uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Sareh Taebi Harandi

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Self-Similar Traffic Modeling and Clos-like Packet Switch Architectures

TITRE DE LA THÈSE / TITLE OF THESIS

T. Hall

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Hanan Anis

Changcheng Huang

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

Self-Similar Traffic Modeling and Clos-like Packet Switch Architectures

by

Sareh Taebi Harandi

A thesis submitted to the Faculty of Graduate and Postdoctoral studies in partial fulfillment of the requirements for the degree of

Master of Applied Sciences

in

Electrical Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

School of Information Technology and Engineering

University of Ottawa

Your file Votre référence
ISBN: 0-494-11424-X
Our file Notre référence
ISBN: 0-494-11424-X

Abstract

Two methods of generating self-similar internet traffic are reported in this document. The first method uses Heavy-tailed Pareto distribution and generates fixed length packets, while the second method focuses on Fractional Gaussian Noise as means of generating variable-length packets. Various testing have been performed on the traffic to corroborate its self-similarity and measure relevant statistics.

Using the self-similar traffic environment, the performance of a sectored three-stage packet switch architecture is evaluated which benefits from both photonic and electronic technologies and provides variable bandwidth between sector pairs by changing the configuration of the photonic centre stage according to arriving traffic. This architecture reduces complexity, at the price of a modest spatial speed-up, which is easy to provide with photonic technology.

As an alternative to reconfiguration, a novel load-balancing Clos-like architecture and method without miss-sequencing is proposed. The architecture involves no central schedulers and is implemented by distributing cross-point queues over all elements of the switch and deploying pollers to select the queues to be served. This method is simple and practical and shows delay performance that approaches that of an ideal output queued switch.

To **my mother**, Farzaneh Shafiei and **my father** Dr. Amir Taebi Harandi
with deepest love and gratitude

Acknowledgements

My most sincere appreciation goes to my supervisor Professor Trevor J. Hall for all the knowledge and experience I gained from him. His valuable support, patience and kindness were my motivations for research progress.

Also, many thanks given to my friend and colleague, Dr. Sofia A. Paredes, for her guidance and positive feedbacks along the way. It was enjoyable working collaboratively with her and learning from her.

And more thanks given to the wonderful Michelle Thorman, for coordinating and organizing all activities the best possible way and being so kind and helpful all the way through.

I would also like to express my heart-felt love and gratefulness to my family; My dear mother who was always there for me at worst times, my great father who has been my supportive guide in life, Saleh and Mahdi whom their humor and kindness has always been cheering and dear Mehrdad for his endless support and encouragement to finish this project.

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) and industrial and government partners, through the Agile All-Photonic Networks (AAPN) Research Network. I would like to express my thanks to the people involved in these two initiatives.

Table of Contents

Figures

Symbols

x	random variable
μ	mean
σ	standard deviation
H	Hurst parameter
α	tail index of a Pareto distributed random variable
β	minimum value of a Pareto distributed random variable
ρ	offered traffic load
t	time slot
N	number of incoming links (ports) to the switch
m	number of crossbar switches in the central stage of FBP switch
M	number of central layers in LBS switch
n	number of ports per sector / number of samples in a trace
l	number of sectors in FBP switch
τ	inter-configuration period
Λ_e	estimated inter-sector traffic matrix
k	spatial speed-up
S	temporal speed-up
r	internal link rate
R	external link rate
Q	Queue in a switch

1. Introduction

Every century has been dominated by a particular technology and the key technology of the 20th century is information gathering, processing and distribution. The merging of computers and communication systems has had a profound influence on the information technology. Computer organizations now consist of the interconnection of separate computers to exchange information, these systems being called *computer networks*. Computer networks enable resource sharing and provide high reliability by having alternate sources of information supply. Overall, they make data communications easier and quicker in every sense. Aside from computer machines themselves, computer networks consist of two main components, transmission lines and switching elements [TANE96]. Transmission lines move data in bits between machines. The connection between computers in a network can range from a copper wire to fiber optics, microwaves and wireless satellites. Switching elements connect two or more transmission lines and choose the outgoing line to forward data to. The switching computers are usually called *routers*. Routers are responsible for routing data from a source node (computer, network) to a destination node by passing different nodes and eventually reaching the destination. At the heart of the routers are switches responsible for data forwarding between incoming and outgoing links.

Switching of data can be done in two forms, circuit and packet switching. *Circuit switching* builds on the idea of today's telephony systems. A dedicated connection is set up between two end users to transport messages between each other. In *packet*

switching, data is split into discrete units of data called *packets*. There are no channels dedicated to a message and bandwidth is acquired and released as needed, so packets are forwarded to destinations separately. The long waiting times of long messages imposed on shorter messages is avoided in packet switching. Packet switches also deploy the idea of buffering where packets contending over the same output at the same time are to be queued according to a discipline for that output link to become available. The queueing disciplines along with various scheduling policies allow for avoidance of *contention* inside a switch.

The Internet has become very large in the past decade and the amount of traffic it carries has also subsequently increased. The IP protocol uses packet switching technology, which is advantageous in two respects. Packet switching is very efficient in terms of bandwidth and it also allows variable bit rate connections.

The disadvantage of packet switching is the out of order packet delivery at some times requiring re-sequencing of packets. While mis-sequencing is allowed and common in Internet, it is of benefit if routers do not miss-sequence packets belonging to the same application flow. A *flow* is a train of consecutive packets with the same source and destinations. There are two approaches to avoid mis-sequencing in a switch/router; to prevent packet from becoming mis-sequenced anywhere in the router or to bound the amount of mis-sequencing. However, these solutions can require high computational complexity and impractically in some cases.

The widespread deployment of optical fiber links has now motivated research into the role that optics can play at the routing nodes themselves. Currently, basically all switches are implemented electronically. This is due to the lack of efficient optical buffering technology and the extensive availability of electronic memory. On the other hand, deploying optics in the switches can offer data-rate transparency and fast format-independent transmission. Deploying optics in switching elements is under extensive research at the moment.

Integrated broadband networks are expected to support various traffic types such as data, voice, image and video. Traditional modeling tools and techniques, both theoretical and practical have been able to characterize and understand the behaviour of broadband traffic to a rather limited extent. The problem of teletraffic modeling first

arose in the context of telephone traffic congestion and was considered by Erlang. The Poisson process was used as a model for telephone call arrivals, where enough evidence was provided that the call arrivals follow a Poisson model. As the gap between telecommunications and data communications was reducing in the late 80's and early 90's, the concept of integrated broadband networks carrying multimedia traffic emerged. All data would now be carried in fixed length packets called *cells*. The motivation then aimed toward modeling of teletraffic with the objective of performance evaluation [KRIS03].

In the telephony system, statistical behaviour of call arrivals was enough to determine the required resources to provide Quality of Service (QoS) where required bandwidth was fixed. In contrast, multimedia traffic has been characterized by a high variability and burstiness in the amount of required bandwidth. Therefore, the statistical properties of a data flow during the call would be also of interest [KRIS03]. This calls for a more sophisticated statistical model for the cell arrival process. After the discovery of long-range dependence (LRD) in Ethernet traffic at Bellcore [LELA94], a new model for internet traffic called *Self-similar traffic model* was found. Conventional models were basically short range dependant and unable to explain the fractal like behaviour. The nature of congestion produced by self-similar traffic was worse than that of Poisson based models. LRD was also proven to have a significant impact on queuing performance, dimensioning of buffers and real time bandwidth allocation. The findings were important for engineering of switches, multiplexors and real-time control of broadband networks, therefore an approximate tractable model which can characterize the traffic would now be worthwhile.

1.1 Thesis Contributions

The contributions made by this study are as follows:

- Two approaches of synthesizing self-similar network traffic have been studied:
    - Fixed-length packet generation by the aggregation of several Pareto

distributed ON-OFF sources. This method has been applied to the performance evaluation of two novel switching architectures.

- A novel variable-length traffic generation using Fractional Gaussian Noise (FGN) process with packet length distributions determined according to realistic TCP/IP-based traffic.

- The performance of a novel Clos-like packet switch with an Optical Core deploying Flexible Bandwidth Provision (FBP) algorithm under self-similar realistic traffic conditions is evaluated. Realistic performance evaluation measures are important in determining switch design practicality as well as superiority compared to similar architectures.

- A novel load-balancing scheme in Clos-like packet switches is proposed without the need for packet resequencing. The idea is to use multiple switching layers and spread incoming packets evenly to the layers to achieve a balanced load at the center stage switches, then use distributed cross-point queues and specific schedulers to guarantee the output order of packets. The proposed architecture is shown to be effective via simulations and, in comparison to related art in this field, shows superiority in performance.

1.2 Structure of Document

The fundamentals of Internet traffic, self-similar mathematical bases, models for self-similarity and various testing schemes for self-similarity are presented in **chapter 2**. These shall help the reader understand the concept of self-similarity applied to network traffic thoroughly both in terms of generic and mathematical modeling and thus identify practical self-similar traffic modeling concepts.

Chapter 3 will discuss two means of generating self-similar traffic using the concepts introduced in chapter 2. The traffic generated by these two schemes can be used as the test bed for various switching architectures. These methods have been tested

and verified for degree of burstiness and long-range dependence.

A brief explanation of a sectored packet switch with an optical core will be presented in **chapter 4**. The performance of the switch under self-similar traffic conditions will be evaluated thoroughly. The benefits of this architecture in terms of simplicity and scalability with use of optics will be addressed.

In **chapter 5**, a packet switch architecture will be presented which performs switching by load-balancing in order to spread non-uniform incoming traffic evenly throughout the switch. It is shown how this architecture is universal in the sense that other proposed load-balancing architectures are derived from it. Performance evaluations in terms of average delays are also presented.

Chapter 6 concludes by presenting the scientific innovations and contributions that resulted from this research project. Also presented are some of the possible projects considered for future research to further enhance current methods.

Appendices have been included to help streamline the main body of this document. Included is a list of acronyms and the expansion of some of the mathematical evaluations.

2. Self-similar Traffic

The term *self-similar* was first created by Mandelbrott [MAND68] and used for modeling of geological and hydraulic problems [MAND69]. Later it was discovered that the Internet traffic possesses a *self-similar* or *fractal-like* behavior, and, unlike telephone traffic does not smooth when averaged over large timescales, making it bursty over a wide range of time scales. This chapter is focused on a comprehensive discussion of self-similar network traffic. Fractals are introduced, a brief history on the discovery of presence of self-similarity in data traffic is presented and the mathematical models for self-similarity are discussed. At the end of this chapter, various testing schemes are brought up to test the presence of self-similarity in a stochastic process.

2.1 Fractals

Self-similarity is a characteristic associated mainly with fractals and chaos. Fractals are objects that have a similar geometric appearance when viewed at different scales. Many mathematical processes possess this characteristic and many examples are also found in nature, e.g. the structure of a fern leaf. The universal presence and significance of self-similarity has been emphasized in a memorable quote by Manfred Schroeder [SCHR91]:

"The unifying concept underlying fractals, chaos, and power laws is self-similarity. Self-similarity, or invariance against changes in scale or size, is an attribute of many laws of nature and innumerable phenomena in the world around us. Self-similarity is, in fact, one of the decisive symmetries that shape our universe and our efforts to comprehend it."

Fig. 2.1 shows Sierpinski triangle as an example of a fractal. Its structure is repeated over all degrees of magnification. Therefore, a phenomenon that is self-similar looks the same or behaves statistically similar when viewed at different scales. The scale could be dimensions (length, width) or time. It is quite surprising to state that the importance of self-similarity has only been recently emphasized and its key application in data communications traffic discovered. In the following section we focus on stochastic processes that exhibit self-similarity with respect to time in order to interpret data traffic and as means of generating synthetic traffic traces.

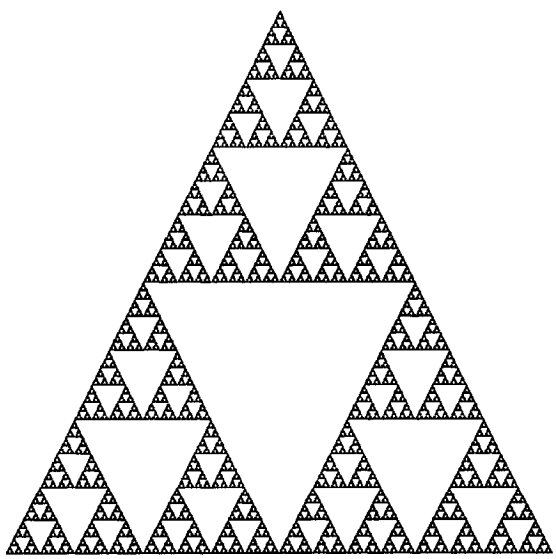


Fig. 2.1 Sierpinski triangle as an example of a fractal

2.2 Self-similar Data Traffic

Traditional analyses of switch performance have used Poisson or Markovian arrival processes to model traffic. Whilst these models proved adequate for voice traffic they have been found inadequate for data traffic. In the 1990s various studies and measurements of Ethernet traffic and World-Wide Web traffic proved that Internet traffic possesses a *self-similar* or *fractal-like* behavior and thus displays long-range dependency or burstiness over a wide range of time scales. The actual network traffic has high degree of traffic variance over all time scales, and, unlike telephone traffic does not smooth when averaged over large timescales. The degree of self-similarity can be measured by the *Hurst parameter (H)* which increases in the interval $0.5 \leq H \leq 1$ as the process becomes more self-similar.

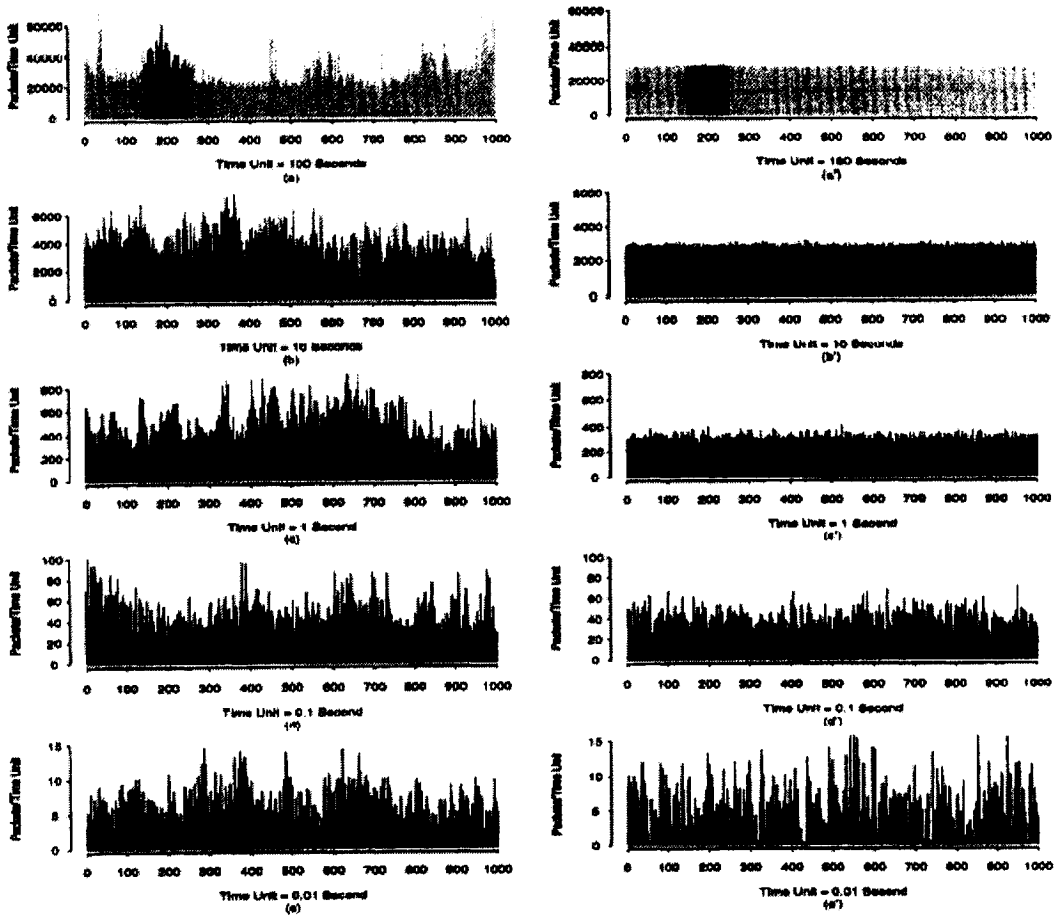


Fig. 2.2 The discovery of self-similar traffic from actual Ethernet data (left column) and the its behaviour compared to synthesized Poisson traffic (right column). [LELA94]

The discovery of self-similarity in network traffic was made by Leland et al in [LELA94]. Using a massive amount of actual data and by careful statistical analysis, this paper proved that the Poisson traffic model is an inadequate description of Ethernet LAN traffic and a new modeling and analysis approach is required. The data collection in that study consisted of a detailed, high-resolution (time accuracy of 20μs) collection of Ethernet traffic measurements conducted between 1989 and 1992. Evaluating LAN data traffic throughout this time emphasized the presence of self-similarity at all times. In addition, the schematic view of the Ethernet data for a measurement set from 1989 at Bellcore labs is shown in Fig. 2.2, which indicates the number of packets transported per unit time. Each plot on the top of column is an aggregation of the plot below it. Viewed generally, all the plots involve a fair amount of burstiness and thus look statistically the same. This can be interpreted in the sense that Ethernet traffic tends to look the same at large scales (hours, minutes) and at small scales (seconds, milliseconds). On the other hand, the plots on the left column show synthesized Poisson traffic. As can be seen, the Poisson model provides bursty traffic for smaller time scales which tends to smooth over larger times and therefore this model does not provide an accurate representation of the Ethernet traffic.

World_Wide Web (WWW) traffic has also been proven to possess self-similarity in a study performed by Crovella et al [CROV97]. In this paper the requests for Web documents have been analyzed and the study shows that the data traffic generated by the Web browsers is self-similar. Each web browser was modeled by an ON/OFF process, where the ON/OFF duration data fit well to a Pareto distribution. The authors did some tests in order to explain the self-similar behaviour of Web traffic, where they hypothesize that Web traffic reflects a random selection of files for transfer and that the files available via the Web over the Internet have a size distribution that is heavy tailed. That is, along with plenty of small files on the Internet, there exist large and very large files, like multimedia files, available for transfer over the Web.

To investigate the characteristics of this type of traffic, another study of TCP traffic, as well as FTP and TELNET traffic carried over TCP connections, has been performed by Paxson et al [PAXS95a]. It is well proven by visualization that commonly used Poisson models seriously underestimate the burstiness of TCP traffic over a wide

range of time scales. It was also found that for FTP and TELNET traffic, only the *connection* arrivals are well modeled by a Poisson distribution, but the Poisson model for data packet arrivals significantly underestimates the burstiness of the traffic.

Given the discussion above, it is appropriate to consider self-similarity as a stochastic process and to find its statistical properties. For a self-similar process, we can state that the statistics of the process do not change with a change in the time scale, while an exact representation of the signal is not performed; i.e. a self-similar process is not a deterministic process and does not preserve the exact shape at all time scales.

2.3 Mathematical Models for Self-similarity

2.3.1 Probability Backgrounds

Before we proceed with mathematical interpretations of self-similar process, some basic probability and stochastic definitions will be reviewed to ease the understanding of later discussions. For every continuous-valued stochastic process, the continuous-valued random variable $x(t)$ for a fixed value of t is usually described by a probability distribution and a probability density with the following structure:

Cumulative Distribution Function:

$$F(a;t) = P[x(t) \le a] \quad F(-\infty;t) = 0 \quad F(+\infty;t) = 1 \quad (P \text{ stands for probability}) \tag{2.1}$$

Probability Density Function:

$$f(x;t) = \frac{\partial}{\partial x} F(x;t) \qquad \int_{-\infty}^{\infty} f(y;t)dy = 1 \tag{2.2}$$

The mean and variance of a stochastic process are defined below where the mean defines the process average value and the variance is the amount of fluctuations in random variable values around the mean (average). In general the mean and variance of

a stochastic process are functions of time.

