Traffic modeling in a Multi-media environment

Selvakumaran N. Subramanian

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montréal, Québec, Canada

1996

ISBN 0-612-18446-3

ABSTRACT

Traffic modeling in a Multi-media environment

Selvakumaran N. Subramanian

The evolving *Broadband Integrated Service Digital Networks* (**B-ISDN**) provide bearer service capabilities supporting real time traffic such as voice and video traffic along with jitter tolerant traffic such as data traffic. These wide spectrum of traffic sources exhibit a diverse mixture of traffic characteristics and have varied quality of service requirements. Designing and managing these evolving networks requires predictions of network performance. Traffic source models capture enough essential properties of the source, that a trace of equivalent traffic can be generated artificially. These models could then be used to investigate many of the open issues in the evolution of B-ISDN. Appropriate models that characterize the variability and correlations in the aggregate (or integrated) traffic are imperative and play an important role in the successful engineering of the future broadband networks.

In this thesis a traffic generator that can represent the behaviour of multi-media traffic for the purpose of evaluating some queueing systems, is developed. The statistical issues involved in the packet arrival process in multi-media networks are explained. These aspects are considered in the development of the traffic generator. A new traffic model called the PMPP (Pareto modulated Poisson process) is proposed. This model can capture the self-similarity and long range dependence characteristics. The traffic generator developed uses MMPP (Markov modulated Poisson process) to characterize voice and video traffic. The long range dependent data traffic is characterized by the proposed PMPP model. The developed traffic generator is used to study the queueing characteristics of long range dependent traffic and its effect on multi-media traffic.

Keywords: traffic model, long range dependence, self-similarity.

To my Parents,

Mr. V.Subramanian and Mrs. Lalitha Subramanian,
with love.....

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my supervisor Prof. Tho Le-Ngoc for his invaluable guidance, encouragement and support throughout the course of this work. He contributed a great deal of his time, effort and ideas to the work presented in this dissertation.

I gratefully acknowledge the role of SPAR Aerospace and CRC (Communication Research Centre, Canada) in funding this project. Special thanks are due to Dr. Reza Soleymani, of SPAR and Dr. Chun Loo of CRC for their constructive criticism and comments through the course of this work.

I also wish to acknowledge all the teachers and individuals who have contributed to my knowledge during the period of my studies at Concordia.

Thanks to all my friends, especially Vijay Kumar Doradla, for their constant encouragement and moral support.

Finally I wish to thank my parents Mr. V. Subramanian and Mrs. Lalitha Subramanian, brother Manimaran and sister Sindhamani for their love, affection and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Background

The recent years have witnessed significant technical advances in switching, transmission and multiplexing technologies, which have the potential of revolutionizing future voice, video and data communications. There has also been a growing demand for an unified access to more sophisticated and powerful communication services, encompassing a wide variety of applications. Together, these developments have resulted in the increasing interest in the design and deployment of *Broadband Integrated Service Digital Networks* (**B-ISDN**). These networks would ultimately bring the *multimedia services* to the end user's desk-top. These future networks would provide an integrated access supporting a wide variety of services with different characteristics: interactive and distributive services, broadband and narrowband services (e.g., real time voice and video), bursty and continuous traffic (e.g., voice and bursty data services), connection oriented and connectionless services, etc. [1]. All of these different services have different quality of service requirements (e.g., real time traffic, such as voice and video are sensitive to delay and delay jitter, while tolerating some error or loss; on the other hand non real time data traffic are sensitive to loss and have no stringent requirements on the delay).

Moreover, these wide spectrum of traffic sources (such as computer data, variable bit rate (VBR) video, voice, etc.), exhibit a diverse mixture of traffic characterisLics. These contrasting traffic were traditionally carried by separate dissimilar networks. Circuit switched networks were used to carry the delay sensitive voice traffic; a packet switched telephone network (PSTN) was used to carry the loss sensitive, jitter tolerant data traffic and cables or the broadcast media have been used to carry the delay sensitive, loss sensitive video traffic. B-ISDN proposes to integrate all these services on the same network. These present a plethora of challenges and numerous avenues for research.

Thanks to the untiring efforts of many researchers in the past decade, the field of Broadband Communications is steadily progressing towards its goal of providing integrated digital services in an efficient manner. The widespread deployment of fiber in the network, and the standardization of these networks by the SONET (Synchronous Optical NETworks) standards, together with the tremendous advances in the field of optical switching have made available huge bandwidths of the order of several gigabits to the enduser. This was followed by extensive research and study of the multiplexing technology to select a transport mode which would efficiently use the huge bandwidth available to provide integrated services. The **ATM** (Asynchronous Transport Mode) has been selected as the preferred mode of transport for these future networks.

ATM is capable of multiplexing large number of connections efficiently. The basic transport unit in ATM is a 53-byte ATM cell which consists of a 48 byte payload (information) and a 5 byte header. The transmission time of each cell or packet can be considered to be analogous to a *time-slot* (time division multiplexed channel) in a synchronous network. The basic difference between the synchronous and asynchronous transfer mode is that in a synchronous network, a time-slot is dedicated to a user and the position or timing information of the time-slot is used to de-multiplex the channel; in an ATM network, the information in the header of the cell is used to route the packet to the appropriate destination, at the multiplexer. The timing information is no longer important and hence time-slots which would have been unused in a synchronous network could be efficiently used to carry the additional traffic ensuing from the other users. Thus in ATM the available bandwidth is dynamically shared between the various users. ATM is akin to the packet networks of the past (legacy networks), except that the advances in high speed switching and the development of some fast packet switching techniques such as the SMDS (Switched Multi-megabit Data Services) have made it possible to use ATM to carry the integrated services.

ATM is benevolent to both continuous bit rate (CBR) and variable bit rate (VBR) services. In CBR, there is an uninterrupted flow of digital information transmitted at regular intervals i.e., video source coding producing constant bit rates, or, voice where silence periods are transmitted. In VBR, information is generated at variable rates as in voice with silence detection and video coding producing variable bit rates. In the case of CBR, ATM cells are generated at constant intervals and in the case of VBR, ATM cells are generated as and when there is information to be transmitted. Hence, the amount of network resources required by the user changes constantly in proportion to the number of cells generated per unit time. When the resources are shared among different users the amounts required by the users do not simultaneously reach their peak values, so the network can accommodate more load with the same amount of resources. This is called *statistical multiplexing* and the gain in the bandwidth obtained thereof, is called the *statistical multiplexing gain.* This is one of the key features of ATM networks. Achieving a high multiplexing gain requires a good understanding of the nature of the traffic sources involved.

1.2 Motivation

At the time of writing of this thesis there are still many more unanswered questions connected with the evolving B-ISDN. It is currently a hot area of research. Of particular importance are the questions related to the engineering of these future networks.

For example, in a synchronous circuit switched network, a new call is accepted, if there is an availability of a dedicated channel (time-slot), assuring an end to end connectivity between the two end users, else the call is blocked. Once a call is accepted in such a network, the resources are guaranteed through out the entire duration of the call. Grade of Service (GOS) is measured in terms of the blocking, a user may experience in setting up a connection; lower the blocking, better the GOS. However, the B-ISDN networks employing ATM, face a more complex scenario. As mentioned earlier, the bandwidth in these networks are shared dynamically between the users. However, each of the users should at least be guaranteed some fixed GOS (in terms of packet loss and delay) which is negotiated upon, during the call setup phase. Thus the network has an onerous task of deciding whether or not to accept a new call; this decision (called *call admission criterion*), has to be based on the current state (usage) of the network and on the determination of the incremental bandwidth required to achieve the desired GOS, for the new connection. Thus, the challenge in arriving at a suitable call acceptance rule is in gauging the current usage of the network (allocated bandwidth) and determining the incremental bandwidth based on the statistical characteristics of the current users and the new connection respectively.

Once the connection has been accepted into the network, appropriate measures have to be taken to ensure that the user does not exceed the usage agreed upon during setup and trespass into the bandwidth allocated for other connections. These measures, called *congestion control schemes*, are required to ensure that GOS is met for each connection. Congestion control schemes can be *reactive* or *pro-active* (preventive). Reactive schemes, as their name suggests, help in controlling the congestion in the network, after they occur. These are more suited to low speed networks. On the other hand pro-active congestion control schemes help avoid the onslaught of congestion by utilizing bandwidth enforcement mechanisms. These are especially suited to high speed networks. Pro-active schemes may use *source policing* mechanisms such as the *leaky bucket* to force the traffic to conform to the values contracted during call set up. Some pro-active schemes estimate the traffic from the current users, based on their statistical characteristics. Currently many congestion control schemes are being proposed for ATM networks. However, good congestion control schemes need an accurate source characterization.

In order to meet the GOS for a call that has been accepted, the buffers at the multiplexers should be engineered accordingly to limit the packet loss. The required dimension of the buffers at these multiplexers depends on the profile of the traffic it is catering to. Appropriate statistical traffic models are required to engineer these networks suitably. Also of importance in such networks are the various switch architectures that may be used. Numerous architectures have been proposed. To evaluate these various switch architectures we need accurate traffic models.

If ATM traffic were carried by satellite networks, then we have an additional level of complexity involved. The satellite networks use a shared access to the various satellite channels. Multiple access schemes are used to have an efficient access to this shared medium. Various multiple access schemes have been studied and proposed, so far. However, these schemes have previously been studied with just voice or data traffic or both. But given the integrated environment of the evolving B-ISDN, more light can be shed on these various multiple access schemes and their effectiveness re-evaluated, if appropriate traffic models characterizing the sources involved, were available.

Thus as outlined above most of these issues involving B-ISDN, first require a traffic characterization of the sources involved. Source models capture enough essential properties of the source that a trace of equivalent traffic can be generated artificially. These models could then be used to investigate many of the open issues in the evolution of B-ISDN. Appropriate models that characterize the variability and correlations in the aggregate (or integrated) traffic are imperative and the successful engineering of the future broadband networks depends solely on the success of these traffic models. This is the motivation behind this thesis on traffic modeling.

1.3 Overview and scope of the thesis

Designing and managing of the evolving broadband networks require prediction of network performance. Analytical techniques, computer simulation, projections from existing data are methods that are used to evaluate and design networks. Actual measurements of traffic from the broadband networks would only be available once these networks are in place and the applications that are envisioned to use these networks take shape. However, in order to engineer these networks successfully, a fairly good understanding of the behaviour of the traffic sources is necessary. Researchers have circumvented this vicious circle, by using existing measurements of traffic, to come up with models that could predict the traffic in these future networks. The traffic models may then be used in traffic generators to generate traffic for simulation studies of broadband networks. This thesis focusses on the development of such a traffic generator, that can represent the behaviour of multimedia traffic.

The traffic generation that we are interested in, is not concerned with either the simulation of an actual broadband network or with the nitty-gritties of generation of cells in these networks. We are interested in modeling and capturing the statistical characteristics of the packet traffic in these networks. Thus the traffic generator need only characterize enough statistical characteristics of the traffic in such networks, such that the synthetic traffic fed to a queue produces the same queueing behaviour as that produced by the actual traffic. Hence the stress all along this thesis is on the capture of statistical characteristics of the actual traffic.

The Poisson model was used by the teletraffic engineers of the past to characterize the call arrival in a circuit switched network. The simple closed form solutions (such as the Erlang-B formula), derived for the circuit switched networks, can be attributed to the simplicity and analytical tractability of the Poisson model. These are considered fairly accurate for the, conventional circuit switched telephone networks, to date. But, ever since the advent of the packet networks and transmission of voice digitally over the network, traffic studies proved that the Poisson assumption may no longer be valid for the packet arrival process in these networks. This is so, because with the arrival of packet networks, the focus has been on modeling the packet arrivals to the queue as opposed to the call arrivals to the system. In the circuit switched systems, these meant one and the same; i.e., a call arrives to the system and engages a circuit for a particular duration of the call called the *holding time* of the call. However with packet switching a single transmission line is shared among various calls (or connections). Hence the modeling shifts to a lower level. Though the call arrival to such circuits may still be completely random and thus obey a Poisson law in such systems, the packet arrival is not. The packet arrival process is correlated and is thus no longer Poisson.

The amount of correlation in the packet arrival process depends on the type of the traffic, i.e., voice, video or data. Voice and video traffic possesses correlation that extend only over short durations. However data traffic, as shown by recent studies in LAN data traffic, exhibits long-range dependence (where the correlations extend over longer durations) and self-similar (or fractal) characteristics, i.e., the traffic exhibits " burstiness" across a wide range of time scales ranging from milliseconds to hours.

Ever since the failure of Poisson assumptions in characterizing packet traffic, a wealth of traffic models have been proposed in the literature to characterize the various classes of traffic. These models such as the Batch Poisson process, layered Markov process, Markov modulated Poisson process still retain some form of Markovian structure either in the way the arrival processes are modulated or in the arrival processes themselves, for reasons of mathematical tractability. Also these models incorporate only short term correlations.

There have also been a few long-range dependent models like the fractional Brownian traffic model [2], Chaotic maps [3] that have been added to the list of traffic models. These models aim at characterizing long-range dependent traffic. In this thesis, we underline the relevant issues in modeling packet traffic and provide an overview of all the models proposed in the literature from the standpoint of synthetic traffic generation.

Realizing the need for a simple and efficient model for the generation of long-range traffic, we also propose a new long-range dependent model called the PMPP (Pareto modulated Poisson process) [4]. The PMPP is a special class of doubly stochastic Poisson processes and are thus extremely simple to generate. The PMPP is versatile in capturing the long-range dependence and self-similarity.

Finally, a traffic generator is developed in OPNET. The traffic generator developed is capable of capturing different level of correlation and dependency and is extremely simple and versatile from the standpoint of synthetic traffic generation. The developed traffic generator is then used to study the queueing performance of a queue fed by the aggregate multi-media traffic.

The thesis is organized as follows: In **Chapter 2**, we outline the important issues in the statistical characterization of traffic and the reasons for the failure of the Poisson model. We also provide the statistical definition of long-range and short-range dependent processes and distinguish between them.

In **Chapter 3**, we present a study of the traffic models proposed in the literature for voice and video traffic. The strengths and weaknesses of these models are discussed and a suitable traffic model is selected for representing voice and video traffic respectively in the traffic generator.

Chapter 4 presents a study of data traffic models. The new model for long-range dependent traffic proposed in this thesis, the PMPP model, is also presented here. The statistical characteristics of this model found by simulation, which demonstrate its ability in capturing the long-range dependent correlations are presented here and are also backed by an analytical study of the model.

Chapter 5 presents the traffic generator developed. The rationale for the selection of the various models in the traffic generator is also presented. Finally the developed traffic generator is used to study the characteristics of a queueing system fed by aggregate multi-media traffic. The simulation results of the queueing behaviour are presented and discussed.

Chapter 6 summarizes the results of this thesis and provides directions for future work.

The process models and program listings of the traffic generator, developed on OPNET are provided in the Appendix.

Chapter 2

Characterization of packet traffic

In this chapter, we explain in detail the statistical terms and notions that are important from the perspective of traffic modeling. First we take a look at the Poisson model and discuss the limitations of the model in characterizing packet traffic. Next, we look at the commonly used measures of "burstiness" in the packet arrival process. The last section of this chapter introduces the notions of long-range dependent, short range dependent and self-similar processes.

2.1 Poisson model and its limitations

Traditionally, the Poisson model has been used as the model for characterizing telephone traffic in circuit switched networks. The Poisson model has been very successful because of its simplicity; it has just a single parameter, namely the arrival rate λ. Given the arrival rate λ, the probability distribution of the number of arrivals X_t, in a given time interval t may be given by the relation

$$P(X_t = k) = \frac{\exp((-\lambda t))(\lambda t)^k}{k!} \tag{2.1}$$

The Poisson model besides being simple lends itself to analysis easily. In fact the Poisson process plays a central role in queueing theory for this reason. The analytical simplicity of the Poisson process may be attributed to its intimate relationship with the exponential distribution. For Poisson arrivals the inter-arrival time (time between arrivals) has an exponential distribution as given below:

$$A(t) = \lambda \exp((-\lambda t)) \tag{2.2}$$

The exponential distribution is the only distribution to possess the *memoryless* property; i.e., the past history of a random variable that is distributed exponentially plays no role in predicting its future. This memoryless (Markovian) property forms the basis of analysis of queueing systems involving Poisson arrivals and leads to closed form solutions.

After their successful use in modeling traffic in circuit switched systems, the Poisson models were also employed to model traffic in early packet data networks and subsequently in packet voice networks. However, the Poisson process is used to describe events (or arrivals) that occur completely randomly. Any correlation in the arrival process is not captured by the Poisson process. In circuit switched systems, where the Poisson process was used to model the call arrivals to the system, this was not a problem, since calls arrived from various users completely randomly and engaged the circuits for a particular duration called the *holding time of the call.* However with packet networks, a single transmission line is shared between various calls (or connections). Hence the modeling shifts to a lower level. Though the call arrivals to such circuits may still be completely random and thus obey a Poisson law, in such systems, the packet arrival is not. Recent studies have shown that packet traffic is highly correlated and possesses strong dependencies among successive packet interarrivals [5, 6, 7, 8, 9, 10]. Intuitively, given a source (voice, video or data) has just emitted a packet, there is a higher probability of another arrival from the same source. Hence, as deemed by the Poisson process, the packet arrivals are not totally random but possess some degree of correlation. The amount of correlation may vary between the type of source (voice, video, data) and the coding method (for voice and video) used. These studies also suggest that packet traffic is "bursty", i.e., it has higher variability than a Poisson process. We shall explore the characterization of this "burstiness" in packet networks in the next section.

