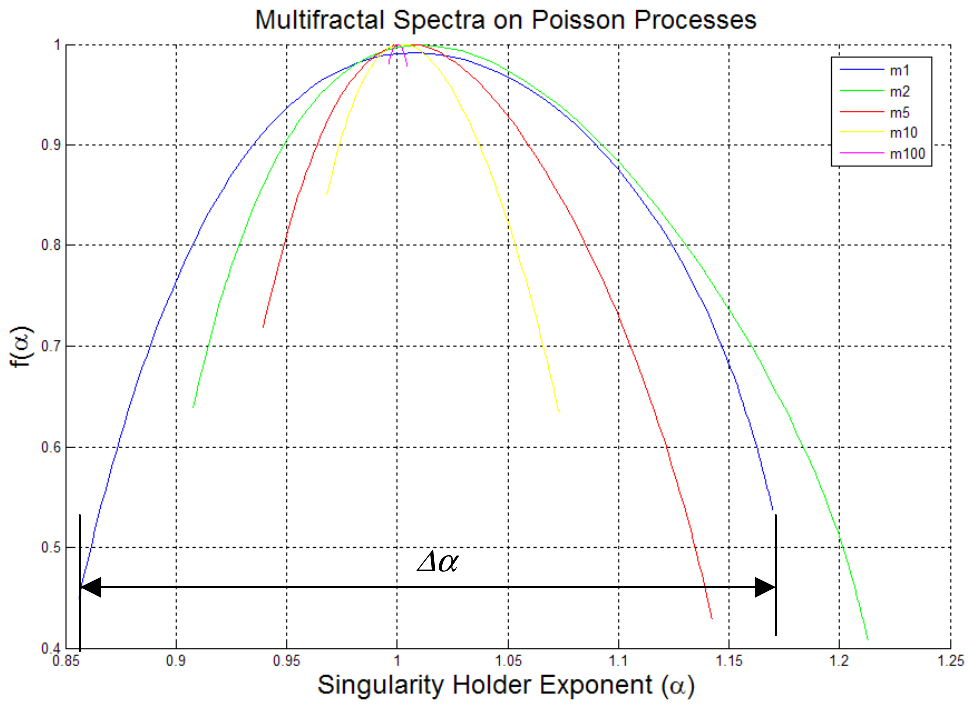


Figure 3-12 Poisson process multifractal spectra (2).

(2) ***Narrow***: In MFA plots of Poisson processes, α varies in small ranges: $\Delta\alpha$ ~ 0.2 for the Poisson process with mean = 5; and $\Delta\alpha$ ~ 0.35 for the Poisson processes with mean = 1. As the mean of a Poisson process increases, the range of α narrows (Figure 3-12).

(3) ***Identical on fractality***: Unlike binomial cascades (Figure 3-8), these MFA plots look very similar to each other. For example, all are almost centered at $\alpha_0 \sim 1.0$. The plots (labeled '$m1_1$', '$m1_2$', '$m1_3$' in Figure 3-11) are three separated Poisson processes with the same mean of 1. They are very close to each other on multifractal spectra.

3.2.2 Multifractal spectra on network traffic processes

3.2.2.1 Simulated network traffic

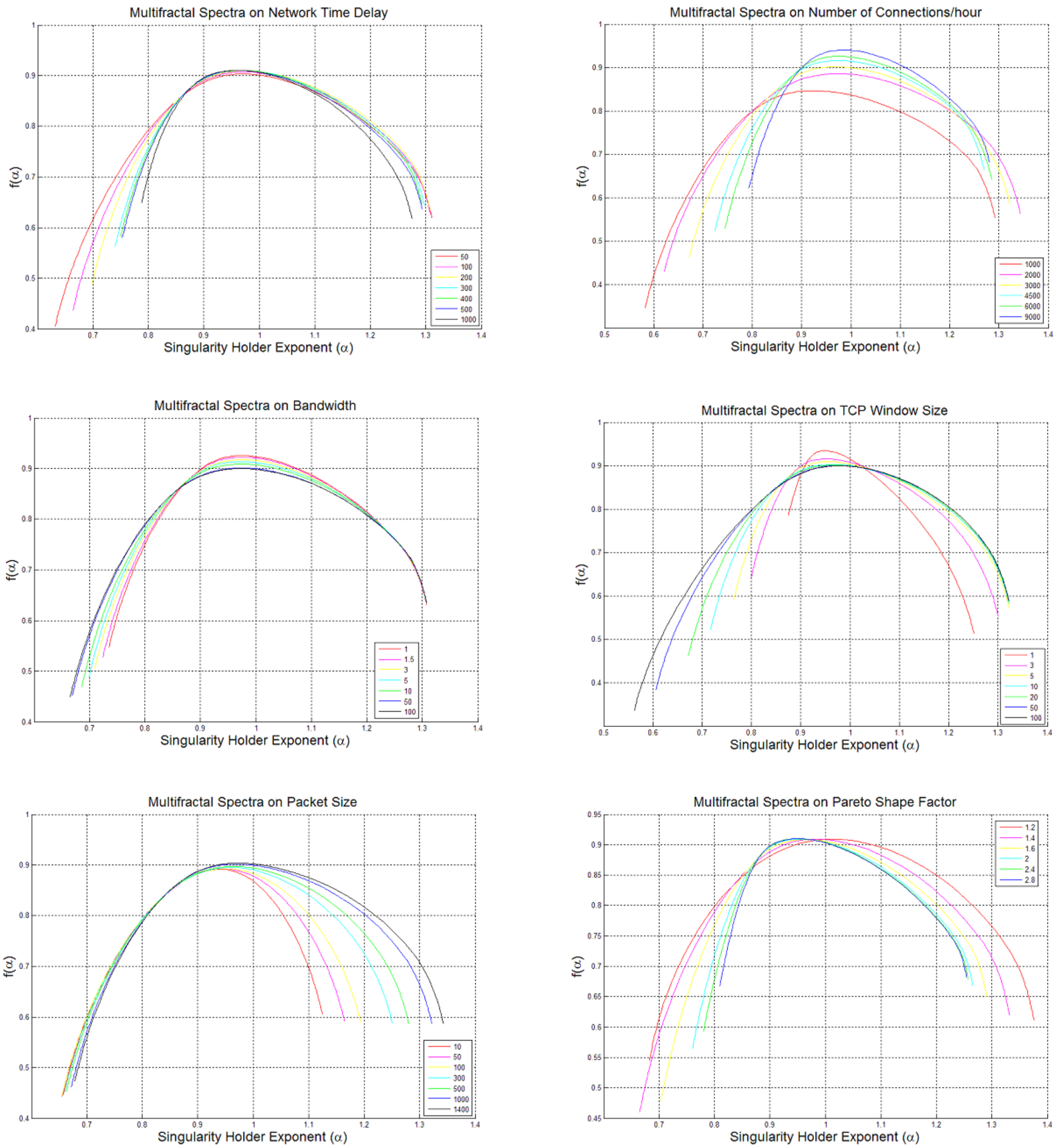


Figure 3-13 Multifractal spectra of network traffic (using NS-2 simulations) on small timescales (SRD). Minimum timescale is 1.024ms.

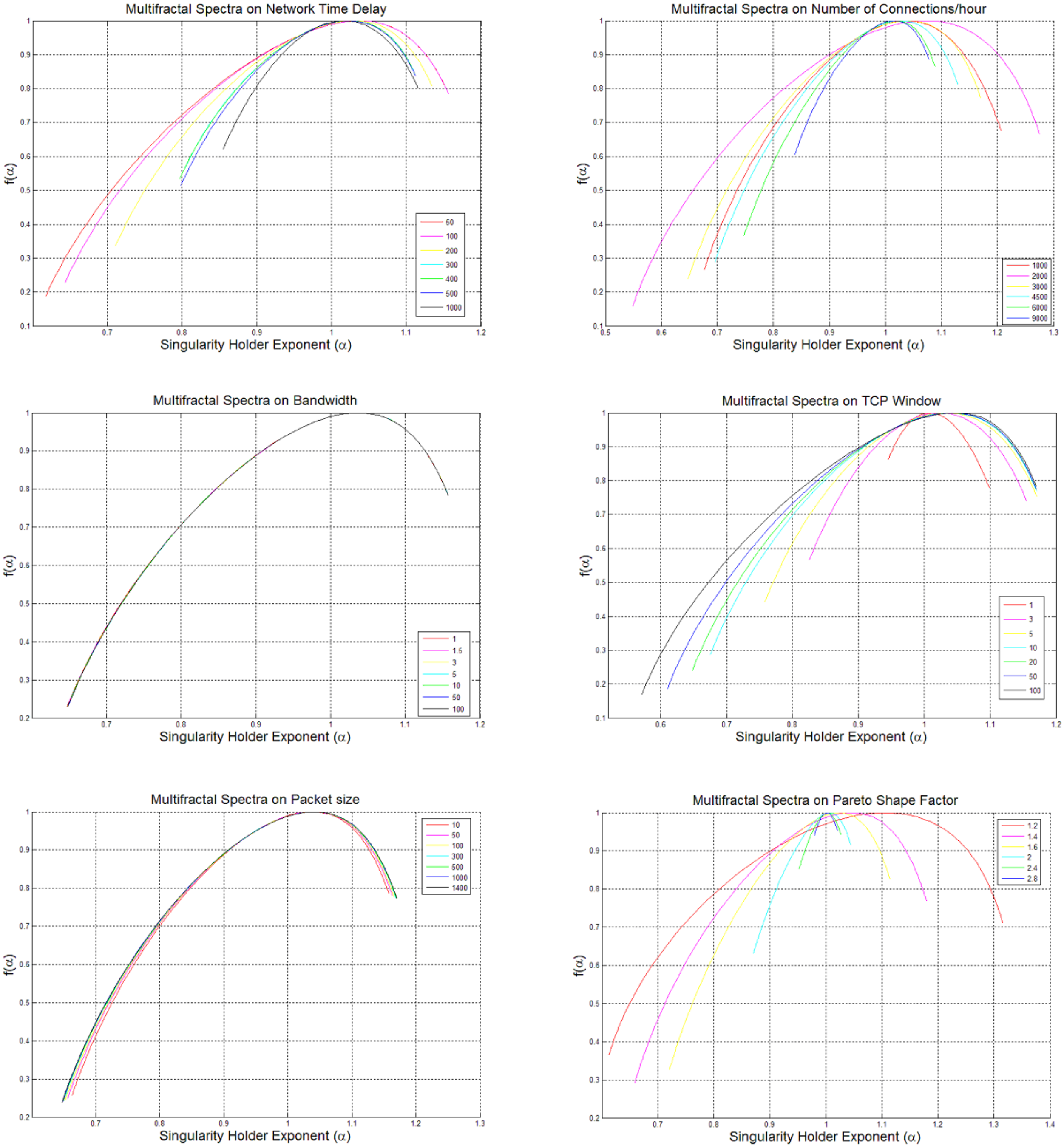


Figure 3-14 Multifractal spectra of network traffic (using NS-2 simulations) on large timescales (LRD). Minimum timescale is around 0.5s.

In this test, multifractal analysis is applied to the same simulated network traffic datasets presented in Chapter 2. Multifractal analysis allows us to examine the data processes in more detail. Two versions of results are presented: (i) MFA on small timescales (SRD) in Figure 3-13; and (ii) MFA on large timescales (LRD) in Figure 3-14. The following can be found:

(1) Shorter network delay results in burstier traffic. This is shown in that the multifractal spectrum becomes wider as the delay decreases, especially on the left side of the spectrum.