Mean:

$$E[x\{t)] = \mu(t) = \int_{-\infty}^{\infty} xf(x;t)dx \tag{2.3}$$

Variance:

$$Var[x(t)] = \sigma^2(x(t)) = E[(x(t) - \mu(t))^2] = E[x^2(t)] - \mu^2(t) \tag{2.4}$$

Parameter σ is called the *standard deviation* of a stochastic process which is the square root of variance.

An important concept in our discussion is the autocorrelation function which is the joint moment of 2 random variables at times t_1 and t_2.

Autocorrelation Function:

$$R(t_1, t_2) = E[x(t_1)x(t_2)] \tag{2.5}$$

The autocorrelation function is a measure of the relationship between the two time instances of a stochastic process. A related measure is the autocovariance:

Autocovariance:

$$C(t_1, t_2) = E[(x(t_1) - \mu(t_1))(x(t_2) - \mu(t_2))] = R(t_1, t_2) - \mu(t_1)\mu(t_2) \tag{2.6}$$

Up to this point, the important properties of random processes such as mean, correlation and variances have been introduced. It would be useful to be able to show whether or not the statistical properties of a sequence or group hold true for the entire random process. To do this, the concept of stationary processes has been developed. A stationary process is a random process where its statistical properties do not vary with

time. Stationary processes are of different degrees. The *strict-sense stationary* process is one which the probability density function remains constant at all times. To relax this constraint, *wide-sense stationary process* is defined. In general, a stochastic process is stationary in the wide sense if its expected value is a constant and its autocorrelation function depends only on the time difference:

$$E[(x(t)] = \mu \text{ and}$$
$$R(t,t+\tau) = R(t+\tau,t) = R(\tau) = R(-\tau) \text{ for all t}$$

(2.7)

Further, the variance and autocovariance functions of a Wide-sense Stationary Process change as follows:

$$Var[x(t)] = R(t,t) - \mu^2(t) = R(0) - \mu^2$$
$$C(t,t+\tau) = R(t,t+\tau) - \mu(t)\mu(t+\tau) = R(\tau) - \mu^2 = C(\tau)$$

(2.8)

The autocorrelation function of a wide-sense stationary process $R(\tau)$ measures the degree of dependency of a stochastic process at one time instant on other time instants. If $R(\tau)$ goes towards zero rapidly as τ becomes larger, there is little dependency of the process at one time instant on distant time instants, but if $R(\tau)$ decays slowly for large τ, the process at different time instants are related and the process has a long memory.

Another important concept defined here is the *spectral density* or the *power spectrum* of a stationary random process which is the Fourier transform of its autocorrelation function:

Power spectrum:

$$S(w) = \int_{-\infty}^{\infty} R(\tau)e^{-jw\tau}d\tau \qquad w \text{ is the angular frequency}$$

(2.9)

$S(w)$ is the power (energy per unit time) of the random process $x(t)$ falling within the neighborhood of w.

2.3.2 Self-similar Stochastic Processes

A stochastic process is considered to be *exactly second-order self-similar* with parameter β $(0 < \beta < 1)$ if for all $m = 1,2,...$ the following applies:

$$Var(x^{(m)}) = \frac{Var(x)}{m^{\beta}} \quad \text{Variance}$$

$$R_{x(m)}(t) = R_x(t) \quad \text{Autocorrelation} \tag{2.10}$$

For a stationary time series $x(t)$, $x^{(m)} = \{x_t^{(m)}, t = 0,1,2,...\}$ is the m-aggregated time series obtained by aggregation of the original time series over non-overlapping, adjacent blocks of size m, i.e.

$$x_t^{(m)} = \frac{1}{m} \sum_{i=tm-(m-1)}^{tm} x_i \tag{2.11}$$

This could be viewed as a time average of process x.

The parameter β is related to H, the Hurst parameter by $H = 1 - (\beta/2)$. The **Hurst parameter** or the **self-similarity parameter** is a key measure of the degree of self-similarity in a process and as H increases, the degree of persistence or long-range dependence increases.

A weaker condition is that a process is said to be *asymptotically second-order self-similar* if for all t large enough:

$$Var(x^{(m)}) \sim \frac{Var(x)}{m^{\beta}} \quad \text{Variance}$$

$$R_{x(m)}(t) \to R_x(t) \text{ as } m \to \infty \quad \text{Autocorrelation}$$

The above definitions state that the autocorrelation of the aggregated process has the same form as the autocorrelation of the original process. This would imply that the degree of variability, or burstiness, would be the same at different time scales. It is now time to define two concepts related to self-similarity: long-range dependence and heavy-tailed distributions. These two concepts are independent of one another and of self-similarity, although a self-similar process must have both characteristics.

2.3.3 Long-Range Dependence

Long-range dependence of a stationary process with autocovariance function $C(\tau)$ is described as follows:

$$C(\tau) \sim |\tau|^{-\beta} \text{ as } |\tau| \to \infty,\ 0 < \beta < 1 \tag{2.12}$$

Therefore a long-range dependant process has a hyperbolically decaying autocovariance and $\sum_{\tau} C(\tau) = \infty$. In general, long range dependence implies the existence of clustering and bursty characteristics at all time scales in self-similar processes. Long-range dependence could be stated in the frequency domain as well. The power spectral density obeys a power law near the origin:

$$S(w) \sim \frac{1}{|w|^{\gamma}} \text{ as } w \to 0,\ 0 < \gamma < 1 \tag{2.13}$$

Where $\gamma = 1 - \beta = 2H - 1$. In contrast, for short-range processes, the autocovariance decays rapidly, for example, like an exponential and $\sum_{k} C(k)$ is finite. The spectral density of these processes remains finite as $w \to 0$.

2.3.4 Heavy-Tail distributions

A random variable with a heavy-tailed distribution exhibits infinite variance, and possibly an infinite mean. The result of such random variable is samples with many relatively small values and a few relatively large samples. The cumulative distribution *F(x)* of a random variable *x,* it is said to be heavy tailed if:

$$1 - F(x) = P(X > x) \sim \frac{1}{x^{\alpha}} \text{ as } x \to \infty, \alpha > 0$$

(2.14)

Heavy-tailed processes are of matter in traffic modeling, because they explain the behaviour of self-similar traffic [WILL97]. The individual traffic sources exhibit characteristics that cover a wide range of time scales and have high-variability. A very popular heavy tailed distribution is the *Pareto distribution* with parameters α , β and a cumulative distribution given by:

$$F(x) = 1 - \left(\frac{\beta}{x}\right)^{\alpha}, (x > \beta, \alpha > 0)$$

(2.15)

The probability density function is therefore:

$$f(x) = \frac{\alpha}{\beta}\left(\frac{\beta}{x}\right)^{\alpha+1}$$

(2.16)

The mean of the distribution is:

$$E[x] = \frac{\alpha}{\alpha - 1}\beta, (\alpha > 1)$$

(2.17)

Parameter α describes the tail of the distribution and determines the mean and variance of the random variable. If $\alpha \leq 2$ the distribution has infinite variance, and if $\alpha \leq 1$, it has infinite mean and variance. In order to obtain a self-similar process, $1 < \alpha < 2$ and only the variance will be infinite. Fig. 2.3 plots the Pareto distribution for different α and compared to the exponential distribution. It can be seen that the tail of the Pareto distribution decays much more slowly than the exponential, consistent with its description as a *heavy-tail*ed distribution.

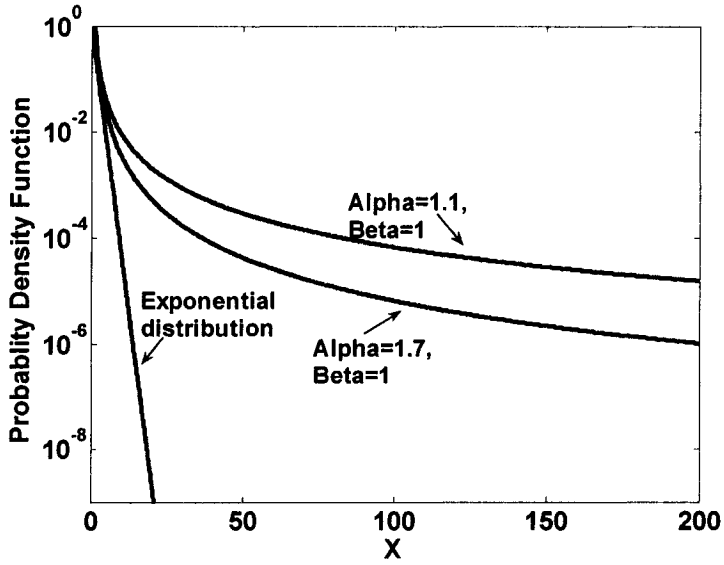


Fig. 2.3 Pareto distribution versus Exponential distribution with $\lambda = 1$.

2.4 Tests for Self-similarity

Test procedures are required in order to determine whether a time series of synthetic or actual data is self-similar and if so, to estimate the Hurst parameter *H*.. Some of the more popular procedures are described in the following sub-sections.

2.4.1 Variance-Time Plot

For a self-similar process, the variance of the aggregated time series $x^{(m)}$, obeys:

$$Var(x^{(m)}) \sim \frac{Var(x)}{m^{\beta}}$$

(2.18)

for large m where β is related to the Hurst parameter by $H = 1-(\beta/2)$. Taking the logarithm, (2.18 may be re-written as:

$$\log[Var(x^{(m)})] \sim \log[Var(x)] - \beta \log(m)$$

(2.19)

Hence for self-similar process, a plot of $\log[Var(x^{(m)})]$ versus $\log(m)$, will yield a straight line with slope of $-\beta$. The plot is easily generated by computing the aggregate process of the data series $x(t)$ at different levels of aggregation m and then obtaining the variance. Resulting negative slope values between -1 and 0 suggest Hurst parameters between 0.5 and 1 and therefore the self-similarity of the trace.

2.4.2 R/S Plot

For a stochastic process $x(t)$ defined at discrete time instances $\{X_t, t = 0,1,2,...\}$, the rescaled range of $x(t)$ over a time interval T is defined as the ratio *R/S*:

$$\frac{R}{S} = \frac{\max\limits_{1 \le j \le T}\left[\sum\limits_{k=1}^{j}(X_k - M(T))\right] - \min\limits_{1 \le j \le T}\left[\sum\limits_{k=1}^{j}(X_k - M(T))\right]}{\sqrt{\frac{1}{T}\sum\limits_{k=1}^{T}(X_k - M(T))^2}}$$

(2.20)

Where $M(T)$ is the sample mean over the time period T :

$$M(T) = \frac{1}{T}\sum_{j=1}^{T} X_j$$

(2.21)

The numerator in (2.21 is a measure of the range of the process and the denominator calculates the standard deviation of the sample. According to ref [STAL02] for a self-similar process, the ratio has the following characteristic for large T:

$$R/S \sim (T/2)^H \text{ with } H > 0.5$$

Taking a logarithm this may be rewritten as

$$\log[R/S] \sim H\log(T) - H\log(2)$$

(2.22)

And therefore for a self-similar process a plot of $\log(R/S)$ versus $\log(T)$ will yield a straight line with slope H.

2.4.3 The Periodogram

The spectral density of a stochastic process $x(t) = \{X_k, k = 1,2,...\}$ defined at uniformly spaced discrete times can be estimated by applying a discrete Fourier Transform over an interval containing n samples, as follows:

$$I_n(w) = \frac{1}{2\pi n}\left|\sum_{k=1}^{n} x_k e^{-jkw}\right|^2$$

(2.23)

The resultant spectral density is called a *periodogram* and may be used to reveal long-Range dependence. The Periodogram plot is obtained by plotting $\log 10(I_n)$ versus $\log 10(w)$. The spectral density of a stochastic process is the Fourier transform of its autocorrelation function, therefore, if the autocorrelation is summable, near the origin the samples of the periodogram should be scattered randomly around a constant. On the other hand, if the autocorrelation is non-summable and therefore Long-Range

Dependant, the points of the sequence are scattered around a negative slope. An estimate of the Hurst parameter is given by $H = (1 - \hat{\beta})/2$ where $\hat{\beta}$ is the slope [JEON98]. Samples of the periodogram are shown in Fig. 2.4 for different values of H. As the degree of burstiness increases the scattering of sample points around the origin are more vivid.

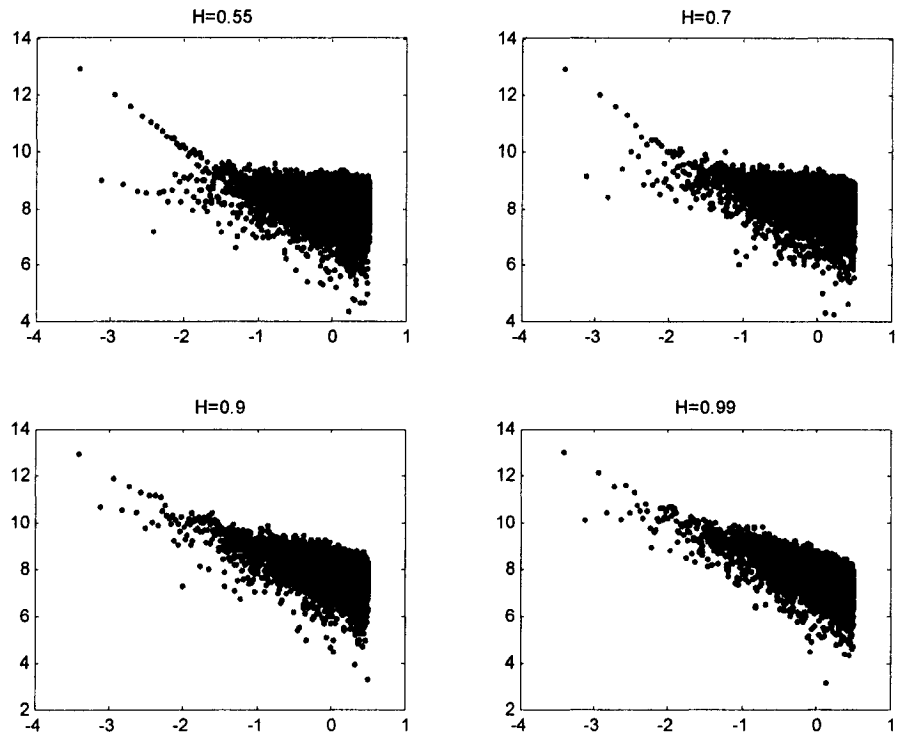


Fig. 2.4 Periodogram for different values of H and n=8192 samples.

3. Self-Similar Traffic Generation

The previous chapter defined self-similarity, introduced its mathematical properties and described its relevance to real Internet traffic. This chapter considers means of generating self-similar traffic. Synthesized self-similar traffic is important in evaluating the performance of various switch and network architectures under realistic conditions with the goal of devising systems which cope well with self-similar traffic in terms of acceptable delays and finite queue lengths. Therefore, this chapter focuses on two methods of generating self-similar traffic with specified load, Hurst parameter, and number of input/output ports. The first method deploys heavy-tailed Pareto distribution to generate fixed length packets, while the second method focuses on Fractional Gaussian noise as means of generating variable-length packets.

3.1 Pareto Traffic Generation

The self-similar traffic generated by this method consists of fixed-length packets with packet duration equal to a *time-slot*. The self-similar traffic is approximated by the aggregation of N alternating renewal processes, known as "ON-OFF" processes; i.e., each of the traffic sources (inputs to a switch) is an ON-OFF process. The ON and OFF times are Pareto-distributed random variables, which are heavy-tailed (i.e., with infinite variance). The aggregation of these sources yields a self-similar process with long-range dependence in the limit of large N [WILL97]. Since fixed length packets are used, the

ON times correspond to the *packet train size*, i.e., the number of packets transmitted contiguously and the OFF times correspond to the silent times, where no packets arrive at that input. Both are defined in time slots. For every new train of packets, a destination port is chosen at random with a uniform distribution. As stated in section 2.3-4, the probability density function f for a Pareto distributed random variable x is:

$$f(x) = \alpha\beta^{\alpha} x^{-\alpha-1}, x \geq \beta, \beta > 0$$

(3.1)

With mean:

$$E(x) = \frac{\alpha\beta}{\alpha - 1}$$

(3.2)

β is the minimum value of the random variable x, and α is the tail index; $1<\alpha<2$ for a heavy-tailed distribution. Longer tails in the distribution are obtained as $\alpha \to 1$. To generate the traffic traces, for the ON times, a set of parameters α_{on} and β_{on} are used, which differ from α_{off} and β_{off} used for the OFF times. From (3.2 the mean ON-time of a source will be:

$$E_{on}(x_{on}) = \frac{\alpha_{on}\beta_{on}}{\alpha_{on} - 1}$$

(3.3)

Similarly, the mean OFF-time of a source will be:

$$E_{off}(x_{off}) = \frac{\alpha_{off}\beta_{off}}{\alpha_{off} - 1}$$

(3.4)

The load is therefore given by the source mean ON-time, in total time:

$$\rho = \frac{E_{on}}{E_{on} + E_{off}}$$

(3.5)

Using equations above, α_{on}, α_{off} and β_{on} may be specified and the parameter β_{off} calculated to yield a desired load, ρ. The parameter β_{on} is intuitively unity, since there are control packets only one-packet long. However, if packet segmentation from higher network layers is involved, β_{on} could become larger.

To computer-generate random values that fit the Pareto distribution, it is necessary to map samples, u from a uniform distribution over [0,1] which may be generated with relative ease by computer, to samples x of the Pareto distribution. To find the mapping $x(u)$, the cumulative distribution function of the Pareto distribution $F(x) = 1 - (\beta/x)^{\alpha}$ is equated to the cumulative distribution function of the uniform distribution, $F(u) = u$ $(0 \le u \le 1)$ yielding

$$1 - (\beta/x)^{\alpha} = u$$

and hence:

$$x = \frac{\beta}{(1-u)^{1/\alpha}}$$

There is no change if u is substituted in place of (1-u) as both variables are uniformly distributed between between 0 and 1 , yielding:

$$x = \frac{\beta}{u^{1/\alpha}}$$

(3.6)

The packet generation process begins by having $\sim \rho N$ sources ON and $\sim (1-\rho)N$ sources OFF, N being the number of input/output ports. An ON-time is a Pareto distributed random variable with parameter α_{on} and β_{on} obtained from (3.3. When an ON-time is finished, an OFF-time is obtained by inserting α_{off} and β_{off} in (3.4. A destination is chosen randomly each time a new train of packets (ON-time) starts. All the packets in this train will have the same destination.

The values generated for random variable x have their decimal part truncated so that an integer number is obtained since train sizes and silent times are measured in timeslots, as a whole unit. Different measurements indicate that this truncation does not significantly affect the actual overall load obtained for the traffic traces.