2.2 Characterization of burstiness

Various measures have been used to characterize "burstiness" in packet networks. All these measures are used to indicate the extent by which the packet arrival process deviates from the "smooth" arrivals, as modelled by a Poisson process. Intuitively the term "burstiness" can be interpreted as denoting the variability in the packet arrival process, leading to "bunching" of packet arrivals as opposed to much smoother arrivals in the case of Poisson processes. Capturing of this variability is very important in ATM networks, as this variability in packet arrivals is connected to the queueing delays the packets experience [5]. Some of the commonly used measures of burstiness include

- Peak to average bit rate ratio (PAR).
- Coefficient of variation.
- Indices of dispersion for intervals and counts (IDI and IDC)

PAR is the ratio of the peak to the average bit-rate or bandwidth of traffic from a source. Since in ATM networks, statistical multiplexing is used, and sources are allocated bandwidth around their mean bit-rate, this measure gives an intuitive notion of the factor by which the given traffic may exceed its mean bit-rate and, therefore, can be viewed as a measure of the burstiness of the traffic. Higher the PAR, higher the burstiness of the traffic. However, the PAR depends on the length of the interval over which the measurements are made.

Coefficient of variation is defined as the ratio of the standard deviation to the mean of the number of packet arrivals. This measure incorporates some second order characteristics and captures the variability in the arrival process. Though the coefficient of variation is a better measure of burstiness than PAR, in terms of capturing the second order properties in the arrival process, this measure again depends on the duration over which the measurements are made and as such cannot be used as an absolute measure of burstiness of a process.

The Indices of dispersion have long been known in the statistical community as a powerful tool in the analysis of the second order-properties of point processes. Now, they have also been used as a measure for characterizing burstiness in the packet arrival process [11] [5]. There are two indices of dispersion depending on whether the point process is viewed from an interval characterization or as a counting process. They are *Index of Dispersion of Intervals (IDI)* and *Index of Dispersion of Counts (IDC)*.

Let $\{L_k, k \geq 1\}$ represent the sequence of packet interarrival times (i.e., L_1 is the time elapsed between the arrival of the first and second packet, L_2 is the time elapsed between the arrival of the second and third packet,.....etc.). Then the Index of dispersion of intervals (IDI), also called the k interval squared coefficient of variance sequence, is the sequence $\{J(k), k \geq 1\}$ defined by

$$J(k) = \frac{k\, Var\{L_1 + L_2 + \ldots + L_k\}}{E[\{L_1 + L_2 + \ldots + L_k\}]^2}$$

Assuming that $L_k, k \geq 1$ is stationary we note that the $E(L_1) = E(L)$, $Var(L_1)$ $= Var(L)$ and the sum $L_1 + L_2 + \ldots + L_k = L_{i+1} + L_{i+2} + \ldots + L_{i+k}$. Denoting this sum by S_k we have

$$\begin{aligned} J(k) &= \frac{k\, Var(S_k)}{[E(S_k)]^2} \\ &= \frac{Var(S_k)}{k[E(L_1)]^2} \\ &= \frac{k\, Var(L_1) + \sum_{i,j=1;i\neq j}^{k} Cov(L_i, L_j)}{k\,[E(L_1)]^2} \end{aligned}$$

$$= C_J^2 + \frac{2\sum_{j=1}^{k-1}(k-j)\,Cov(L_1 + L_{1+j})}{k\,[E(L_1)]^2} \qquad k \geq 1$$

$$= C_J^2 \left[1 + 2\sum_{j=1}^{k-1}(1 - \frac{j}{k})\,\varepsilon_j \right] \qquad k \geq 1 \qquad (2.3)$$

where $C_J^2 = \frac{Var(L_1)}{E(L_1)} = \frac{Var(L)}{E(L)}$ is the squared coefficient of variation of intervals and $\varepsilon_j = \frac{Cov(L_i, L_{i+j})}{var(L)}$ is the autocorrelation coefficient. For $k = 1, J(k) = C_J^2$ is the squared coefficient of variation of a single interarrival time. For $k > 1, J(k)$ measures the cumulative covariance (normalized by the square of the mean) among k consecutive inter-arrival times. It is the dependency of the IDI on the autocorrelation coefficient as shown in Eqn. 2.3 that makes IDI useful in describing arrival processes and burstiness.

For a Poisson process the successive interarrival periods are independent and identically distributed with an exponential distribution and hence the ε_j vanishes for $j \geq 1$ and $C_J^2 = 1$. Thus for a Poisson process the IDI is identically equal to one for all lags k.

For stationary point processes with positive correlation coefficients, the IDI increases monotonically with increasing lags. The limit of Eqn. 2.3 when it exists is proportional to the sum of all correlation coefficients, that is ,

$$lim_{k \to \infty}\, J(k) = C_J^2 \left[1 + 2\sum_{j=1}^{\infty} \varepsilon_j \right] \qquad (2.4)$$

Thus when applied to packet arrivals, the behaviour of IDI with increasing lags can be suggestive of the existence of correlation between successive interarrivals and the scale in which they exist. The lags at which the IDI reaches the limit give an indication of the duration over which the correlation between interarrival times are pertinent. (The correlation coefficients of interarrival times beyond this lag decrease to 0). Also the rate at which the IDI increases with lags indicates the amount of positive correlation existing between interarrivals. The presence of a higher positive correlation would result in interarrival times shorter than the mean interarrival time and interarrival time longer than the mean interarrival time to occur together in separate bursts. Hence, higher the correlation between arrivals, higher is the IDI and burstier the traffic.

The same second order properties captured by the IDI can be captured by analyzing the packet arrival process from the perspective of packet counts, i.e., the number of packets in an interval. This gives rise to the Index of dispersion of counts (IDC) and is often preferred over IDI since it is easier to visualize the packet arrival process as a counting process than analyzing its interarrival characteristics. The Index of dispersion of counts (IDC) at time t is given by the ratio of the variance of the number of arrivals in an interval of length t to the mean number of arrivals in the same interval. If we let $\{X_t, t \geq 1\}$ represent the sequence of packet counts in successive intervals, then the Index of dispersion of counts (IDC) is given by

$$\begin{aligned}
I(t) &= \frac{Var\{X_1 + X_2 + \ldots + X_t\}}{E[\{X_1 + X_2 + \ldots + X_t\}]} \\
&= \frac{t\,Var(X_1) + \sum_{i,j=1;i\neq j}^{t} Cov(X_i, X_j)}{[E(X_1)]} \\
&= \frac{Var(X)}{E(X)} + \frac{2\sum_{j=1}^{t-1}(t-j)\,Cov(X_1 + X_{1+j})}{[E(X)]} \qquad t \geq 1 \\
&= \frac{Vcr(X)}{E(X)}\left[1 + 2\sum_{j=1}^{t-1}(1 - \frac{j}{t})\,\rho_j\right] \qquad t \geq 1 \qquad (2.5)
\end{aligned}$$

where Var(X) and E(X) are the common variance and mean of the X_t and ρ_j is the correlation coefficient at lag j. As in the case of IDI, the IDC also depends on the autocorrelation function (of counts, however) and this dependency makes IDC a powerful measure in characterizing an arrival process.

For Poisson, the counts in disjoint intervals are totally independent of each other, hence the correlation coefficient ρ_j is zero for all lags j. Hence the second term in Eqn. 2.5 vanishes. Also, for a Poisson process, the variance equals its mean and thus from Eqn. 2.5, IDC for a Poisson process is identically equal to 1 for all lags.

For processes having positive correlation between counts in disjoint intervals, the IDC monotonically increases with increasing lags, as seen from Eqn. 2.5. The limit of Eqn. 2.5 when it exists, is proportional to the sum of all correlation coefficients, i.e.,

$$lim_{t\to\infty}\ I(t) = \frac{Var(X)}{E(X)}\left[1 + 2\sum_{j=1}^{\infty}\rho_j\right] \tag{2.6}$$

Further, it can also be proved that the limit of the IDC is equal to the limit of the IDI [12]. i.e.,

$$lim_{k\to\infty}\ J(k) = lim_{t\to\infty}\ I(t) \tag{2.7}$$

Thus as in the case of IDI, the behaviour of IDC with lags can be used to gain an understanding on the burstiness of the process. A monotonically increasing IDC suggests the existence of positive correlation between packet arrivals in disjoint intervals. The lag at which the IDC reaches its limit gives the duration over which the positive correlation are existent (for lags greater than this duration, the autocorrelation coefficient ρ_j equals 0). The rate at which the IDC increases with lags indicates the amount of positive correlation existing between counts in disjoint intervals. The presence of a higher correlation would result in intervals with packet counts higher than the mean packet count and intervals with packet count less than the mean packet count to bunch together separately. Thus higher the correlation between arrivals in disjoint intervals, higher the IDC and burstier the traffic.

As seen from Eqn. 2.6, the IDC has a limit only when the second term converges, i.e., the autocorrelation coefficients ρ_j are summable. This is only true if the autocorrelation coefficients ρ_j approach zero faster (than an exponential). Such processes for which the autocorrelation coefficients ρ_j die down over a period are called *short range dependent* processes. The processes like MMPP (Markov modulated Poisson process), batch Poisson processes [1] constructed from Poisson processes are

[1] These processes will be dealt with in detail in the forthcoming chapters

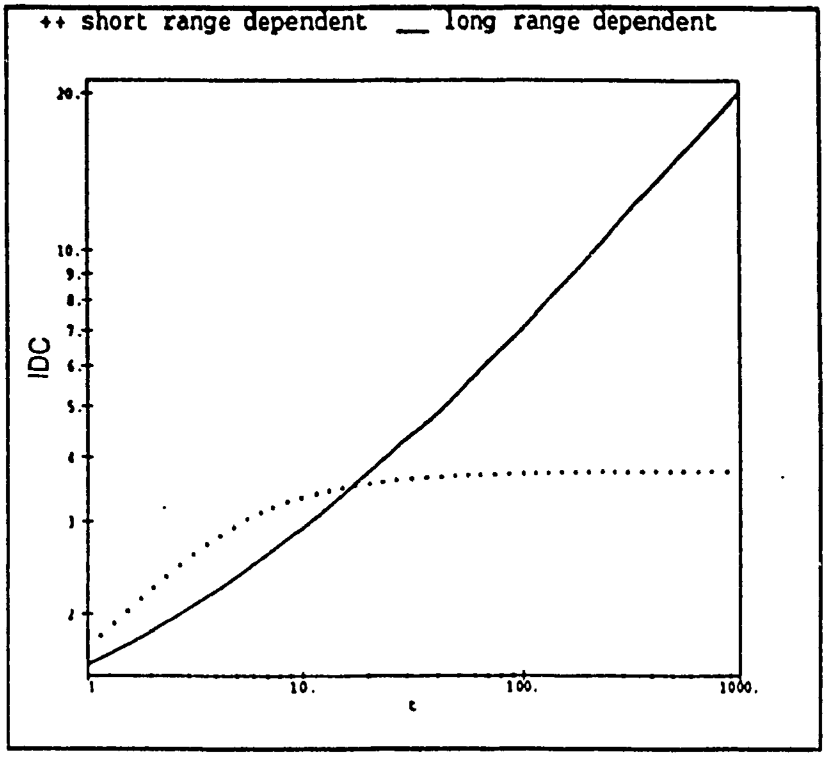


Figure 2.1: Typical IDC characteristics of short range dependent and long range dependent processes

all short-range dependent process. The IDC for these processes settles down after increasing monotonically over a duration. The typical IDC characteristic for such processes is plotted in a log-log plot in Figure 2.1.

There are other processes for which the autocorrelation coefficients ρ_j are not summable, they die down very slowly and exist over longer durations. For such processes the IDC has no limit and increases monotonically forever. Such processes for which the autocorrelation coefficients ρ_j exist over longer intervals are called *long term dependent* processes.

In particular there are some processes for which there are relatively large amount of very-long-term variation. for which the autocorrelation function $\rho_j =$

$O(j^{-\beta})$ with $0 < \beta < 1$, i.e., the autocorrelation function dies down as a fractional power of the lag. For these processes the IDC in the limit will behave as

$$I(t) \approx Kt^{1-\beta} \tag{2.8}$$

where K is a constant. For such long range dependent processes the fractional component β gives additional information about these processes. The IDC is no longer a meaningful measure for such processes; since the IDC keeps increasing with time. Plotting the IDC in a log-log plot, the value of the fractional component β can be obtained from the slope of the curve. Such processes whose autocorrelation dies down as a fractional power of time possess *self-similar* characteristics. (We shall discuss in detail about the self-similar characteristics in the next section). Self-similarity is measured by the Hurst parameter H. The Hurst parameter H is related to β by the relation

$$H = 1 - \frac{\beta}{2} \qquad 0 < \beta < 1 \tag{2.9}$$

Thus Eqn. 2.8 for self-similar processes can be rewritten as

$$I(t) \approx Kt^{2H-1} \tag{2.10}$$

Since $0 < \beta < 1$, the Hurst parameter H is $0.5 < H < 1$. Lesser the value of β, slower the rate of fall of the autocorrelation function, higher the burstiness. Thus high values of Hurst parameters correspond to bursty traffic and low values of H correspond to less bursty traffic. Hence in the case of the self-similar traffic burstiness is better captured by the Hurst parameter rather than the IDC.

2.3 Short range dependent, long range dependent and self-similar processes

In the previous section, the short range dependent, long range dependent and self-similar processes were introduced. In this section we give a more rigorous and mathematical definition for these processes. Statistically long range dependent and short range dependent processes may be distinguished as shown below [13].

Consider a second order stationary process $\{X_t\}$ with autocovariance function

$$\begin{aligned}\gamma_h &= Cov(X_t, X_t + h)\\ &= E[(X_t - E[X])(X_{t+h} - E[X])]\end{aligned}$$

variance $v = \gamma_0$, autocorrelation function $\rho_h = \gamma_h/\gamma_0$ and power spectral density function $g(\omega)$, where for $-\pi < \omega < \pi$,

$$g(\omega) = \frac{1}{2\pi}\sum_{h=-\infty}^{\infty} \gamma_h \exp(-ih\omega)$$

$$\gamma_h = \int_{-\pi}^{\pi} \exp(ih\omega)g(\omega)d\omega$$

Also, for each $m = 1, 2, \cdots$ let $\{X_t^{(m)}\}$ denote the new series formed by averaging the original series in non-overlapping blocks of m, replacing each block by its mean. That is

$$X_t^{(m)} = \frac{X_{mt-m+1} + \cdots + X_{tm}}{m} \tag{2.11}$$

Now, the new series is second order stationary with autocovariance function $\gamma_h^{(m)}$, variance $v_m = \gamma_0^{(m)}$ and associated variance sequence $v_k^{(m)}$.

For the process $\{X_t\}$ to be short range dependent the following conditions apply:

(i) $\sum_{h=0}^{\infty} \rho_h$ is convergent.

(ii) $g(0)$ is finite.

(iii) v_m is for large m asymptotically of the form $\frac{v'}{m}$. (where v' is a constant)

(iv) the averaged process $\{X_t^{(m)}\}$ tends to second order pure noise as $m \to \infty$.

As seen already if the autocorrelation coefficients (or equivalently the autocovariance coefficients) of a process vanishes bevond a certain lag it results in the process being short range dependent. This is denoted by the condition *(i)* which requires that the autocorrelation function be summable. (With most short range dependent processes such as the MMPP (Markov Modulated Poisson process), the autocorrelation function decays eponentially fast, thus giving rise to summable autocorrelation function). Since the power spectral density and the autocovariance function are related, the condition *(ii)* is a frequency domain manifestation of condition *(i)*. Condition *(iii)* states that the variance of the sample mean decreases linearly proportional to the sample size.

On the contrary processes with long range dependence, possess the following properties

(i)' $\sum_{h=0}^{\infty} \rho_h$ is divergent or non-summable.

(ii)' $g(\omega)$ is singular near $\omega = 0$.

(iii)' variance of the sample mean v_m does not decrease linearly proportional to the sample size.

(iv)' the averaged process $\{X_t^{(m)}\}$ does not tend to a second order pure noise as $m \rightarrow \infty$.