(2) With respect to connection rate (or the number of connections per hour), we expect traffic to be burstier when there are fewer connections. This is because there is less traffic load on the network, so an individual traffic flow becomes burstier. Also, the fractal structure changes, which is not clear in self-similarity analysis reported in Chapter 2.

(3) With respect to bandwidth, traffic seems to be faster only at small timescales (SRD) as the bandwidth increases. With large timescales (LRD), the behavior is the same (Figure 3-14). This will be discussed further in Chapter 5.

(4) The TCP sender's congestion window is another factor which affects greatly the burstiness of TCP traffic. For both SRD and LRD, with larger window sizes, the multifractal spectrum shows that traffic is burstier. However, with the smaller windows, the spectrum becomes close to a Poisson distribution's: $\Delta\alpha$ shrinks, and the two sides of the spectrum are

uneven. The right side of the spectrum is larger, i.e., traffic is constrained at the sender side.

(5) Increasing packet sizes does not affect the fractal property in LRD but it does affect SRD. In SRD, the right side of the spectrum gets wider.

(6) With respect to the Pareto shape factor, traffic becomes burstier when the factor decreases to 1. It is the same as the results seen from self-similarity analysis.

In general, these traffic processes belong to the same class of random process with the following common features:

(1) The ranges of α are mostly between 0.7 and 1.2 in LRD and between 0.7 and 1.3 in SRD.

(2) The left sides of multifractal spectra are much larger, especially in LRD. This is a clear difference from a Poisson distribution's. It indicates that network traffic is burstier than a Poisson process.

(3) The centers of the spectra α_0 are around 0.9 ~ 1.0 in SRD and around 1.0 ~ 1.1 in LRD. Evidently, three factors affecting α_0 are connection rate, TCP window size and Pareto shape factor.

3.2.2.2 Internet traffic traces

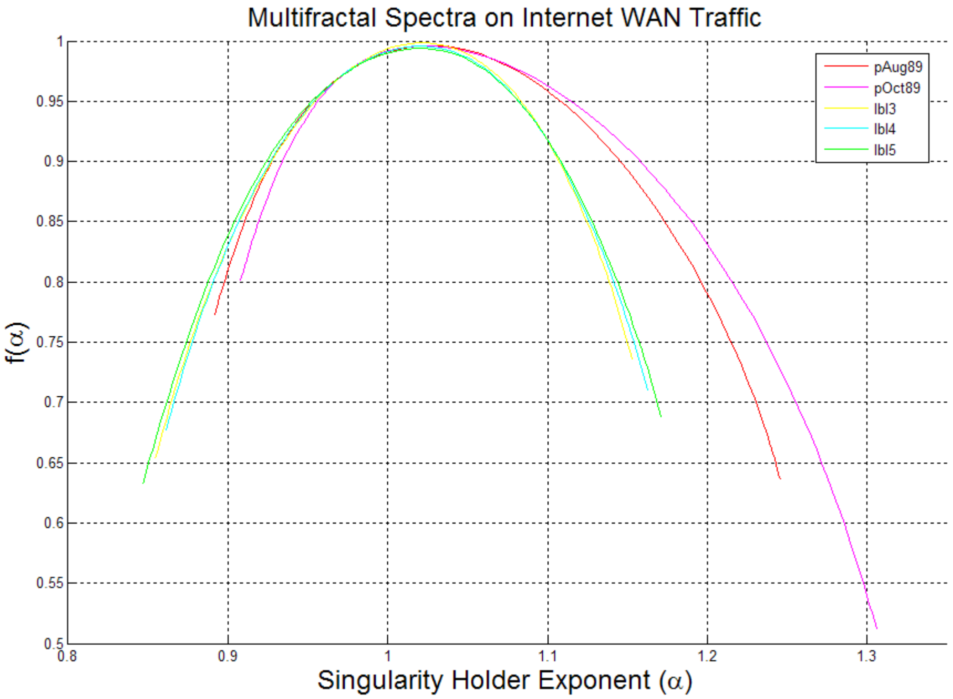


Figure 3-15 Multifractal spectra of Internet traffic (from Internet Traffic Archive).

In this test, I use five Internet traffic traces downloaded from Internet Traffic Archive[59]. The two traces 'pAug89' and 'pOct89' are WAN traffic from Bellcore. Each contains 1 million Ethernet packets. 'pAug89' began at 11:25am on August 29, 1989, and ran for about 3142.82 seconds. 'pOct89' began at 11:00am on October 5, 1989, and ran for about 1759.62 seconds. There are also three traces from Lawrence Berkeley Laboratory (LBL). The trace 'LBL3' is a two-hour WAN TCP traffic, time from 14:10 to 16:10 on Thursday, January 20, 1994, capturing 1.8 million TCP packets. The packet drop rate is about 0.0002. The trace 'LBL4' ran from 14:00 to 15:00 on Friday, January 21, 1994, and 'LBL5' ran from 14:00 to 15:00 on Friday, January 28, 1994. Each captured 1.3 million TCP packets, dropping about 0.0007 and 0.0005, respectively. The LBL traces were made using *tcpdump* on a Sun SPARCstation. Their timestamps have microsecond precision.

Figure 3-15 shows the multifractal spectra for these WAN traffic traces. We can see the three 'LBL' traces are very similar to each other, especially for 'LBL4' and 'LBL5'. This can be understood since all three are TCP traffic captured under the similar conditions (e.g., similar network and relatively the same time period of a day/week). The two 'Bellcore' traces are different from "LBL", particularly on the right side. This indicates unevenly distributed packet sizes, similar to the simulated scenario in Section 3.2.2 - (5). Since these two traces included all Ethernet packets (unlike 'LBL', which only contains TCP packets), the packet size distribution will be different. So, the difference in packet sizes could be one of the causes.

In addition, the multifractal spectra from the Internet traffic traces are very close to the simulated traffic results described in Section 3.2.2. In particular, the ranges of α *are between* 0.8 and 1.3, and the centers, α_0, are between 1.0 and 1.1.

3.2.3 Summary

Multifractal spectra analyzes the functional properties of data processes. By using binomial cascade and Poisson processes as examples to compare with network traffic processes, the following results are found from this study:

(1) With respect to LRD, increasing the bandwidth does not change the traffic throughput (or network performance). The two factors affecting the traffic throughput are network delay and TCP window size.

(2) Also With respect to LRD, more traffic load with a larger number of connections makes traffic less burstier, i.e., there is smoothing by multiplexing. Reducing

packet sizes also reduces the traffic load, but the burstiness in LRD is not affected. This might suggest that in order to improve network efficiency we need to reduce the packet sizes and split traffic into multiple connections.

(3) The burstiness caused by Pareto shape α is inherent to traffic.

(4) With respect to SRD, these six factors all affect traffic burstiness to some extend. But, because three factors, bandwidth, TCP window size, and network delay, do not improve throughput in long term, we need to control them to reduce burstiness, especially by using the tradeoffs between the bandwidth and the network delay.

(5) In general, network traffic processes have values of the Hölder exponent α between 0.7 and 1.2 in LRD and between 0.7 and 1.3 in SRD. They are different from Poisson processes. The center exponents of the spectra α_0 are typically around 0.9 ~ 1.0 in SRD and around 1.0 ~ 1.1 in LRD. Three factors affecting α_0 are connection rate, TCP window size and Pareto shape factor. Since the Pareto shape is inherent in traffic, we can use other parameters to counteract it and reduce the burstiness.

CHAPTER 4

APPLICATIONS FOR EFFICIENT TRAFFIC CONTROL

Self-similarity and long-range dependence present new challenges to network engineering. With these two properties, data traffic exhibits fractal behaviours, which produce high peak-to-average bandwidth ratios. Therefore, those problems make it much more difficult to predict, quantify, and control data traffic, in contrast to traditional voice traffic in telephone networks. Especially in the recent years, many new bandwidth-consuming applications with multimedia content have become increasingly popular, which causes network traffic to be even more fractal in nature. Multimedia content requires network providers to improve network efficiency and to improve the quality of services to customers. How to make networks more efficient while still maintaining a good quality of service becomes the key issue in the network-engineering domain. This chapter will introduce a novel solution on efficient traffic control based on the results from multifractal analysis.

4.1 Network performance evaluation

4.1.1 Basic parameters of networks

First, let us review a few basic concepts of network performance evaluation. A network can be viewed as a queueing system with three parameters:

Traffic Load (X), the number of packets put into a network during a time interval. For Poisson traffic in traditional telephone networks, this can be represented by a predetermined mean value. However, for data traffic, packet arrival rates vary dramatically from time to time. The mean value of traffic is less accurate as a representation for traffic load.

Throughput (S), the number of bytes transmitted through a network during a time interval. Throughput will increase with traffic load until reaching a maximum point, referred to as the Capacity (C). Because of rapid fluctuations of data traffic, it is almost impossible to utilize the full capacity of a network. So, there is another metrics, the network efficiency, which is the ratio between throughput and capacity.

Delay (D), the amount of time that a packet takes in transmission in a network. While there is a propagation delay for signals through the physical media (wire or wireless), the majority of the time a packet spends in a network is due to buffer delay. For instance, a connection between points in the same continent will typically have a RTT in the order of hundreds of milliseconds while its propagation time is only a few tens of milliseconds. The buffer delay is also determined by network bandwidth, traffic load, and throughput. For fractal traffic, burstiness increases buffer delays, and it can be so severe that buffers overflow and packets are lost. *Loss* (L) is defined as the percentage of dropped packets.

4.1.2 Connection time factor

A network object, such as a Web page, contains multiple packets. Packet delay only tells the time for a single packet. In order to monitor the delay as a group, we need to modify the previous parameter for traffic delay.

Packets in fractal traffic are correlated at both the packet level and the connection level. The measurement of individual packet delay cannot show how fast or slow a network object such as a Web page is retrieved. To compare delays, we should use the connection time. However, durations of connections vary from each other and cannot be simply unified. For example, an average connection time may be interpreted as representing either of two things: the average duration of connections or the speed of a network.

For comparison of group traffic delays, a parameter, *connection time factor* (Γ), is used, which is a ratio between the tested connection time and the benchmark time.

$$\Gamma[i] = \frac{\text{Tested Connection Time}[i]}{\text{Benchmark Time}[i]},\ i \text{ is the connection index.}$$

Here, the benchmark measurement is referred to a test with special settings, e.g., there is no bottleneck link, so Γ generally is greater than 1.

4.2 Efficient bandwidth of fractal traffic

The goal in the control of fractal traffic is to achieve the same or a better throughput by controlling the available bandwidth for fractal traffic flows. This optimum bandwidth is referred to as the *efficient bandwidth* (EB), noted as $E_b[n]$. $E_b[n]$ is associated with a timescale n. For a different timescale, there is a different value of efficient bandwidth.

4.2.1 Formulation of efficient bandwidth

In a positive fractal process, a fractal measure is represented by

$$\mu = \delta^{\alpha}. \tag{4.1}$$

We can think μ as a fraction of the total mass 1. On the right hand side of (4.1), δ is the unit, related to a scale of the measure μ, e.g., $\delta = 2^{-n}$. We can choose any δ. α is the Hölder exponent, which describes the fractal structural information and may vary from point to point. According to MFA in Chapter 4, a fractal process can be characterized by its multifractal spectrum $f(\alpha)$. Let S_α be a subset of a fractal process S. So, we have

$$S_{\alpha} = \bigcup_{(\alpha-\varepsilon)\le\xi\le(\alpha+\varepsilon)} S_{\xi},\ \varepsilon \to 0. \tag{4.2}$$

Now, let V be the volume of traffic (in the number of bits) during a period T. So, we can derive the efficient bandwidth $E_b[n]$ for the fractal traffic flow as

$$E_b[n] = \frac{\mu[n]V}{\delta[n]T} = \frac{(\delta[n])^{\alpha}V}{\delta[n]T}. \tag{4.3}$$

If we apply $\delta[n] = 2^{-n}$, we have

$$E_b[n] = 2^{(1-\alpha)n}\, {V}\!/\!{T}. \tag{4.4}$$

We recognize V/T is the average bandwidth of the flow. Therefore, the efficient bandwidth in a scale n is $2^{(1-\alpha)n}$ times of the average bandwidth. Furthermore, α is the Hölder exponent corresponding to the partition entropy ($\alpha_S = -\mathrm{H}(\delta)/\ln\delta$) of the fractal traffic flow.