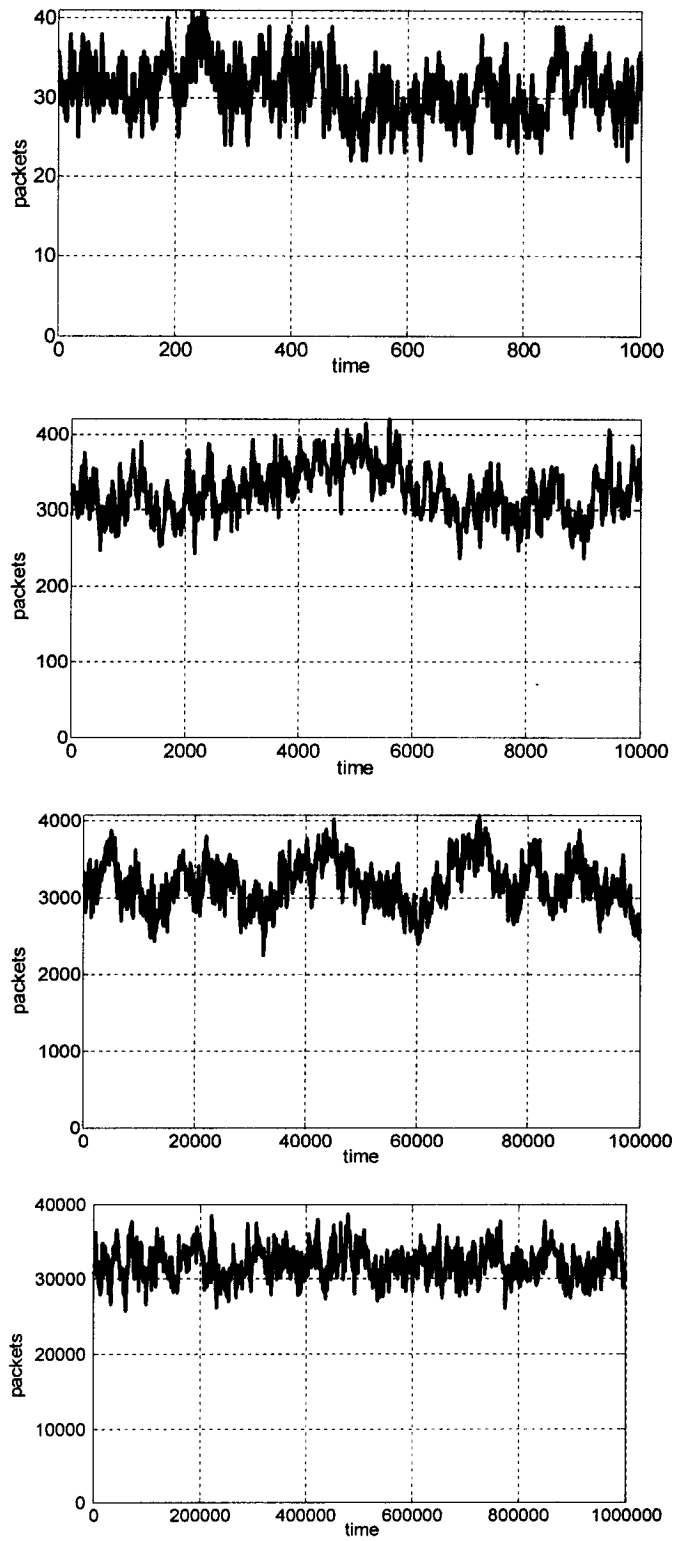


Fig. 3.1 Pareto-generated traffic traces with the aggregation scale of 1, 10, 100 and 1000 time slots respectively.

Fig. 3.1 shows a sample of traffic generated with this method for *N=64* sources, $\alpha_{on} = \alpha_{off} = 1.1$, $\beta_{on} = 1$ and a load of $\rho = 0.5$. The total number of arrivals at the switch is aggregated over 1, 10, 100, and 1000 time slots. The trace shows visual signs of self-similarity and does not smooth with aggregation. Aside from visual inspection, the traffic traces have also been tested for self-similarity using the testing schemes described in section 2.4. Fig. 3.2, Fig. 3.3, and Fig. 3.4 respectively show the VTP test, R/S plot and Periodogram test performed on the traffic traces of Fig. 3.1. The estimated Hurst parameter is about $H = 0.9$ in all 3 testing schemes confirming a self-similar process. Also, the scattering of data points away from the origin in Fig. 3.4 mentions the long-range dependency of traffic traces, another visual sign of self-similarity. In general for all Pareto-generated traffic traces with $1 < \alpha < 2$, Hurst parameter is expected to be $0.5 < H < 1$ for a bursty long-range dependant trace.

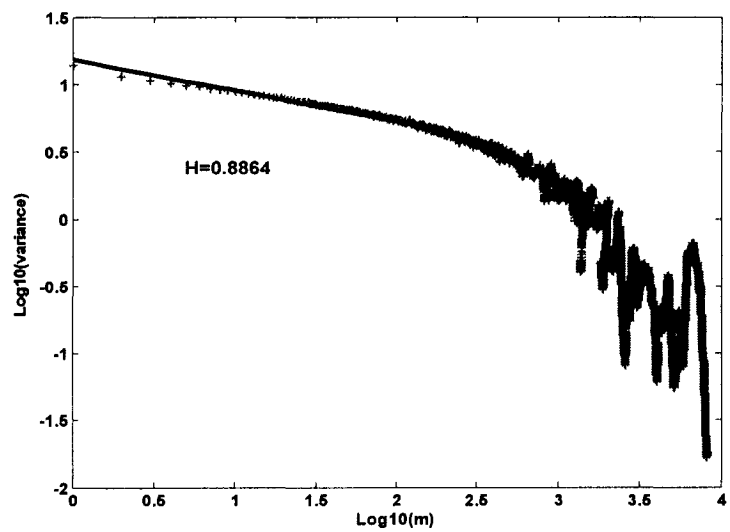


Fig. 3.2 VTP test for the traffic trace of Fig. 3.1 , H=0.8864.

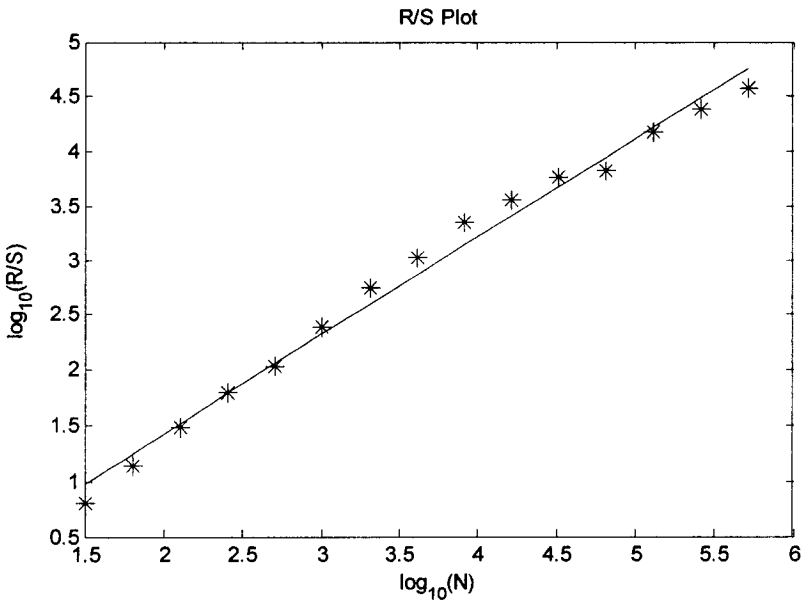


Fig. 3.3 R/S Plot test for a traffic trace of Fig. 3.1, H=0.8965.

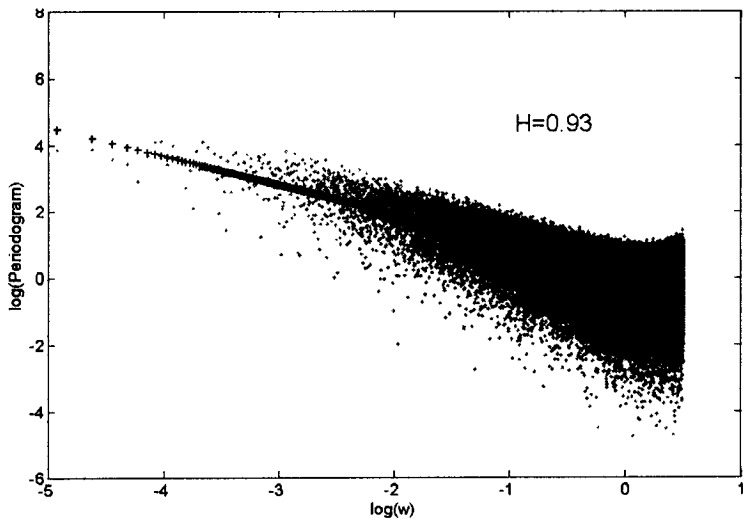


Fig. 3.4 Periodogram of Fig. 3.1, H=0.93.

Fig. 3.5 shows how the Hurst Parameter changes with a change in the tail index (α) and minimum train size (β) for a constant load of $\rho = 0.9$. The measures of H in this plot come from R/S plot estimates over several runs. The R/S testing technique was chosen due to its computational speed without loss of accuracy. It can be seen that the degree of self-similarity (Hurst parameter) decreases with increasing α, consistent with the relationship in [LELA94],

$$H = \frac{(3-\alpha)}{2} \tag{3.7}$$

Also as the minimum train size β increases, a burstier, 'Hurstier' trace is expected, but the effect of β is not as influential as α on the degree of self-similarity. Greater realism may be achieved by setting different loads for each source and defining their average as the overall load.

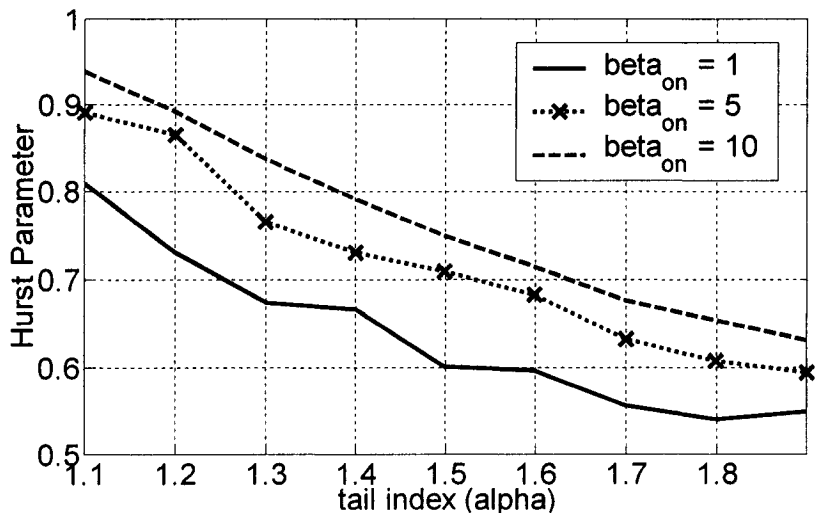


Fig. 3.5 Hurst Parameter vs. α for 90% load

The effect of parameters α_{on} and β_{on} on a train size or ON-time is best visualized in Fig. 3.6. It can be seen that values of α_{on} closer to unity produce greater ON-time burst sizes. This is due to a longer tail in the distribution and therefore higher a

probability of longer burst. The β_{on} parameter or minimum train size has greater effect on burst size: a large β_{on} would result in larger trains. This result is consistent with (3.3 and the fact that the choice of α_{on} and β_{on} determines mean burst length.

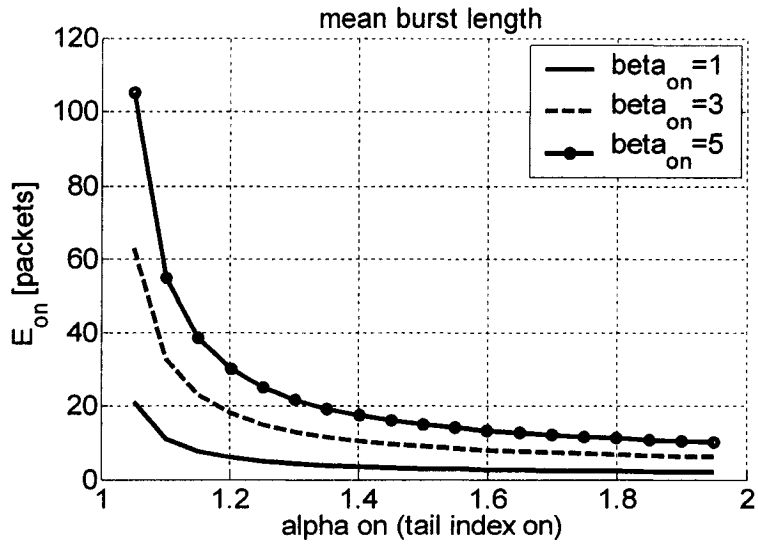


Fig. 3.6 Mean burst length E_{on} vs. α_{on} for different values of β_{on}

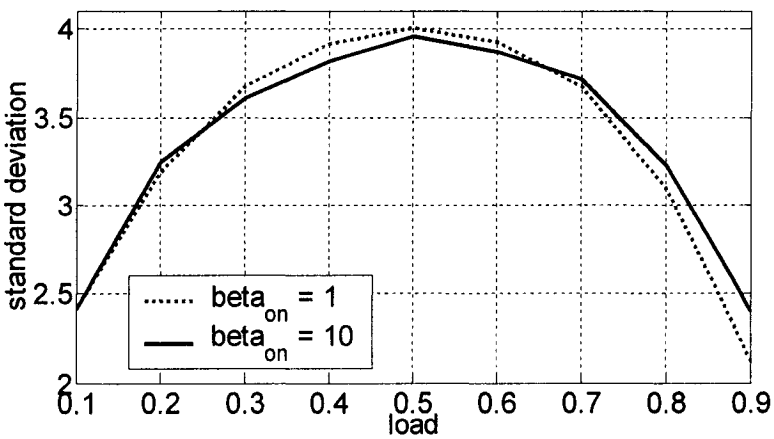


Fig. 3.7 Standard deviation vs. load for α_{on}= α_{off}=1.2

Fig. 3.7 shows the standard deviation of packet counts relative to load. The highest standard deviation occurs at 50% load, and the graph is symmetrically

distributed around 50% load. This measure is affected mainly by the load, so the tail index (α) and the minimum value of train size (β_{on}) have minimal effects on this measure.

3.2 FGN Traffic Generation

In addition to generating fixed-length packets, it is of interest to introduce more realism by generating variable length packets with self-similar statistics. The packets length distribution may be inferred from measurements of actual TCP/IP network traffic. The choice then remains of which traffic model to use for generating the self-similar time traces. The best known mathematical models that exhibit self-similarity are Fractional Gaussian Noise (FGN), integrated FGN which generates Fractional Brownian Motion (FBM) and Fractional ARIMA. Due to its popularity and related research in this area [PAXS95b] and [LEDE00], Fractional Gaussian Noise was chosen in this work for the purpose of producing streams of variable-length packets with self-similar statistics.

Fractional Gaussian Noise (FGN) is a stationary Gaussian process $\{x = x_k : k = 0,1,2,...\}$, and among the most well-known processes with self-similar characteristics. FGN is characterized by its mean μ, variance σ^2 and Hurst parameter H with autocorrelation function:

$$r(k) = 1/2\left(|k+1|^{2H} - |k|^{2H} + |k-1|^{2H}\right) \quad k = 1,2,3,... \tag{3.8}$$

Asymptotically, $r(k) \sim H(2H-1)|k|^{2H-2}$ $(k \to \infty, 0 < H < 1)$

Also, the resulting aggregated processes $x^{(m)}, (m = 1,2,3,...)$ all have the same distribution as x, for all $0 < H < 1$. Thus according to section 2.3.2, fractional Gaussian noise is exactly second-order self-similar with self-similarity parameter H, as long as $1/2 < H < 1$.

3.2.1 Fast Fourier Transform for FGN Self-similar Trace Generation

To produce the samples of an approximate FGN self-similar process, the algorithm presented by Paxson in [PAXS95b] is used. The algorithm uses a *Fast Fourier Transform (FFT)* method and is based on synthesizing sample paths that have the same power spectrum as that of Fractional Gaussian Noise process. The power spectrum is constructed using the Whittle estimator. The Whittle estimator is normally a method which allows the estimation of the Hurst parameter from an empirical data set. However the Whittle estimator may be reversed and used to generate the desired power spectrum with the chosen value of H as input. Once the power spectrum of the FGN process has been successfully estimated, the Inverse Fourier Transform is used to obtain the time trace of the FGN.

The method starts by estimating the power spectrum of the FGN process as:

$$f(\lambda;H) = A(\lambda;H)[|\lambda|^{-2H-1} + B(\lambda;H)] \quad 0 < H < 1\ ,\ -\pi \le \lambda \le \pi \tag{3.9}$$

Where:

$$A(\lambda;H) = 2\sin(\pi H)\Gamma(2H+1)(1-\cos\lambda)$$

$$B(\lambda;H) = \sum_{j=1}^{\infty}[(2\pi j + \lambda)^{-2H-1} + (2\pi j - \lambda)^{-2H-1}]$$

The difficulty in computing the power spectrum is provided by the infinite summation of $B(\lambda;H)$. To overcome this issue, Ledesma et al [LEDE00] have used ideas from Paxson in [PAXS95b] and modified them by approximating certain values in order to speed computations. Ledesma's method to estimate the infinite summation is discussed in more detail in Appendix B.

Once the power spectrum is known, samples of the amplitude spectrum may be chosen as Rayleigh distributed random complex variables with zero mean and variance equal to the corresponding sample of the power spectrum. (Note that a Rayleigh distributed random complex variable has a negatively distributed squared magnitude and

a uniformly distributed phase). This yields a frequency domain sample path represented by a sequence of complex numbers $\{z_i\}$, i.e. the Fourier transform of a Gaussian process. An inverse Fourier transform yields a time series $\{x_i\}$ with the same autocorrelation function as the FGN process since it shares the same power spectrum. In detail:

The inputs to this method are the Hurst parameter, H and the desired even number of observations n. The self-similar samples are obtained as follows:

1. Construct all the values of power spectrum as $\{f_1,...,f_{n/2}\}$, where $f_i = \tilde{f}\left(\frac{2\pi i}{n}; H\right)$, accounting for frequency changes ranging from $2\pi/n$ to π.
2. Multiply each $\{f_i\}$ by an independent negative-exponential random variable with mean of unity to give a new sequence $\{\hat{f}_i\}$.
3. Build a sequence of complex values as $\{z_1,...,z_{n/2}\}$ such that $|z_i| = \sqrt{\hat{f}_i}$ and the phase of z_i is uniformly distributed between 0 and 2π. Random phase assignment preserves power spectrum and ensures that different sample paths are independent.
4. Build the expanded version of sequence z as $\{z'_0,...,z'_{n-1}\}$:

$$z'_i = \begin{cases} 0, i = 0 \\ z_i, 0 \le i \le \frac{n}{2} \\ \bar{z}_{n-i}, \frac{n}{2} < i < n \end{cases}$$

   $\{z'_i\}$ has the same power spectrum as $\{z_i\}$, but is symmetric around $z'_{n/2}$, and hence corresponds to the Fourier transform of a real-valued signal.
5. Take the Inverse-Fourier Transform $\{z'_i\}$ to obtain the approximate FGN sample path $\{x_i\}$.

The FGN samples $\{x_i\}$ are to possess self-similar characters and are therefore tested for self-similarity. VTP testing scheme has been used here to estimate the Hurst

parameter of every trace. In Fig. 3.8, a least-squared fit was used to obtain a straight line and thus a constant slope corresponding to β and derive H using (2.19.

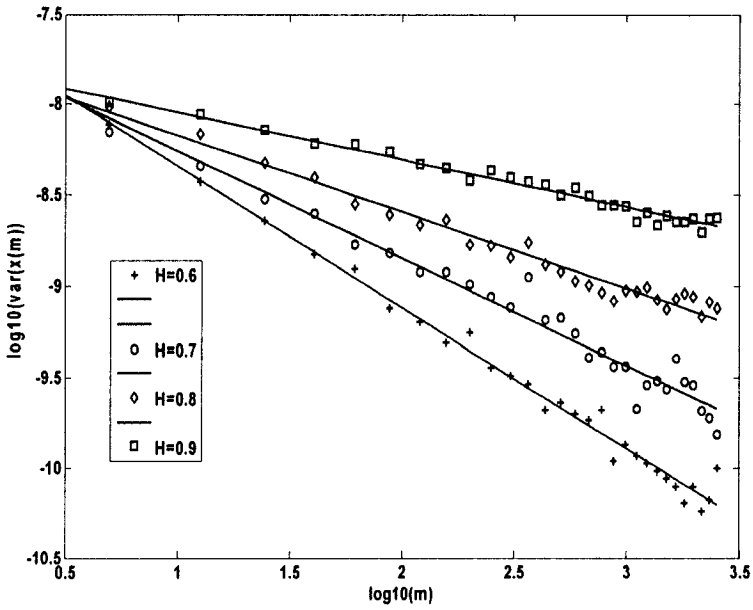


Fig. 3.8 VTP for FGN traces to test their self-similarity and estimate their Hurst parameter

Table 3.1 summarizes some of the measurements of H' calculated using VTP and compared to the desired H for FGN traffic traces. The estimated H' is the average of 10 traces with a different random seed. In general the degree of burstiness is almost as intended. It can be seen that for $H \leq 0.75$, the results are close enough to the intended result. For $H \geq 0.75$, the results are usually lower (The VTP is usually steeper-sloped) than expected. The reason for this could be that VTP is based on an asymptotic relationship, so this anomaly may be due to stronger long-range dependence for higher values of H, requiring longer sample paths to show long-range dependence. The number of samples was $n = 1024$ in Table 3.1.