Some long range dependent processes also display another property called *self-similarity*. Intuitively, self-similar phenomena display structural similarities across all (or at least a wide range of) time scales. This self-similar property was found in the packet arrival processes in recent measurements of Ethernet LAN data traffic [8, 9, 10]. In the case of Ethernet LAN data traffic, self-similarity is manifested in the absence of a natural length of a "burst"; at every time scale ranging from a few milliseconds to minutes and hours, bursts consist of bursty subperiods separated by less bursty subperiods. Figure 2.2 reproduced from [14] helps explain the concept of self-similarity pictorially. Figure 2.2 (sets of figures on the left) shows a sequence of simple plots of the packet counts for five different choices of time units. Starting with

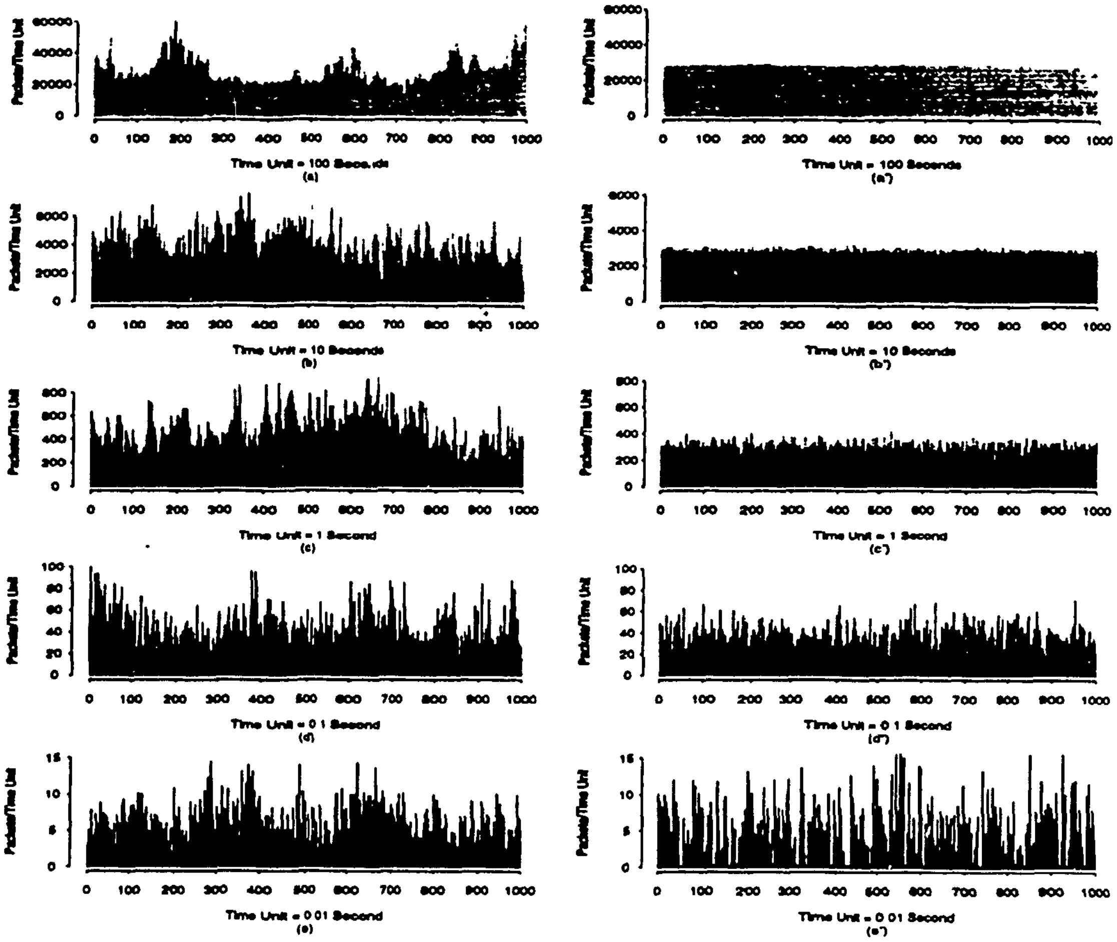


Figure 2.2: Ethernet traffic (packets per unit time) on five different time scales (left side). Synthetic traffic from compound Poisson model (right side).

a time unit of 100 seconds (a), each subsequent plot is obtained from the previous one by increasing the time resolution by a factor of 10 and by concentrating on a randomly chosen sub-interval (as indicated by the darker shade). The time unit corresponding to the finest time scale is 10 milliseconds. From Figure 2.2 it is evident that

a) plots of traffic measurements at various time scales look intuitively similar to one another (statistical self-similarity).

b) plots are distinctively different from white noise.

c) plots show that at every time scale ranging from milliseconds to minutes and hours, bursts consists of bursty sub-periods separated by less bursty sub-periods.

These processes are markedly different from the traditional short range dependent processes. This is pictorially illustrated in Figure 2.2 (right side) which shows the traffic plots generated by a batch Poisson model, viewed at different time scales. As is characteristic of a short range dependent process, the plots appear more smooth like white noise as the time scale in which the observations are made is increased.

Thus the most striking feature of the self-similar processes is that their aggregated process $\{X_t^{(m)}\}$ possesses a non-degenerate correlation structure as $m \to \infty$. This behaviour is precisely the intuition illustrated with the sequence of plots in Figure 2.2.

Statistically a process $\{X_t\}$ as defined earlier, is said to be self-similar if it exhibits the following properties (a_1, a_2, a_3 given below are all constants):

(i)" as $h \to \infty, \rho_h \approx a_1 h^{-\beta}$ $(0 < \beta < 1)$. i.e., the autocorrelation function decays hyperbolically implying a nonsummable autocorrelation function.

(ii)" as $\omega \rightarrow 0, g(\omega) \approx a_2\omega^{-(1-\beta)}$, with $(0 < \beta < 1)$. i.e., the spectral density obeys a power law near the origin. This property is also called the *1/f noise.*

(iii)" as $m \rightarrow \infty, v_m \approx a_3 m^{-\beta}$ $(0 < \beta < 1)$. i.e., the variance of the sample mean decreases more slowly than the reciprocal of the sample mean, a property called *slowly decaying variances.*

(iv)" for exactly (second-order) self-similar processes as $m \rightarrow \infty$, $\rho_h^{(m)} = \rho_h$ and for asymptotically (second-order) self-similar processes, as $m \rightarrow \infty$, $\rho_h^{(m)} \rightarrow \rho_h$ i.e., the process $\{X_t\}$ and the averaged process $X_t^{(m)}$ have identical correlational structure.

The degree of self-similarity is measured by the Hurst parameter H (named after the hydrologist, H. E. Hurst who originally found the occurrence of such processes in river flow time series [15]). The Hurst parameter H is related to β by the relation

$$H = 1 - \frac{\beta}{2} \qquad 0 < \beta < 1 \tag{2.12}$$

The Hurst parameter of the process may be estimated by obtaining the parameter β, either from the IDC plots (as explained in the previous section) or by plotting the Variance - time curves (i.e., by plotting the variance of the aggregated process $\{X_t^{(m)}\}$ against the aggregation level m; by property *(iii)"*, the slope of this curve in a log-log plot gives β).

Since β is between 0 and 1 $(0 < \beta < 1)$, the Hurst parameter H is between 0.5 and 1 $(0.5 < H < 1)$. It can be seen that when $\beta = 1$, $H = 0.5$ and the conditions *(i)"* to *(iii)"* reduce to the conditions *(i)* to *(iii)* for the short range dependent processes. Thus H = 0.5 corresponds to short range dependent models. On the other extreme $H = 1$, denotes very long range dependent, self-similar processes. Lower the value of β, (high values of H) more the long range dependence in the process and burstier is the arrival process. Hence for self-similar process Hurst parameter may be used to characterize the burstiness of the process, as explained before.

Chapter 3

Traffic models for Voice and Video

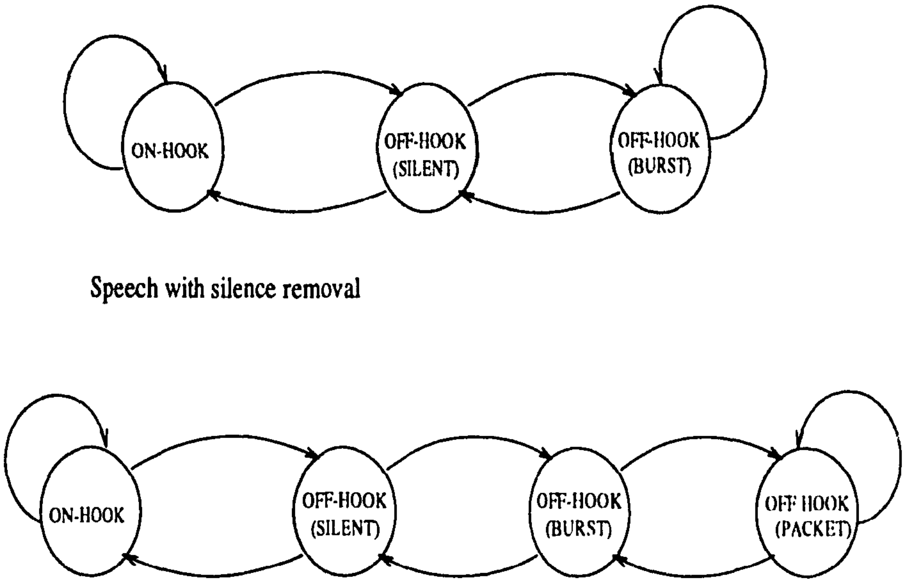
With the advent of B-ISDN, significant effort has been devoted to supporting real time communication application such as real time voice and video along with jitter tolerant data in a packet switched environment. In such a multiplexing environment, the packets from many sources are statistically multiplexed on to a single high speed link in order to exploit the bursty nature of the sources. Such a statistical multiplexing introduces different delays to packets. Real time traffic (like voice and video) are delay sensitive (loss insensitive) while data traffic is loss sensitive (delay insensitive). Hence in packet networks supporting real time traffic delay is bounded at the expense of some loss. However, in order to meet a required grade of service the loss of packets have to be kept within a certain limit. This necessitates that the buffer used to queue the packets in the statistical multiplexer be engineered to keep the delay and packet loss within specified limits. In order to do so, a thorough understanding of the packet arrival process to the statistical multiplexer and simple but accurate models to analyze such a system are required. Traditionally a Poisson approximation has always been adopted to characterize the packet arrival process. But recent studies indicate that the packet arrival process to the multiplexer is highly correlated and that the Poisson approximation for the arrival process results in erroneous results since it fails to account for these correlations. This chapter presents a study of the models that have been proposed for voice and video traffic. A suitable traffic model for voice and video traffic is also selected to be used in building the traffic generator.

3.1 Nature of voice traffic

Traditionally the voice source (telephone) in a circuit switched environment has been viewed as a 2 state process. The source is either in the *OFF* state (on-hook) or *ON* state (off-hook). The length of an off-hook period corresponds to the duration of the *call* and is called the *call holding time*. It should however be noted that during each call, the user (talker) is not always talking; one talker pauses while the other speaks. Also, even when one talker is speaking, pauses occur between utterances and there are times when the circuit is idle. Thus, active speech signals are present on a transmission channel for only a fraction of the total conversation time; in fact, actual measurements show that speech is present on a typical telephone channel approximately 40 percent of the time [16]. (The fraction of the time speech is present on the transmission line is called the *activity factor*). Hence during a call there are periods of *silence* between successive *bursts* or *talk-spurts*.

With the digitization of speech and the introduction of packet voice networks, Digital Speech Interpolation (DSI) techniques [17] and speech activity detectors have been employed to remove these silent periods; voice packets are transmitted only when there is a speech activity. Such a system can be modeled by 3 states: on-hook, off-hook and burst. If each burst is packetized for transmission, a fourth state is needed which represents the state of a packet transmission during the burst state, as shown in Figure 3.1.

The above model characterizes the single source at a higher level (i.e., at a



Speech with silence removal

Packetized voice with silence removal

Figure 3.1: Call level models for a single voice source

call level). Different approach has been followed to characterize the single source at the packet level. In this model the voice source is *active* when there is speech activity (i.e., the talker is actually speaking) and during these times the voice source periodically generates fixed length packets. A voice source is *inactive* when the speaker is silent (during the course of the call) and during these times the voice source does not generate packets, Figure 3.2. Experimental results have proved that the duration of the active periods fits the exponential distribution very well, while the duration of the inactive period is not as well approximated by the exponential distribution [18, 16]. However for analytical simplicity the silence periods have always been modeled as exponentially distributed.

Single voice source - Model 1

If T ms is the packetization time then, the packet stream from a single voice source is characterized by arrivals at fixed intervals of T ms during talkspurts and no

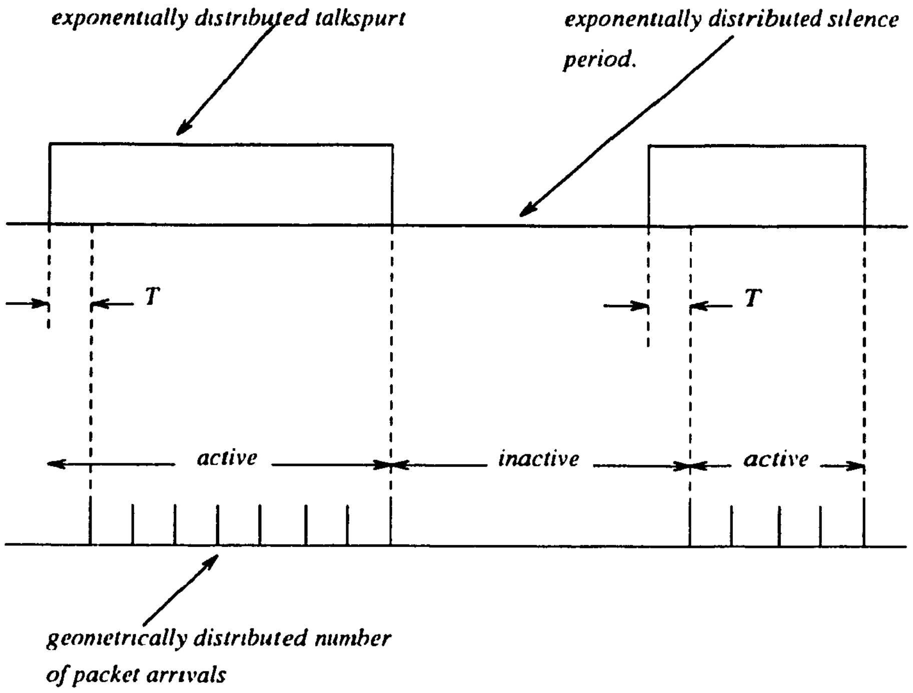

exponentially distributed talkspurt
exponentially distributed silence period.
T
T
active
inactive
active
geometrically distributed number of packet arrivals

Figure 3.2: The packet arrival process from a single voice source

arrivals during silences. The talkspurts are assumed to be exponentially distributed with mean α^{-1} generating a geometrically distributed number of packets of mean α^{-1}/T. The silent periods are assumed to be exponentially distributed with mean β^{-1}. Under these assumptions, the packet arrival process can either be treated as a renewal process (since the talkspurt and silence periods are independent and identically distributed and alternate each other) or as a 2 state (ON/OFF) discrete time (or continuous time) Markov chain with the transition rates from ON to OFF state equal to α and from OFF to ON state equal to β, Figure 3.3.

For a packet of 64 bytes, coded with 32 Kbps ADPCM T = 16 ms. Typical values of α^{-1} = 352 ms (with a mean of 352/16 = 22 packets) and β^{-1} = 650 ms [5]. The interarrival period for such a source is T ms for most of the packets and occasionally greater than T ms, when there is a silence period in between. Hence the probability density function of the interarrival period, as shown in Figure 3.4 ([5])is as given below,

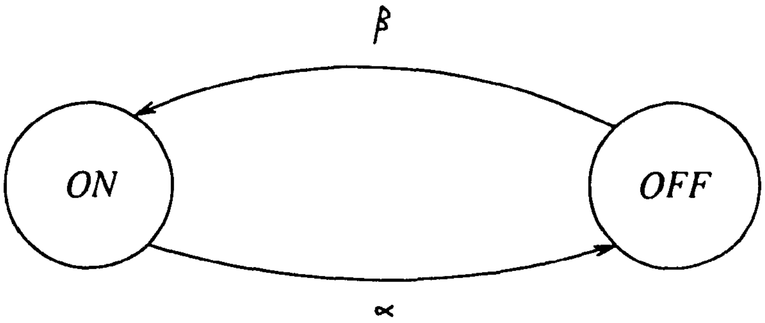


Figure 3.3: Two state continuous time Markov chain model

$$f(t) = p.\delta(t-T) + (1-p).\beta \exp(-\beta(t-T))$$

where p is the probability that a packet is followed by another packet after T ms and is given by p = $\exp(-\alpha T) \approx 1-\alpha T$. Therefore,

$$f(t) = (1-\alpha T)\delta(t-T) + \alpha T\beta \exp(-\beta(t-T)) \qquad (3.1)$$

The cumulative distribution function for the interarrival time F(t) is obtained by integrating f(t) and is given by

$$F(t) = [(1-\alpha T) + \alpha T(1-\exp(-\beta(t-T)))]U(t-T) \qquad (3.2)$$

where U(t) is the unit step function.

The number of packets per talkspurt is geometrically distributed with mean equal to $1/\alpha T$, and the distribution is given by

$$P_i = (1-\alpha T)^{i-1}\alpha T \qquad i = 1,2,3,\ldots$$

The squared coefficient of variation (variance divided by the square of the mean) of an interarrival time is given by

$$c_1^2 = (1-p^2)/[T\beta + (1-p)]^2 \qquad (3.3)$$

$c_1^2 = 18.1$ (with typical values given before). Hence the packet arrival process from a single voice source is highly bursty as is reflected by the high value of c_1^2 compared to that of a Poisson process which has a $c_1^2 = 1$.