4.2.2 Properties of *EB*

In the formulation of EB (4.4), S_α is a concentration set as α corresponds to the partition entropy of a fractal set. This assure that $M(S_\alpha)$ asymptotically converges to $M(1)$ as $n \to \infty$.

Let α be in a range of $[\alpha_s-\varepsilon, \alpha_s+\varepsilon]$. In a simple binomial cascade case with $p_0 < p_1 < 1$, the measure $M(S_\alpha)$ becomes the maximum at $\alpha = \alpha_s = -\frac{p_0 \ln p_0 + p_1 \ln p_1}{\ln 2}$ and decays at a rate of $n^{-1/2}$ near α_s. As $n \to \infty$, ε is close to zero, and $M(S_\alpha)$ is close to $M(1)$. Therefore, the set S_α is the concentration of the entire fractal set.

For a fractal dataset, α must be less than 1, because $E_b[n]$ is ascendant as $n \to \infty$. We can easily observe the following on E_b:

(1) When n is very small ($n \to 0$), E_b is close to V/T.

(2) When n is large ($n \to \infty$), the timescale is close to zero, and E_b is much greater than V/T.

Therefore, to determine an efficient bandwidth $E_b[n]$, we need to choose a time scale n, and then measure fractal exponents α. The computation of α is very simple and can be the number of bits that represents the traffic at a time t.

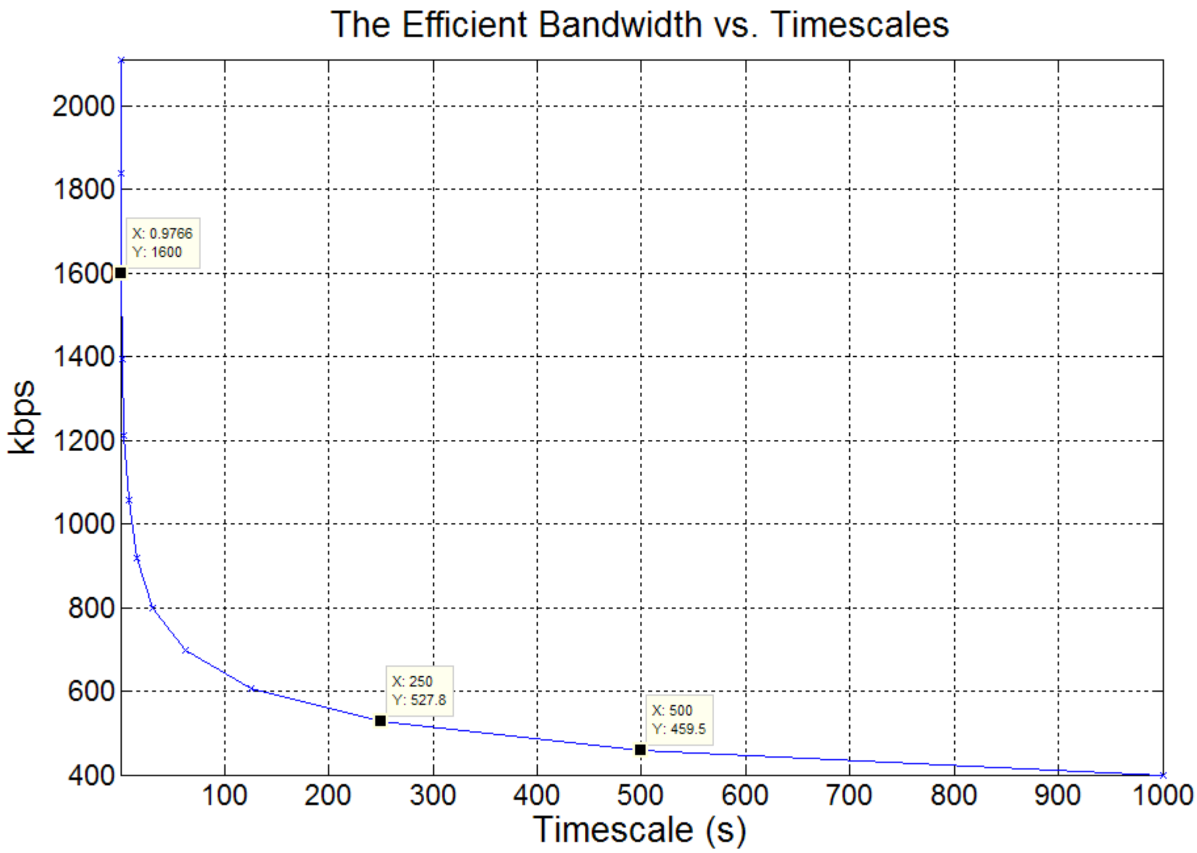


Figure 4-1 An example of efficient bandwidth vs. timescales.

Example 4.1 Suppose a traffic stream of 400Mbits with $\alpha = 0.8$ to be sent in a period of 1000 seconds. Figure 4-1 shows the needed bandwidth vs. timescales. Here, the average bandwidth for this traffic stream is 400kb/s. But, below the timescale of 1s, the required bandwidth is 1.6Mb/s. With even smaller timescales, $\boldsymbol{E_b}$ increases exponentially with a rate $2^{0.2n}$.

Note that the efficient bandwidth is not a traffic burst bandwidth. Taking into account the overall random structure, an efficient bandwidth "averages" burstiness at a given timescale in fractal space. By applying the efficient bandwidth for a certain timescale, we see that traffic spreads out uniformly within the timescale. Above that timescale, the traffic flow appears to be the same. That is, the efficient bandwidth removes burstiness of traffic in small scales while maintaining the fractal structure at higher traffic level, e.g., the connection level and the user session level. Therefore, this traffic smoothing will not degrade the quality of services to users.

4.3 TCP traffic

In this section, an experimental study of the fractal traffic control mechanism will be presented using the concept of *efficient bandwidth* on TCP traffic. Normally, Web, email, and FTP sessions are carried by TCP traffic. These are not time-critical applications. Choosing a timescale of 100ms or even a second should not affect services. But, packet bursts originate at millisecond or sub-millisecond timescales. Removing traffic burstiness will greatly increase the network efficiency.

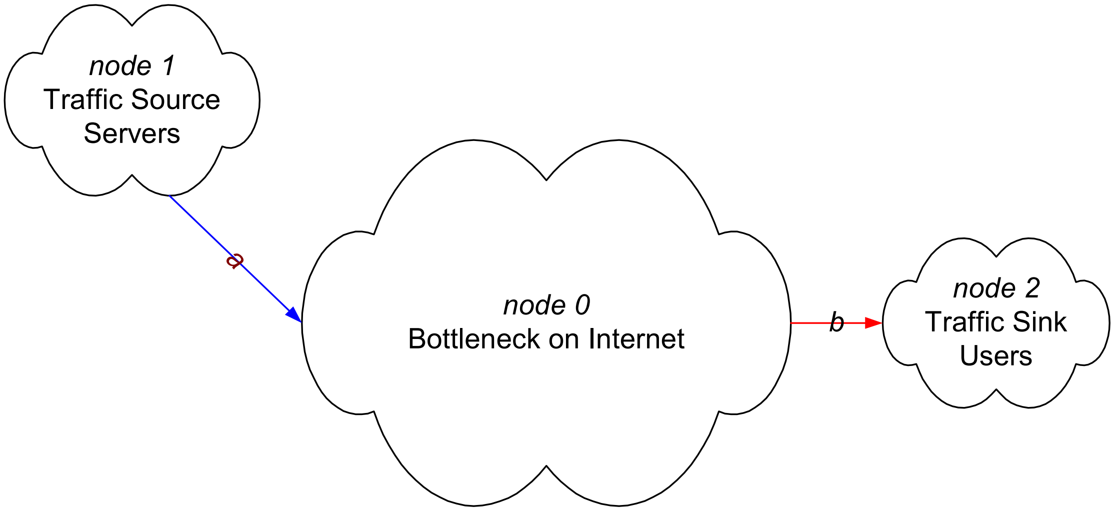


Figure 4-2 The General Network Model

The experiment set-up is similar to those in Chapter 2 and Chapter 3. A large number of TCP flows simulated in NS-2 originate from “node 1” to “node 2”. In the middle is a bottleneck “node 0”. This simplified network model emulates a typical

Internet traffic scenario and is shown in Figure 4-2. “node 1” represents an Internet server domain, which typically has a high bandwidth for downstream data transfers. “node 2” represents a user group, which receives packets from the server group. “Node 0” acts as an Internet bottleneck, at which traffic experiences limited bandwidth, buffer delay and packet loss.

4.3.1 An experiment

In this experiments, four tests were conducted with the controlled bandwidth at link (b) at 0.3, 0.5, 1, 10Mb/s. Table 4-1 lists the experiment parameters.

Table 4-1 Experiment parameters

Items	Contents
Traffic Source	- Poisson connections with Pareto distributed size - Number of connections: 3000 - Connection arrival time: 0.6s - Connection Pareto size: 12 - Connection Pareto shape: 1.5 - Traffic type: TCP
Links and Bandwidth	- Link (a): 0.3, 0.5, 1, 10 (Mb/s) - Link (b): 1Mb/s, the bottleneck link with one-way time 75ms and buffer capacity of 10 packets.

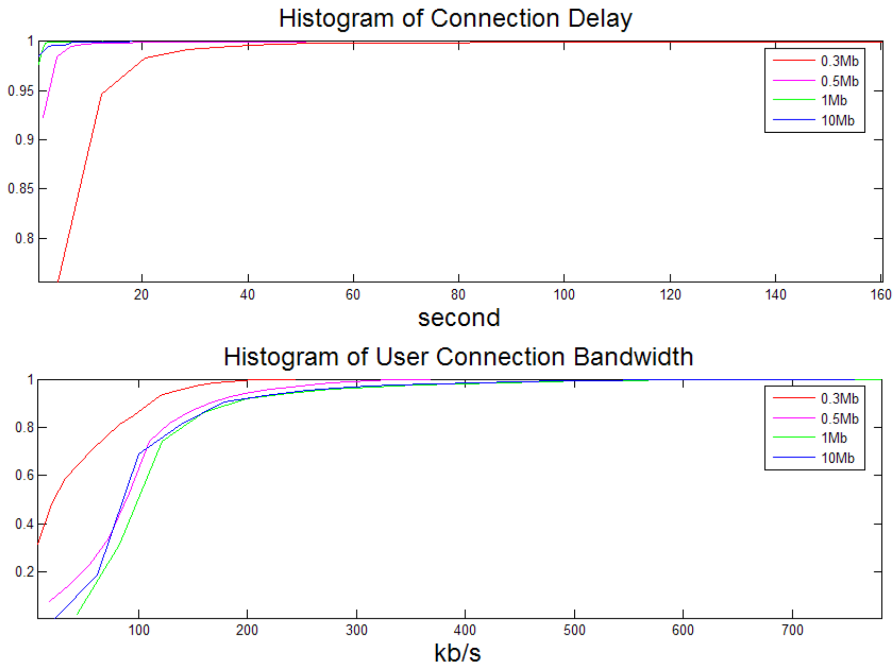


Figure 4-3 Experiment results (1)

Figure 4-3 shows histograms for delay and bandwidth. From these histograms, we can find the case of link “a” with 1Mb/s is the fastest one among the four cases. The 10Mb/s link is a little slower because it has no bandwidth control and has packet drops at the bottleneck. The case with only 0.5Mb/s performs very well and has moderate throughput, but for the 0.3Mb/s link the performance is greatly degraded.