Input H	Average H'	Range of H'
0.55	0.53	0.48-0.58
0.6	0.59	0.54-0.67
0.65	0.62	0.58-0.68
0.7	0.67	0.62-0.71
0.75	0.7	0.65-0.75
0.8	0.74	0.7-0.81
0.85	0.79	0.72-0.85
0.9	0.82	0.79-0.87
0.95	0.81	0.75-0.9

Table 3.1 Estimation of the Hurst parameter of FGN traces using VTP.

3.2.2 Variable Length Packet Generation from FGN Trace

A sample FGN trace is shown in Fig. 3.9 for $H = 0.95$. This trace has a small standard deviation ($\sigma = 0.003$) which is then normalized to a normal standard deviation $\sigma = 1$.

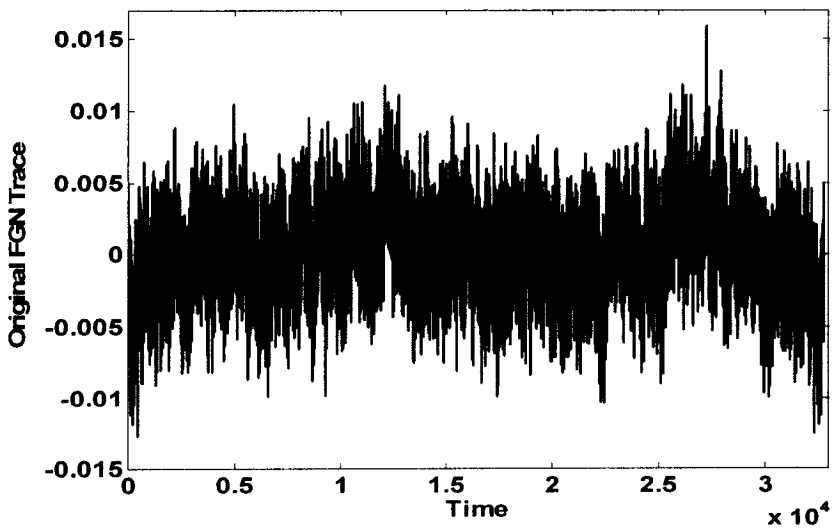


Fig. 3.9 Sample self-similar trace generated using FFT, H=0.95.

What has been obtained up to this point is a Gaussian Noise process with mean $\mu = 0$, normal standard deviation $\sigma = 1$ and a LRD correlation structure which may be scaled and a constant added to produce a process with a desired mean and standard deviation with the same correlation structure. The aim is to obtain variable length sample packets with self-similar structure from the trace. From [WILL97] it is known that this process follows from an aggregation of N Pareto distributed ON-OFF processes with rate R/N in the limit of infinite N, where R is the link rate in bytes per second. Each ON-OFF process therefore contributes either zero or R/N to the sum. The resultant process, a must therefore be bounded $0 \le a \le R$. However, FGN may go out of these bounds. The probability of this is small if $\sigma << \mu$.

Considering a fixed link rate of R bytes/second, the mean μ and standard deviation σ of the self-similar trace will change according to our demand. The mean of the trace is directly related to the traffic load ρ, say

$$E(x) = \mu = \rho.R \tag{3.10}$$

The standard deviation is chosen to be quite smaller than the mean, so that the samples, a will not exceed bounds with significant probability, preserving the condition $0 \le a \le R$:

$$STD = \sigma << E(x) \tag{3.11}$$

However, the trace may still obtain values out of range. The process could be hard-clipped at this point to force it to fit the bound. The number of values out of bound are very limited and have a small probability of occurrence if $\sigma << E(x)$. The hard-clipping in this case is not too worrisome and reflects a small error inherent in the approximate analysis being undertaken. As an example, the trace of Fig. 3.9 is modified considering a link rate of $R = 10^5 B/s$ and a load of $\rho = 0.8$. The standard deviation is chosen to be $\sigma = \mu/12$ to hard-clip the least number of samples. For this example, 67 sample points

exceeded the link rate and were forced to fit the bounds. This can be clearly seen in Fig. 3.10. Every sample now represents a flow in number of bytes/second. The flow of bytes should be fragmented into different packets with a standard packet length distribution.

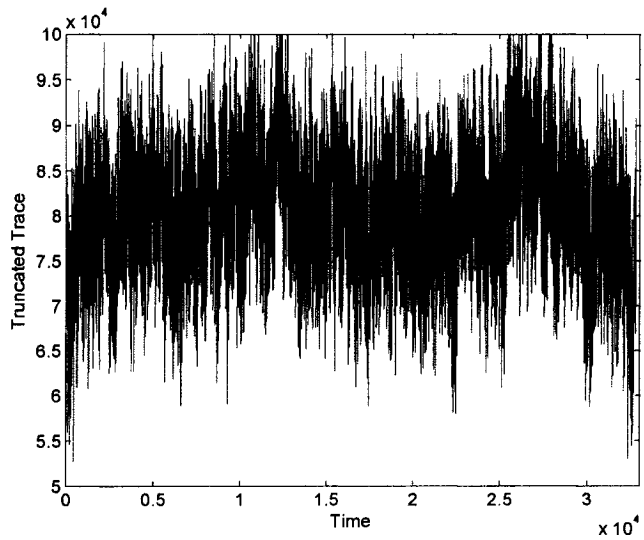


Fig. 3.10 The modified trace of Fig. 3.9 according to link rate (R) and desired load.

It is worth mentioning that the modified trace of bytes per second has preserved the Hurst parameter of the original FGN trace and therefore changing the mean and variance of the self-similar trace does not affect its burstiness and degree of self-similarity. The bounded points are very limited in amount; therefore they do not have a significant effect on the self-similar or Hurst parameter. This has been verified by the testing schemes mentioned in section 2.4.

The FGN trace with desired mean, standard deviation and Hurst parameter has been built, where every point $a(t)$ in the trace would be considered as an instantaneous byte rate. Therefore:

$$0 \leq a(t) \leq R \tag{3.12}$$

And the cumulative number of bytes received in [0, t] are $A(t)$:

$$0 \leq A(t) \leq Rt \tag{3.13}$$

Where $A(t)$ is thought of as containing a large number of flows each containing variable length packets distributed according to $f_P(L)$, the packet length probability density function. Parameter $q(L)$ determines the percentage of every packet length in a flow determined by the packet length distribution:

$$q(L) = \frac{f_P(L).L}{\sum_L f_P(L).L} \tag{3.14}$$

From these flows $A(t)$, fictitious flows $A(t;L)$ of packets with fixed length L are considered where $A(t;L)$ determines the number of bytes in the traffic trace dedicated to every packet length L:

$$A(t;L) = q(L).A(t) \tag{3.15}$$

Imagine each of these flows feeding an aggregation buffer. Then the cumulative arrival of packets ready for departure in every buffer holding different length packets is:

$$A_L(t) = \left\lfloor \frac{A(t;L)}{L} \right\rfloor \tag{3.16}$$

The virtual aggregation buffers holding fixed-size packets and the concept of filling and serving them are shown pictorially in Fig. 3.11. A buffer is ready when it has at least one packet ready to depart. At time t, a buffer is selected uniformly and at random from amongst the ready buffers. A packet is dispatched at the head of that

buffer. Correspondingly, another packet is dispatched randomly from a non-empty buffer at another random time $t' \geq t + \frac{R}{L}$. The departure rate is therefore R bytes/second while at least one buffer is ready and zero otherwise which corresponds to an effective cumulative service of Rt. Once the bytes are segmented into variable length packets and placed in the correct buffer, they have to be served according to a scheme. Since:

$$\sum_{L} A_L(t).L \leq \sum_{L} A(t;L) \leq A(t) \leq Rt \tag{3.17}$$

The buffers are stable, and the result is a variable length stream of non-overlapping packets with the desired statistics (in the form of an ON-OFF process). The destination address distribution can be assigned as required.

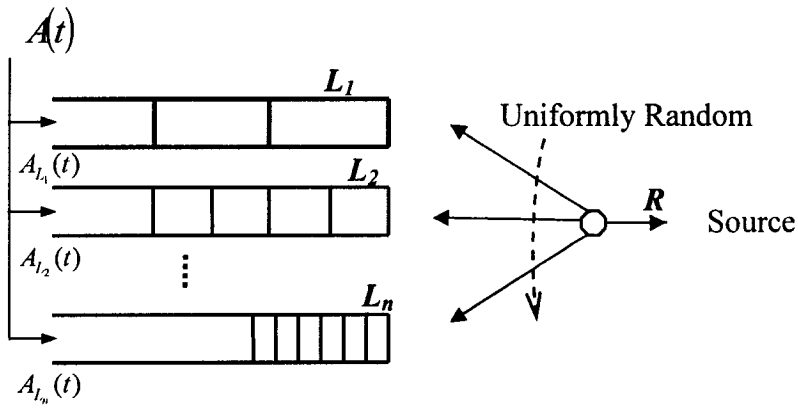


Fig. 3.11 Variable length packet generation scheme.

In this study $f_P(L)$ has been considered as being the packet length distribution of IP traffic in the network. IP traffic usually contains a number of more frequent packet length and their associated distribution. Using data from CAIDA (http://www.caida.org/analysis/AIX/plen_hist/) a sample packet length distribution has been used for all the simulations as follows:

$$f_p(L) = \begin{cases} 0.18, L = 1500 \quad \text{Bytes} \\ 0.18, L = 576 \quad \text{Bytes} \\ 0.18, L = 552 \quad \text{Bytes} \\ 0.408, L = 40 \quad \text{Bytes} \end{cases}$$

(3.18)

The majority of the packets seen in (3.18 are one of three sizes due to the way common TCP implementations divide a data stream into packets:

- 40 byte packets (the minimum packet size for TCP) which carry TCP acknowledgements but no payload.

- 1500 byte packets (the maximum Ethernet payload size) from TCP implementations that use Maximum Transmission Unit (MTU).

- 552 byte and 576 byte packets from TCP implementations that do not use MTU.

In addition most of the data is carried in packets of size 1500 bytes or more.

The served packets are the packets that build the traffic being fed into a switch or a network. These packets are stored in a file along with their generation time and packet length. Random destinations could be assigned to these packets according to the number of destination ports. It is well worth mentioning that for most switch implementations, every packet would have to be segmented into a fixed size packet (a cell) before entering a switch. The packet segmentation scheme is a topic of further work.

This packet generation scheme actually produces an ON-OFF process. A histogram of the ON-times is shown in Fig. 3.12 for the trace with $\rho = 0.8$ and $H = 0.95$. The histogram is consistent with the Pareto density function by visual inspection. However, it is important to note that packet trains in this case not only mix variable length packets but these packets have, in general, different destinations. The traffic is therefore more accurately describes an aggregation of traffic sources.

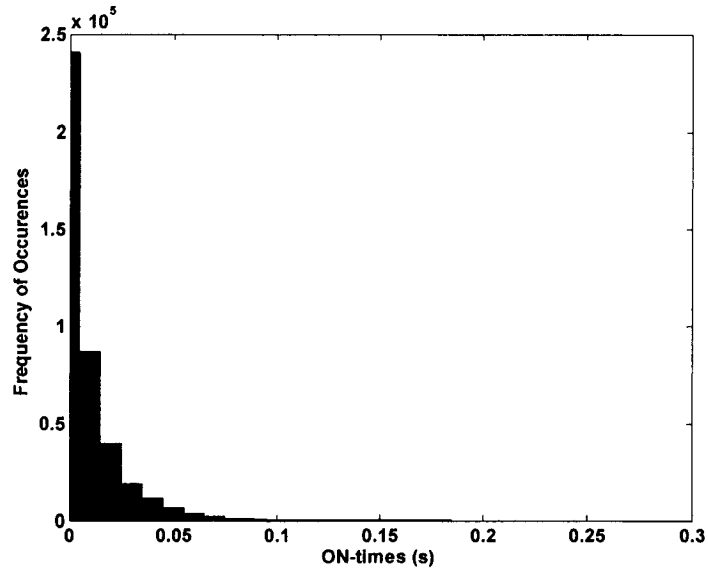


Fig. 3.12 Histogram of ON-times for a load of 80% and H=0.95, n=4096 samples

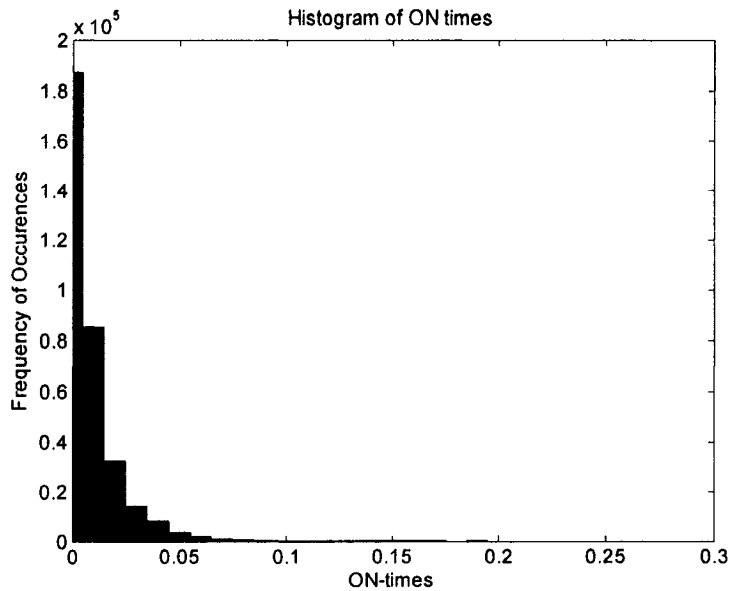


Fig. 3.13 Histogram of ON-times for a load of 60% and H=0.95, n=4096 samples.

Fig. 3.13 shows the histogram ON-times for a lower traffic load than for Fig. 3.12. The difference with Fig. 3.12 is in the frequency of occurrences, which is fewer. This means that the higher the traffic load, the more packet trains present in the traffic.

Therefore Load (ρ) is the main parameter effecting the shape of the histogram. Hurst parameter will mainly affect the number of sample points truncated to keep the trace within limits, ($0 < a(t) < R$).

The statistics of the generated packets are collected in a file to estimate the packet length distribution. The pie plot of Fig. 3.14 shows the result of the statistics and confirms that the packets were actually generated according to the desired distribution.

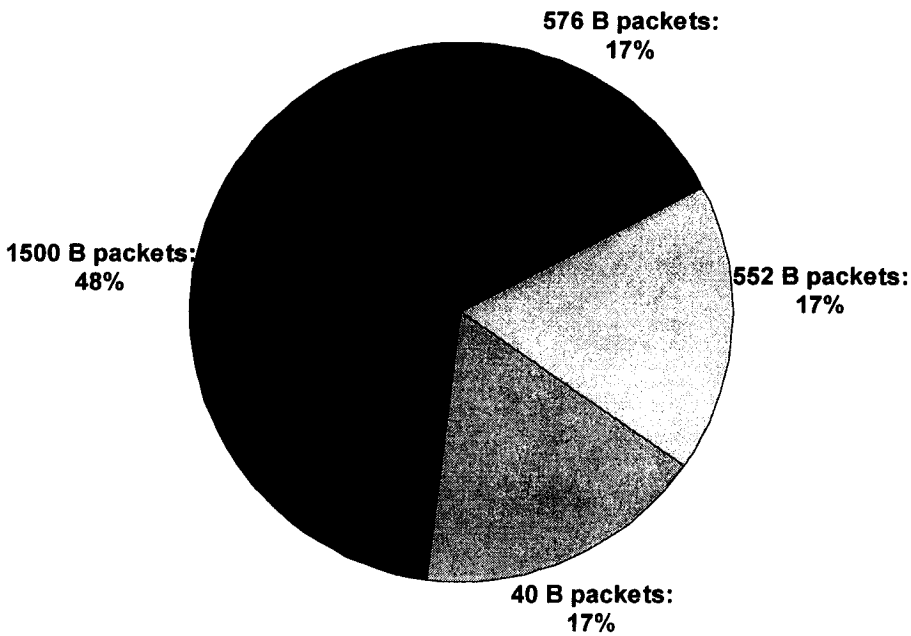


Fig. 3.14 Packet length distribution of generated variable-length traffic

Having generated packet traces with self-similar statistics, the actual test bed for various switching architectures is now obtained which allows for bursty traffic delay measurements and buffer occupancy observance of switches. In the following chapter, we'll be evaluating the performance of a 3-stage packet switch using the fixed-length Pareto packet generation with various statistics.

4. Clos-like Packet Switch with an Optical Core

This chapter mainly evaluates the performance of a hybrid Clos-like packet switch architecture. The chapter starts by touring the evolution of packet switches and the position of optics in packet switch architectures. After that a three-stage hybrid switch architecture is described which exploits the advantages of electronics as a memory technology and photonics as a communication technology while accommodating the slow reconfiguration times of the optical center stage. This architecture and method are later verified by simulation using a self-similar Pareto generated traffic and the switch performance in terms of queue length and delays is inspected.

4.1 Historical Evolution of Switches and Role of Optics

Packet switching in general involves two tasks, scheduling and data-forwarding. Scheduling means selecting the packet to be sent among the packets destined for the same destination during one time-slot. Data forwarding comes after a packet has been scheduled and consists of means of delivering a packet to the output port. Packet switches may become source of bottleneck in a packet switching network due to speed and buffer capacity limitations. Various architectures have been conceived to overcome these issues. The *ideal output queued switch (IOQS)* is the ideal architecture to perform

switching the best possible way.

This architecture shown in Fig. 4.1 consists of N input ports, a switch fabric (crossbar), N output ports and N output queues, one for every output port. There is no input queueing, but the switch fabric runs N times faster than the incoming links, making it capable of transferring up to N packets from the incoming links to the output queues in each time slot. This switch architecture is impractical due to limitations in switch fabric speeds [CHUA99]; but it has the lowest delay possible [HLUC88] and a 100% throughput for all kinds of traffic [KARO87], providing the reason to use its performance as a point of reference.

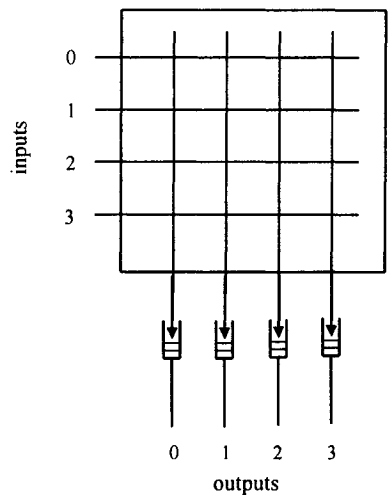


Fig. 4.1 Output queued packet switch

To overcome the impracticality of IOQS in terms of switch fabric speed, the queues can be placed in the input and the switch operates at line rate. This is called the *input queued switch.* The problem with this switch is that packets arriving at the same input destined to different outputs would be able to proceed to the desired output only in turn, even if their output port is free. This is called *Head of Line Blocking (HOL)*, and should be avoided because it decreases throughputs greatly [KARO87]. To avoid the head of line (HOL) blocking, *virtual output queues* (VOQ) may be deployed [ANDE93]. This method for input queueing consists of maintaining one queue for each output at each input. This method is practical with the current low memory cost.

Another important non-blocking switch architecture is the *Clos switch* [CLOS53] which is a re-arrangeable three stage circuit switch built from smaller switching nodes. It is non-blocking if it is capable of setting a circuit between any input-output port pair at any time. For a *strictly non-blocking* switch, the condition $2n-1 \leq m$ must be satisfied, where n is the maximum number of inputs to the switch and m is the number of central stage switches. Also the Clos switch is said to be *re-arrangeably non-blocking* if a connection can be made between any pair of idle ports if existing connections are suitably re-arranged. The condition for this is $n \leq m$ as stated in [BENE65]. Some of the principle of Clos switches may also be applied to packet switches.