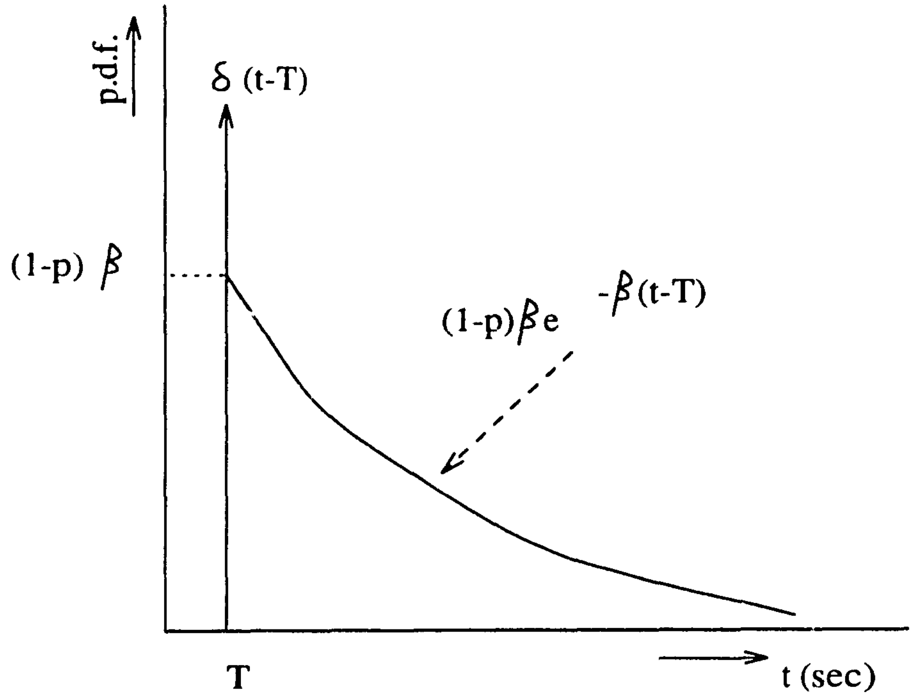


Figure 3 : Probability density function for packet interarrival time from a single voice source.

Single voice source - Model 2

Another approach followed in characterizing an individual voice source is by approximating it as an Interrupted Poisson Process (IPP). Here again the talkspurt and silence period are assumed to be exponentially distributed, but the arrivals during the talkspurt are Poisson with a rate λ, rather than deterministic [19]. This process can be visualized as a Poisson Process which is alternately turned ON for an exponential period of time and then turned OFF for another independent exponential period of time - hence the name Interrupted Poisson Process.

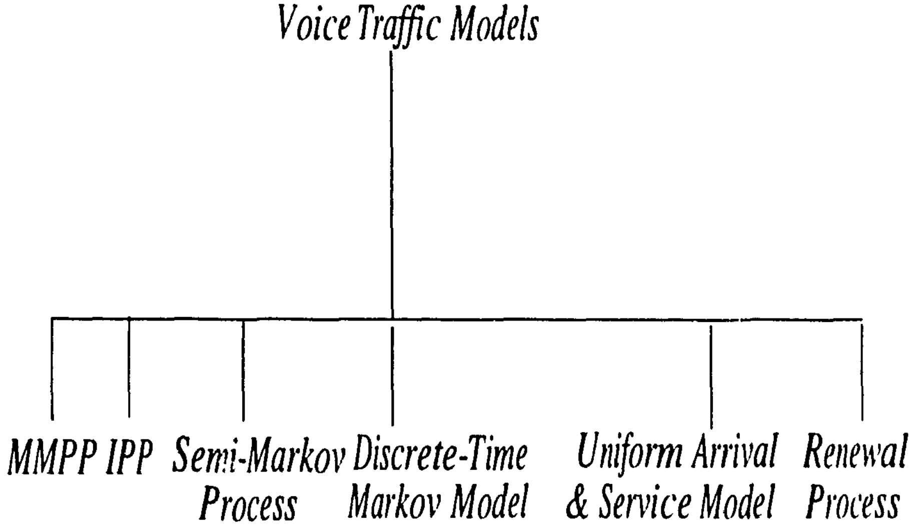


Figure 3.5: Models for voice traffic

3.2 Modeling of Statistically multiplexed voice

The superposition of arrival processes of many voice sources possesses strong correlations in the number of arrivals in adjacent time intervals and, therefore, a Poisson approximation to the aggregated arrival process will underestimate the delays experienced by the actual voice packets. Several studies ([20, 5, 6, 11, 21, 22, 19, 23, 24, 25, 26, 27, 28]) have dealt with the issue of characterizing the superposition of voice sources and analyzing the behaviour of the resulting queue. All of them concur that superposition process is not Poisson but they differ in their approaches in modeling the aggregate process. The basic objective in these various models has been to approximate the superposition of many voice sources, by a suitable process and obtain various performance measures such as the probability of loss, mean packet delay etc. Table 3.1 gives an overview of the various models proposed in the literature to characterize the superposition arrival process of voice, and the respective performance measures, the models were used to evaluate.

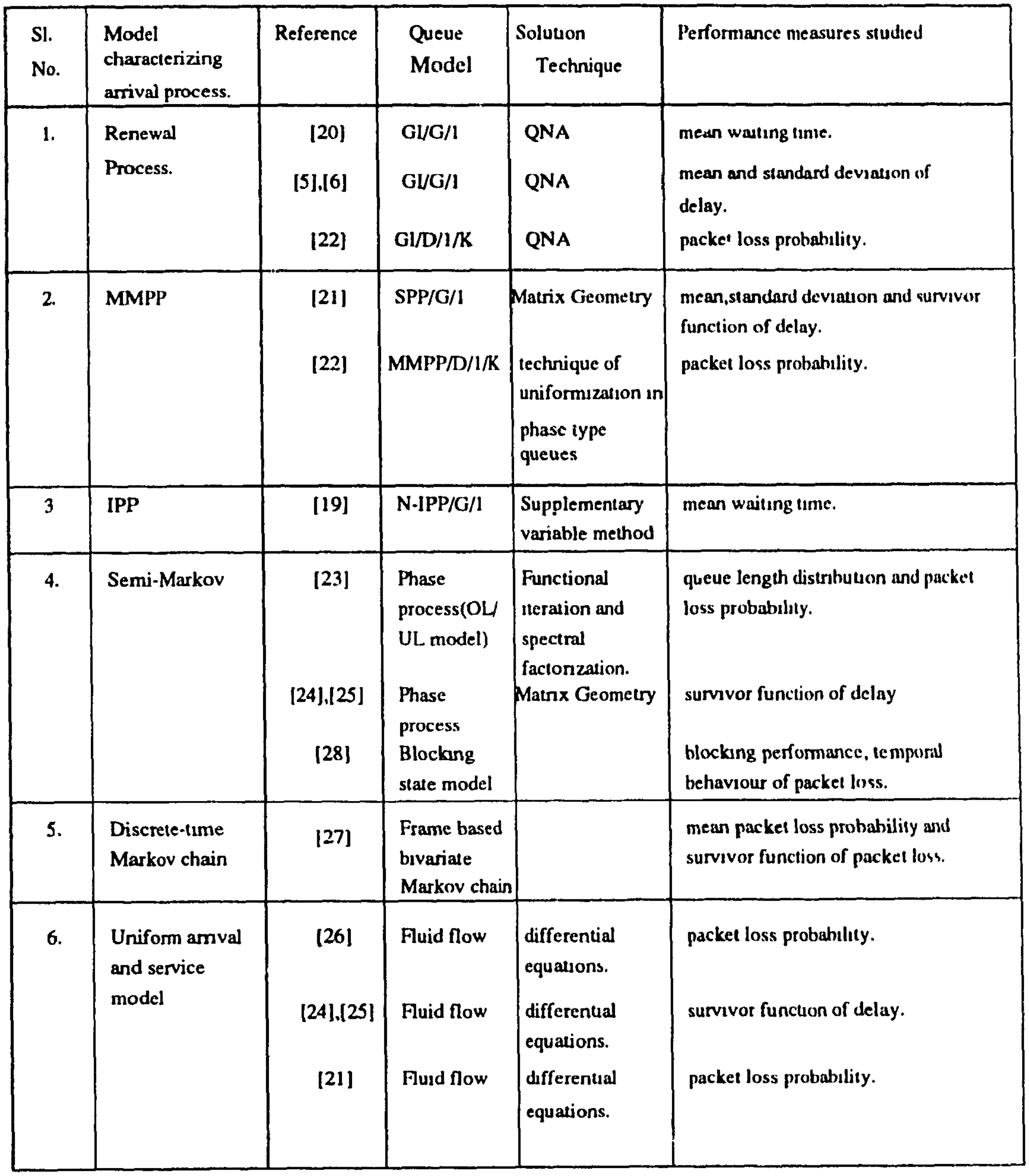
The models like the MMPP (Markov Modulated Poisson Process) and IPP

(Interrupted Poisson Process) try to modify the Poisson process in order to capture the correlations in the superposed packet voice traffic while maintaining the simplicity of the Poisson process. These models have become quite popular both as candidates for synthetic traffic generators and also as analytical models because of the wealth of analytical tools added to the literature in the past couple of decades, to analyze the queues fed by such models. Another advantage of such models is that, once the parameters for these approximating processes (for eg. MMPP) are obtained from the actual superposed traffic, the approximating process can be fed to any system and the performance measures obtained either by a simulation or analysis of the system under consideration. Yet other models like Semi Markov models, Discrete Markov chain models, Uniform arrival and service process models and Renewal process qualify more as analytical models only. These models attempt at approximating the superposition process together with the underlying queueing system under consideration and their applicability is limited to such systems only.

A brief description of each of the models used to characterize the superposition arrival process is given in the following sections.

Markov Modulated Poisson Process

Since the superposed process consists of the aggregation of many on-off processes, the aggregate voice packet arrival rate is a *modulated* process obtained by modulating the individual voice source rate by the number of voice sources in their talkspurt. Hence at a given time, the packet arrivals from the superposed process may be approximated by a Poisson process whose rate is determined by the number of voice sources in their ON state, thus giving rise to a Markov modulated characterization of the aggregate process. Markov modulated Poisson process (MMPP) is a nonrenewal, doubly stochastic Poisson process where the rate process is determined by the state



Sl. No.	Model characterizing arrival process.	Reference	Queue Model	Solution Technique	Performance measures studied
1.	Renewal Process.	[20] [5],[6] [22]	GI/G/1 GI/G/1 GI/D/1/K	QNA QNA QNA	mean waiting time. mean and standard deviation of delay. packet loss probability.
2.	MMPP	[21] [22]	SPP/G/1 MMPP/D/1/K	Matrix Geometry technique of uniformization in phase type queues	mean,standard deviation and survivor function of delay. packet loss probability.
3	IPP	[19]	N-IPP/G/1	Supplementary variable method	mean waiting time.
4.	Semi-Markov	[23] [24],[25] [28]	Phase process(OL/ UL model) Phase process Blocking state model	Functional iteration and spectral factorization. Matrix Geometry	queue length distribution and packet loss probability. survivor function of delay blocking performance, temporal behaviour of packet loss.
5.	Discrete-time Markov chain	[27]	Frame based bivariate Markov chain		mean packet loss probability and survivor function of packet loss.
6.	Uniform arrival and service model	[26] [24],[25] [21]	Fluid flow Fluid flow Fluid flow	differential equations. differential equations. differential equations.	packet loss probability. survivor function of delay. packet loss probability.

Table 1: Models for superposition for voice sources

Table 3.1: Models

of a continuous time Markov chain. In other words in state k of the underlying Markov chain, arrivals occur according to a Poisson rate λ_k.

In [21], [22] the superposition arrival process is modeled as a 2 state MMPP and it has been shown that the 2 states of the MMPP are enough to capture the correlations, if the parameters of the approximating MMPP are obtained by matching several of its statistical characteristics with the original superposition . The MMPP is a correlated non-renewal stream and hence it can account for the correlations of the input stream. Here, we shall derive the equation for the IDC of the 2-state MMPP and show that it captures correlations over short durations. The equation of IDC for the 2-state MMPP has already been derived in [21] but here we will take a slightly different approach to arrive at the same equation.

The 2-state MMPP is a special class of the random hazard function considered in [29] and [30]. The statistics of the MMPP like mean, variance, IDC(Index of Dispersion for Counts) may be derived from the probability generating function of the process. Generally state such processes alternate between two levels λ_1 and λ_2, with the sojourn times in each state forming an alternating renewal process with interval p.d.f.s $f_1(x)$ and $f_2(x)$ respectively. If ν_1 (ν_2) is the average sojourn time in state 1 (state 2), $f_1^*(s)$ ($f_2^*(s)$) is the Laplace transform of the p.d.f of the sojourn time in state 1 (state 2) and if $R_1^*(s)$ ($R_2^*(s)$) is the Laplace transform of the survivor function in state 1 (state 2), then the Laplace transform of the probability generating function $\phi^*(z,s)$, of the number of arrivals $N(t)$ in time t is given by [30]

$$\begin{aligned}\phi^*(z,s) &= \frac{1}{\nu_1+\nu_2}\left(\frac{\nu_1}{s+\lambda_1(1-z)}+\frac{\nu_2}{s+\lambda_2(1-z)}\right)\\ &\quad -\frac{(\lambda_1-\lambda_2)^2}{\nu_1+\nu_2}\left(\frac{(1-z)^2}{(s+\lambda_1(1-z))(s+\lambda_2(1-z))}\right)\\ &\quad \mathrm{x}\left(\frac{R_1^*(s+\lambda_1(1-z))R_2^*(s+\lambda_2(1-z))}{(1-f_1^*(s+\lambda_1(1-z)f_2^*(s+\lambda_2(1-z)))}\right)\end{aligned} \tag{3.4}$$

Differentiating the above expression partially with respect to z and setting z = 1, gives the Laplace transform of the average number of packets generated.

$$\mathcal{L}\{E[N(t)]\} = \frac{\lambda_1\nu_1 + \lambda_2\nu_2}{(\nu_1+\nu_2)s^2}$$

Inverting the above equation, the mean of the counting process is obtained as

$$E[N(t)]\} = \left(\frac{\lambda_1\nu_1 + \lambda_2\nu_2}{\nu_1+\nu_2}\right)t \tag{3.5}$$

The Laplace transform of variance of $N(t)$ can be obtained by differentiating Eqn. 3.4 twice and setting z = 1.

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1\nu_1 + \lambda_2\nu_2}{(\nu_1+\nu_2)s^2} + \frac{2(\lambda_1-\lambda_2)^2}{(\nu_1+\nu_2)^2}\frac{\nu_1\nu_2}{s^2} \\ &\quad \text{x}\left[\frac{1}{s} - \left(\frac{\nu_1+\nu_2}{\nu_1\nu_2}\right)\left(\frac{R_1 * (s)R_2^*(s)}{1-f_1^*(s)f_2^*(s)}\right)\right] \end{aligned}\tag{3.6}$$

An explicit equation for the variance may be obtained by inverting the above equation depending on the sojourn time densities $f_1(t)$ and $f_2(t)$.

Now, for MMPP

$$\begin{aligned} f_1(t) &= r_1\exp(-r_1t) & f_2(t) &= r_2\exp(-r_2t) \\ R_1(t) &= \exp(-r_1t) & R_2(t) &= \exp(-r_2t) \\ \nu_1 &= 1/r_1 & \nu_2 &= 1/r_2 \end{aligned}$$

The corresponding Laplace transforms are

$$\begin{aligned} f_1^*(s) &= \tfrac{r_1}{s+r_1} & f_2^*(s) &= \tfrac{r_2}{s+r_2} \\ R_1^*(s) &= \tfrac{1}{s+r_1} & R_2^*(s) &= \tfrac{1}{s+r_2} \end{aligned}$$

Substituting the above in Eqn. 3.6, we have the Laplace transform of variance as

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1 r_2 + \lambda_2 r_1}{(r_1+r_2)s^2} + \frac{2(\lambda_1-\lambda_2)^2}{(r_1+r_2)^2}r_1r_2 \\ &\quad \text{x}\left[\frac{1}{s^3} - \frac{r_1+r_2}{s^3(s+(r_1+r_2))}\right]\end{aligned}$$

Inverting we have

$$\begin{aligned} Var[N(t)] &= \left(\frac{\lambda_1 r_2 + \lambda_2 r_1}{r_1 + r_2}\right) t + \frac{2(\lambda_1 - \lambda_2)^2}{(r_1 + r_2)^3} r_1 r_2 t \\ &\quad - \frac{2(\lambda_1 - \lambda_2)^2}{(r_1 + r_2)^4} r_1 r_2 (1 - \exp(-(r_1 + r_2)t)) \end{aligned} \tag{3.7}$$

From Eqn. 3.5 and Eqn. 3.7 the IDC of 2-state MMPP may be obtained as

$$\begin{aligned} I(t) &= 1 + \frac{2(\lambda_1 - \lambda_2)^2 r_1 r_2}{(r_1 + r_2)^2 (\lambda_1 r_2 + \lambda_2 r_1)} \\ &\quad - \frac{2(\lambda_1 - \lambda_2)^2 r_1 r_2}{(r_1 + r_2)^3 (\lambda_1 r_2 + \lambda_2 r_1) t} (1 - \exp((-(r_1 + r_2)t)) \end{aligned} \tag{3.8}$$

The behaviour of IDC with time is indicative of the correlations in a process. For MMPP, as seen from Eqn. 3.8, the IDC would reach an asymptote for longer lags, after increasing initially, over small lags. The asymptote of the IDC is

$$I(\infty) = 1 + \frac{2(\lambda_1 - \lambda_2)^2 r_1 r_2}{(r_1 + r_2)^2 (\lambda_1 r_2 + \lambda_2 r_1)} \tag{3.9}$$

The IDC approaches this asymptote at the rate of $r_1 + r_2$ as seen from Eqn. 3.8. When plotted in a log-log scale, the IDC would initially increase linearly over certain lags and then settle down. A linear behaviour of IDC with time (in a log-log plot) indicates the presence of serial correlations. For MMPP, these correlations are captured only for a certain interval, after that the correlations do not exist. This is due to the fact that the autocorrelation function of a MMPP falls off exponentially and is summable and hence the covariances are negligible for longer lags. As seen in the previous chapter, these conditions make MMPP a short-range dependent model. However capture of these short-term correlations are enough to model voice packet traffic, as demonstrated in [21] [22].