The benchmark test has a 10Mb/s bandwidth without the bottleneck. Compared to the benchmark trace, the maximum connection delays are 164.356s (0.3Mb), 52.504s (0.5Mb), 12.986s (1Mb), and 18.536s (10Mb), respectively. In Figure 4-4, for the average of connection time factors, both the 1Mb and 10Mb cases are close to 1, the 0.5Mb is just over 2, and the 0.3Mb is greater than 9. From those numbers, it is obvious that the 0.3Mb link is congested, the bandwidth at 10Mb/s is too fast for the bottleneck, and 1Mb/s is sufficient enough but may not be efficient (note that the difference on throughputs between the source's and the user's in Figure 4-4 upper right). So, how much bandwidth actually is needed for efficiently carrying a fractal traffic flow?

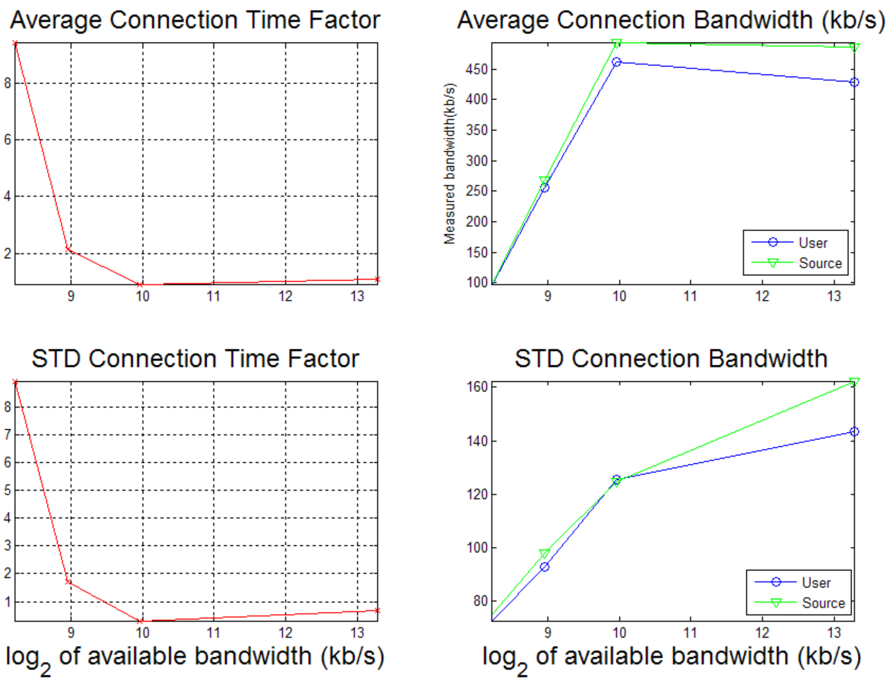


Figure 4-4 Experiment results (2)

Figure 4-5 shows the multifractal spectra. They are consistent with the above observations. The 1Mb link (yellow line) is slightly over the 10Mb link (cyan line), the 0.3Mb link is throttled at 0.3Mb/s, and the 0.5Mb link is in the middle. Hölder singularities of α_S are calculated as 0.9822 (0.3Mb), 0.9647 (0.5Mb), 0.9467 (1Mb), 0.9484 (10Mb), and 0.9229 (the "benchmark" run), respectively. So, using (4.4), Figure 4-6 plots the efficient bandwidth. For timescales less than 1s, the 0.3Mb link reaches the maximum of 0.3Mb/s so traffic is congested below this timescale. The 0.5Mb link has no problem under the 1s timescale. Its peak bandwidth is about 400kb/s at the smallest timescale of 10ms. Comparably, both the 1Mb link and the 10Mb link are close to

480kb/s at the maximum. Increasing bandwidth from 0.5Mb/s to 1Mb/s only has a small gain in throughput (400kb/s to 480kb/s). So, it is implied that the 0.5Mb link is the best among the four cases. However, in reality, choosing the right bandwidth between 0.3Mb/s to 1Mb/s is not tightly defined. The best value also depends on the network environment, i.e., network traffic loads from other traffic flows. Using efficient bandwidth give an accurate estimation on how to multiplex fractal traffic flows to share the bandwidth at a common link.

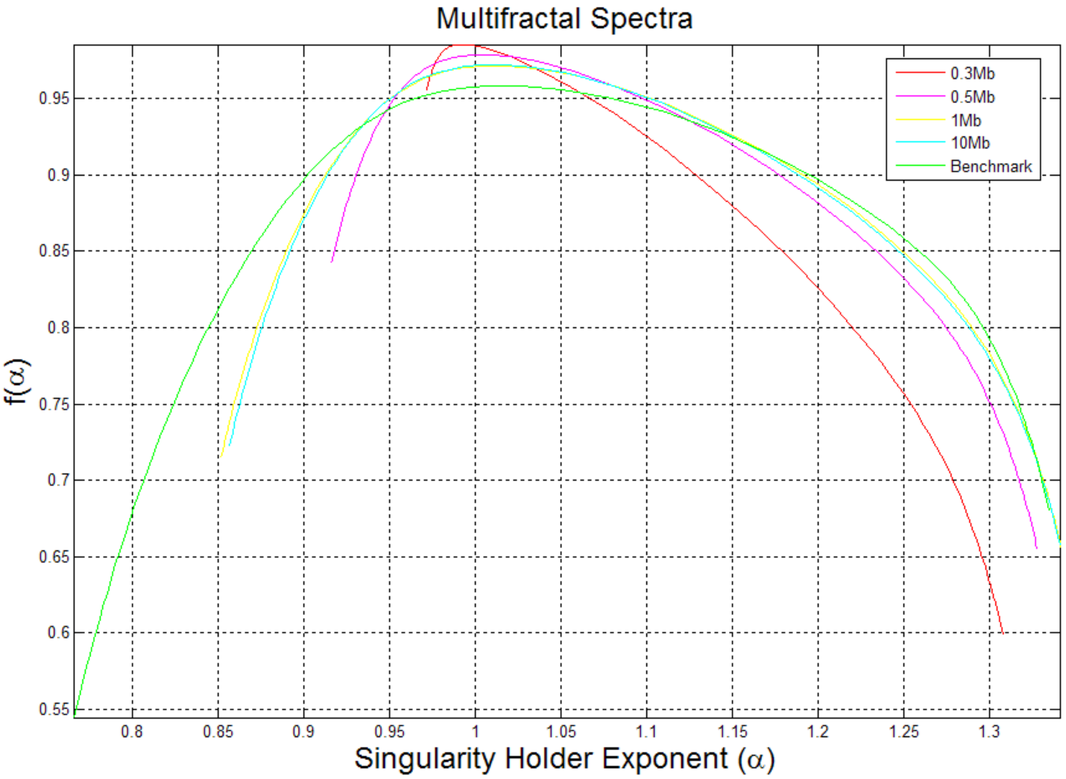


Figure 4-5 Multifractal spectra of traffic

Figure 4-7 give the detail of the connection time factor and the packet loss. The outline of the connection time factor in the 0.3Mb/s case (red) resembles a fractal image

(Von Koch curve) in a random fashion, and that in the 0.5Mb/s case also looks similar but in a smaller scale. This demonstrates the traffic fractal behaviors in another aspect: when fractal traffic is constrained (by bandwidth), network delays increase nonlinearly. This finding invalidates the formula: $bandwidth = traffic / time$. The figure shows delay will increase exponentially with decreasing bandwidth. Figure 4-8 zoom in to show the variations in the connection time factor in finer detail, and the above property remains similar.

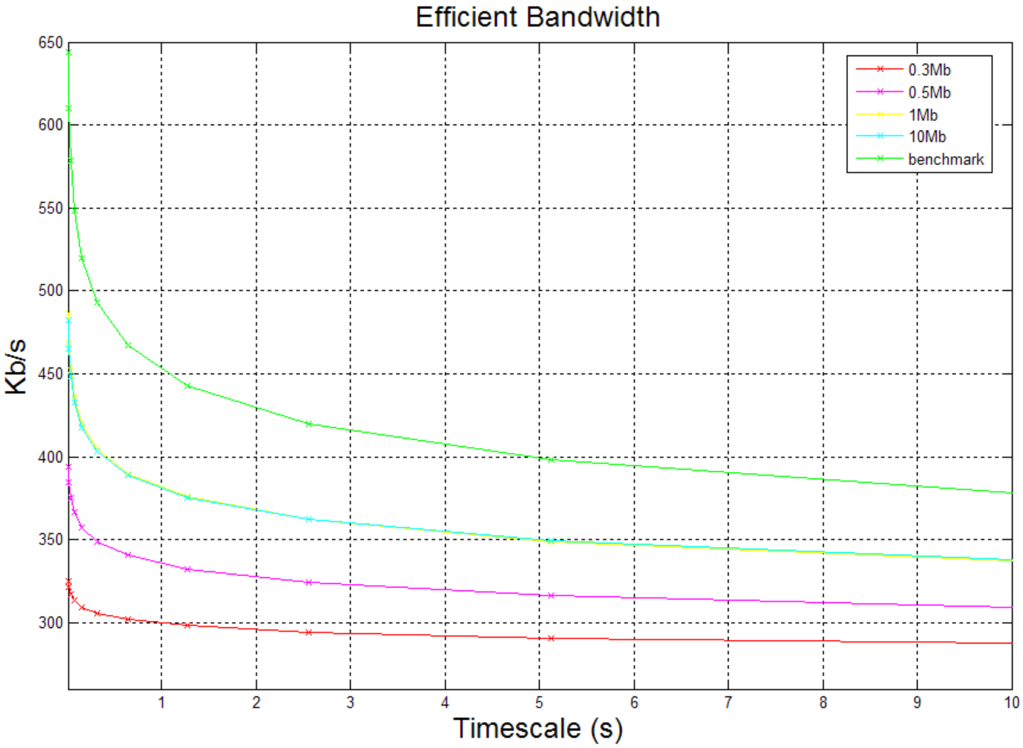


Figure 4-6 The efficient bandwidths

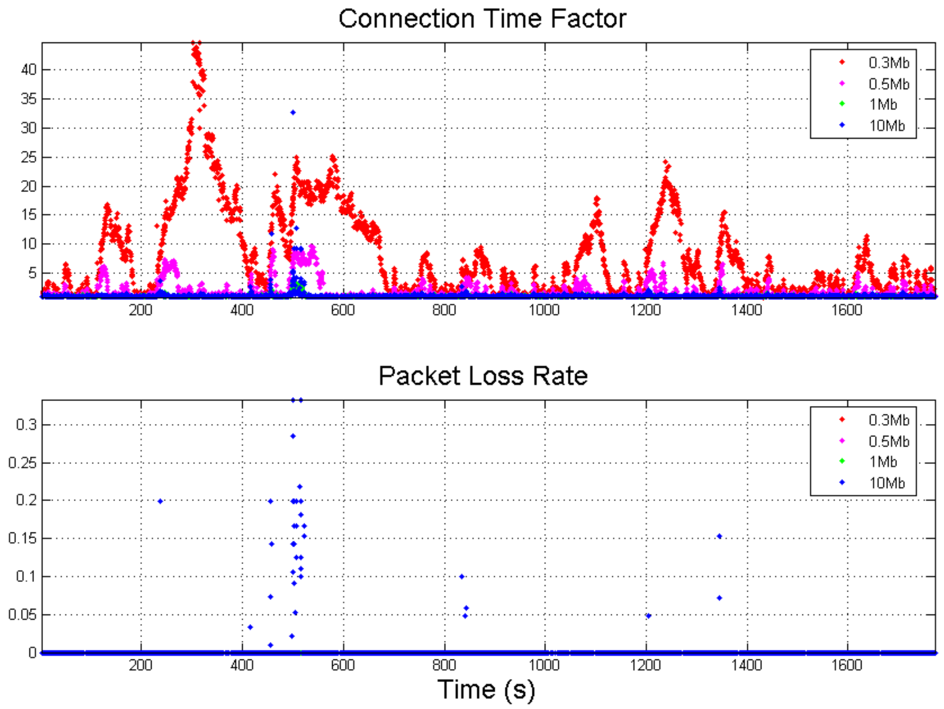


Figure 4-7 Connection time factors and connection packet loss rate vs. time.