An issue requiring consideration is the size of the switch fabric. Most of the current switches use several single-stage switching technologies. Single-stage switches are quite simple and usually deploy electronic technologies. In order to increase the switch size, the size of the basic switch element must be increased. This would result in a cooling limitation imposed by the high density packaging and low power consumption. The solution would be the use of multistage Clos switch architectures. This approach expands the switch size easily in a distributed manner.

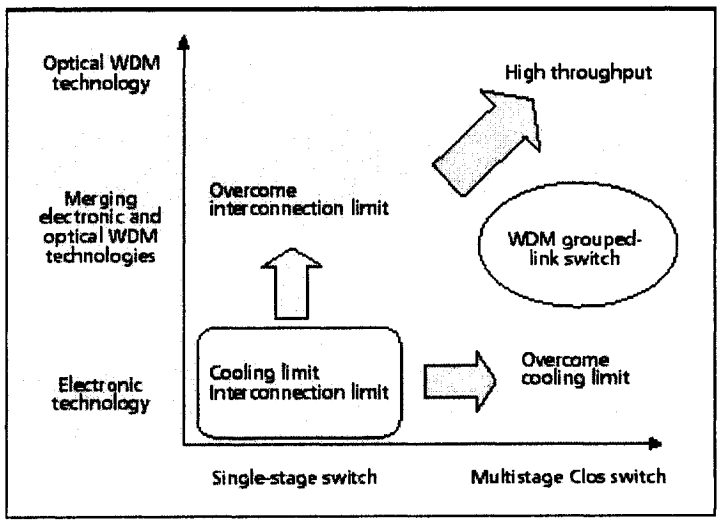


Fig. 4.2 Approaches for single-stage and multi-stage Clos switches [OKI03].

Another limitation present for both the single-stage and multi-stage switch architectures is the interconnection between different switching devices. As the switch size and port speed increases, a larger number of high-speed signal interconnections are required. To solve this problem, optical wavelength division multiplexing (WDM) for the interconnection between basic switches can be used which simplifies the interconnection system between basic switches. Fig. 4.2 shows the single and multi-stage switching approaches taken from [OKI03].

On the other hand, the widespread deployment of optical fiber links and WDM networks has motivated research into the role that photonic technology can play within routers that exist at the nodes of a packet switched network. A router consists of a packet switch which is responsible for switching packets from incoming links to outgoing links. Today the packet switches are basically implemented all-electronically. Electronic switches are opaque but offer fast switching and extensive buffering. Optical switches on the other hand are transparent and offer high data rates, but are very difficult to buffer and slow to reconfigure. It is therefore an advantage to deploy optics as transportation means and electronic technology for memory purposes.

In what follows, a hybrid 3-stage Clos-like switch architecture with an optical core will be described which benefits from both optical and electronic technologies. The switch performance in terms of queue length and delays will also be shown under self-similar traffic conditions.

4.2 Hybrid Packet Switch Architecture

The three-stage packet switch architecture described in this section consists of a reconfigurable optical interconnection as the central stage placed between two electronic buffering stages which have been *sectored* to alleviate memory access contention as shown in Fig. 4.3. The first stage of the architecture contains l centralized shared memory packet switch modules, referred to here as *input sectors*, that switch and buffer packets between n ingress and m egress ports. The third stage contains l *output sectors*, switching and buffering packets between m ingress and n egress ports. The

buffering stages are suited for electronic implementation. The overall switch dimension is $N = n \times l$. The central stage consists of m crossbar switches of dimension $l \times l$ with no buffering, therefore suitable for implementation in photonic technology. The packets are assumed to be fixed-sized and a time slot is equal to the duration of a packet. All links transport one packet per time slot, therefore speed-up is provided spatially and is equal to $k = m/n$, i.e. there are kN paths available in the central stage.

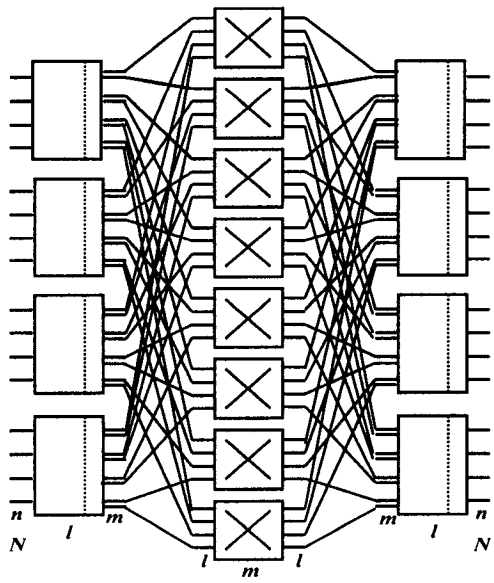


Fig. 4.3 Sectored packet switch architecture with an optical core

The input sectors maintain Virtual Output Sector Queues (VOSQ); i.e., one queue for each output sector within each input sector. The number of queues in each input sector is therefore l . An example of the input sector is shown in Fig. 4.4. Queue Q_{ij} is the VOSQ for traffic going from input sector i to output sector j.

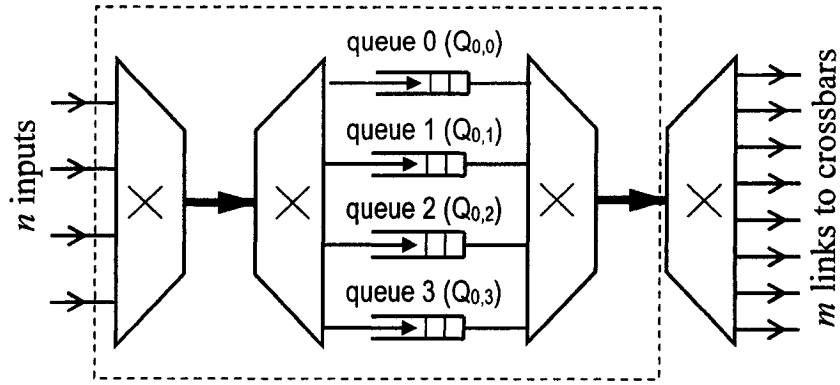


Fig. 4.4 Input sector 0, a_0, for the switch in Fig. 4.3. N=16, n=4, l=4, m=8.

The output sectors maintain Output Queues (OQ); i.e., one queue for each of its output ports within every output sector, therefore maintaining n queues for every output sector. An example of the output sector is shown is Fig. 4.5. Q_q is the dedicated queue for traffic going to output q, $0 \le q \le N-1$. The total number of queues to be managed in the switch is therefore $l^2 + N$.

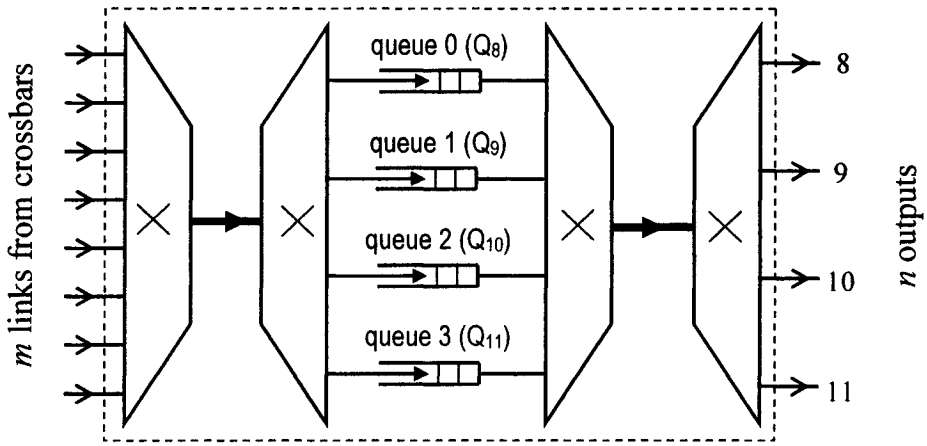


Fig. 4.5 Output sector 2 for the switch in Fig. 4.3. N=16, n=4, l=4, m=8.

The central stage consists of optical cross-connects which have the ability to reconfigure as desired. The sectors are equipped with opto-electronic interfaces that

convert electrical signals within the sectors to optical signals within the central stage and vice versa. The use of optics enables the switch to be scaled beyond the limit determined by memory addressing constraints (e.g. a single sector). It also suggests how the same architecture in distributed form is equivalent to a network with an optical core and electronic edge nodes. Clearly, the third stage will receive the optical signals, convert them to electronic domain and store them in buffers, ready to be served. The opto-electronic conversion is crucial, because practical optical memory and processing technology is not available.

In the next section, an algorithm will be reviewed based on [PARE05] which changes the configuration of the central optical cross-connects according to traffic demands for every output port. This algorithm and scheduling policy in the centre stage will eliminate internal blocking of packets in the switch to provide performance close to that of IOQ switch.

4.2.1 Flexible Bandwidth Provision

To accommodate all traffic demands at the input ports without any bottlenecks inside the switch, a *Flexible Bandwidth Provision* method is used that redistributes the available paths in the optical interconnect to form higher (or lower) capacity logical connections between different sectors as dictated by the incoming traffic. The switch is represented by a regular bipartite graph in which the input and output sectors correspond to the vertices and the edges represent the interconnections between them. The graph is shown in Fig. 4.6 and is built from an *intersector service matrix*, S, whose integer entries specify the bandwidth that will be provided between input/output sector pairs, measured in number of paths between them. The sum of each row and column must equal the number of available cross-connects m. By varying the number s_{ij} of paths between input sector i and output sector j, Flexible Bandwidth Provision (FBP) is achieved.

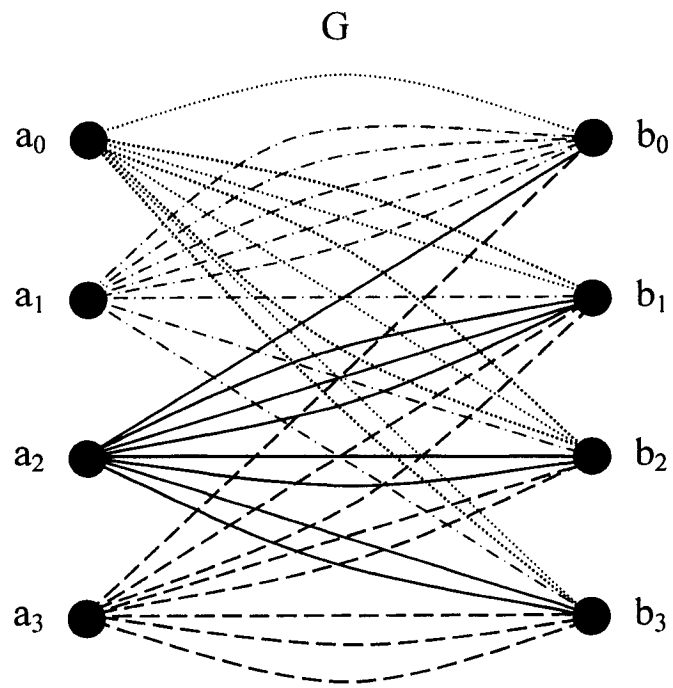


Fig. 4.6 Representation of the switch with a regular bipartite graph

The inter-sector service matrix is calculated from the *estimated intersector traffic matrix*, whose elements correspond to the number of packets to be transported between input sector i and output sector j. The traffic matrix may be derived from rate requests or estimated traffic arrivals. To achieve 100% throughput, it suffices to ensure that the service rate of the VOSQs (set by the configuration of the central stage crossbars) exceeds the arrival rate (set by the traffic statistics). In this context, switching of packets occurs within the sectors, while the central stage only transports packets across the switch.

Below is an example of traffic matrix Λ_e obtained for $l=4$, $n=4$ *and* $\tau= 20$, where τ is the inter-configuration period, and the incoming arrivals are measured after every τ time-slots:

$$\Lambda_e = \begin{bmatrix} 0 & 3 & 17 & 0 \\ 90 & 0 & 1 & 2 \\ 12 & 2 & 9 & 1 \\ 0 & 4 & 0 & 16 \end{bmatrix}$$

The service matrix calculated for $m = 8$ from Λ_e according to the algorithm in [PARE05] will be:

$$S = \begin{bmatrix} 1 & 2 & 3 & 2 \\ 5 & 1 & 1 & 1 \\ 1 & 3 & 2 & 2 \\ 1 & 2 & 2 & 3 \end{bmatrix}$$

The configuration of paths in the central stage is now found by solving an edge-coloring problem on the bipartite graph defined by the service matrix and shown in Fig. 4.6. The matrix S is decomposed into m permutations. Using these permutations, the m switches in the central stage and the read operations in the input sectors are configured [PARE05].

4.3 Performance Evaluation of FBP Switch

Having introduced FBP packet switch, it is now appropriate to consider the evaluation of the performance of this switch under self-similar traffic conditions which is one of the objectives of this thesis. The Pareto traffic generation method is used here to feed packets into the switch ports for testing the switch performance. The aim is to show how the number of paths (bandwidth) provided between sector pairs with the FBP algorithm adapts to the changing traffic statistics to obtain 100% throughput and how, as a consequence of this adaptation, the delays are kept close to the ideal performance.

The performance of the sectored switch architecture with the FBP method is compared mainly to the performance of an ideal output queued switch. The simulations emulate a delay between the time when the traffic is estimated and the time when

reconfiguration is applied. This delay has been made equal to the inter-configuration period, although it could be different. The buffer size of the sectors has been set to infinity with the aim of obtaining measures of average delay that have not been distorted by packet loss.

The following results were obtained for a *64x64* switch with *8* sectors, *8* inputs/outputs per sector and *16* switches in the central stage, hence a speed up of two. The performance of this switch has been compared to an ideal Output Queued Switch (OQS), which has been simulated with a *64x64* centralized shared memory switch.

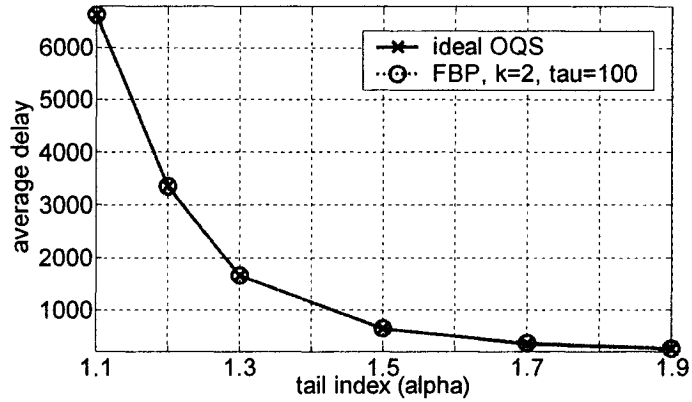


Fig. 4.7 Steady state average delay vs. tail index for 90% load and β_{on}=1

Fig. 4.7 shows the steady-state average delay versus tail index α. Load is 90%, $\beta_{on} = 1$ and τ=100 which emphasizes that even for high values of burstiness ($\alpha \to 1$) and a high load, the switch performance is very close to the OQS switch, with a difference of only 1 to 2%.

The delay performance of switch versus load is shown in Fig. 4.8 for various values of α, where increasing the load will increase the average delay in an exponential fashion. The effect of burstiness of traffic should be noted as well; given that train sizes are longer as α shifts toward unity, the average delay will increase as a result of the backlogs in the output sector queues.

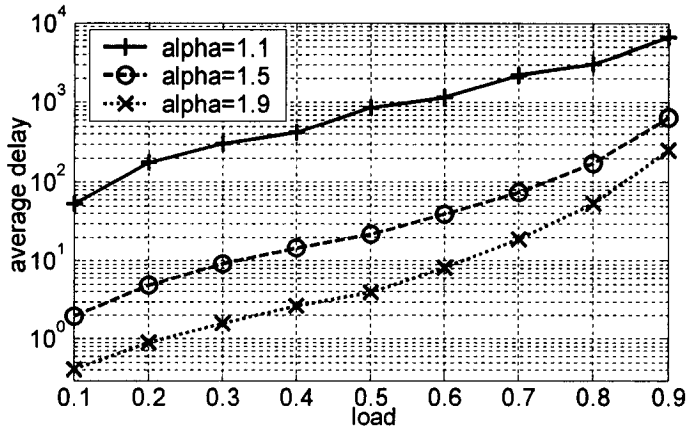


Fig. 4.8 Steady state average delay vs. load for various values of α and β=1.

The effect of the parameters α, the tail index and β_{on}, the minimum burst size, for a constant load of 80% is shown in Fig. 4.9, where an increase in the tail index toward the maximum limit of 2 provides less bursty traffic and therefore less delays. Considering the definition of β, larger values of this parameter will cause greater delays due to the backlogs caused by longer packet trains.

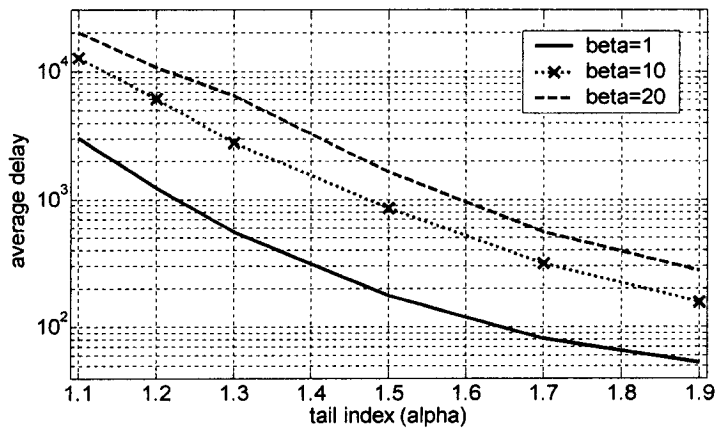
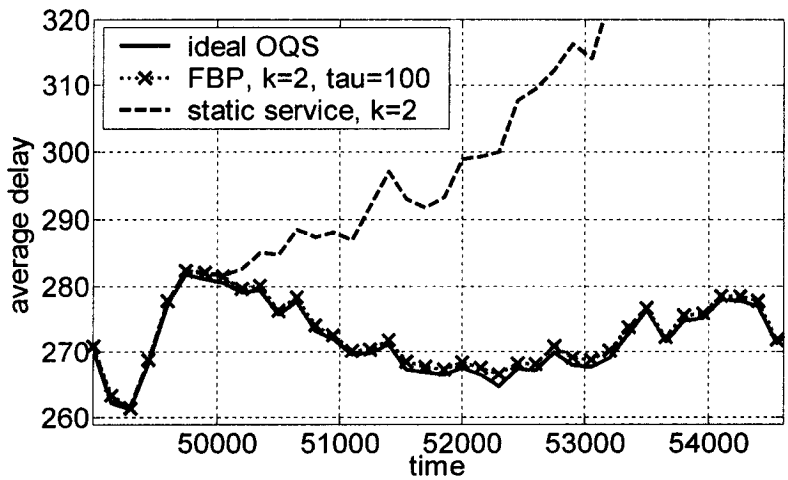


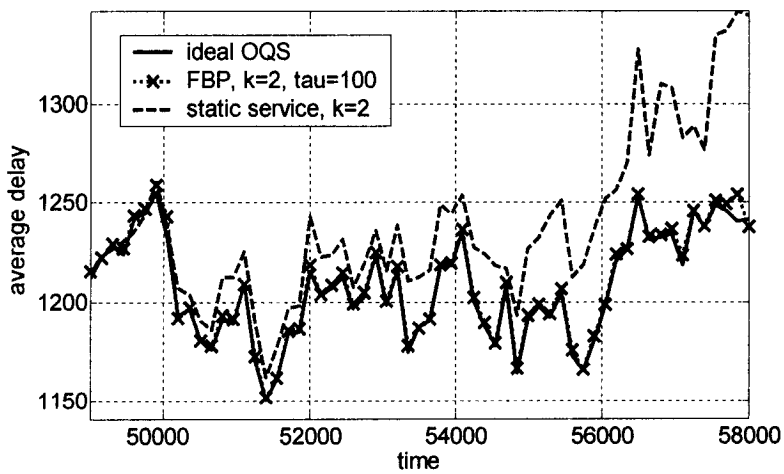
Fig. 4.9 Steady state average delay vs. α for load of 80%.