In [21] the approximating MMPP is chosen in such a way that several of its characteristics identically match with those of the original superposition. There are 4 parameters for the 2 state MMPP chosen, namely, the mean sojourn times in states 1 and 2,r_1^{-1} and r_2^{-1}, and the Poisson arrival rates in states 1 and 2 λ_1 and λ_2 respectively. In order to de ermine these 4 parameters of the model, the following 4 characteristics of the model are matched with those of the superposition process:

1. the mean arrival rate,
2. the variance to mean ratio of the number of arrivals in an interval $(0, t_1)$,
3. the long term variance to mean ratio of the number of arrivals and
4. the third moment of the number of arrivals in $(0, t_2)$

First, all the above characteristics are determined for the superposition arrival process, in [21] as follows. Consider the single voice source as a renewal process (single voice source - model 1, described earlier), then the interarrival distribution is as given by Eqn. 3.1. Taking the Laplace Stieltjes transform (LST) of Eqn. 3.1 we have

$$\tilde{f}(s) = \int_0^\infty \exp(-st)\, dF(t) = [1 - \alpha T + \alpha T\beta/(s + \beta)]\exp(-sT) \qquad (3.10)$$

Expected interarrival time of a single source $= -\tilde{f}'(0) = T + \alpha T/\beta$.
Equivalently, the mean packet arrival rate λ is given by

$$\lambda = 1/(T + \alpha T/\beta) \qquad (3.11)$$

Now let $A(0, t)$ denote the number of arrivals of a stationary renewal process in the interval $(0, t)$ and let

$$M_r(t) = E[A^r(0, t)]$$

be the rth moment of arrivals in $(0, t)$ and let

$$M_r(s) = L[M_r(t)]$$

where L(.) denotes the Laplace transform. Using the results of the renewal process we have

$$
\begin{aligned}
M_1(s) &= \lambda/s^2 &(3.12)\\
M_2(s) &= \frac{\lambda}{s^2}\frac{1+\tilde{f}(s)}{1-\tilde{f}(s)} &(3.13)\\
M_3(s) &= \frac{\lambda}{s^2}\frac{1+4\tilde{f}(s)+\tilde{f}^2(s)}{\left(1-\tilde{f}(s)\right)^2} &(3.14)
\end{aligned}
$$

But $M_1(t) = \lambda t$. Using Eqn. 3.11 for λ gives

$$M_1(t) = t/(T + \alpha T/\beta)$$

The Index of dispersion for counts, I(t), satisfies

$$lim_{t\to\infty} I(t) = lim_{t\to\infty} \frac{Var[A(0,t)]}{M_1(t)} = \frac{Var(X)}{E^2(X)}$$

where X is the interarrival time. Therefore

$$lim_{t\to\infty} \frac{Var[A(0,t)]}{M_1(t)} = \frac{1-(1-\alpha T)^2}{(\alpha T + \beta T)^2}$$

The values of $M_2(t)$ and $M_3(t)$ can be obtained by numerical transform inversion of Eqn. 3.14 and Eqn. 3.14.

For the superposition process, the number of arrivals is given by

$$A^s(0,t) = \sum_{i=1}^{N} A_i^s(0,t)$$

where $A_i(0,t)$ is the number of arrivals during the interval from source i.

Hence,

$$M_1^s(t) = E[A^s(0,t)] = n\,M_1(t) \qquad (3.15)$$

$$\frac{var[A^s(0,t)]}{E[A^s(0,t)]} = \frac{Var[A(0,t)]}{E[A(0,t)]} \qquad (3.16)$$

The third central moment of the superposition process is given by

$$
\begin{aligned}
\mu_3^s(0,t) &= E\{[A^s(0,t) - E(A^s(0,t))]^3\}\\
&= n[M_3(t) - 3M_2(t)M_1(t) + 2M_1^3(t)] &(3.17)
\end{aligned}
$$

Now for the MMPP, from [21] we have the probability generating function of the number of arrivals in an interval

$$g(z,t) = \pi \exp\{[\mathbf{R} + (z-1)\mathbf{\Lambda}]t\}\mathbf{e} \tag{3.18}$$

where

$$\pi = \frac{1}{r_1+r_2}(r_2, r_1) \text{ (equilibrium probability vector)}$$

$$\mathbf{e} = (1,1)^T$$

$$\mathbf{R} = \begin{bmatrix} -r_1 & r_1 \\ r_2 & -r_2 \end{bmatrix}$$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

If A_t is the number of arrivals in the stationary 2 state MMPP over the interval $(0,t)$, then

$$\overline{A_t} = E[A_t] = \frac{\lambda_1 r_2 + \lambda_2 r_1}{r_1 + r_2}t \tag{3.19}$$

$$\frac{Var(A_t)}{\overline{A_t}} = 1 + \frac{2(\lambda_1-\lambda_2)^2 r_1 r_2}{(r_1+r_2)^2(\lambda_1 r_2 + \lambda_2 r_1)} - \frac{2(\lambda_1-\lambda_2)^2 r_1 r_2}{(r_1+r_2)^3(\lambda_1 r_2 + \lambda_2 r_1)}.(1-\exp(-(r_1+r_2)t))$$

$$lim_{t\to\infty}\frac{Var(A_t)}{\overline{A_t}} = 1 + \frac{2(\lambda_1-\lambda_2)^2 r_1 r_2}{(r_1+r_2)^2(\lambda_1 r_2 + \lambda_2 r_1)} \tag{3.20}$$

$$\text{3rd moment of number of arrivals in}(0,t_2) = g^{(3)}(1,t_2) \tag{3.21}$$

Equations 3.15, 3.16, 3.17 are equated against equations 3.19, 3.20, 3.21 respectively to determine $\lambda_1, \lambda_2, r_1$ and r_2. Once these are known, Matrix Geometric Techniques [31] can be used to solve the resulting MMPP/G/1 queue as dealt in detail in [21]. In [21] the model was used to evaluate the average delay of an infinite buffer, voice multiplexer with good accuracy. The method however did not work well for the finite buffer case.

In [22] two MMPP models were used to study the performance of fixed buffer multiplexers. The first model was used to evaluate the packet loss of moderate to large buffers while the second model was used for large buffer. Here, the arrival process is considered to consist of an underload and overload period. An overload state occurs when the number of sources in talkspurt exceeds the capacity of the system.

In the first model of moderate to large buffers, the variance in the arrival process during the overload states are considered, since the packet loss in such buffers are expected to occur in overload states only. The parameters that are matched are as follows

1. Value of $E[A_H(0,t)]/t$ at $t = 0$, with that of the superposition process, where $A_H(0,t)$ is the number of arrivals in time t for the MMPP given that the process started in the high arrival rate (H) at $t = 0$.

2. $E[A_H(0,t)]/t$ at $t = \infty$ with that of the superposition process.

3. The derivative of $E[A_H(0,t)]/t$ at $t = 0$ with that of the superposition process.

4. The value of the $Var[A_H(0,t)]$ at $t = t_m$, with that of the superposition arrival process. The value of t_m is chosen so that $Var[A_H(0,t)]$ match well over a period of one second, which is the average ON/OFF period of the voice source.

For the second model discussed in [22], the first 3 parameters matched are as given above and in addition $Var[A_L(0,t)]$ at $t = t_m$ is matched with that of the superposition arrival process. Simulation results of [22] suggest that this model performs better than [21] for finite buffer case.

Hence, as seen above the MMPP can successfully capture the correlations present in the superposition arrival process from many voice sources. The biggest advantage of MMPP is that, once its parameters are obtained from the actual superposition, the model can be fed into any system and the system can be either analyzed or simulated. Also, the MMPP model is more suited for synthetic traffic generation becau^e of its simplicity and ease of implementation.

Interrupted Poisson process

IPP is a special case of a 2 state MMPP, where one state is an ON state with associated positive Poisson rate, and the other state is an OFF state with associated rate zero. As discussed in an earlier section, such models have been used to characterize the packet arrival process from a single source. In a similar fashion the aggregated arrival process can be approximated by the superposition of IPPs, called the N-IPP. The N-IPP is an MMPP. If we denote the state of the N-IPP at time t as $J(t)$ where $J(t) = j$ is the number of IPPs in their ON state, then $J(t)$ is an $(N + 1)$ state continuous time Markov chain (a birth and death process). The arrival in state j of the Markov chain is Poisson with rate $j\lambda$ while the birth and death rates are $(N - j)\gamma$ and $j\omega$ respectively. (where γ^{-1} and ω^{-1} are the mean ON time and OFF time of the model). [19] adopts this approach for modeling the superposition arrival process.

In [19], unlike the MMPP approach, the component process (arrival from a single voice source) is characterized rather than the superposition process. Hence, the packet arrival process from a single source is approximated as an IPP with 3 defining parameters, namely, the mean arrival rate in the ON state λ, the mean ON time γ^{-1} and the mean OFF time ω^{-1}. To determine these 3 defining parameters of the IPP from the statistical characteristics of the packet arrival process from a single voice source, 3 methods are considered in [19].

1) The mean interval method: The mean ON time γ^{-1}, OFF time ω^{-1} and the mean arrival interval during ON time $1/\lambda$ of the IPP are matched with the mean talkspurt α^{-1}, mean silence period β^{-1} and the packet arrival interval period T, during talkspurts.

2) 3 moments method: The first 3 moments of interarrival time distributions of the IPP are matched with those of the packet arrival process. If m, c, k are the mean, the coefficient of variation and the third central moment of the packet interarrival time of the packet arrival process respectively, then from [19] we have

$$\lambda = \frac{2(k - 3c^2 + 1)}{(2k - 3c^4 - 1)m} \tag{3.22}$$

$$\omega = \frac{3(c^2 - 1)}{(k - 3c^2 + 1)m} \tag{3.23}$$

$$\gamma = \frac{9(c^2 - 1)^3}{(k - 3c^2 + 1)(2k - 3c^4 - 1)m} \tag{3.24}$$

3) 2 moments and peakedness method: An important characteristics of the arrival stream is the peakedness. The exponential peakedness function $Z_{exp}(\mu)$ is defined as the variance to mean ratio of the number of busy servers in a fictitious infinite exponential server system with service rate μ, to which the arrival stream is hypothetically offered. For the packet arrival process from a single source, $z_{exp}(\mu)$ is given by (from[19])

$$Z_{exp}(\mu) = (1 - [1 - \alpha T + \frac{\alpha T \beta}{\mu + \beta}] \exp(-\mu T)^{-1} - \frac{\beta}{\mu(\alpha + \beta)T}$$

Hence, in this method as the name suggests, the first 2 moments of the interarrival time distributions and a peakedness are matched to yield ([19])

$$\lambda = \frac{1}{m} + \frac{(c^2 - 1)(z - 1)\mu}{c^2 + 1 - 2z} \tag{3.25}$$

$$\omega = \frac{2(z-1)\mu}{(z-1)(c^2-1)\mu m + c^2 + 1 - 2z} \qquad (3.26)$$

$$\gamma = \frac{2\mu^2 m(c^2-1)(z-1)^2}{[\mu m(c^2-1)(z-1) + c^2 + 1 - 2z](c^2 + 1 - 2z)} \qquad (3.27)$$

After having approximated the arrival process from a single source with the parameters $(\gamma, \omega, \lambda)$ determined by one of the above methods, the superposition can be analyzed as a N-IPP/G/1 queue as outlined in [19]. Simulation results in [19] show that the 2 moments and peakedness method is the most accurate of the 3 methods discussed.

Semi-Markov Process

In [23], [24], [25] and [28] the superposition of on-off process (i.e., individual voice sources) is approximated by a Semi-Markov process or a two dimensional Markov chain.

As discussed before, each of the active sources feed packets to the multiplexer at the rate of V packets per second and these are removed by the multiplexer at the rate of VC packets per second. The number of packets arriving to the multiplexer depends on the number of sources in their active state. Hence, the number of active sources as a function of time $(J_{(}t)$ can be modeled as a continuous time Markov chain as shown in Figure 3.6 ([23]). It is called the phase process in [23]. In [23] an approximate generating function of the probability density function of the queue length is computed by focusing on instants of completion of an overload/underload (OL/UL) cycle, which is defined as follows. Let J_0 be the smallest integer greater than C (the channel capacity) and let $J_u = J_0 - 1$. Then overload starts at the instant the number of active voice sources changes from J_u to J_0 (since, in such

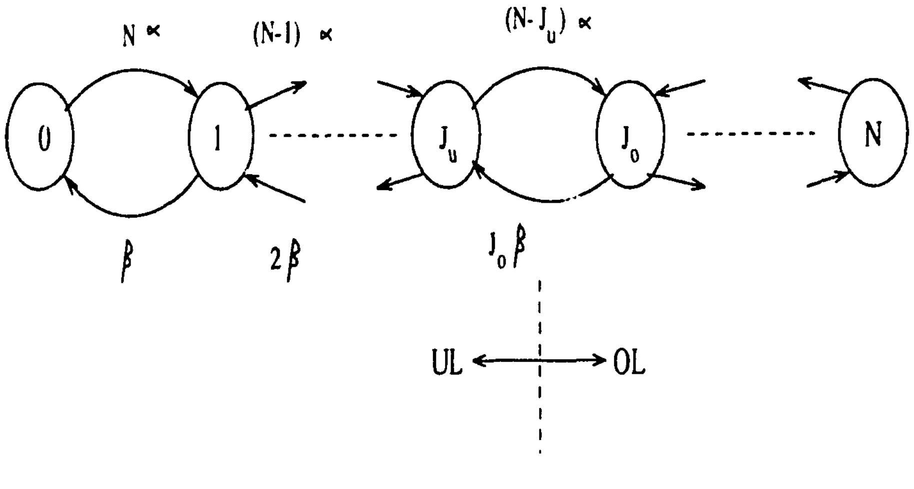


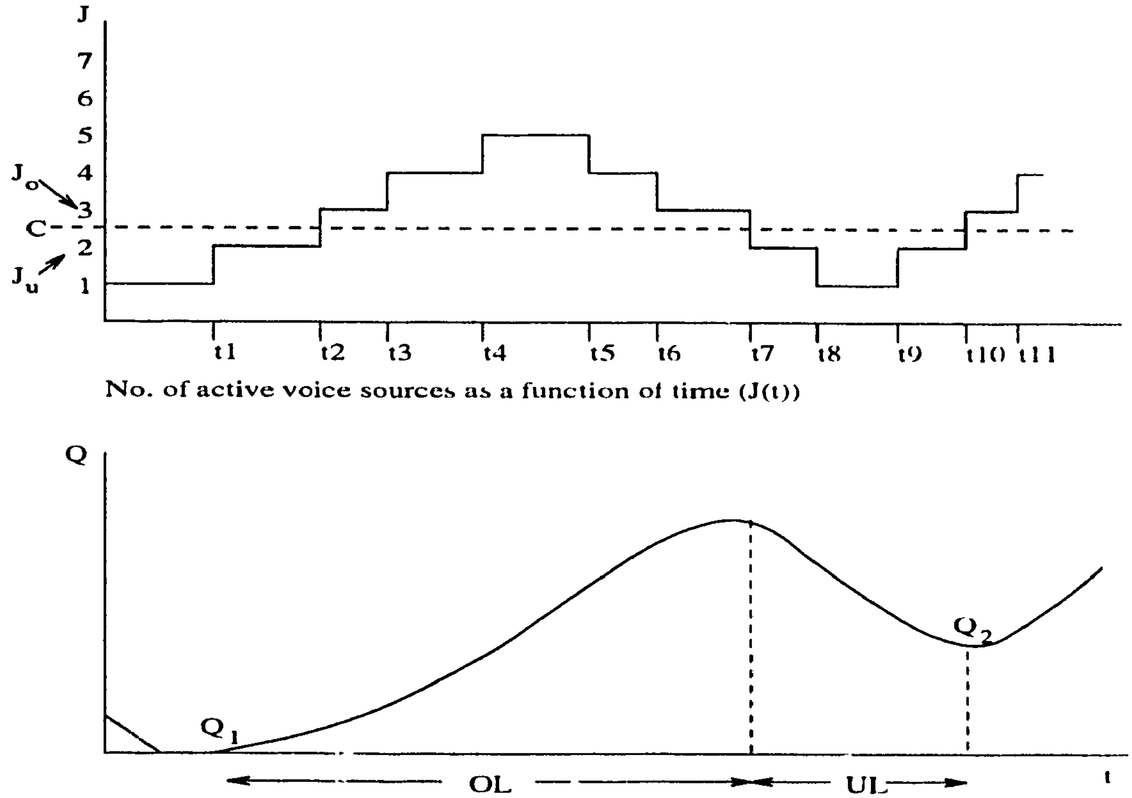
Figure 3.6: Phase process

a condition more than C packets arrive in a frame of C transmission slots) and ends at the instant when the number of active voice sources change from J_o to J_u. Underload begins at this time and persists till overload starts again. The period between the start of successive overload is called the OL/UL cycle. The number of packets in the queue at the end of the n th OL/UL cycle is denoted by Q_n and the queue at the end of $n+1$th OL/UL cycle by Q_{n+1} (Figure 3.7 [23]). Then,

$$P\{Q_{n+1}|Q_n, Q_{n-1}, \ldots\} = P\{Q_{n+1}|Q_n\}$$

Therefore the sequence Q_n can be viewed as the states of a Semi-Markov chain whose state transition intervals correspond to the OL/UL cycle times, which are random variables. [23] discusses two methods - functional iteration and spectral factorization to determine the probability generating function of the probability density function of the queue length. However [23] does not evaluate the stochastic equilibrium distribution of the queue length.