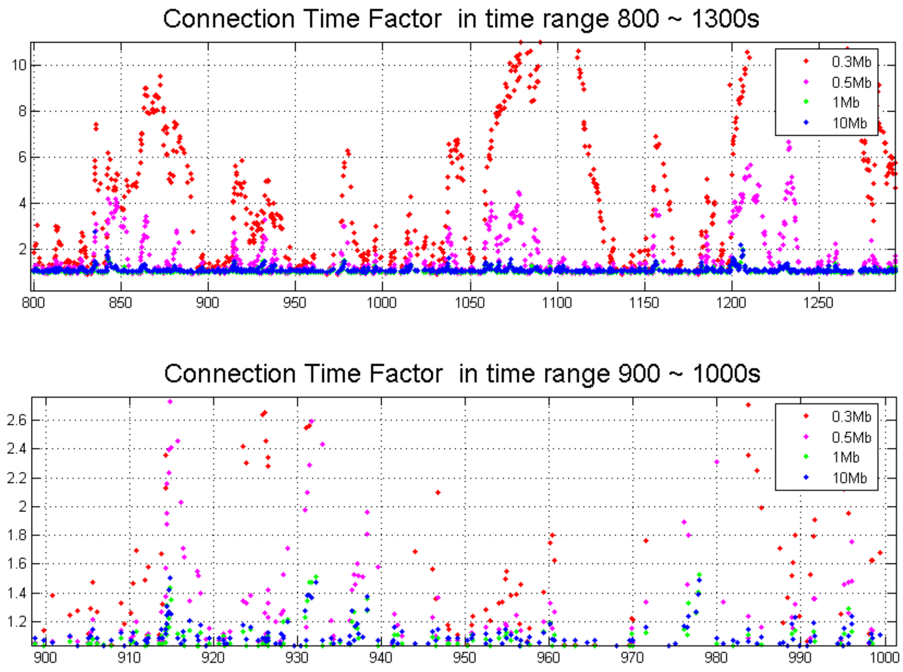


Figure 4-8 Connection Time Factor in smaller time ranges

Furthermore, in Figure 4-7 bottom plot, we will also notice that the 10Mb link (blue dots) has packet losses in some connections. Large connection time factors in the 10Mb link are caused by their packet losses. The 10Mb case is the only one suffering packet losses in this experiment. It is apparent that without bandwidth control traffic is very bursty at the packet level, and those packet losses in fractal traffic occur in bursts which are non-uniform.

4.3.2 Discussion

In network engineering, efficiency is a key objective in network provisioning. Network efficiency is determined by traffic characteristics, e.g., the peak-to-average rate ratio. Dramatic traffic burstiness affects network utilization. Without traffic control, fractal traffic is bursty in multiple timescales. Simple multiplexing does not smooth out such burstiness rather than causes traffic congestions and packet losses. Fractal traffic needs to be measured and controlled to achieve higher network efficiency.

A fractal traffic flow has an *efficient bandwidth* (EB). Above this bandwidth, there is limited improvement of performance. Large packet-level bursts may overflow the buffer at the bottleneck link and cause packet losses. Using EB can improve the network performance in the following aspects:

(1) It reduces the degree of traffic burstiness, and more fractal traffic flows can be multiplexed without interference. Thus, the network efficiency can be improved.

(2) EB is more accurate in measuring a fractal traffic flow bandwidth than the other conventional methods, such as the mean. Because of the scaling of fractal traffic, a flow with a small mean could have a large EB at a small

timescale. The scaling exponent plays a very important part in fractal traffic.

(3) It is easy to calculate EB and to measure it in real-time to provide adaptive control.

In addition, we can also find the following aspects of EB:

(1) As $\alpha \to 1$, EB increases slowly with decreasing timescale. So for a given timescale, if α is close to one, there is a higher probability that the traffic rate is less than EB.

(2) Decreasing EB, on the other hand, will increase the delay exponentially for a given α. We can no longer use the formula $bandwidth = traffic / time$ to calculate a network bandwidth to carry fractal traffic.

4.4 Reliable UDP traffic

Traffic analysis is proven even more useful for real-time applications where time is more crucial. With self-similarity analysis and multifractal analysis, we understand that a traffic process has a random structure, which is completely characterized by its singularity. The last section showed a method to derive the efficient bandwidth for TCP traffic from the singularity. Conversely, the singularity reflects a probability. If the efficient bandwidth is known, we can use the bandwidth to derive the probability that a real-time packet will be delivered so that it provides the reliability without acknowledgement in UDP traffic.

Below is a study for efficient and reliable transport of UDP game traffic on HFC cable networks[48]

4.4.1 Introduction

In recent years, the hybrid fibre coax (HFC) cable system for broadband access from household users has gained popularity dramatically. Besides typical data services, such as email, and Web browsing, many other types of applications have been developed on HFC networks. Among them, the interactive type of gaming application is the most popular one, which is attracting many customers. To operate such an application, a game server resides at a digital head-end (HE) and broadcasts messages to a group of users. Cable modem clients will respond with user messages to the server in real-time.

On HFC cable networks, there are some challenges to this interactive application. First, a data session for a single transaction is usually very short but very bursty, e.g., a large number of clients exist on the same network but the interaction is short (there could be one packet from each client). Packets from different clients will interfere each other because of contention on the upstream channel. Second, the response packet intervals are spontaneous and have comparatively large time ranges. Bandwidth reservation is not efficient for this type of network operation. Third, each packet has a real-time constraint with a hard deadline. A packet received after a deadline is useless.

In practice, HFC networks are heavily contention-based. A large number of users could coexist in a single collision domain, such as 500 – 2000 users sharing one upstream channel. For such an application, the MAC protocol is operated at the HFC *immediate mode*, which resembles a Slotted-Aloha scenario, to reduce the inefficiency of using time-slot reservations. Without reservation of time slots, packet collisions at the upstream channel could be a very serious problem. Furthermore, on the HFC MAC layer, senders cannot directly sense a packet collision unlike on Ethernet. Loss discovery requires other mechanisms. At the transport layer, though TCP is capable of recovering lost packets, it is very ineffective for the real-time requirement due to its extremely slow response time. To use UDP, we will have to implement an efficient transport scheme with reliability.

Nowadays, despite the popularity of online gaming, few papers have directly addressed the issues of interactive real-time applications on HFC networks. Related work can be found in [51], which focuses more on the signalling design to provide reliability. The transport layer performance of TCP on HFC network is studied in [52]. The related MAC layer issue is addressed in [53].

4.4.2 The network architecture and the game application

4.4.2.1 The simulation model

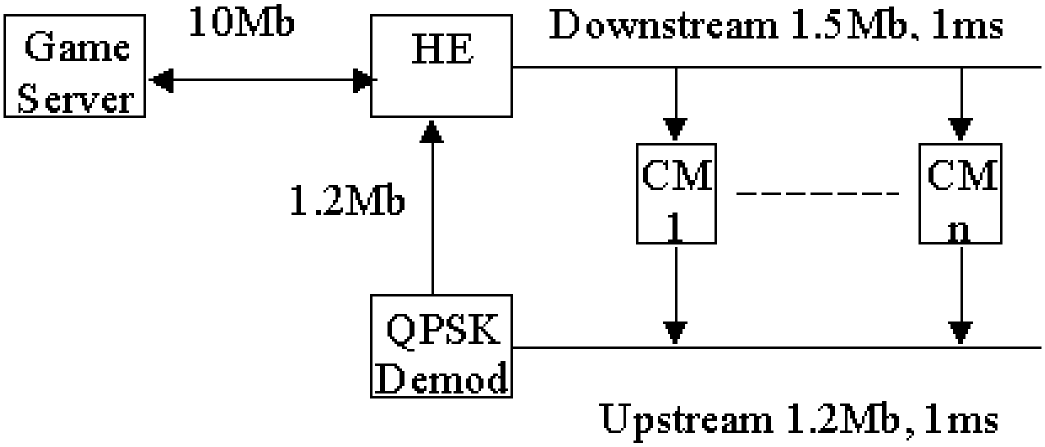
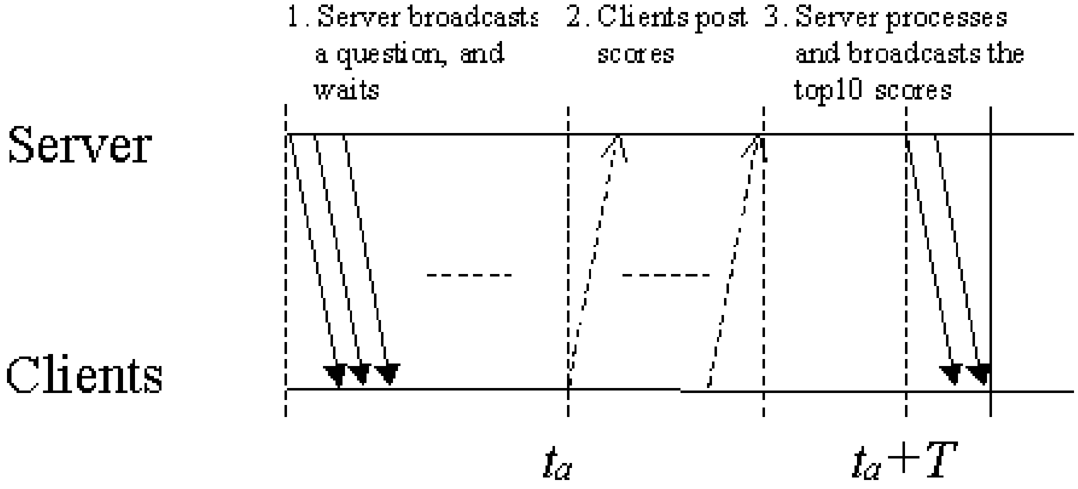


Figure 4-9. The Abstract Network Topology

The abstract network topology for the simulated HFC system is shown in Figure 4-9. (NS-2 is used for this simulation.) There are several nodes representing the game server, the head-end (HE), the QPSK demodulator for an upstream channel, and 500 cable modem users (CM $_{1\text{-}500}$), respectively. The downstream channel (DS) is a broadcast link from the head-end to cable modems. The upstream channel (US) is contentious by the packets from a large number of simultaneous game users and traffic from other applications, such as Web and email. The upstream channel is assumed in the HFC immediate mode, in which the MAC protocol uses *Slotted-Aloha*. Therefore, in the simulation, we will study the packet delay that occurs mainly due to contentions on the upstream channel.