The transient response of the switch is shown in Fig. 4.10 for a change in traffic destinations distribution at *t=50000* and an inter-configuration period of *τ=100* timeslots. With FBP, the configuration of the central stage changes to adapt to the new traffic matrix and the delay remains close to the ideal performance. These figures also

show the performance of a switch with the same architecture but with a static service matrix; i.e., the configuration of the central stage does not change. With no adaptation, the queues in the input sectors will build up, resulting in an ever increasing delay. In Fig. 4.10.a the burst sizes are smaller due to larger α and therefore adaptation of the FBP switch to changes in traffic statistics happens in less time. On the other hand, in Fig. 4.10.b, the FBP switch takes more time to adapt to changes in traffic and follow the ideal OQ switch; this is due to burstier traffic, where the queues take time to empty and therefore the change in center stage will not take effect until the backlog of previous traffic arrivals has been served.



(a)



(b)

Fig. 4.10 Transient response for 90% load, βon=1. a) αon= αoff =1.7, b) αon= αoff =1.3

The delay measurements obtained by simulation show that, at the modest price of a speed-up of two, it is possible to obtain a performance practically equal to that of the "ideal" architecture even for high loads and high degrees of burstiness. Moreover, reconfiguration does not need to be on a per-packet basis.

Cells 512 bit long transported at a rate of 2.5Gbit/s yield a 204.8 ns time slot. An architecture with n=16, m=32, l=16 is feasible with today's memory bandwidth (80Gbit/s) and optical space switch dimensions (16x16). These numbers translate into an overall switch capacity of 640 Gbit/s. Higher capacities may be achieved by employing this switch as the sectors of a larger switch with the same architecture.

To sum, a 3-stage hybrid switch architecture is the solution which exploits the advantages of electronics as a memory technology and photonics as a communication technology while accommodating the slow reconfiguration times of the optical center stage by implementing suitable scheduling algorithms. The advantage of this hybrid optics/electronics implementation is that packets do not contend over resources such as shared buses. There is a penalty paid for the hybrid architecture and that is the repeated optic-to-electronic and electronic-to-optic conversions. However, development of recent optical technologies like the VCSEL technology would bring the cost down. The next chapter will introduce an alternate means of accommodating traffic demands in the switch inputs without packet contention in any section of the switch aside from using the concept of Load-balancing in switches.

5. Load-Balanced Switches

Load-balancing in general means distributing, processing and communicating the traffic load evenly across a device so that no single section is overwhelmed, especially where it is difficult to predict the number of requests that will be issued to a server. Load-balancing can therefore be applied to the network switching devices to avoid congestion in any section of a switch.

This chapter will briefly review the prior art of load-balancing switch architectures and then introduce a new method for load-balancing without miss-sequencing that builds on earlier research on Clos-like packet switches which are suitable for implementation in hybrid electronic / photonic technology.

5.1 Review of the Prior Art of Load-Balancing

Earlier work reported in Chapter 4 demonstrated how memory access contention may be alleviated by partitioning memory into input / output sectors and providing variable bandwidth between the sectors according to the traffic demand by reconfiguration of the photonic centre-stage switch elements in a Clos-like architecture. An alternative to reconfiguration is to balance the load by evenly distributing the traffic across the switch ports by, for example, preceding the switch with a crossbar performing a cyclic shift. However, this introduces miss-sequencing of packets within flows, which is undesirable. Solutions to this problem [KESL02] [CHAN02b] involved impractical

complex scheduling algorithms, until the appearance of the work [KESL03a], [KESL03b] and [ASLA04] reviewed in the following subsections.

5.1.1 Load-Balanced Birkhoff-Von Neumann Switch

The load balanced Birkhoff-von Neumann switch [CHAN02a] consists of two identical switching stages which surround a middle stage of buffering (Fig. 5.1). These buffers are partitioned into N separate First-In-First-Out (FIFO) queues at each intermediate input, one buffer per output, and are called Virtual Output Queues (VOQ). VOQs prevent head-of-line blocking [KARO87]. The first stage performs load balancing and spreads traffic over all VOQs, while the second stage is an input-queued crossbar switch in which each input is served at a fixed rate. Each switching stage is called a *fixed, equal-rate switch.*

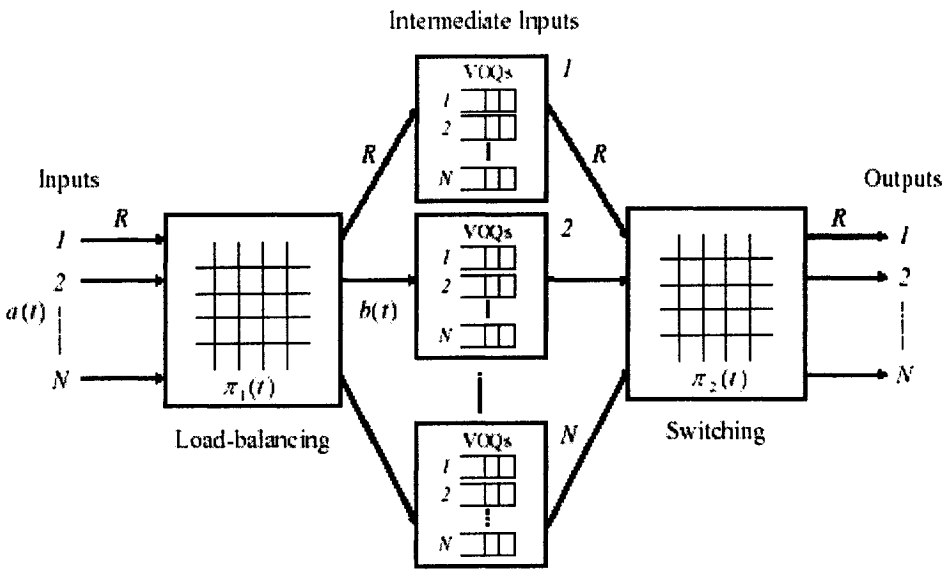


Fig. 5.1 Load balanced Birkhoff-von Neumann switch

The switching stages are not configured according to the queue occupancies, but both switching stages walk through a fixed sequence of configurations. This means that

at time t, input i of each switch fabric is connected to output $[(i+t) \bmod N]+1$. Therefore the configuration is a cyclic shift, and each input is connected to each output exactly $1/N^{th}$ of the time, irrespective of arriving traffic.

When a packet arrives at the input of the first stage, this packet is immediately switched to one of the intermediate inputs and placed in a VOQ according to its destination port. The intermediate input that the packet will go through depends on the current configuration of the load-balancer. This packet will then be switched to the appropriate output by the second fixed, equal-rate switching stage.

With the function described, the load balanced Birkhoff-von Neumann switch now has the following advantages:

Scalability: The on-line complexity of the scheduling algorithm in the switch is $O(1)$ since the scheduling policy is simply periodic.

Low hardware complexity: The switch fabric requires two crossbars and buffers between them. These two crossbars can be built using Banyan networks. The switch does not need internal speedup or rate estimation inside the switch.

Low average delay in heavy load and bursty traffic: The load-balancing stage of the switch has not only proven to convert non-uniform traffic into uniform traffic, but also achieve burst reduction in the incoming traffic. This is very effective in reducing delay, and the average delay of the load-balanced Birkhoff-von Neumann switch has proven to converge to that of an output-buffered switch under heavy load [CHAN02a].

Efficient buffer usage: The reducing bursts in the input traffic permits smaller buffer sizes to be employed and more efficient buffer usage. It has been shown that when both this load-balanced switch and the corresponding output-buffered switch are allocated the same finite amount of buffering per port, the packet loss probability is much smaller for the load-balanced switch than that in an output-buffered switch when the buffers are large [CHAN02a].

Aside from the advantages, there are four main problems to tackle with the load-balanced Birkhoff-von Neumann switch as follows. (1) The switch fabric has to be rapidly reconfigured, making it difficult and expensive to use an optical switch fabric. This problem is important in implementing the load-balanced Birkhoff-von Neumann

switch in optics. (2) The basic Birkhoff-von Neumann switch has the problem of miss-sequencing packets, because it spreads packets evenly to the intermediate stage despite their arrival ports. Therefore, packets from one flow will be sent to two different intermediate VOQ stages with different occupancies and might not arrive in the right order. A flow in this context means all packets arriving at one input port and destined for the same output port. This can only be corrected by placing re-sequencing buffers at the output ports which introduces additional and unbounded delay. Re-sequencing queues are not work-conserving and hence degrade the delay performance relative to an Ideal Output Queued Switch. (3) Pathological periodic traffic patterns can make throughputs randomly small. (4) The switch does not work when some intermediate stages are missing or have failed. Solutions to each one of the mentioned problems have been investigated in [KESL03] to identify an architecture with predictable throughput and scalable capacity which avoids packet miss-sequencing. The architecture may be folded into a single electronic stage interconnected via an optical transpose interconnection, which is made reconfigurable to provide restoration against failure and operates at a timescale commensurate with available optical switch technology. To alleviate miss-sequencing of a packet, the queues in the input sector are organized as Virtual Output Queues (VOQ) which are scheduled using a Full Ordered Frames First (FOFF) algorithm. FOFF bounds the different in lengths of VOQs in the second stage hence bounding the delay due to resequencing which is provided by finite length re-sequencing buffers at the third stage.

The basic idea of FOFF is that the input stage keeps N FIFO queues per port. When a packet arrives it is placed in the queue associated with its destination.. FIFOs containing at least N packets are served first in round-robin order... N packets are read from the FIFO being served and each packet is sent to a different intermediate stage. In this way, the packets are spread uniformly over the second stage and all the VOQs that receive packets belonging to a flow will face the same delay, and will not be miss-sequenced.

Miss-sequencing can happen when no buffer holds N or more packets; in this case non-empty buffers are served in round-robin order. The miss-sequencing is bounded and will be corrected in the third stage by the re-sequencing buffers that are organized as

virtual intermediate sector queues. Details are not provided on precise means of re-sequencing beyond proving that the re-sequencing backlog is bounded.

5.1.2 Parallel Packet Switch with Virtual Input Queues

The switch described in this section [ASLA04] builds on the idea of parallel packet switches [IYER03] and therefore the parallel packet switch (PPS) shown in Fig. 5.2 is described. It performs load-balancing by distributing packets to M internal switch planes at the input de-multiplexors in round robin fashion for every flow. The switch achieves no packet miss-sequencing by using Virtual Input Queues (VIQs) which sort packets into buffers by their source, therefore acting as 'come from' queues, and which provide the re-sequencing function in the output multiplexors. The internal switch planes deploy FIFO-OQs. Load balancing is provided by employing pollers to spread each flow across a layered co-ordination buffer with each layer corresponding to an internal switch plane.

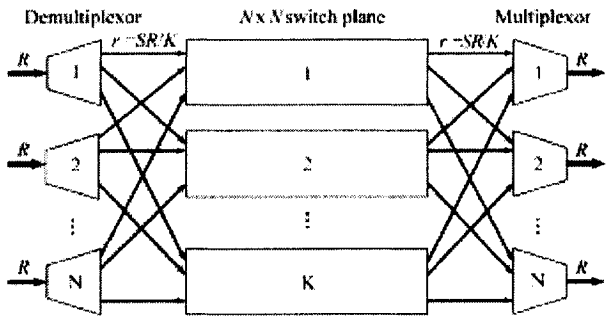


Fig. 5.2 Parallel Packet Switch Architecture

5.2 Load-Balanced Switch with Cross-point Queues

In this section, a novel load-balanced 3-stage Clos-like packet switch architecture and method is described [TAEB05] that was conceived prior to the appearance of the architecture and method described in [KESL03] and [ASLA04] which it includes as special cases. The early origins of the method may be traced to [HALL02]. The

architecture is derived starting from a centralised shared memory switch by expanding its 1D (one dimensional) output-queue structure into a 3D (three dimensional) layered cross-point queue structure that is distributed across sectored memory. This derivation renders trivial the proof of no miss-sequencing and provides intuition on approaching or even emulating ideal Output Queued switch (OQS) behaviour. This architecture is universal in the sense that, depending upon how it is controlled, it can emulate a wide range of architectures that have been hitherto considered distinct.

5.2.1 Architectural Derivation

The starting point is a centralised shared memory switch that behaves as an ideal output-queued switch. The first step is to transform the 1D output queue structure (i.e. one queue Q_j per output port j) into a 2D cross-point queue structure (i.e. one queue Q_{ij} per input-port output-port pair (i, j)). A suitable work-conserving scheduler; e.g., oldest cell first (OCF) or simple round-robin (RR), selects between packets at the head of cross-point queues sharing the same output port. For an OCF scheduler and FIFO cross-point queues this exactly emulates an ideal FIFO-OQ switch.

The 2D cross-point queues Q_{ij} are then expanded into 3D layered-cross-point queues Q_{ijk} with each layer indexed by k. The layers associated with a single cross-point queue are addressed by a pair of RR pollers. Arriving packets are appended to the queue pointed to by a first poller that is then advanced and packets depart from the queue pointed by a second poller that is then advanced. The pointers are not advanced if there is no arriving packet or packet at the head of the queue respectively. Initially both pollers point to the same queue and, as they cycle around the layers in the same direction, packets depart the layered cross-point queue in the same order that they arrive. The pollers therefore ensure that the layered cross-point queues behave logically as a single cross-point queue and that all flows are evenly spread across the layers.

The individual queues Q_{ijk} are then split into three parts, tail Q^1_{ijk}, body Q^2_{ijk} and head Q^3_{ijk} logically connected in series as shown in Fig. 5.3. The tails are organized into

groups of adjacent input ports and assigned to a dedicated input stage memory sector that serves those inputs. The bodies are organized into groups of adjacent layers and assigned to a dedicated intermediate stage memory sector that serves those layers. The heads are organized into groups of adjacent output ports and assigned to a dedicated output-stage memory sector that serves those outputs. The reorganization of queues introduces a transpose interconnection between the input and intermediate stages and the intermediate and output stages that preserves the 1:1 connection between the tail, body and head parts of the same cross-point queue. The tails, bodies and heads may be considered to form layered VOQs, layered cross-point queues and layered VIQs.

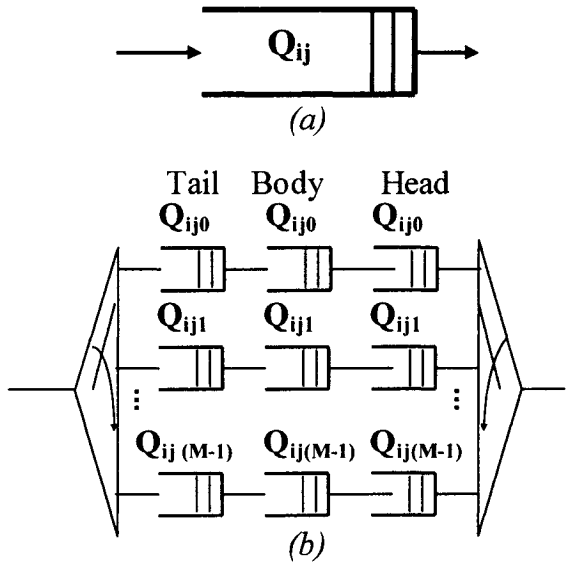


Fig. 5.3 The intuition of a load-balancing architecture by logically splitting a cross-point queue Qij (a), between switching stages and (b) in layers .

Without loss of generality, it will be assumed that there is one input port only associated with each input sector, one layer only associated with each intermediate sector, and one output port only associated with each output sector. The indices (i, j, k) then refer to sectors. The result is a three-stage architecture (Fig. 5.4) in which layered cross-point queues are distributed from input sector i to output sector j via intermediate sector k. The preservation of a 1:1 logical interconnection between the tail,

body and heads of the same layered cross-point queues and the action of the pollers to ensure that the layered cross-point queues behave logically as a single cross-point queue, guarantees the in-order departure of packets belonging to the same flow. The pollers ensure that the flows are evenly distributed between the layers and, as a consequence, across the physical links between the stages. The *N.M.N* logical links between stages can be shared using suitable work-conserving schedulers (e.g. OCF or RR) between the *N.M* and *M.N* internal physical links that run *M* times slower than the external line rate. The sectoring of memory alleviates memory contention thus enabling the switch to scale to higher capacities or alternatively provides the same capacity with lower cost memory.

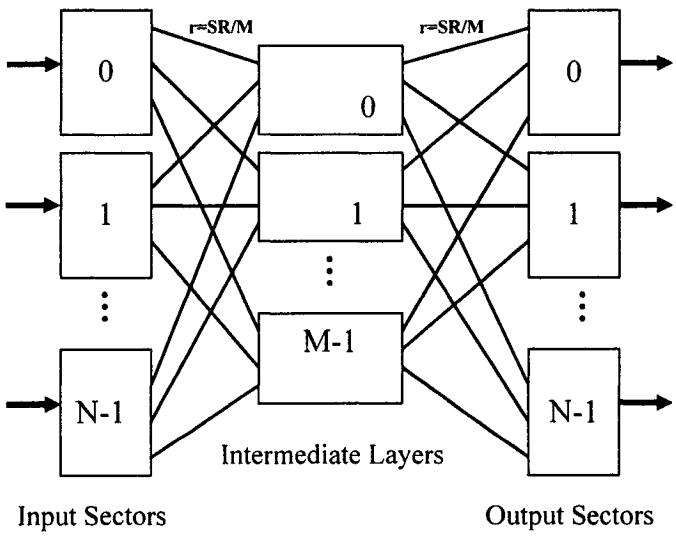


Fig. 5.4 Cross-point queue load-balanced architecture

5.2.2 Resultant Switch Architecture

As a consequence of the above, it is possible to think of a three-stage Clos-like packet switch with a buffered central stage as a parallel packet switch, i.e. a means of scaling-up the intermediate switch elements to higher line rates by connecting them in parallel. The switch contains *N* sectors in the input and output stages. The central stage contains *M* sectors. The external link rate is *R*. The external line rate is *M* times faster

than the internal line rate, resulting in an internal line rate of $r=R/M$. If an internal temporal speed-up S is used, then the internal line rate would become $r=SR/M$.

The main task of the input stage is to spread the traffic uniformly among the M sectors in the central stage. For this reason, each input sector contains M VOQs for every output port resulting in a total of $N.M$ queue tails as shown in Fig. 5.5. These collectively constitute a co-ordination buffer with a bounded backlog [KESL03] [ASLA04]. There are N pollers P_{ij}^{1} (layer pointers), one per set of layer-destination VOQs to indicate the queue where the next arriving packet should be placed. The pointer advances only after queueing a packet in the position indicated.

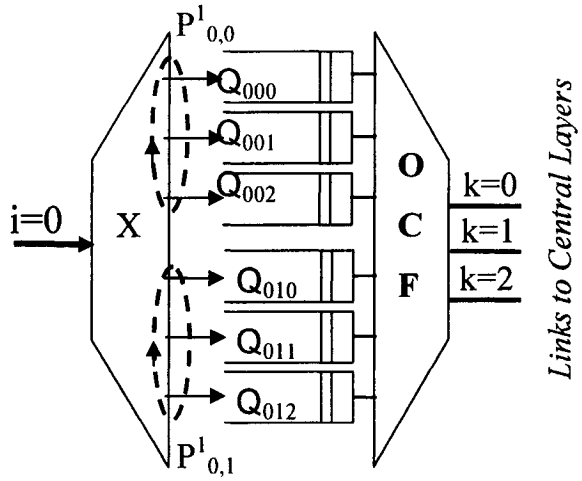


Fig. 5.5 Input sector i=0 (N=2, M=3)

Considering the action of the input sectors, the intermediate stage may be thought of as the 'load-balanced stage' where all the intermediate layers carry almost the same share of the traffic. Each intermediate sector contains $N.N$ cross-point queue bodies, one for every source/destination pair (Fig. 5.6).