[24] and [25] discuss a method to determine the stochastic equilibrium distribution of the multiplexer queue by approximating the superposition arrival process by a semi-Markov chain. The semi-Markov process approximated is as described below. Consider the phase process as shown in Figure 3.6. The following approximations are made



No. of active voice sources as a function of time (J(t))

The multiplexer queue as a function of time

Figure 3.7: No. of active voice sources and multiplexer queue length as a function of time

a) when $J(t) < C$ (corresponding to the underload state UL), the length of the queue (when it is non-empty) decreases at the rate of $V(C - J(t))$ packets per second. If the queue is empty it remains so as long as $J(t) < C$. No queue increment is allowed .

b) when $J(t) = C$ (this is possible only if C is an integer), the rate of change of the queue length is zero.

c) when $J(t) > C$ (corresponding to the overload state OL), the length of the queue increases at the rate of $V(J(t) - C)$ packets per second. No queue decrement is allowed.

Let the states of the process be denoted by (q_t, v_t) where $v_t = J(t)$, the number of sources in talkspurt at time t and q_t is the number of packets in the queue. Transitions from (i,j) to $(i, j-1)$ or $(i, j+1)$ are called phase transitions, since the queue length does not change. The transitions from (i,j) to $(i+1, j)$ is a queue increment and to $(i-1, j)$ is a queue decrement. The process is shown in Figure 3.8. It can be observed that the transition probabilities for the process shown depend on the current state of the process. Hence there exists a Markov chain embedded at the instants of phase state changes, queue increments and queue decrements. Also the expected sojourn time in any state depends only upon the state. Therefore the process is a semi-Markov process. The parameters of this semi-Markov process are the packet generation rate, the mean talkspurt and silence periods, the communication link capacity and the total number of voice sources.

To compute the equilibrium probability $p_{i,j}$ that $q_t = i$ and $v_t = j$ the following equation from renewal theory is used in [24] and [25]

$$p_{i,j} = \frac{q_{ij}\, m_{ij}}{\sum_{k=0}^{\infty} \sum_{l=0}^{N} q_{kl} m_{kl}} \tag{3.28}$$

where

q_{ij} = equilibrium probability for the embedded Markov chain

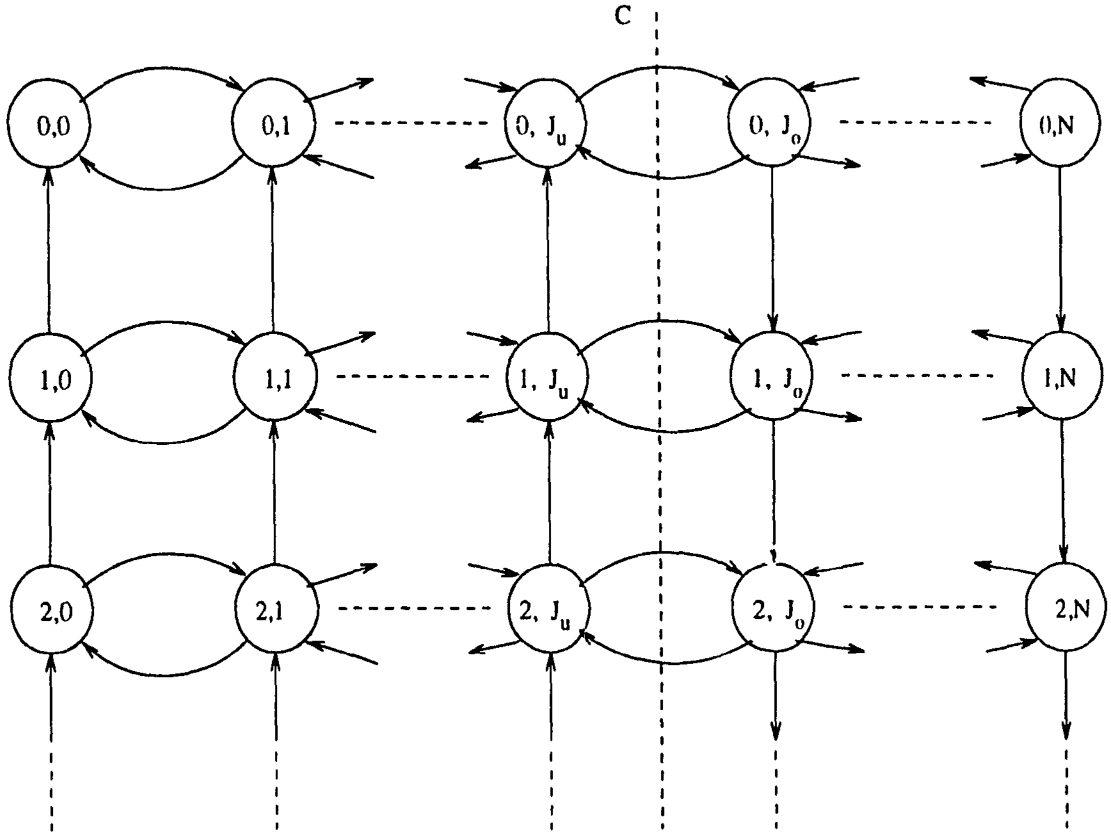
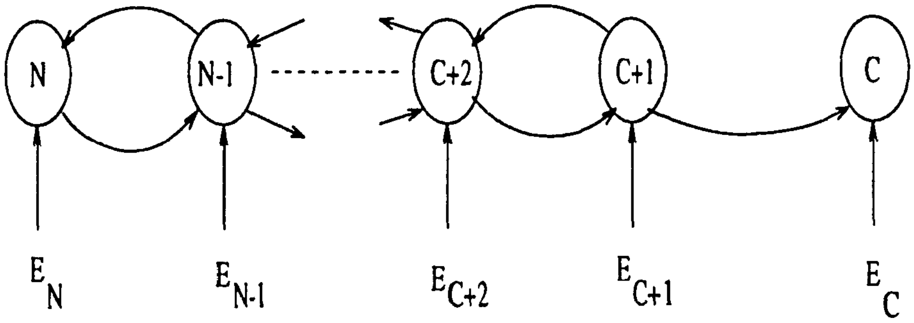


Figure 3.8: Semi-Markov process



E_i = the probability that a blocking period starts in state (i,K)

Figure 3.9: Blocking state diagram

m_{ij} = expected sojourn time in state (i,j)

Matrix Geometric method is used to solve Eqn. 3.28 in [24] and [25]. Comparison of the results obtained by this approach with the simulation shows that the model overestimates the probability that the queue is empty. This is due to the approximations underlying the model.

[28] also approximates the superposition as a semi-Markov chain. However, the system considered is a finite buffer one and the emphasis is placed on the packet loss which is incurred only when the buffer is full. If K is the total buffer capacity in packets and $\pi_{i,K}$, the equilibrium probability of i voice calls in talkspurt when the buffer is full, then we have $\pi_{i,K} = 0$ for $i \leq C$, since the buffer will not be full when the service rate is greater than the arrival rate. Hence the packets would be lost only for states greater than C. [28] considers these states alone and calls it the blocking states (Figure 3.9 [28]). Focusing on the blocking states analytical expressions are derived in [28] for the temporal behaviour of packet loss. Results show that the packet loss rate changes slowly and has large fluctuations. Increasing the buffer size merely extends the non-blocking periods and thereby reduces the overall average packet loss rate. However, once a blocking period occurs, the length of the period as well as the packet loss within this period becomes irrelevant to the buffer size.

Discrete time Markov model

In [27], the aggregation of on-off processes is approximated by a a Discrete time Markov model. Here again the state of the process is the tuple consisting of the number of sources in talkspurt and the number of packets in the queue. But the sampling is done after every frame and hence the process is in discrete time domain. The state of the system at the beginning of the nth frame is given by (t_n, b_n) where t_n is the number of users in talkspurt and b_n is the queue length.

Such a system is studied in [27] for a finite buffered voice multiplexer. Two schemes for discarding the packets are considered. In the first scheme a buffer of size K is properly selected so that all the packets within the buffer can be transmitted within their time (delay) constraint. All the packets arriving after the buffer is full are discarded. In the second scheme, all the arriving packets are stored in the buffer and at the end of a frame, the system randomly selects a packet to drop from the arrivals in the frame. This process is repeated until all the remaining packets meet their delay constraint. This scheme balances the packet loss for each user.

For both the schemes the transition probability,

$$p_{ij,kl} = Pr\{t_{n+1} = k, b_{n+1} = l | t_n = i, b_n = j\} \qquad 0 \le i, \quad k \le N \quad 0 \le j \quad l \le K$$

and the equilibrium state probability

$$\pi_{nm} = Pr\{t = n, b = m\} \qquad 0 \le n \le N, \quad ; 0 \le m \le K$$

in both the schemes are determined by considering the queue length transitions from b_n to b_{n+1} for the following four cases

- *case 1:* $b_n \ge C$ and $t_n \le K - b_n + 1$; enough packets in the queue to keep the server busy and not too many arrivals to cause overflow.

- *case 2:* $b_n < C$ and $t_n \le K - b_n + 1$; not enough packets in the system to keep the server busy and not too many arrivals to cause overflow.

- *case 3:* $b_n \geq C$ and $t_n > K - b_n + 1$; the server keeps busy a nd overflow may occur. Due to overflow some packets will be discarded. Let the number of packets discarded $D_1 = d$. Then D_1 is a random variable with probability density function $\Psi_{D_1}(d)$.
- *case 4:* $b_n < C$ and $t_n > K - b_n + 1$; the server may go idle and overflow may occur. Let D_1 be the number of packets discarded and R the number of packets served in the frame (then the server is free for $C - R$ timeslots during the frame). Then R and D_1 are random variables with a joint probability density function $\Theta_{R,D_1}(r,d)$.

[[27] discusses the computation of the pdfs $\Psi_{D_1}(d)$ and $\Theta_{R,D_1}(r,d)$ for both the schemes. Results show that scheme 2 performs better than scheme 1 as it spreads the packet loss across the users.

Uniform Arrival and Service model

The Uniform Arrival and Service (UAS) model, which assumes that the information flow in and out of the buffer is uniform rather than in discrete packets was used by [23] for modeling data traffic. In the UAS model the source generates information to the transmitter at a rate of one unit of information per unit time and the server removes information from the buffer at a uniform rate not to exceed C units of information per unit of time. As in the semi-Markov process of [24], while the system is in state $J(t) = j > C$, the buffer content increases at the rate of $j - C$ units of information per unit of time (if the queue reaches its limits it will stay on its limit) and when the system is in state $J(t) = j < C$, the buffer content reduces at the rate of $C - j$ units of information per unit of time as long as the buffer is nonempty(if the buffer becomes empty it will stay empty)[26],[24] and [25] approximate the superposition arrival process by this model.

In [26] the UAS model is used to model a finite buffer multiplexer. The equilibrium distribution is des^ribed by a set of differential equations, which together with a set of boundary conditions can be solved to yield the equilibrium distribution of delay and packet loss. The method is briefly outlined below.

If $P_i(t,b)$ be the probability that at time t there are b packets in the queue and i lines are in their talkspurt, where $0 \leq i \leq N, t \geq 0$ and $0 \leq b \leq K$. If δt be a small time interval, then from Figure 3.6 we have

$$\begin{aligned} P_i(t+\delta t, b) &= P_{i-1}\{t, b-(i-C)\delta t\}p(i-1,i)\delta t \\ &+P_{i+1}\{t, b-(i-C)\delta t\}p(i+1,i)\delta t \\ &+P_i\{t, b-(i-C)\delta t\}(1-p^*(i)\delta t)+O(\delta t) \qquad (3.29) \end{aligned}$$

where

$$\begin{aligned} &p^*(i) = p(i,i+1)+p(i,i-1) \\ &p(i,i+1) = (N-i)\beta \qquad i \neq N \\ &p(i,i-1) = i\alpha \qquad i \neq 0 \end{aligned}$$

Dividing Eqn. 3.29 by δt and letting $\delta t \to 0$ we get

$$\begin{aligned} \frac{\partial P_i(t,b)}{\partial t} + (i-C)\frac{\partial P_i(t,b)}{\partial b} &= p(i-1,i)P_{i-1}(t,b) \\ &+p(i+1,i)P_{i+1}(t,b) \\ &-p^*(i)P_i(t,b) \qquad 0 < b < K \quad (3.30) \end{aligned}$$

To find time independent equilibrium probability $lim_{t\to\infty}P_i(t,b)$ define $F_i(b) = lim_{t\to\infty}P_i(t,b)$, then Eqn. 3.30 becomes

$$(i-C)\frac{dF_i}{db} = p(i-1,i)F_{i-1}(b)+p(i+1,i)F_{i+1}(b)-p^*(i)F_i(b) \; ; \qquad 0 < b < K \qquad (3.31)$$

Eqn. 3.31 can be written in matrix form as

$$\mathbf{D}dF(b)/db = \mathbf{M}F(b) \qquad 0 < b < K \qquad (3.32)$$

where

$$\mathbf{D} = diag\{-C, 1-C, 2-C, \cdots\cdots N-C\}$$

$$\mathbf{M} = \begin{bmatrix} -p^*(0) & p(1,0) & & & & & \\ p(0,1) & -p^*(1) & p(2,1) & & & & \\ & p(1,2) & -p^*(2) & p(3,2) & & & \\ & & \cdot & & & & \\ & & & \cdot & & & \\ & & & & \cdot & & \\ & & & & & \cdot & \\ & & & & p(N-2,N-1) & -p^*(N-1) & p(N,N-1) \\ & & & & & p(N-1,N) & -p*(N) \end{bmatrix}$$

The solution to the differential Eqn. 3.32 is

$$F(b) = \sum_{k=0}^{N} \exp(z_k b) a_k \phi_k \qquad 0 < b < m \tag{3.33}$$

$$z_k = \text{eigen value of } D^{-1}M$$

$$\phi_k = \text{right eigen vector of } D^{-1}M$$

The a_k are coefficients got by solving boundary conditions. For an infinite buffer case, closed form expressions exist for z_k, ϕ_k and a_k, as given in [23]. In the case of finite buffers [26] discusses a method of formulating the boundary equations to solve for ϕ_k, z_k and a_k.

Renewal Process

As observed earlier, the packet arrival process from a single source can be modeled as a renewal process with exponentially distributed talkspurts alternating with exponentially distributed silence periods. In [20, 5, 6] the superposition arrival process is approximated as a renewal process with inflated coefficient of variation for the interarrival time. In [5] it has been noted that the main cause for deviation of the superposed packet arrival process from a Poisson process is due to the high variability of the interarrival times between the arrivals. This variability is captured in the approximating renewal process by a high coeffecient of variation. Then, a 2 parameter approximation technique as in [32, 33] called the Queueing Network Analyzer (QNA) approach is adopted. In this method the superposition arrival process is characterized by 2 parameters; one is the average arrival rate (λ) and the other is the squared coefficient of variation of the interarrival time (c_a^2). The squared coefficient of variation of the interarrival time of the renewal process may be approximated from the original superposition process by one of 2 methods:[32]

- *Stationary interval method*:Here the moments of the renewal interval is approximated with the moments of the stationary interval in the superposition arrival process.

- *Asymptotic method*: In this method the moments of the renewal interval is determined by matching the asymptotic behaviour of the moments of the sum of successive intervals.

The formula for the squared coefficient of variation of the interarrival time distribution (c_a^2) in the approximating renewal process for the aggregate packet arrival process is as given below ([5])

$$c_a^2 = w\,c_1^2 + (1 - w) \tag{3.34}$$

where

c_1^2 = squared coefficient of variation of a single voice source

$w = 1/[1 + 4(1 - \rho)^2(N - 1)]$

ρ = traffic intensity

N = number of sources multiplexed

The QNA approximation as given above selects an increasingly higher squared coefficient of variation c_a^2 as N increases (when ρ is kept constant), to directly capture the effect of covariance. (See Figure 5 of [5]).

The other parameter λ of the approximating renewal process can be found as $\lambda = N\,\lambda_1$ where λ_1 is the mean arrival rate of a single source.

Let τ and c_s^2 be the mean and squared coefficient of variation associated with the packet service time. ($c_s^2 = 0$ if the service time is constant)

Now, given the mean and squared coefficient of variation of the interarrival and service times $(\lambda, c_a^2, \tau, c_s^2)$, the congestion measures for the queue such as the mean and standard deviation of delay can be obtained by regarding it as a GI/G/1 queue (with renewal arrival process). See [34] for specific formulas. The mean delay calculated by this model in [5] seems to agree well with simulation results especially at high traffic intensities, where the Poisson approximation fails.

[22] also uses this renewal process model with QNA approximations but introduces an additional heuristic needed to handle finite buffers. Given the distribution $P(Q_\infty = i)$ (probability that the queue length is equal to i) for an infinite buffer case, $P(Q_k = K)$ for a multiplexer with K buffers is approximated in [22] by

$$P(Q_k = K) = \frac{P(Q_\infty = K)}{P(Q_\infty \leq K)} \tag{3.35}$$

where $P(Q_\infty = K)$ is obtained as outlined before. An approximate method for solving Eqn. 3.35 is given in [22].