Figure 4-9 also shows the basic system parameters. These parameters are estimated based on following situations. The QPSK channel bandwidth is 1.5Mb/s, which will be reduced by an overhead of the packet conversion between Ethernet frames and cable system frames on upstream. Therefore, the available upstream bandwidth is about 1.2Mb/s. The one-way propagation delay on a typical cable network approximates 1ms.

4.4.2.2 The game application


1. Server broadcasts a question, and waits
2. Clients post scores
3. Server processes and broadcasts the top10 scores
Server
Clients

------
t_a
t_a+T

Figure 4-10. The Game Procedure

In the simulation, the transaction of the game application is relatively simple (Figure 4-10). It exemplifies an application such as a TV game. During the game, the game server first broadcasts a question through downstream channels to client cable modems. Game players have t_a seconds to answer the question. Nearly at the end of t_a seconds, a client program will score the answer and post the user's new score to the game server. Only one packet will hold the score from each client, and the packet size is small. After t_a seconds, the server starts receiving the client packets. The server waits until another T seconds when all the packets arrive. Then, the server will sort all scores and broadcast the best 10 scores to the clients. To avoid a long latency, the server must set a limit on the deadline T for all client packets to come.

During the response period of T, packets could be lost due to a collision at the upstream channel. If a packet loss is detected, the client needs to retransmit the packet. For every retransmission, the packet latency increases. Therefore, the response period T is a critical time. In this design, for customer's satisfaction, we must keep the overall packet loss rate negligible in a given response time T. On the other hand, the period T also limits the number of retransmissions (in the worst case, multiple losses occur for the same client). Furthermore, the application must adapt to varied traffic loads. Those problems have to be considered for the implementation of the transport scheme.

4.4.3 SCRA algorithm

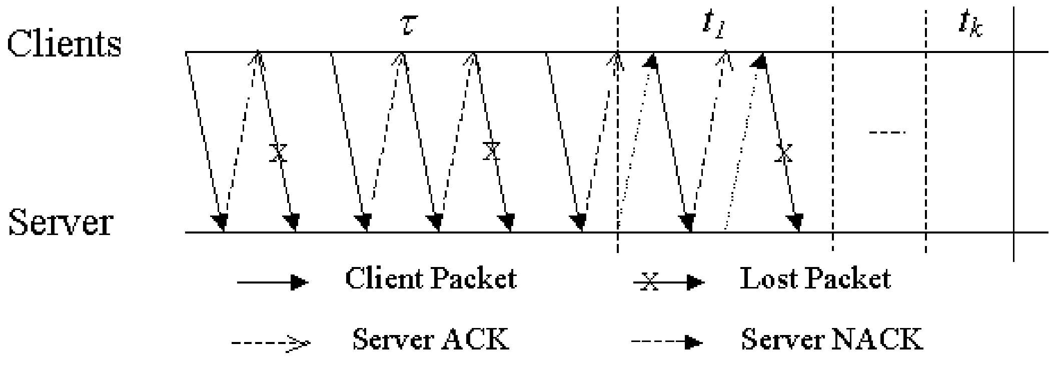


Figure 4-11. SCRA packet transactions

SCRA is a server-initiated collision resolution and avoidance scheme for real-time packet transport using UDP. Because of using UDP, SCRA must be able to recover the lost packets. Figure 4-11 illustrates SCRA packet transactions. SCRA uses a server-initiated retransmission. The server maintains a timer to track client's packets. For each received packet from a client, the server replies an ACK. At a timeout, the server will send a request (NACK) packet to those "missing" clients. On the client side, after sending a packet, it waits for an acknowledgement. If ACK is received, the client does nothing. If NACK arrives, the client will resend the packet.

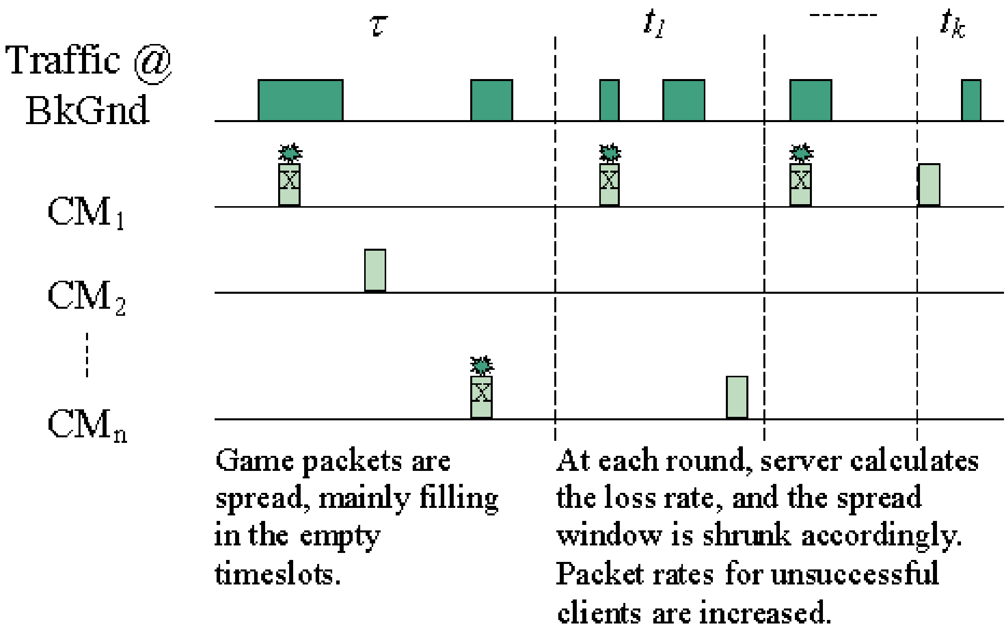

τ
t_1
------
t_k
Traffic @ BkGnd
CM_1
CM_2
CM_n
Game packets are spread, mainly filling in the empty timeslots.
At each round, server calculates the loss rate, and the spread window is shrunk accordingly. Packet rates for unsuccessful clients are increased.

Figure 4-12. SCRA Algorithm

SCRA is not only a scheme for packet loss recovery, but also an algorithm that can effectively avoid and resolve collisions for the game clients. Figure 4-12 shows the mechanism. At the beginning, we estimate the aggregated traffic load *G* from all clients. Then, spread the packets out along an initial time window of τ. For a large τ, it contains more timeslots than the number of game users so the total game traffic has approximately a uniform distribution with an average rate less than one packet per timeslot. So, contentions among game clients are greatly reduced. Because all stations on HFC are synchronized, this could be implemented by each user independently running a random delay generator. However, it is still possible that the packets from game clients will collide with the packets from other applications, as well a small chance of collisions from other game clients. Therefore, after the first transmission period τ, a fraction of the packets will successfully arrive at the server. For those missing clients, the server will send the request packets (NACK) during a time t_1. t_1 is proportional to the number of missing users, $t_1 = \tau * L$, (L is the loss rate), so the retransmission time window is shrunk at each time, such that $\tau > t_1 > \ldots > t_k \rightarrow 0$. In this way, the gaming traffic load G is unchanged. For an unlucky client, however, if the collisions occur repeatedly, the individual packet rate g from this client will increase and converge to G exponentially, $g \propto t_k = \tau \prod_i^k L(t_i)$, where $L(t_i)$ is the average loss rate within the *ith* retransmission. (If in this process, $L(t_i)$ is assumed to be constant, then, $L(t_i) \sim L$, and $t_k \sim \tau . L^k$.) Therefore, at a heavy load condition, L will be greater, and the time window shrinks less quickly. Oppositely, at a low traffic load, the window could shrink rapidly. In all, the transmission time will adapt to the background traffic, but the overall traffic load from the game clients will remain constant during the transmission period, since G is not growing with time. This assures that a cable network has less interference by the burstiness of the game traffic, and this prevents the network from reaching an unstable state.

SCRA is designed systematically based on the analysis of Slotted-Aloha MAC protocol. Under a certain traffic load, Slotted-Aloha produces a steady throughput. By spreading the client game packets, it de-correlates traffic, minimizes interference with background traffic, and achieves an optimum transmission rate.

4.4.4 Simulation results

In the simulation, the applied traffic load *G* is designed at 160kb/s from 500 game clients. This traffic is less than 15% of the upstream link capacity at 1.2Mb/s, and the upstream game packets are spread out in a 2s window at the first transmission. SCRA is also tested under background traffic loads of 300, 600, and 900kb/s, respectively.

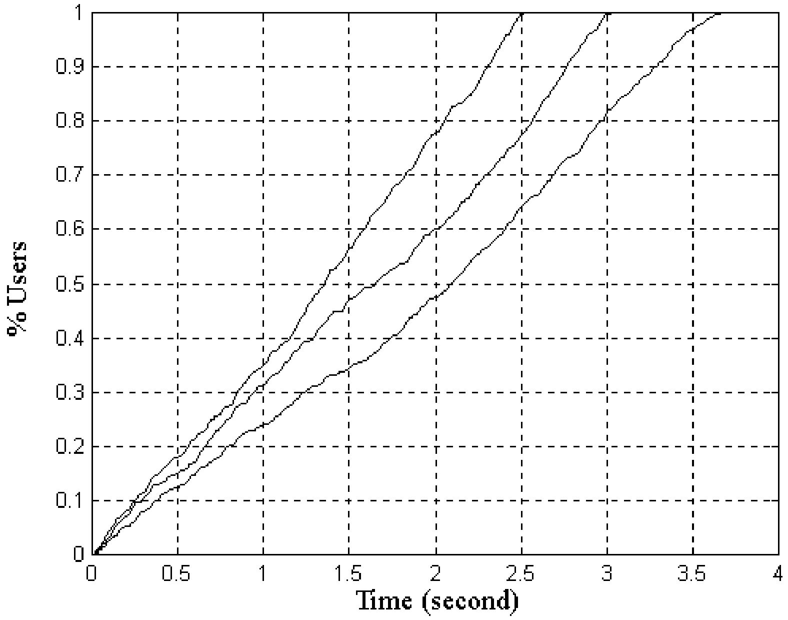


Figure 4-13. The percentage of finished users vs. time with different background traffic loads (The three plots, from left to right, corresponds 300, 600, 900 kb/s).

Figure 4-13 shows the percentage of finished users versus time. We find under traffic loads of 25% (300kb/s), 50% (600kb/s), and 75% (900kb/s), the finish times are nearly linear. That indicates that the success rate of packet transmission is steady. This is an expected result, because it indicates the interference between the game traffic and the background traffic is minimal. The three lines are distinguished by different skew rates, which are determined by the background traffic loads.