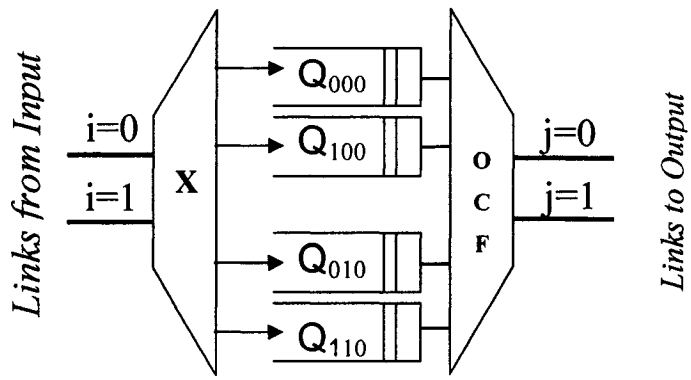


Fig. 5.6 Central layer k=0 (N=2, M=3)

There are M queue heads for each VIQ in the output stage. This results in a total of $N.M$ queues in each output sector (Fig. 5.7). These collectively constitute a realisation of a re-sequencing buffer with a bounded backlog [KESL03] [ASLA04].There are N pollers P_{ij}^3 (layer pointers) that point to the next packet in sequence and a scheduler (OCF, RR,...) selects among the pollers that point to a non-empty queue. A poller will advance only if it points to the packet that is selected by the scheduler.

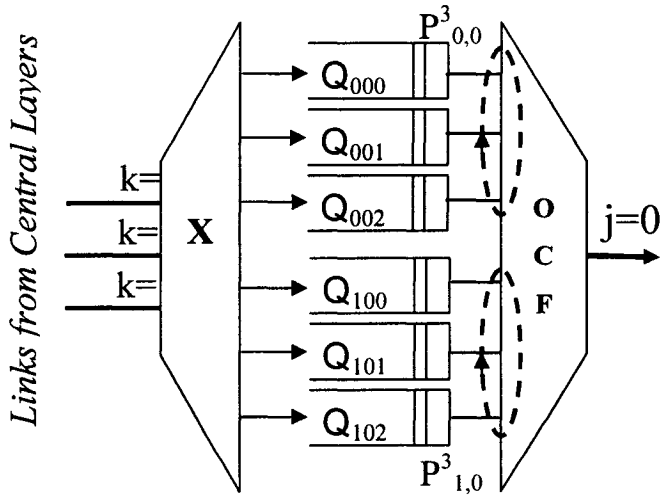


Fig. 5.7 Output Sector j=0 (N=2, M=3)

It is important to remark that if packets are transported across the inter-stage interconnects in a timely fashion, IOQ switch behaviour may be emulated. To allow packets with greater priority to be served first, they can be stamped on arrival. Within each sector, the packet with the earliest time to depart is served first among those ready to depart for the same sector in the following stage. If the arrival time is used as the stamp, this is equivalent to OCF scheduling. Whether a practical scheduler can be found that guarantees IOQ emulation is a topic of further research.

The queue structure and schedulers in the input sector and output sectors of [ASLA04] and [KESL03] may be considered as special cases of the structure described here. The input sector in [ASLA04] is functionally equivalent to the structure described here (layered-VOQs combined with an OCF scheduler); it is a consolidation into layer-only queues, which eliminates the need for the OCF scheduler. The input sector in [KESL03] is also functionally equivalent to layered-VOQs, but combined with an FOFF scheduler and hence it is a consolidation of the structure into VOQ-only queues.

The output sector queue structure in [ASLA04] corresponds to the polled layered-VIQs structure described here, but combined with a RR scheduler. An OCF scheduler has better delay performance. The layered-VIQ–OCF structure described here may be considered a specific implementation of the re-sequencing buffers of [KESL03].

The queues within the central stage layers are FIFO cross-point queues with outputs merged via a work-conserving scheduler. If the latter uses the OCF schedule, it is tempting to conclude that the result is equivalent to OQ enabling the consolidation of the layered-cross-point queues into layered-OQs as in [KESL03] and [ASLA04]. However, the cross-point queues with OCF and OQs are only equivalent when the time-stamp occurs on the packet arrival at that switch element. Here the time-stamp occurs when the packet arrives at the input sector. The cross-point queues with OCF therefore behave as push-in first out queues and permit packets delayed in the input sector configuration buffers to catch-up. Although the improvement with centre-stage cross-point queues in comparison with output queues appears marginal from simulations that measure only the average delay, the ability to catch-up may be crucial to the improvement of other performance measures and is likely to be essential to IOQ switch emulation.

5.2.3 Performance Evaluation

Simulation experiments have been conducted to evaluate the proposed load-balanced switch architecture using three traffic models, the *uniform / non-uniform Bernoulli traffic* and the *self-similar (Pareto) traffic* model. These traffic models consider fixed-sized packets with the packet length equal to an external time-slot. Bernoulli traffic corresponds to a simple Bernoulli process at each input port with destinations distribution either uniform or modulated by a non-uniform distribution. The latter method generates non-stationary admissible traffic with a non-uniform destinations distribution that changes approximately every T timeslots. Pareto traffic generation has been explained thoroughly in chapter 3.

The Cross-Point Queue load-balanced switch (CPQ-OCF) model has been implemented using C++. Infinite length buffers are employed throughout the switch simulation to enable an accurate analysis of packet delay undistorted by packet drops. The performance of CPQ load-balanced switch has been compared to that of an Ideal Output Queued Switch (IOQS). A comparison is also made with a simulation of the VIQ-PPS switch architecture described in [ASLA04].

All the results presented were obtained for a 16×16 switch with the following parameters, $N = 16$ and a speed-up $S = 1$ unless otherwise stated. For non-uniform Bernoulli traffic, $T = 16{,}000$. For self-similar traffic, $\beta_{on} = 1$ and $\alpha_{on} = \alpha_{off} = 1.4$. The performance metric mainly used is the average delay measured in external time slots.

Fig. 5.8 states the delay comparisons for the mentioned switches under uniform Bernoulli traffic for *M=10* central stages. The delay measurements obtained in this case are an exact match to the delays given in [ASLA04] and confirm the correct implementation of the PPS. The average delay difference between VIQ-PPS and CPQ-OCF arises only from the output stage delay difference as shown in Fig. 5.9. The expanded architecture of the Cross-point queues in CPQ-OCF allows for less re-sequencing delays in the output stage, thus upgrading the performance, making the switch less non-work conserving compared to VIQ-PPS.

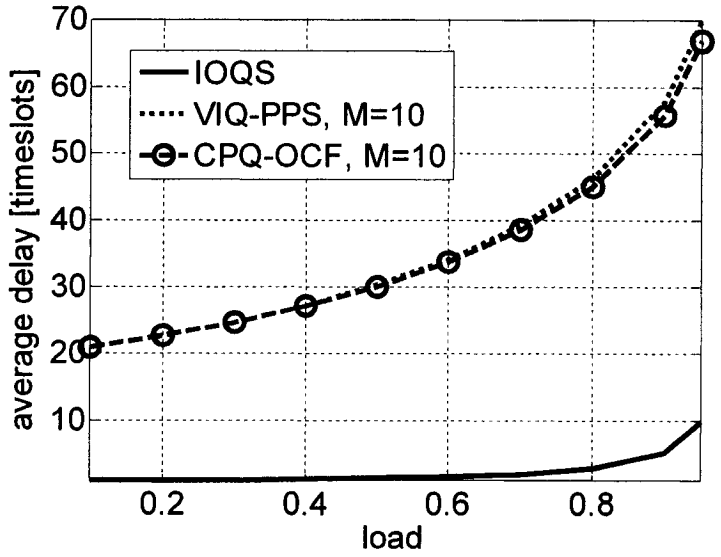


Fig. 5.8 Delay performance for CPQ-OCF for uniform Bernoulli traffic

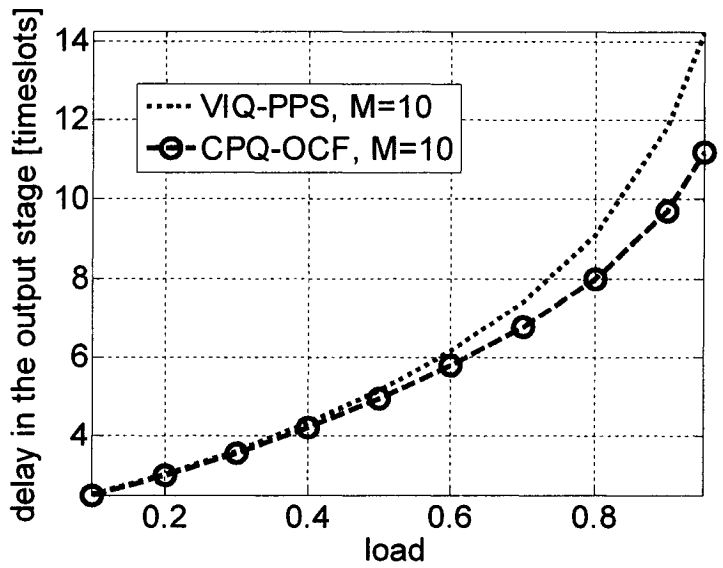


Fig. 5.9 Delay in the output stage for Uniform Bernoulli traffic

Fig. 5.10 illustrates the average delay of the switch under non-uniform Bernoulli traffic versus the offered load. This analysis shows that CPQ-OCF outperforms VIQ-PPS in terms of average delay. It is interesting to note that the performance superiority of CPQ-OCF is more noticeable for traffic with non-uniform destinations, emphasizing the action of this switch as a load-balancer under unbalanced input traffic conditions.

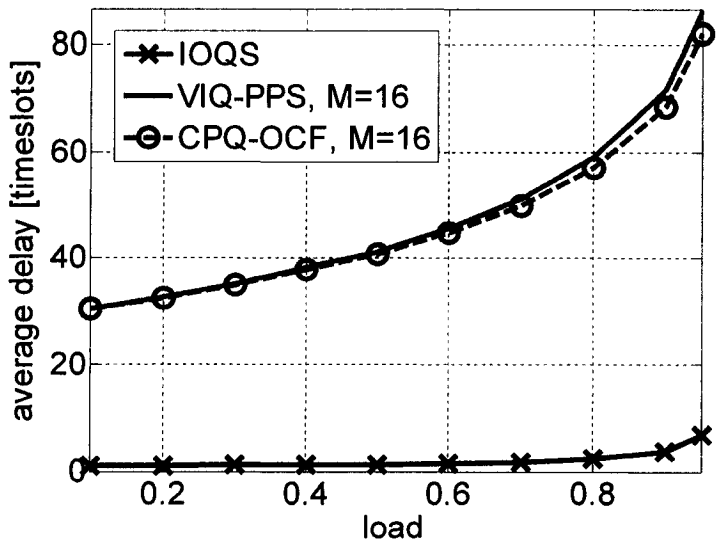


Fig. 5.10 Delay performance for CPQ-OCF for non-uniform Bernoulli traffic

The switch performs well and is stable under self-similar (Pareto) traffic as shown in Fig. 5.11. The larger magnitude of the delays in this plot is caused by the over-subscription of the queues for prolonged periods which is characteristic of self-similar traffic.

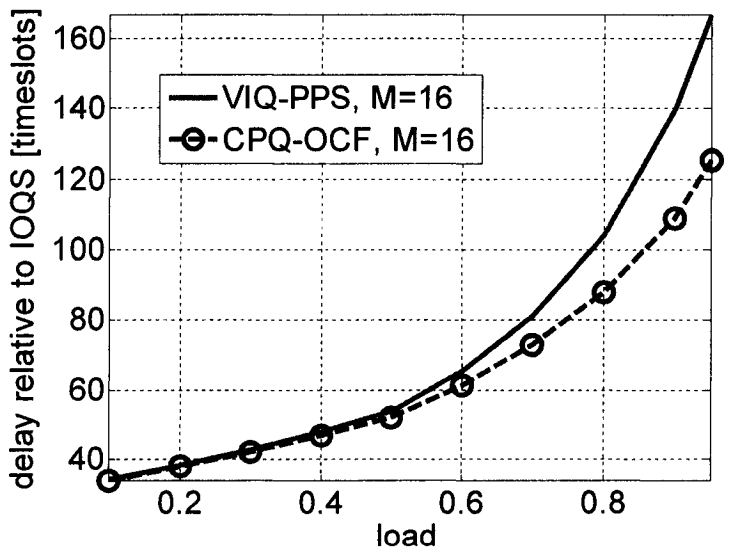


Fig. 5.11 Delay performance relative to IOQS for Pareto Traffic

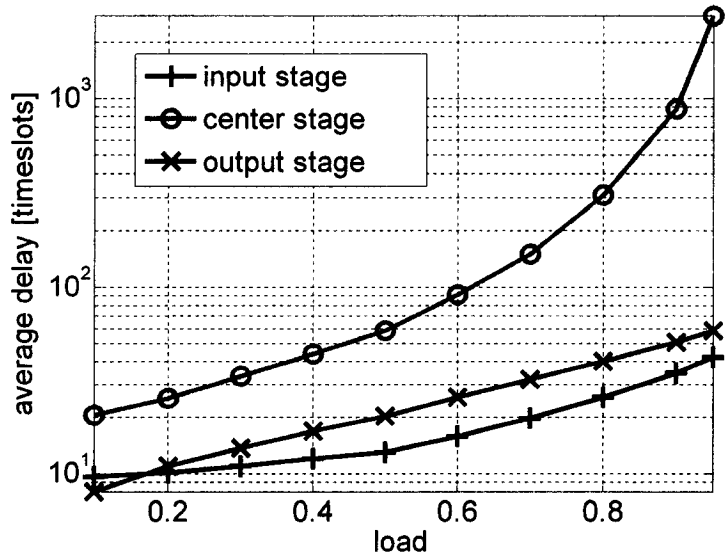


Fig. 5.12 Average delay of each switching stage for Pareto traffic

Fig. 5.12 compares the average delay in each switching stage, where the centre stage buffers provide the congestion queueing and the input and output stage buffers provide co-ordination and re-sequencing queueing respectively. Speed-up is kept at unity.

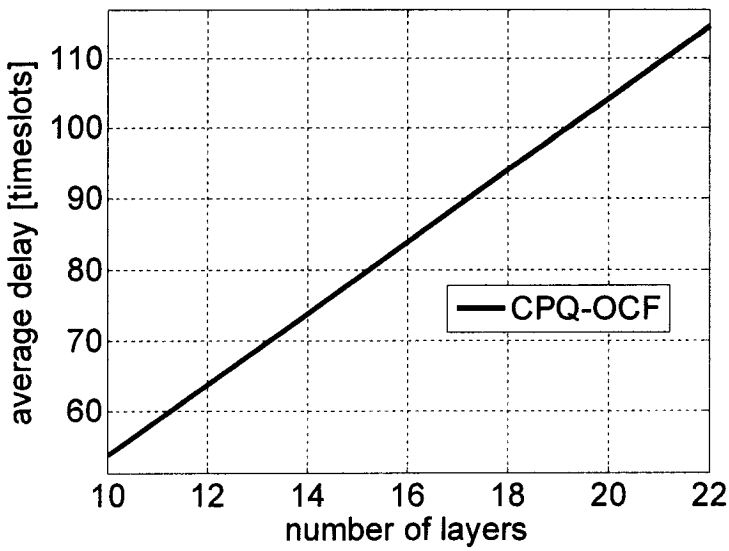


Fig. 5.13 Effect of change in the number of layers on average delay. $\rho = 0.9$, Uniform Bernoulli traffic.

Increasing the number of sectors (layers) in the centre stage permits the reduction of the internal line rate for the same speed-up. This expedient however increases the delay as shown in Fig. 5.13 and a compromise must be found. In the limit of one layer the architecture is nearly equivalent to an ideal output queued switch (the delay in the input and output stages become close to zero and the internal links run at the same speed as the external).

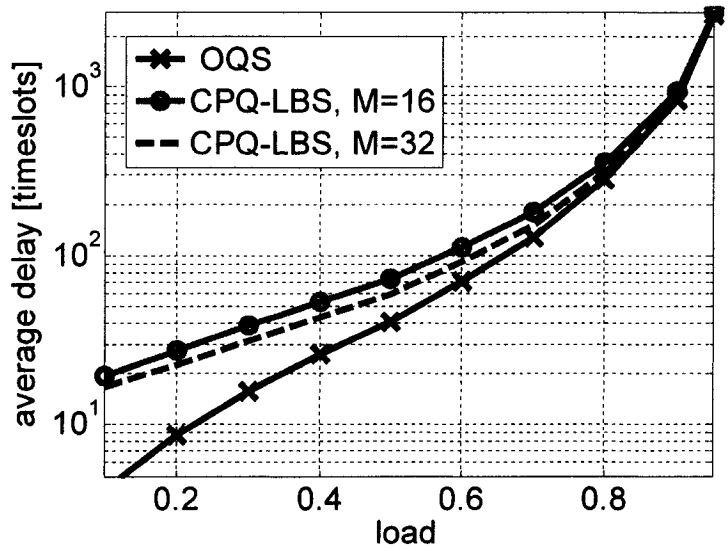


Fig. 5.14 Spatial speed-up experiment for CPQ-LBS using Pareto traffic

Fig. 5.14 shows the delay obtained with increased internal *spatial* speed-up, which is achieved by increasing the number of layers while keeping the internal line rate constant ($r=R/N$). The internal spatial speedup reduces average delays but with diminishing benefits beyond a speed-up of two ($M=32$). On the other hand, providing speed-up in this switch will move the congestion queues towards the output sectors, thus reducing the buffering requirement in center stage, alleviating the challenge of an implementation of the centre stage in photonic technology. This is to be further investigated.

Another measure of interest here is the relative delay of every packet entering the packet switch compared to that of the IOQ switch. This way a better measure of the switch performance can be conducted to observe the switch architectures in terms of

capability of emulating IOQS behaviour. To obtain this measure, every packet is stamped with an *ideal time to depart* from the IOQ switch. The same packet is departed from a different architecture and its departure time is calculated, and then subtracted from the ideal time to depart resulting in packet relative delay. A sample histogram of relative delays is shown in Fig. 5.15 for both CPQ-OCF and VIQ-PPS. The performance superiority of CPQ-OCF can be better visualized in this figure where the packets relative delays can be much larger for VIQ-PPS. The relative delays for CPQ-OCF are far more bounded around zero delay where the relative delays for VIQ-PPS can become very large. The main reason to the superiority of performance in CPQ-OCF is the use of Oldest Cell First scheduling scheme which compared to Round Robin facilitates the packets to reach the destinations on time and therefore speeds up re-sequencing of packets in the last stage.

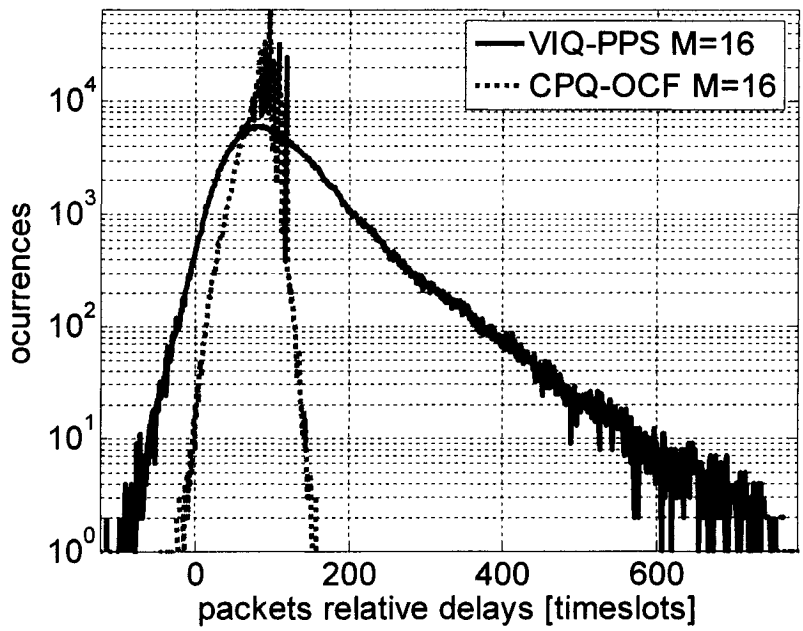


Fig. 5.15 Histogram of individual delays relative to IOQS for $\rho = 0.9$, and 100,000 sample packets

5.2.4 Load-balancing Switch Properties

A load-balanced switch architecture with distributed cross-point queues has been described. The key properties of this architecture are:

1. Packets within a flow are transported to their correct destinations in sequence.
2. 100% throughput is guaranteed even for very high loads and bursty traffic with a buffer requirement only greater than an IOQ switch by a modest constant bound.
3. The load is distributed equally well among layers providing load-balancing in the center stage.
4. There are no central schedulers; making the switch less complex and easier to implement without a central point of failure.
5. The switch may be made survivable; i.e. in case of a failed layer, the packets can be distributed among the remaining layers. Therefore, the pointers will skip over queues labeled by the failed layer and only serve the remaining queues.
6. This architecture is universal in the sense that, depending upon how it is controlled, it can emulate a wide range of architectures that have been considered distinct up to now.
7. It has been shown how the queues may be consolidated into a smaller number of queues in certain special cases. However, it may be that the full advantage of the architecture will only be manifest using schedulers that require the full 3D queue structure. The complexity of managing a very large number of short queues relative to a smaller number of longer queues must then be addressed.