3.3 Selection of a voice traffic model

In the previous section, the various modeling approaches used to model the superposition of voice sources was presented. As seen there, three main approaches have been adopted for the representation of the superposition of (on-off) voice sources. One approach explicitly takes into account the individual components of each of the sources (this is the approach adopted in modeling the aggregate arrival process as IPP, Semi-Markov process and Discrete state Markov process). The second approach is that of matching few of the statistical parameters of the aggregate arrival process with that of a suitably chosen arrival process such as that of MMPP. The last approach resorts to a fluid flow approximation (UAS models). The first approach has the limitation that the computational complexity dramatically increases in practical cases. The fluid flow approach cannot account for the packet level. The second approach is more elegant and versatile. This approach approximates the superposition arrival process by a simpler model. However, the correlations present in the aggregate arrival process are well captured by this model (MMPP). As was illustrated in the previous section, the MMPP is a short-range dependent model and captures correlations in the arrival process over short intervals. Since correlations in the voice traffic also exist only over short intervals [21], the MMPP is ideally suited to capture these correlations. Once the parameters of the 2-state MMPP are derived from the actual superposition, the MMPP model may be used to study the queueing behaviour of the aggregated arrival process. Also, being a very simple model, the MMPP is an excellent candidate for synthetic generation of voice traffic. Hence in building the traffic generator we select the 2-state MMPP model for characterizing the arrival from the superposition of many voice sources.

3.4 Nature of Video traffic

The introduction of BISDN/ATM technologies to broadband networks and the advancements in source coding algorithms for video, have made feasible the use of variable bit rate coding for video transmission. This would engender a flexible communication network with a high efficiency, as network resources can be shared dynamically by numerous users.

A VBR video codec produces a variable bit rate output by adapting the generated bit rate to the the local and temporal image complexity, while maintaining a

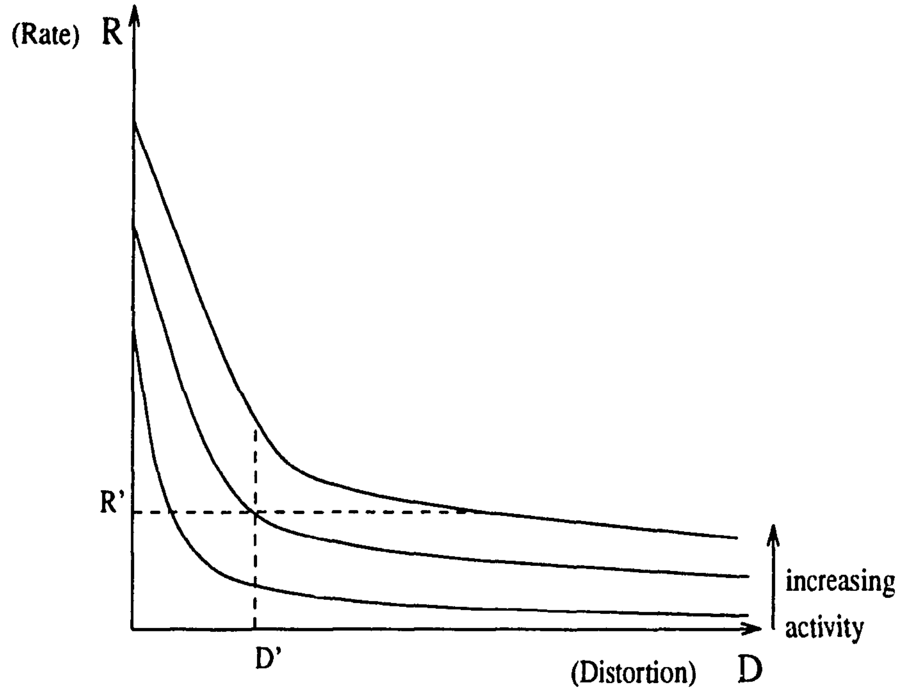


Figure 3.10: Rate Distortion Curves

constant image quality. This can be observed from the rate distortion curves shown in Figure 3.10. These curves depict the variation in the output bit rate as a function of distortion in the output. From the figure it is evident that in order to maintain a low distortion (or high quality) in the output, a higher bit rate codec is required, however a lower bit rate codec produces high distortion in the output. While a constant bit rate coder, produces a constant bit rate output at the expense of quality, a VBR codec maintains a constant quality by varying the bit rate.

The advantages of employing VBR video codecs are many. First of all at low bit rates, use of constant bit rate video codecs, produces a highly varying picture quality which is particularly annoying to the viewer. Use of VBR video codecs helps maintain a constant quality. Secondly, at high bit rates use of VBR video yields high bandwidth gains by using channel sharing among multiple users. In certain cases VBR coding also alleviates the need of sophisticated coding algorithms, as the same effects in picture quality could be achieved by using higher bit rates.

VBR video sources are highly bursty. The burstiness of VBR video sources is a subjective measure [35] that depends on the content of the video (e.g., picturephone, teleconference, broadcast television etc.,) and the encoding scheme used (DCT, Motion compensated DCT, DPCM, MPEG etc.). As the video signals are expected to occupy most of the bandwidth in the future broadband networks, accurate modeling of a VBR video source based on its statistical characteristics is required.

The characteristics of VBR video depend on the information content of the picture and the encoding algorithm used. The bit rate of the coded video is dependent on the motion activity in the scene, namely low, medium and high motion. Due to the continuity of motion within a scene only small portion of the picture changes from frame to frame. Hence variations in bit rate are smaller within a scene. The bit rate of the coder also depends on the changes in the content of the video (like cuts, scene changes, etc.). Highest bit rates arise during scene changes and last only one or two frames depending on the coding algorithm. However, the data rate output of a VBR video encoder does not actually reflect the changes in the information content of original video signal, since the compression of the bit rate achieved by various algorithms are different.

The data buffering scheme used by the encoder also influences the bit rate variations of the encoder. For example in an encoder that uses frame buffering, all the variations arising from the locality of an image within a frame are smoothed, whereas in a multi-frame buffered codec variations in bit rate between frames are also smoothed.

There is a strong correlation among the bit rates of successive frames due to the nature of actual video scenes and interframe coding. Correlations that occur because data on part of an image is highly correlated with data on the same part on the next line are called spatial correlations. Correlations that occur because data on one part of an image is highly correlated with data on the same part of the next image are called temporal correlations. Spatial and temporal correlations together

Type	Time scale	Causes	Characteristics
Long term variability (multiple scenes)	Several seconds	Scene changes	Discontinuous variation,differing statistical characteristics before and after the change
Short term variability (intrascene)	Between 1 frame period and several seconds.	Subject motion, camera motion, pattern variation.	Smooth variations with temporal correlations,with occasional large variations due to subject and camera motion.
Intraframe variability	Less than 1 frame period	Spatial variation of the characteristics within an image.	Variations that have a periodicity due to image scanning or block processing.

Table 3.2: Classification of bit rate variations

with the encoding scheme greatly influence the bit rate output of VBR video codec.

Table 3.2 adapted from [35] summarizes the bit rate variations that occur in a VBR video codec and the corresponding time scale they occur. Hence modeling a VBR video source is a difficult and complex task as the bit rate process possesses a high degree of variability at different levels. Thus, modeling of a VBR video source may be done at one of 3 levels, namely at a scene level, frame level or intraframe level, as depicted in Figure 3.11.

3.5 Modeling of video traffic

Various models have been proposed in the literature to characterize VBR sources with scene changes and without scene changes (at a frame level), Figure 3.12. However the characteristics of VBR video at the intraframe level have not been well understood. The following sections give a brief overview on the various models

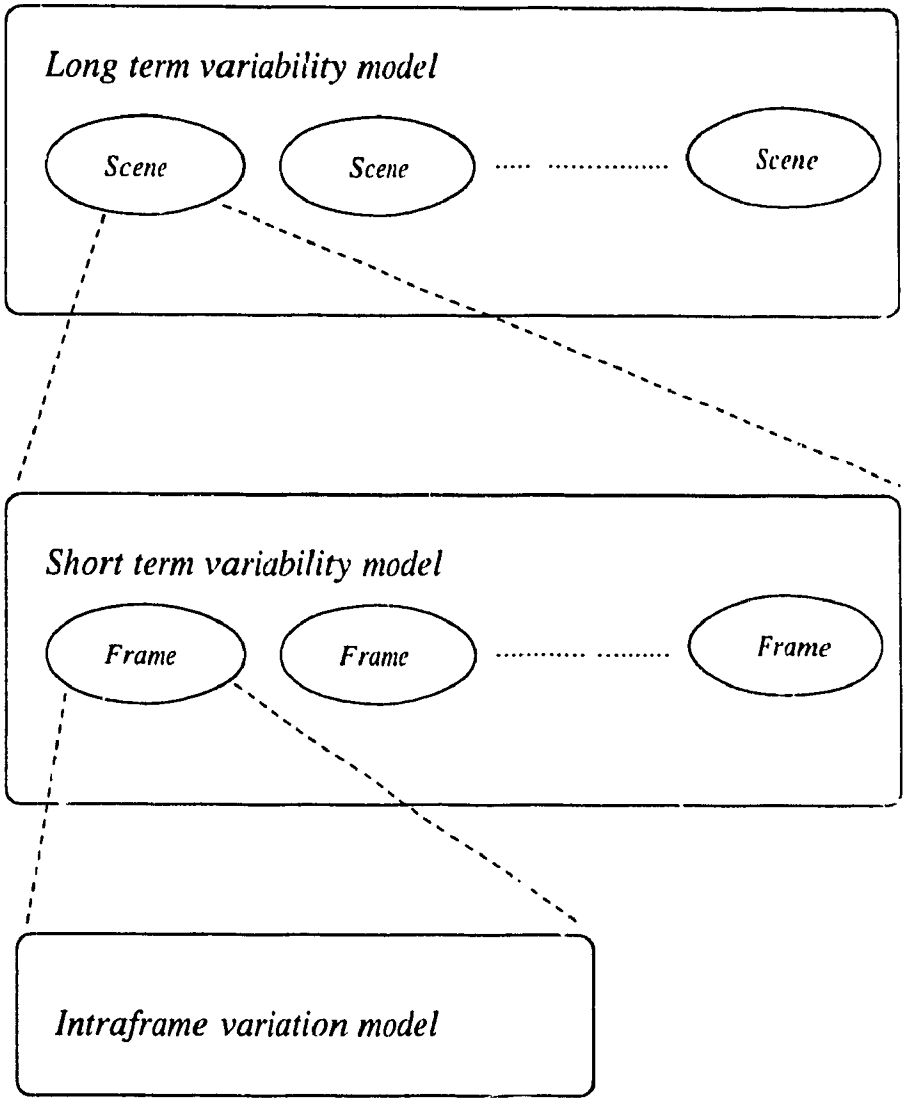
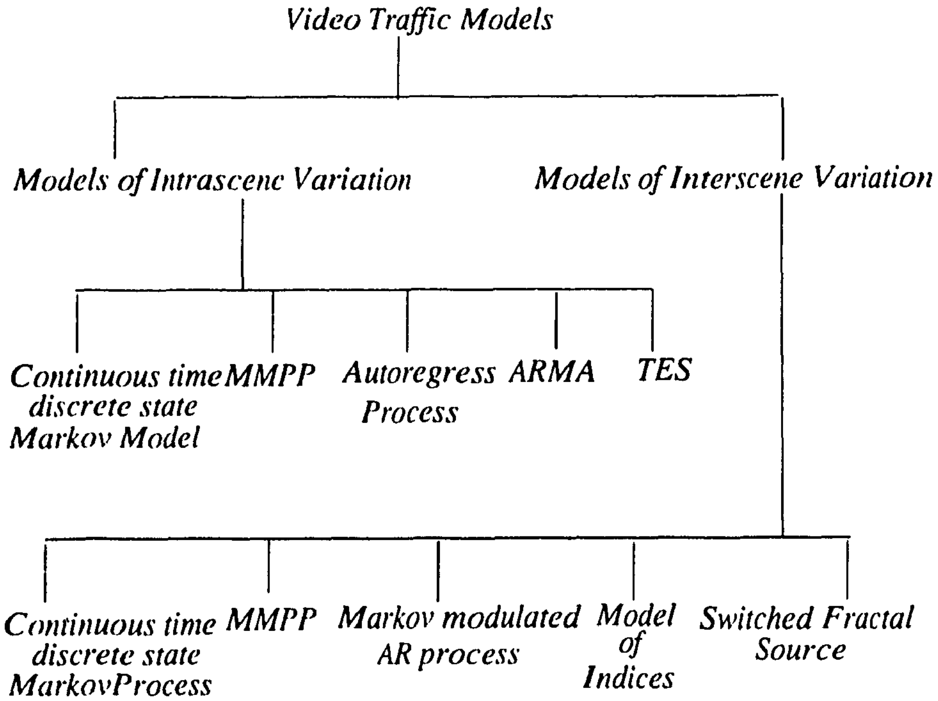

Long term variability model
Scene
Scene
Scene
Short term variability model
Frame
Frame
Frame
Intraframe variation model

Figure 3.11: Modeling of VBR video



Video Traffic Models
Models of Intrascene Variation
Models of Interscene Variation
Continuous time discrete state Markov Model
MMPP
Autoregress Process
ARMA
TES
Continuous time discrete state MarkovProcess
MMPP
Markov modulated AR process
Model of Indices
Switched Fractal Source

Figure 3.12: Models for video traffic

proposed to characterize interscene and intrascene variations.

3.5.1 Models of Intrascene variations (i.e. without scene changes)

These models are applicable to video scenes with relatively uniform activity levels, with few scene changes like video conference scenes showing a person talking. Under these circumstances the variations in bit rate is small and the bit rate proce"s possesses short term correlations only. In fact the study of such bit rate processes have shown that they possess bell shaped nearly normal distributions [35], [36], [7], [37], [38],[39]. The autocorrelation function of the bit rate process closely resembles a negative exponential (for a frame buffered codec). Based on these a few models have been suggested to characterize intrascene variations.

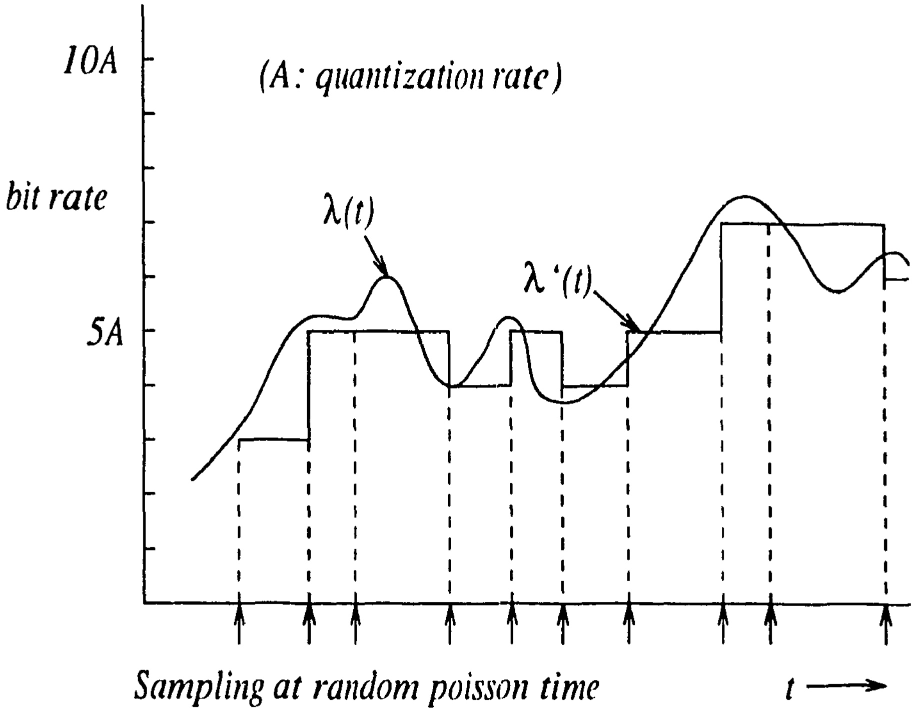


Figure 3.13: Poisson sampling and quantization of the source rate

Continuous time discrete state Markov model

The bit rate process $\lambda(t)$ from a video source is modeled as a continuous time, discrete state, Markov model in [7],[10]. The spectrum of possible values of bit rates from the video source is quantized into M discrete levels (where state M corresponds to peak bit rate level) of step-size A and these $M+1$ levels (including 0) correspond to the state space of the Markov process. Now, the continuous process $\lambda(t)$,[1] describing the bit rate of the video source at time t is sampled at random points in the time domain, and is quantized into the nearest level $\lambda'(t)$ (Figure 3.13). Hence the process can be seen as switching between different states (as determined by the value of $\lambda'(t)$), spending exponentially distributed time periods in each state (due to the Poisson sampling). Since the process being modeled is of uniform activity, only state transitions to nearest neighbour states are allowed. The approximation of

[1] since the bit rate is of the order of several Mbps and the packet length is small, this model assumes the data as a continuous bit stream, ignoring the effects of packetization

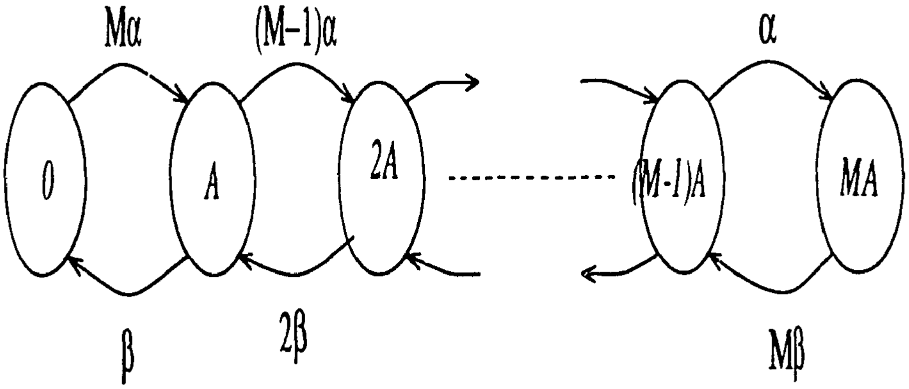


Figure 3.14: State transition rate diagram of Continuous time, discrete space, Markov process

$\lambda(t)$ by $\lambda'(t)$ can be improved by decreasing the quantization step size A (and thus increasing M) and increasing the sample rate.