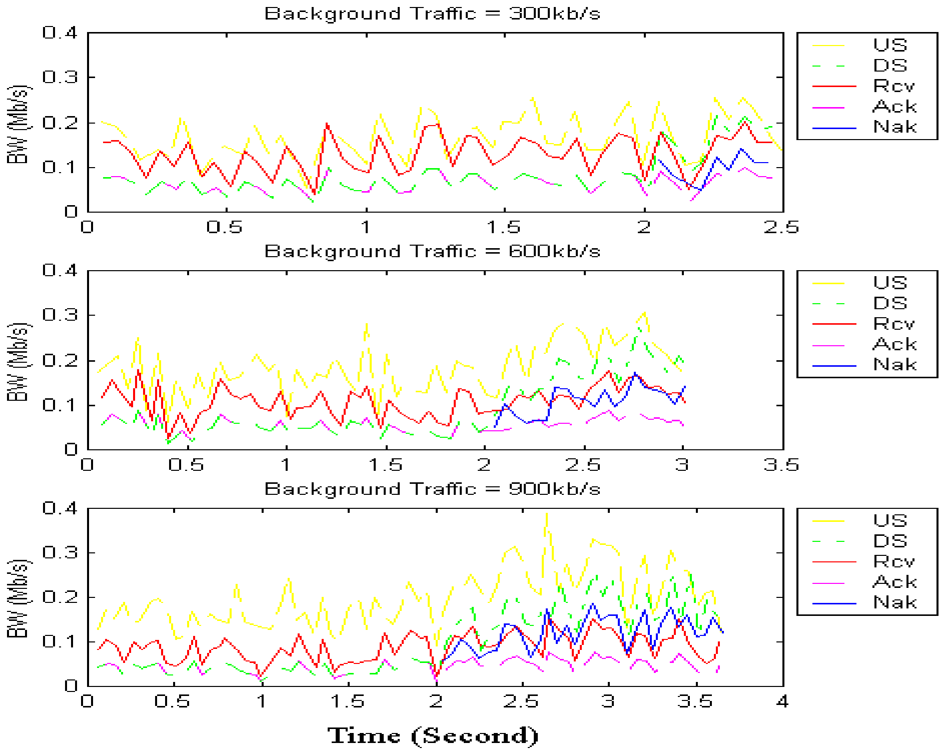


Figure 4-14. Traffic from the game application

US: total applied game traffic at upstream; DS: total game traffic at downstream; Rcv: the actually received traffic at upstream; Ack: the downstream traffic by the server sending ACK packets. Nak: the downstream traffic by the server sending NACK packets

Figure 4-14 shows the game traffic bandwidth under the three traffic loads. The labeled 'US' is the total bandwidth that the clients try to send packets on upstream, and 'Rcv' is the received bandwidth by the game server. From the plots, we see that the traffic rates are not very bursty. Especially at the loads of 25% and 50%, the upstream traffic appears even everywhere. This proves that the network is steady during the packet transactions, and SCRA does not overload the network to meet the deadline.

4.4.5 System analysis

Referred to the Section 4.4.2.2, in the implementation of this SCRA algorithm, we have the following parameters: the response time *T*, the number of clients on the network *N*, the typical traffic load *X*, and the expected packet missing rate P_{miss}. Therefore, the purpose of this system analysis is to derive a set of design formulae using those parameters.

4.4.5.1 Throughput and packet loss rate

Given a channel, a timeslot could be one of the three situations: empty, packets collided, and a successful packet transmission. Assuming the packet arrival is a Poisson distribution, the probability of a timeslot empty is $1/e^X$ [56], where *X* is the fractional traffic load.

The traffic load from the game clients $G = \sum_{n}^{N} g_n$, g_n is the traffic load from the *n*th client. To avoid collision from different game users, a random number for each client is assigned to specify in which timeslot a client may send a packet. The packets are spread randomly in a time window τ, so $g_n \sim g$, $G = gN$. *G* is designed to be only a portion of link bandwidth, and constant.

The background traffic load *X* is steady during the game traffic transaction period. So, the packet loss rate due to collision is approximately invariant, $L(t) \sim L$.

> G *is chosen to be comparably small and uniform so that any negative impact of gaming traffic to the network will be negligible. For example, when* G *is 0.1 and the fractional traffic load* X *is 0.2, the throughput for* X *is* $0.2 \times e^{-0.2} \approx 0.164$*, of which 10% (or 0.0164) will be collided with the game traffic. Therefore, the throughput of game application,* S_g*, is proportional to the applied traffic. It can be shown as*

$$S_g = Ge^{-X} . \qquad (4.5)$$

For the game clients, the packet loss rate from collisions would be

$$L = 1 - e^{-X} . \qquad (4.6)$$

If τ is small enough, we can assume the background traffic X is constant. The analysis of Slotted-Aloha indicates that there are at least a percentage of $1/e^X$ timeslots empty. In SCRA, the gaming traffic is independent from the background traffic, and is distributed randomly with uniform probability. So, we expect that the throughput for gaming traffic is proportional to the available timeslots under traffic load X. Therefore, this argument justifies (4.5) and (4.6).

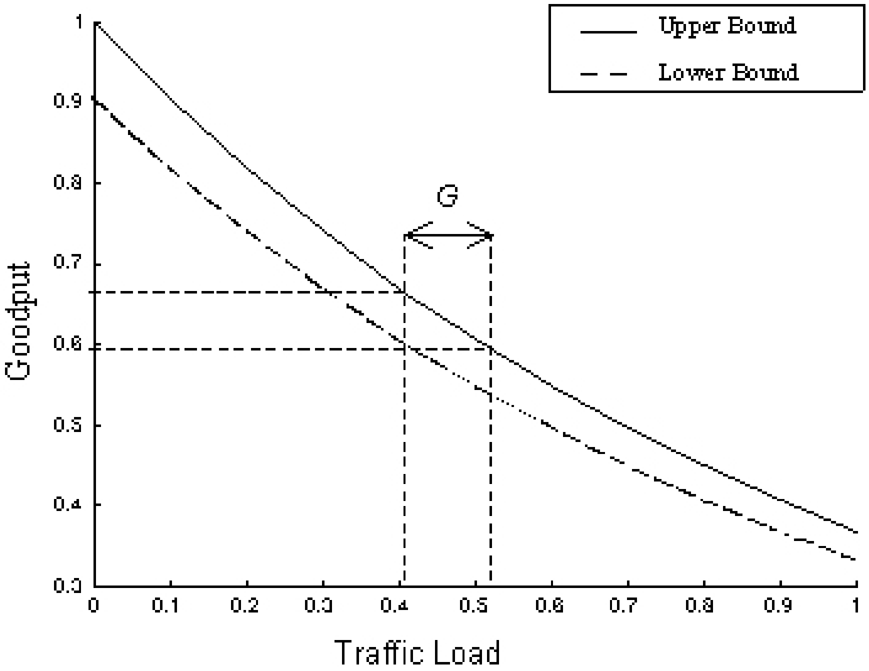


Figure 4-15. Goodput vs. traffic load

In SCRA, G is a design parameter. The larger the value of G, the higher throughput for the game traffic will be, but its impact may no longer be negligible. When a larger G is used, there is more interference with background traffic. So, it requires a balance to determine a proper *G* so that it can minimize the contention with background traffic and also meet the required deadline. Figure 4-15 plots the goodput vs. traffic load with *G* = 0.1. The lower bound of goodput corresponds to $e^{-(X+G)}$, and the upper bound of goodput is e^{-X}, the case without the impact of *G*. The plots will be useful to decide an operational range for the game traffic under a certain background traffic condition. For example, if the typical background traffic load is around 0.4, and we need a goodput of no less than 0.6 to meet the time requirement, the proper *G* is 0.1. We should only choose *G* in the small range where the packets could meet their deadline *T*.

4.4.5.2 Packet latency and probability of missing a deadline

The latency and the missing-deadline probability are QoS parameters. They are related to the game traffic throughput. Whether a sender can meet the deadline depends on the number of retransmissions due to the collisions. The packet latency can be found

$$D(K) \geq \sum_{i=0}^{K} \tau L^i = \tau \frac{1-L^{K+1}}{1-L}. \tag{4.7}$$

K is the number of transmissions. *L* is the game packet loss rate. τ represents the initial time window size and can be calculated by $\tau = \eta NB/GC$. *N* is the number of clients; *B* is the packet size; *C* is the link capacity; and $\eta > 1$ is a constant.

Given a time deadline *T*, we can solve *K* by inverting (4.7):

$$K = \left\lfloor \log\left\{1 - \frac{(1-L)T}{\tau}\right\} \Big/ \log L - 1 \right\rfloor. \tag{4.8}$$

Therefore, *K* is the maximum number of transmissions before missing the deadline.

The *probability of missing the deadline* can be expressed as:

$$P_{miss}(k) = L(1-L)^{k-1}. \tag{4.9}$$

So, using (4.8) and (4.9), we can estimate the probability P_{miss}, and derive τ and G. If the application requires no more than P_{miss} fraction of packets missing their deadlines, we can find k from (4.9). Compared with the K from (4.8), if K is larger, we have to reduce τ and increase G. Note that, to make the system stable and to reduce the interference from background traffic, we should keep G as small as possible.

4.4.6 Conclusions

The game application is an example of how to reduce traffic burstiness at the user session level. Using UDP, a transport scheme is proposed to solve a contention problem and to meet a real-time requirement.

To design and implement such a scheme, we first determine an efficient bandwidth using the QoS requirements: deadline and packet missing probability. Then, based on the number of customers, packets are spread out to avoid collisions. As results, the SCRA scheme solves the traffic congestion with very limited knowledge between the server and clients – a transmission window size, unlike TCP that has more complicated mechanisms.

SCRA is efficient to meet a hard deadline requirement. It does not generate traffic burstiness and affect the background traffic to improve its throughput. In fact, the traffic rate remains almost constant during the transmission period. So, the packet level burstiness is eliminated. Finally, SCRA is scalable to the different sizes of network and the different conditions of background traffic.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Today, Internet is growing rapidly. Traffic measurement and management are important in every aspect of network engineering. This thesis took on a new unsolved problem in the area: how to analyze the fractal characteristics of network traffic. Two analytical methods were used to study the problem: self-similarity analysis and multifractal analysis. Using a number of experiments, the following results and observations were presented:

1. Self-similarity in traffic is an adaptability of traffic in the network. Many factors are involved in creating this characteristic. A new view of this self-similar traffic structure is interpreted. This view is an improvement over the theory used in most current literature, which assumes that the traffic self-similarity is solely based on the heavy-tailed file-size distribution.

2. The scaling region on traffic self-similarity is divided into two timescale regimes: SRD and LRD. Experimental results show that the network transmission delay

(RTT time) separates the two scaling regions. This gives a physical source of the periodicity in the observed traffic. Also, bandwidth, TCP window size, and packet size have impacts on SRD. The statistical heavy-tailedness (Pareto shape parameter) affects the structure of LRD. In addition, a formula to calculate traffic burstiness is derived from self-similarity.

3. Furthermore, examinations of fractal traffic using multifractal analysis give more interesting and applicable results. (1) In LRD, increasing the bandwidth does not improve throughput (or network performance). The two factors affecting the traffic throughput are network delay and TCP window size. On the other hand, more simultaneous connections smooth traffic, which could result in the improvement of network efficiency. (2) At small time scales, traffic burstiness varies. In order to improve network efficiency, we need to control bandwidth, TCP window size, and network delay to reduce traffic burstiness. There are tradeoffs from each other but nonlinearly related. (3) In general, network traffic processes have Hölder exponents α ranging between 0.7 and 1.3. Their statistics differ from Poisson processes.

4. To apply this prior knowledge from traffic analysis and to improve network efficiency, a notion of the *efficient bandwidth*, EB, is derived to represents the fractal concentration set. Above the bandwidth EB, traffic appears bursty in a way that cannot be reduced by multiplexing. But, below EB, traffic is congested. The important finding is that the relationship between the bandwidth and the transfer delay is nonlinear.

5. Traffic analysis can be applied to various applications, including best-effort TCP traffic and real-time UDP traffic, so that we can remove packet-level burstiness and improve network efficiency.

5.2 Future work

For the future work, fractal traffic analysis and results will be useful to enhance the performance of real-time traffic, especially in multimedia video applications. There are the following reasons why this research is applicable:

a. Video traffic is known as being self-similar and fractal;

b. Video traffic is extremely time-sensitive;

c. Video traffic is bandwidth-consuming.