6. Conclusions and Future Work

The internet and computer networks are becoming an imperative part of people's lives and used widely for daily activities, and therefore extensive research in this field is being carried on to further improve the data access of consumers both in terms of speed and quality of service while maintaining cost effective solutions. In this research, the aim has been on the design of switching architectures which possess simplicity and scalability as well as adaptability to the optical domain. To achieve this, a traffic test bed for the switch design was created to analyze their performance and behaviour under worst case conditions. The self-similar traffic test bed notion came from the most recent studies and observations of the real internet traffic performed in LAN, WAN and WWW. These studies confirmed that Poisson modeling fails to comply with real network conditions and a bursty, long-range dependant, non-smooth traffic is present in the internet over long periods of time.

Two methods of self-similar traffic generation and modeling were presented in this research. Both methods use the fact that aggregation of N source processes with ON-OFF distribution will yield a self-similar process. The first method generates packets based on Pareto distributed ON_OFF times with the ON times representing the packet train sizes and the OFF times representing silent times. The packets generated in this scheme are fixed-length and every packet train has one destination assigned to it.

The second method was based on generation of the Fractional Gaussian Noise self-similar processes and mapping the self-similar trace to actual packet counts with

self-similar statistics. The packet sizes obey the variable length distribution found in actual network traffic and more flexibility in terms of destination assignments and randomness along with burstiness is obtained. For both methods, several testing schemes were applied to ensure the self-similar nature of generated packets along with visual inspection. All the tests confirmed self-similarity of packet traces in both cases. Self-similar traffic modeling and various testing schemes were performed by MATLAB simulation and programming. The generated traffic along with source and destination ports was saved to a text file for later use as inputs to various packet switches.

Later, a sectored, three-stage (Clos-like) switch architecture was described. The configuration of the center stage is found by solving an edge-coloring problem on a bipartite graph defined by intersector service matrix which needs to be repeated only when the statistical pattern of arriving traffic changes. The arriving traffic was the fixed-length Pareto generated traffic. Regular reconfiguration of the center stage helps accommodate the bursts of intersector traffic and thereby keep the input sector backlogs small and the queue length finite. The delay measures obtained by simulation show that, at the modest price of a speed-up of two, it is possible to obtain a performance practically equal to that of the "ideal" architecture even for high loads and high degrees of burstiness. Moreover, reconfiguration does not need to be on a per-packet basis making it suitable for implementation of the center stage in optics where the reconfiguration overhead can be high. The architecture is being built in hardware at the moment.

To alleviate memory access contention, as an alternative to reconfiguration of the center stage, a novel load-balanced 3-stage Clos-like packet switch architecture and method was described in chapter 5. The architecture can be considered as a parallel packet switch performing load-balancing without miss-sequencing that builds on earlier research of Clos-like packet switches and suitable for implementation in hybrid electronic / photonic technology. The architecture involves no central schedulers and is implemented by distributing cross-point queues over all elements of the switch and deploying pollers to select the queues to be served. This architecture is universal in the sense that, depending upon how it is controlled, it can emulate a wide range of architectures that have been considered distinct up to now. By performing simulations in

C++ and MATLAB, throughput performances in this architecture are guaranteed even for bursty self-similar traffic conditions. The buffers although extensively used in the switch are small and manageable. The switch shows signs of adaptability to failures, where the failed sectors can be eliminated by having the pollers skip that layer. This architecture is pending for an invention patent at the moment.

6.1 Future Work

Along with the contributions in terms of self-similar traffic generation and scalable architectures, there are valuable tasks worthy of future research:

Enhancements to self-similar packet generation techniques

- The packet generation techniques are very simple without any considerations of the Quality of Service (QoS) issues and classifications. It would be desirable to consider QoS assignment according to packets' source, destination and packet prioritization.
- The variable length FGN-based traffic is still in early stages of development and has not been deployed for performance measures yet. To make it feasible in a packet switch, packet segmentation into fixed length cells and later re-assembly would have to be performed.

FBP switch advancements

- A more detailed study of delay performance can be performed to decide whether the sectored packet switch can emulate an ideal output-queued switch up to a fixed delay.
- A detailed comparison evaluation of FBP switch compared to the load-balanced switch with cross-point queues can be performed to outline the merits of one switching scheme over the other.

- It would be valuable to map the switch with Flexible Bandwidth Provision scheduling policy to a larger network The impacts of propagation delays on signaling in a distributed architecture would be the main task.

Load-balanced switch advancements

- A more detailed performance evaluation of the load-balanced switch with cross-point queues in terms of relative delays can be performed to see how this switch can emulate the IOQS. Also extending the performance comparisons in simulation to similar architectures like the one described in [KESl03a] is worthwhile.
- The main architectural backbone has been introduced. Consolidation of cross-point queues in any of the switching stages for less delay measures is still to be investigated.
- An *Agile All-Photonic Network* (AAPN) as described by [BESH00], [BOCH04] is in essence a distributed implementation of a packet switch. It would be a valuable study the extension to an agile network of the load balancing concept introduced in this thesis.

Bibliography and References

[ANDE93] T. Anderson, S. S. Owicki, J. B. Saxe, C. P. Thacker, "High-Speed Switch Scheduling for Local-Area Networks", *ACM Transactions on Computer Systems*, Vol. 11, No. 4, pp.319-352, November 1993.

[ASLA04] A Aslam, K J Christensen, "A Parallel Packet Switch with Multiplexors Containing Virtual Input Queues", *Computer Communications*, 27, pp. 1248-1263, September 2004.

[BENE65] V. E. Benes, *Mathematical Theory of Connecting Networks and Telephone Traffic*, New York, Academic Press, 1965.

[BESHA00] M. Beshai, R. Vickers, "Petaweb Architecture", Networks 2000, Towards Natural Networks: 9th International Telecommunications Network Planning Symposium, Toronto, 2000.

[BOCH04] G.V. Bochmann, T.J. Hall, O. Yang, M.J. Coates, L. Mason and R. Vickers, "The Agile All-Photonic Network: An Architectural Outline", 22nd Biennial Symposium on Communications, Kingston, Canada, June 2004, pp. 217-218.

[CHAN02a] C. Chang, D. Lee, Y. Jou, "Load balanced Birkhoff-von Neumann Switches, part I: One-stage Buffering", *Computer Communications*, Vol. 25, No. 6, pp. 611-622, April 2002.

[CHAN02b] C. Chang, D. Lee, C. Lien, "Load balanced Birkhoff-von Neumann Switches, part II: Multi-stage Buffering", *Computer Communications*, Vol.25, No. 6, pp. 623-634, April 2002.

[CLOS53] C. Clos, "A study of non-blocking switching networks", The Bell System Technical Journal, pp 406-424, March 1953.

[CROV97] M. Crovella, A. Bestavros, "Self-similarity in World-Wide Web Traffic: Evidence and Possible Causes", *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 835-846, December1997.

[CHUA99] S.T. Chuang, A. Goel, N. McKeown, "Matching Output Queueing with a Combined Input/Output-Queued Switch", *IEEE Journal on Selected Areas in Communications*, Vol. 17, No.6, pp.1030-1039, June 1999.

[HALL02] T. J. Hall, "IP-capable Switch", 18 April 2002, UK Patent Application GB0208797.1.*'Packet Switching'* International Patent Application PCT/GB2003/001690.

[HLUC88] M. Hluchyj, M. Karol, "Queueing in High Performance Switching", *IEEE Journal on Sel. Areas in Comms.*, vol 6, no. 9, pp.1587-1597, December 1988.

[IYER03] S. Iyer, N. McKeown, "Analysis of the Parallel Packet Switch Architecture", *IEEE-ACM Transactions on Networking*, Vol. 11, No. 2, pp. 314-324, April 2003.

[JEON98] H. D. Jeong, D. McNickle, K. Pawlikowski, "A Comparative Study of Generators of Synthetic Self-similar Teletraffic", *Technical Report, TR-COSC*, March 1998.

[KARO87] M. J. Karol, M. G. Hluchyj, S. P. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch", *IEEE Trans. Commun.*, Vol. 35, No. 12, pp. 1347-1356, December 1987.

[KESL02] I. Keslassy and N. McKeown, "Maintaining Packet Order in Two-stage Switches," *Proc. of the IEEE Infocom*, June 2002.

[KESL03a] I. Keslassy, S. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard and N. McKeown, "Scaling Internet Routers Using Optics" *ACM SIGCOMM 2003*, Karlsruhe, Germany, pp. 189-200, August 2003.

[KESL03b] I. Keslassy, S. Chuang, N. McKeown, "A Load-Balanced Switch with an Arbitrary Number of Linecards". *Stanford HPNG Technical Report TR03-HPNG-080101*, 2003. Available at http://yuba.stanford.edu/techreports/TR03-HPNG-080102.pdf

[KRIS03] M. Krishna. P, V. M. Gadre, U. B. Desai, *Multifractal Based Network Traffic Modeling*, Kluwer Academic Publishers, 2003.

[LEDE00] S. Ledesma, D. Liu, "Synthesis of Fractional Gaussian Noise Using Linear Approximation for Generating Self-Similar Network Traffic", *Computer Communication Review*, Vol.30, pp.4–17, April 2000.

[LELA94] W. E. Leland, M.S. Taqqu, W. Willinger, D.V. Wilson "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, Vol.2, No. 1, pp. 1-15, Febuary1994.

[MAND68] B.B. Mandelbrot, J. W. Van Ness, "Fractional Brownian Motions,

Fractional Noises and Applications", *SIAM Rev.* , Vol. 10, pp. 422-437, 1968.

[MAND69] B. B. Mandelbrot, J. R. Wallis, "Computer Experiments with Fractional Gaussian Noises", *Water Resources Research*, Vol. 5, pp. 228-267, 1969.

[OKI03] E. Oki, N. Yamanaka, K. Nakai, N. Matsuura, "Multi-Stage Switching System Using Optical WDM Grouped Links Based on Dynamic Bandwidth Sharing", *IEEE Communications Magazine*, pp. 56-63, October 2003.

[PARE05] S.A. Paredes, T.J. Hall, "Flexible Bandwidth Provision and Scheduling in a Packet Switch with an Optical Core", *Journal of Optical Networking*, Vol. 4, No. 5, pp. 260 – 270, April 2005.

[PAXS95a] V. Paxson, S. Floyd, "Wide Area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Transactions on Networking*, June 1995.

[PAXS95b] V. Paxson, "Fast Approximate Synthesis of Fractional Gaussian Noise for Generating Self-similar Network Traffic", *Computer Communications Review,* Vol. 27, No. 5, pp. 5-18, October 1997.

[SCHR91] M. Schroeder, *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise,* W. H. Freeman 1991.

[STAL02] W. Stallings, *High Speed Networks and Internets; Performance and Quality of Service*, Upper Saddle River, N.J.: Prentice Hall, 2002.

[TAEB04] S. Taebi, S. A. Paredes, T. Hall, "Performance of a Packet Switch with an Optical Core under Self-similar Traffic", *IEEE Canadian Conference on Electrical and Computer Engineering*, Niagara Falls, pp. 747-750, May 2004.

[TAEB05] S. Taebi, S. A. Paredes, T. J. Hall, "Load-balancing in Clos-like Packet Switches with Distributed Cross-Point Queues", *HPSR05*, May 2005.

[TANE96] A. S. Tanenbaum, *Computer Networks*, 3rd Edition, Prentice Hall, 1996.

[WILL97] W. Willinger, M. Taqqu, R. Sherman, D. V. Wilson, "Self-similarity Through High Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," *IEEE/ACM Trans. Networking*, Vol.5, No. 1, pp. 71-86, Febuary 1997.

Appendix A

List of Acronyms

AAPN	Agile All Photonic Network
CPQ	Cross-point Queue
FBP	Flexible Bandwidth Provision
FFT	Fast Fourier Transform
FGN	Fractional Gaussian Noise
FIFO	First-in First-Out
FOFF	Full Ordered Frames First
FTP	File Transfer Protocol
HOL	Head of Line Blocking
IOQS	Ideal Output Queued Switch
IP	Internet Protocol
LAN	Local Area Network
LRD	Long Range Dependant
MTU	Maximum Transmission Unit
OCF	Oldest Cell First
OQ	Output Queue
OQS	Output Queued Switch
PPS	Parallel Packet Switch
QoS	Quality of Service
RR	Round Robin
TCP	Transmission Control Protocol
VCSEL	Vertical-Cavity Surface-Emitting Laser
VIQ	Virtual Input Queue
VOQ	Virtual Output Queue

VOSQ	Virtual Output Sector Queue
VTP	Variance Time Plot
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing
WWW	World Wide Web

Appendix B

This appendix describes the approximation to the infinite sum term within the analytic formula for the power spectrum of a FGN process. The approximation is based on the research by Ledesma in [LEDE00] that argues that a linear approximation can be used to accurately calculate the power spectrum. The power spectrum of FGN process is:

$$f(\lambda;H) = A(\lambda;H)[|\lambda|^{-2H-1} + B(\lambda;H)] \text{ For } 0 < H < 1 \text{ and } -\pi \le \lambda \le \pi \tag{B.1}$$

Where:

$$A(\lambda;H) = 2\sin(\pi H)\Gamma(2H+1)(1-\cos\lambda)$$
$$B(\lambda;H) = \sum_{j=1}^{\infty}[(2\pi j+\lambda)^{-2H-1} + (2\pi j-\lambda)^{-2H-1}] \tag{B.2}$$

The aim is to show that $B(\lambda;H)$ is closely approximated by a linear function of λ as j tends towards infinity. Both terms of $B(\lambda;H)$ in (B.2will become constant for $j \ge 3$ and depend only on λ. Therefore $B(\lambda;H)$ can be divided into two terms, the non-linear term for $j=1, 2$ and the linear term $B_{3;\infty}$.

$$B(\lambda;H) = \sum_{j=1}^{2}[(2\pi j+\lambda)^{-2H-1} + (2\pi j-\lambda)^{-2H-1}] + B_{3;\infty} \tag{B.3}$$

Where:

$$B_{3;\infty} = \sum_{j=3}^{\infty}[(2\pi j+\lambda)^{-2H-1} + (2\pi j-\lambda)^{-2H-1}]$$

This infinite summation can therefore be written as a linear function of λ :

$$B_{3;\infty} = D(\lambda; H) = p\lambda + q$$

(B.4)

Values of p and q have to be determined now in order for $p\lambda + q$ to become as close as possible to the value of $B_{3;\infty}$. The best way of achieving this task is by minimizing the mean-squared error:

$$\varepsilon = \int_0^{\pi} [B_{3;\infty}(\lambda; H) - (p\lambda - q)]^2 d\lambda$$

(B.5)

For error minimization the derivative of ε relative to p and q should be zero:

$$\partial\varepsilon / \delta p = 0 \text{ and } \partial\varepsilon / \delta q = 0$$

(B.6)

The two conditions stated above produce the following two equations:

$$\frac{\pi}{2} p + \pi q = F(H)$$
$$\frac{\pi^2}{3} p + \frac{\pi^2}{2} q = G(H)$$

(B.7)

where p and q are given by:

$$p = \frac{-6}{4\pi^2} F(H) + \frac{12}{\pi^3} G(H)$$
$$q = \frac{4}{\pi} F(H) - \frac{6}{\pi^2} G(H)$$

(B.8)

The infinite summations in *F(H)* and *G(H)* are only a function of j and independent of λ:

$$F(H) = \sum_{j=3}^{\infty}\left[\frac{(2\pi j - \pi)^{-2H} - (2\pi j + \pi)^{-2H}}{2H}\right]$$

$$G(H) = \sum_{j=3}^{\infty}\left[\frac{2\pi j}{2\pi j + \pi} + \frac{2\pi j}{2\pi j - \pi} + \ln(2\pi j + \pi) - 2 - 2\ln(2\pi j)\right]$$

(B.9)

At first glance, it seems like the equations above require infinite summations. However, it has been shown in [LEDE00] that $F(H)$ and $G(H)$ can be accurately estimated by finite summations, since they converge to a constant value for values for $j > 100$ and therefore Fig. B.1 and Fig. B.2 show how p and q have a convergent behaviour for $j > 100$.

This approach has been proven in [LEDE00] to be accurate in estimating the FGN power spectrum and to reduce the complexity of the calculation and hence substantially reduce the computational time.

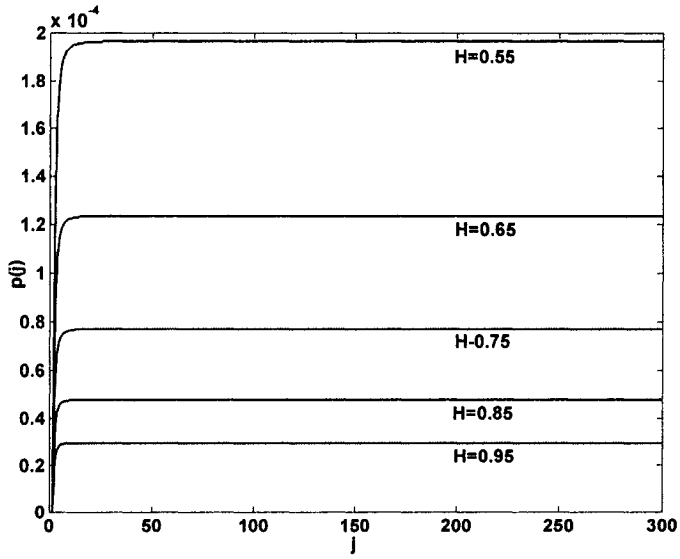


Fig. B.1 P as a function of j. It's clear that p will become constant for j>20.

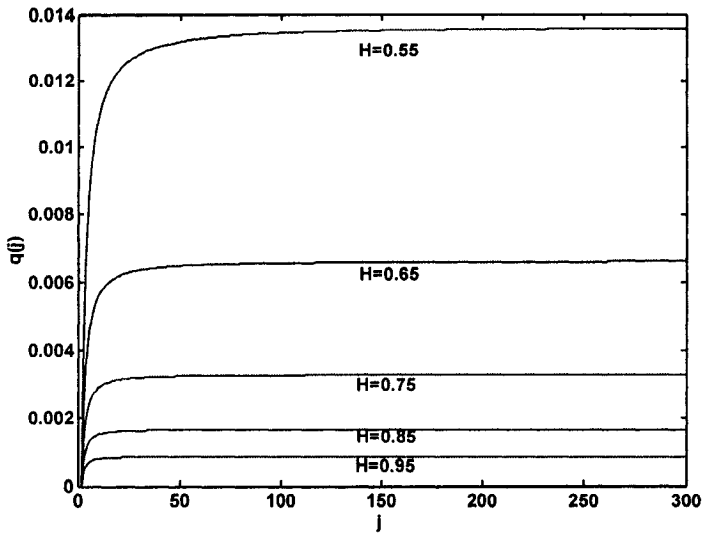


Fig. B.2 q as a function of j. It can be seen that q becomes constant for j>100