To balance bandwidth, delay and network efficiency, it is necessary to study the fractal properties of video traffic and to find the efficient bandwidth.

References

[1] W. E. Leland, M. S. Taqqu, W. Willinger, D. V. Wilson, “On the Self-Similar Nature of Ethernet Traffic,” *IEEE/ACM Trans. Networking* Vol. 2, pp.1-15. 1994.

[2] V.Paxson and S.Floyd, “Wide-area traffic: The failure of Poisson modeling,” *IEEE/ACM Trans. Networking*, vol.3, pp.226-244, 1995

[3] Patrice Abry, Richard Baraniuk, Patrick Flandrin, Rudolf Riedi, Darryl Veitch, “The Multiscale Nature of Network Traffic: Discovery, Analysis, and Modelling,” *IEEE Signal Processing Magazine,* May 2002

[4] M.E.Crovella and A.Besttavros, “Self-similarity in world wide Web traffic – evidence and possible causes,” Proc. *ACM Sigmetrics’96*, pp.160-169, 1996.

[5] K.Park and W.Willinger, Eds., *Self-Similar Network Traffic and Performance Evaluation*. Wiley Interscience, 2000.

[6] K.Park, G.Kim, and M.Crovella, “On the relationship between file sizes, transport protocols, and self-similar network traffic,” *Proc. IEEE International Conference on Network Protocols*, pp.171-180, 1996.

[7] A.Feldmann, A.C.Gilbert, and W.Willinger, “Data networks as cascades: Investigating the multifractal nature of Internet WAN Traffic,” Proc. *ACM/Sigcomm’98*, pp.25-38, 1998.

[8] A.Feldmann, A.C.Gilbert, P.Huang, and W.Willinger, “Dynamics of IP Traffic: A study of the role of variability and the impact of control,” Proc. *ACM/Sigcomm’99*, pp.301-313, 1999.

[9] A. Feldmann, A.C.Gilbert, W.Willinger and T.G.Kurtz, “The changing nature of network traffic: Scaling phenomena.” *Computer Communication Review* 28, No, April 1998

[10] Z.-L. Zhang, V.Ribeiro, S.Moon, and C.Diot, “Small-time scaling behaviors of internet backbone traffic: an empirical study,” in *Proc. IEEE Infocom*, San Francisco, CA, Apr. 2003.

[11] D.R. Cox, “Long-range dependenc: a review.” *Statistics:An Appraisal*. H.A. David and H.T.David, Eds. Iowa State University Press, 1984.

[12] A.M.Makowski, and M.Parulekar, "Buffer Asymptotics for M/G/∞ input processes," *Self-Similar Network Traffic and Performance Evaluation*. K.Park and W.Willinger, Eds. Wiley Interscience, 2000.

[13] C.Xia, and Z.Liu, "Queuing Systems with long-range dependent input process and subexponential service times," *ACM SIGMETRICS 2003*. Preprint

[14] K.Nagarajan, "Fractional ARIMA Processes and its applications in network traffic modeling," A qualifying examination report, Georgia Tech, Aug. 1998.

[15] A. Adas, "Traffic Models in Broadband Networks," *IEEE Communications Magazine*, Jul. 1997. p.82-89.

[16] S. Ma, and C.Ji, "Modeling Video Traffic Using Wavelets," *IEEE Communications Letters*, Vol.2, No.4, Apr.1998.

[17] S. Ma, and C.Ji, "Modeling Heterogeneous Network Traffic in Wavelet Domain," *IEEE/ACM Trans. on Networking*, vol.9, No.5, Oct.2001, p.634-49.

[18] X. Tian, J. Wu and C. Ji, "A Unified View of Heterogeneous Network Traffic: Impact of Network Load", *Proceeding of INFOCOM*, New York, June 2002.

[19] R.Riedi, M.S.Crouse, V.J.Ribeiro, and R.G. Baraniuk, "A multifractal wavelet model with application to network traffic," *IEEE Trans. Inform. Theory*, Vol. 45, no.3, pp.992-1018, Apr.1999.

[20] V. J. Ribeiro, R. H. Riedi and R. G. Baraniuk, "Wavelets and Multifractals for network traffic modeling and inference," *Proceedings ICASSP Salt Lake City, Utah, May 2001.*

[21] S. Sarvotham, R. Riedi, and R. Baraniuk, "Connection-level Analysis and Modeling of Network Traffic,"*Proceedings IEEE/ACM SIGCOMM Internet Measurement Workshop 2001*, San Francisco, CA.

[22] N. Hohn, D. Veitch, and P. Abry, "Cluster Processes, a Natural language for network traffic," *IEEE Trans. on Signal Processing*, special issue on "Signal Processing in Networking", Vol.51, No.8, Aug. 2003. p.2229-2244.

[23] J.Potemans, et.al. "A tunable discrete traffic generator based on a hierarchical scheme of Bernoulli sources," *Proc. Of ICC'03*, Anchorage, Alaska, USA, May 2003.

[24] I.Norros, "A storage model with self-similar input." *Queueing Systems*, 16:387-396, 1994.

[25] I.Norros, "ON the Use of Fractional Brownian Motion in the Theory of connectionless traffic," *IEEE JSAC*, Aug. 1995, pp.953-962

[26] A.Erramilli, O.Narayan, A.L.Neidhardt, and I. Saniee, “Performance Impacts of Multi-scaling in Wide-Area TCP/IP Traffic,” in *Proceedings of IEEE Infocom*, Apr. 2000.

[27] V. J. Ribeiro, R. H. Riedi, M. S. Crouse and R. G. Baraniuk, “Multiscale Queuing Analysis of Long-Range-Dependent Network Traffic,” *IEEE Trans. on Networking*, February 2001.

[28] S. Jaffard, “Some open problems about multifractal functions”, Fractals in Engineering, p2-18, Springer, 1997.

[29] S. Mallat, A Wavelet Tour of Signal Processing, Academic Press, 1999.

[30] R. H. Riedi, “An introduction to multifractals,” Rice University, Version Sept 5, 1999.

[31] J. Feder, Fractals, Plenum Press, New York, 1988.

[32] B.B. Mandelbrot, *The Fractal Geometry of Nature*. W.H.Freeman and Company, New York, 1982.

[33] Lauwerier, H. *Fractals: Endlessly Repeated Geometric Figures.* Princeton, NJ: Princeton University Press, 1991.

[34] Laurent Calvert, Adlai Fisher and Benoit B. Mandelbrot, "The Multifractal Model of Asset Returns." Discussion papers of the Cowles Foundation for Economics, Nos. 114-1166. Cowles Foundation, Yale University, 1997.

[35] M.Barnsley, *Fractals Everywhere*. Academic Press Professional, 1993.

[36] G.W.Wornell, *Signal Processing with Fractals: A Wavelet-Based Approach*. Prentice Hall, 1996.

[37] K.Grochenig, *Foundations of Time-Frequency Analysis*. Birkhauser, Boston, 2001.

[38] L.M.Kaplan, and C.J.Kuo, “Fractal estimation from noisy data via discrete fractional Gaussian noise(DFGN) and the Haar basis,” *IEEE Trans. on Info. Theory*, 41(12), 1993.

[39] L.M.Kaplan and C.J.Kuo, “Extending self-similarity for fractional Brownian motion,” *IEEE Trans. Signal Processing*, vol.42, pp3526-30, Dec.1994.

[40] G.W.Wornell, “A Karhunen-Loeve like expansion for 1/f processes via wavelets,” *IEEE Trans. on Info. Theory*, vol.36, pp.859-61, Mar. 1990.

[41] P.Flandrin, “Wavelet analysis and synthesis of fractional Brownian motion,” *IEEE Trans. on Info. Theory*, vol.38, pp.910-16, Mar.1992.

[42] P.Abry and D.Veitch, “Wavelet analysis of long-range dependence traffic,” *IEEE Trans. Inform. Theory*, vol. 44, pp.2-15, 1998.

[43] D. Veitch, P. Abry, “A wavelet based joint estimator for the parameters of LRD,” "Special issue on Multiscale Statistical Signal Analysis and its Applications" *IEEE Trans. Info. Th.* April 1999, Volume 45, No.3, 1999.

[44] Roughan, Veitch, Abry, “On-line estimation of LRD parameters,” *Proceeding Globecom '98* Sydney Vol.6 pp.3716-3721 Nov 1998.

[45] Roughan, Veitch, Abry, “Real-Time Estimation of the Parameters of Long-Range Dependence (Extended Version),” *IEEE/ACM Transactions on Networking* August, 2000, vol.8, no.4, 467-478.

[46] WaveLab 802, Matlab source toolkit, http://www-stat.stanford.edu/~wavelab. 2006.

[47] X. Zhuang, J. Liu, “WRAPS scheduling and its efficient implementation on network processors,” *Proc. Of HIPC 2001*, Bangalore, India, Dec.18-21, 2001.

[48] J. Liu, B.Raheem, J. Copeland, “Implementation of an Efficient Transport for Real-Time Game Applications on HFC Cable Networks**”,** *Proc. Of ICC’03*, Anchorage, Alaska, USA, May 2003.

[49] J. Liu, J. Copeland, “Comparison of Web and kazaa traffic on self-similarity”, IASTED CSA04, Banff, AB, Canada, 2004.

[50] D. Saha, S. Mukherjee, and S. K. Tripathi, “Multirate Scheduling of VBR Video Traffic in ATM Networks”, *IEEE J. of Selected Areas in Communications*, Vol.15, No. 6, Aug. 1997.

[51] G. Bai; Z. Shen; W. Wang; S. Cheng, “RSTP: A New Lightweight Transport Protocol for VoIP,” International Conference on Communication Technology Proceedings, 2000. Vol. 1.

[52] R. Cohen, S. Ramanathan, “TCP for High Performance in Hybrid Fiber Coaxial Broad-Band Access Networks,” IEEE/ACM Transactions on Networking, Vol.6 Feb. 1998.

[53] J.Limb, D. Sala, “A Protocol for Efficient Transfer of Data Over Hybrid Fiber/Coax Systems,” IEEE/ACM Transactions on Networking, Vol.5 Dec. 1997.

[54] Cidon, I.; Rom, R.; Gupta, A.; Schuba, C. “Hybrid TCP-UDP Transport for Web Traffic” IEEE International Performance, Computing and Communications Conference, 1999.

[55] D. Velten, R.Hinden, J.Sax, IETF RFC-908 Reliable Data Protocol, Jul. 1984.

[56] A. S. Tanenbaum, Computer Networks, 3rd edition, Prentice Hall, 1996.

[57] Fraclab 1.0, INRIA, C. Canus, P. Goncalves, b. Guiheneuf, and J. L. Vehel, http://www-syntim.inria.fr/fractals. 2003.

[58] NS-2 network simulator, UC Berkeley, http://www.isi.edu/nsnam/ns/. 2006.

[59] Internet Traffic Achieve, http://ita.ee.lbl.gov/html/traces.html. 2006.

Vita

Jian Liu was born in 1969 in China. He received B.S. degree in Electrical Engineering in 1991. From July 1991 to July 1995, he was a research engineer at Nanjing Research Institute of Electronic Technology, Nanjing, China. From July 1995 to May 1997, he was a product engineer at Ericsson Communication Ltd. Co. Nanjing, China. He completed MS and Ph.D. degree in Electrical and Computer Engineering at Georgia Institute of Technology in May 2000 and August 2006, respectively. His research interests are in broadband communication systems and networking, particularly using fractal theory on statistical network data analysis and modeling, and multimedia networks.