This model can be used to model both a single video source or a multiplex of N video sources. In the latter case the state space is formed by quantizing the aggregate rate of the multiplex $\lambda_N(t)$ into M discrete levels. As before state changes between nearest neighbours are only allowed. Hence it results in a birth death process whose state transition rate diagram is shown in Figure 3.14. The exponential transition rates between states iA and jA are given by

$$\begin{aligned} \gamma_{i,i+1} &= (M-i)\alpha \qquad i < M & (3.36)\\ \gamma_{i,i-1} &= i\beta \qquad i > 0 & (3.37)\\ \gamma_{i,i} &= 0 & (3.38)\\ \gamma_{i,j} &= 0 \qquad |i-j| > 1 & (3.39) \end{aligned}$$

The birth death process of Figure 3.14 can be considered to represent a population of 'mini-sources', where each mini-source is as given in Figure 3.15, i.e., each mini-source is in one of the states ON or OFF. When ON it generates information at the rate of A bits/sec. Then the probability the system is in state kA is same as the probability that there are k mini-sources out of M mini-sources in their ON

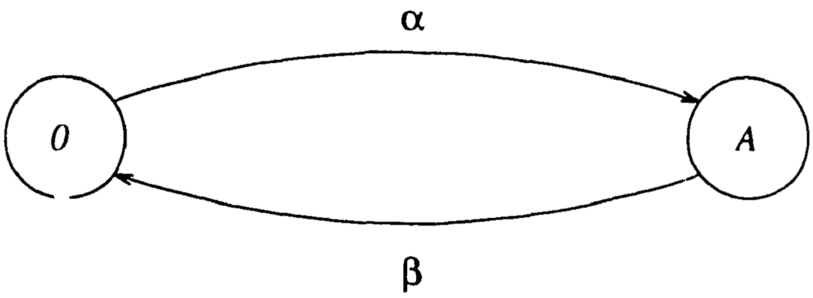


Figure 3.15: Mini-source model

state. It can be shown that $\lambda'_N(t)$ has a binomial distribution

$$P\{\lambda'_N(t) = kA\} = \begin{pmatrix} M \\ k \end{pmatrix} p^k (1-p)^{M-k} \qquad (3.40)$$

where

$$p = \frac{\alpha}{\alpha + \beta}$$

$$E(\lambda'_N) \;=\; M\,A\,p \qquad (3.41)$$

$$C'_N(0) \;=\; M\,A^2\,p(1-p) \qquad (3.42)$$

$$C'_N(\tau) \;=\; C'_N(0)\exp(-(\alpha+\beta)\tau) \qquad (3.43)$$

Here, the parameters of the continuous time, discrete state Markov model namely α, β and A can be determined by matching the mean $E(\lambda'_N)$, variance $C'_N(0)$ and exponential autocovariance $C'_N(\tau)$ as given by Equations 3.41 to 3.43 with the corresponding measured values. The number of mini-sources required for a good approximation of a multiplex of N video sources was found experimentally to be $20N$ [7], [40].

As already mentioned Markov models lead to tractable analytical treatment. In [7], [40] a fluid flow analysis has been carried through to arrive at the survivor function of buffer occupancy.

MMPP model

Markov modulated Poisson process is a doubly stochastic Poisson process, whose Poisson rate depends on the state of the underlying Markov chain. As seen in the section on Voice traffic a MMPP (2-state one in this case) may be used to approximate the superposition of many on-off sources. The same approach may be used to model video traffic, since according to the Continuous time, discrete state Markov model (Maglaris model) discussed above, video traffic may be modelled as emanating from many mini ON - OFF, constant rate sources. Thus video traffic may also be characterized by a MMPP model. MMPP is a correlated non-renewal stream and thus can capture the correlations over certain durations.The parameters of the MMPP may be obtained by matching some of the statistical characteristics of the MMPP with that of the arrival process. Several matching techniques [41] [42] [43] [1] have been proposed to obtain the parameters of the resultant MMPP, when the constituent ON-OFF processes are bursty.

Autoregressive process model

An autoregressive process model of order M (denoted $AR(M)$) is one which predicts the future values of a time series by regressing on the past M sets of values. Such process models have exponentially decaying autocorrelation and a Gaussian distribution. Based on this, AR process was suggested as a model for VBR video in [7], [40], [38], [37].

An autoregressive process model for VBR video is defined as

$$\lambda(n) = \sum_{m=1}^{M} a_m \lambda(n-m) + be(n) \tag{3.44}$$

where $\lambda(n)$ represents the source bit rate during the nth frame,M is the order of the model, $e(n)$ is a Gaussian random process (with mean η and variance 1). $a_m(m = 1, 2, \ldots M)$ and b are constants. For $M = 1$ we have the first order AR process given by

$$\lambda(n) = a\lambda(n-1) + be(n)$$

(Since the value of the sequence depends only on its previous instant it is called a continuous state Auto Regressive Markov model). The parameters of this model are a,b and the mean value η of $e(n)$.

The mean and autocovariance of the AR process are given by

$$E(\lambda) = \frac{b\eta}{1-a} \tag{3.45}$$

$$C(n) = \frac{b^2}{1-a^2}a^n \qquad n \geq 0 \tag{3.46}$$

Hence the parameters a,b and η are obtained by matching the Equations 3.45 and 3.46 with empirical data.

Due to its simplicity and accuracy the AR(1) process is an excellent candidate for modeling VBR video sources. But it does not lend itself to a queueing analysis easily. Hence, this model has its utility limited to simulations.

Another important utility of the AR process is the fact that it can be used to statis'ically characterize a multiplex of video sources [35]. If $\Lambda(n)$ denotes the signal which results from multiplexing N, AR processes, $\lambda_1(n), \lambda_2(n), \lambda_3(n), \ldots \lambda_N(n)$, we have

$$\Lambda(n) = \sum_{i=1}^{N} \lambda_i(n)$$

If $\lambda_i(n)$ are mutually independent then the mean and variance of the resulting multiplex are given by

$$E[\Lambda(n)] = \sum_{i=1}^{N} E[\lambda_i(n)]$$

$$E[\Lambda(n)\Lambda(n+S)] = \sum_{i=1}^{n} E[X_i(n)X_i(n+S)]$$

Hence, if $\lambda_i(n)$ are identical AR processes, the resulting multiplex $\Lambda(n)$ is also an AR process with parameters a and b same as the original AR processes and whose mean and variance are N times those of $\lambda_i(n)$.

Though first order AR processes AR(1) were found to be reasonably accurate in modeling VBR video sources, a better matching may be achieved if the order of regression is increased. In this case the autocovariance of the resultant process is a sum of several exponentials. [37] proposes an alternative solution to achieve the same effect. Here the bit rate per frame $\lambda(n)$ is modeled as a sum of N, AR(1) processes $\beta_i(n)$. i.e.,

$$\lambda(n) = \sum_{i=1}^{N} \beta_i(n)$$

where

$$\beta_i(n) = a_i x_i(n-1) + b_i e_i(n)$$

$e_i(n)$ are Gaussian random processes with mean μ_i and unit variance. It is shown that a choice of N=2 provides a fair accuracy/complexity tradeoff. The method of determining the parameters of the process is discussed in [37]

Autoregressive moving average models

In [44] an Autoregressive moving average (ARMA) model has been proposed for characterizing the output of a non-frame buffered video codec. The ARMA models have autocovariances that exhibit recorrelation. Since the output bit rate from a non-frame buffered video codec also exhibits recorrelation (temporal and spatial). ARMA models serve as a better choice to model the output bit rate process from a non-frame buffered video codec. The ARMA model was used to represent the cell arrival in intervals of typically $100\mu s$. The number of cells in the *ith* interval is modeled by a discrete state, autoregressive moving average process, X_i given by

$$X_i = g(\alpha Z_{i-m} + Y_i + v_i) \qquad \text{with } \alpha < 1 \qquad (3.47)$$

where Y_i and Z_i are a sequence of correlated Gaussian random variables with zero mean (since a white noise sequence σ_i, with zero mean is applied at the filter's input). The moving average part, i,e., the sequence Y_i models frame correlations and the autoregressive part, Z_i, models scene and frame correlations. The sequence of uncorrelated Gaussian random variables v_i, with zero mean, models the white noise stochastic component. $g(.)$ is a Zero memory Non linear (ZMNL) operator which converts the output of the ARMA filter into strictly positive random variables. The method of parameter estimation of the ARMA process is described in detail in [44].

TES models

ΓES (Transform expand sample) [45, 46, 47] is a non-parametric method which can accurately capture the histogram and approximate autocorrelation function of any data set. TES methodology assumes that some stationary empirical time series (such as traffic measurements over time) is available and then it tries to construct a model such that the marginal distribution (or histogram), leading autocorrelation and sample path realizations (histories) matches with the empirical values quite well.

TES processes come in 2 flavours: TES^+ and TES^- process (i.e with positive and negative lag - 1 autocorrelations respectively). TES^+ gives rise to the sequence

$\{U_n^+\}$ given by

$$U_n^+ = \begin{cases} U_0 & \text{if } n = 0 \\ < U_{n-1}^+ > & \text{if } n > 0 \end{cases} \tag{3.48}$$

while TES^- gives rise to the sequence $\{U_n^-\}$,

$$U_n^- = \begin{cases} U_n^+ & n \text{ even} \\ 1 - U_n^+ & n \text{ odd} \end{cases} \tag{3.49}$$

Here, U_0 is distributed uniformly on [0,1); $\{U_n\}$ is a sequence of IID random variables, independent of U_0, called the *innovation sequence* and angular brackets denote modulo - 1 (fractional part) operator $< x > = x - \max \{ \text{ integer } n : n \leq x\}$. The sequences $\{U_n^+\}$ and $\{U_n^-\}$ of the form Eqn. 3.48 and Eqn. 3.49 are called *background sequences* and give rise to a sequence of stationary random variables with uniform marginals on [0,1) and different autocorrelation structures. For practical purposes, transformed TES processes $\{X_n^+\}$ and $\{X_n^-\}$, obtained from Eqn. 3.48 and Eqn. 3.49 by some transformation D (called a *distortion*) are of importance. i.e.,

$$X_n^+ = D(U_n^+); \qquad X_n^- = D(U_n^-) \tag{3.50}$$

The sequences $\{X_n^+\}$ and $\{X_n^-\}$ are called *foreground sequences* . The idea is to create suitable foreground sequences with marginal distributions matching the given (empirical) distribution, by using the inversion method [48]. For a given distribution function F, the inversion method uses distortion $D = F^{-1}$ to generate stationary sequences $\{X_n^+\}$ and $\{X_n-\}$ with marginal distribution F. In the empirical TES methodology, the distortion is effected in two stages. First, in order to "smooth" TES sample paths, a family of transformations called stitching transformations S_ξ, $0 < \xi < 1$ is employed.

$$S_\xi(y) = \begin{cases} \frac{y}{\xi}, & \text{if } 0 \leq y < \xi \\ \frac{1-y}{1-\xi}, & \text{if } \xi \leq y < 1 \end{cases} \tag{3.51}$$

Processes of the form $\{S_\xi(U_n^+)\}$ and $\{S_\xi(U_n^-)\}$ are called *stitched TES processes.* For $0 < \xi < 1$ the effect of S_ξ is to render the sample paths of background TES sequences more "continuous-looking". In th second stage the inversion method is applied to the stitched processes to generate the foreground sequences with matched distributions as the given distribution F. Thus the distortion is given by

$$D = F^{-1}(S_\xi(U_n^-)) \quad \text{or} \quad F^{-1}(S_\xi(U_n^+))$$

However TES methodology models empirical densities as histograms, as is explained in [46].

TES methodology also fits the autocorrelation of the empirical data with that of the model. This is carried out by a heuristic search for a pair (ξ, f_v), (where ξ is a stitching parameter and f_v is an innovation density) such that the autocorrelation function approximates its empirical counterpart. The search can efficiently be carried out using the visual, interactive software environment called *TEStool* [49].

GOB (group of block) level source model, for compressed H.261 standard VBR video over a local area network was constructed in [46, 45]. The GOB is a suitable unit of packet transport. (Each DCT coded frame is divided into 12 group-of-block coded subscreens). At the GOB level the bit rate process is characterized by an autocorrelation that is periodic both at the spatial GOB scan rate and at temporal frame rate. In order to fit a TES model to this data, the raw data is first transformed into a new sequence called the *residual sequence* $\{R_n\}$ which has a faster decaying autocorrelation function. This transformed sequence could effectively and easily be fitted with a TES model as explained in [46, 45].

TES models can be used to generate synthetic streams of realistic traffic to drive simulations of communication networks. However they suffer from the handicap of not leading to tractable mathematical analysis.

3.5.2 Modeling scene changes

These models are useful in describing video sources with high motion and scene changes as in broadcast applications. Models that are proposed for video sources with scene changes must capture both short term and long term correlations. In this section we examine a few models that have been proposed to characterize video sources with scene changes.

Continuous time, discrete state, Markov process model

This model proposed in [50] is an extension of the model by [7] (discussed in section 3.2). Here, as in [7] the source changes between various fixed rate levels, with exponentially distributed times in each level. However, here, the possible data rate levels are built from a linear combination of two basic rates, a higher rate A_h and a lower rate A_l. This model can represent the bit rate from a single video source or an aggregate of N video sources. The state transition rate diagram for an aggregate of N video sources using this model is shown in Figure 3.16. (The labels in each state indicate the data rate in that state). The basic rate A_l corresponds to parameter A in the model [7]. Transitions based on A_l model the short term correlations, while transitions based on A_h model the long term correlations. Hence with no transitions based on A_h, this model reduces to the model of intrascene variations as in [7]. For an aggregate of N video sources, there are $NM + 1$ low rate levels and $N + 1$ high rate levels, where M is chosen arbitrarily. The parameters of the model are determined by matching the theoretical values with measured values. For $N = 1$, the parameters c and d are determined by matching the fraction of time spent in high activity level $q(= c/(c+d))$ and the average time spent in high activity level $\frac{1}{d}$ with the actual measured data. For determining a, b, A_l and A_h, the autocovariance,

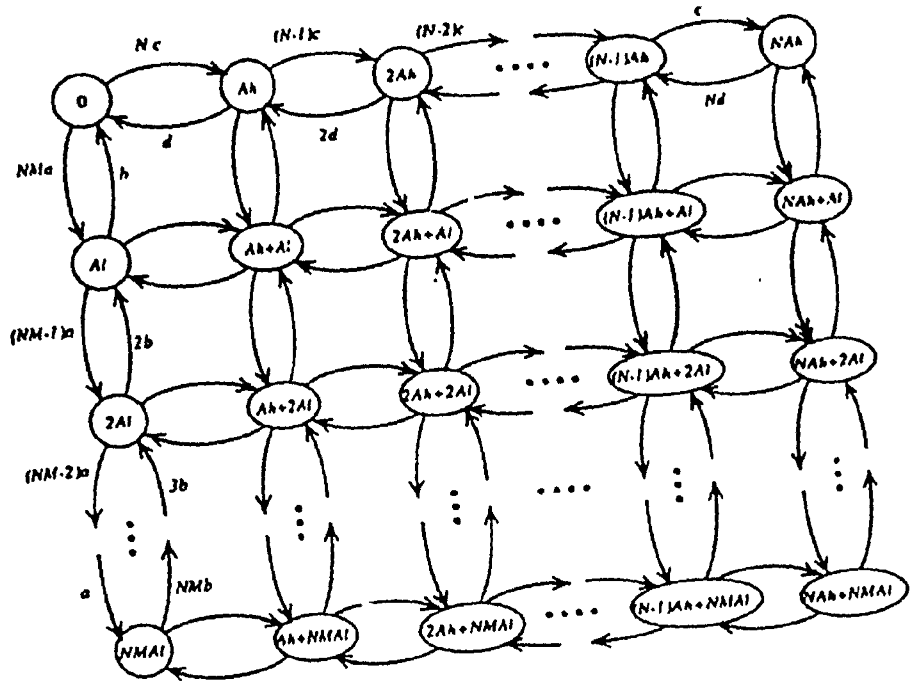


Figure 3.16: Continuous time, discrete state, Markov process model

variance, mean ratio (γ)(ratio of average bit rate in high level to that of low level) and the overall mean bit rate $\overline{\lambda}$ as given by Equations 3.52 to 3.55 are matched with the actual measured values.

$$C(\tau) = C(0)\exp(-(a+b)\tau) \tag{3.52}$$

$$C(0) = Np(1-p)A_l^2 \qquad \text{where } p = \tfrac{a}{a+b} \tag{3.53}$$

$$\gamma = \frac{NpA_l + A_h}{NpA_l} \tag{3.54}$$

$$\overline{\lambda} = NpA_l + qA_h \qquad \text{where } q = \tfrac{c}{c+d} \tag{3.55}$$

For the sake of analysis this model can be viewed as a superposition of simpler ON-OFF mini-processes, NM of the type shown in Figure 3.17a, and N of the type shown in Figure 3.17b, then the state of the aggregate process model is the couplet (i,j) where i, j denote the number of each type of mini-processes which are in the ON state. A fluid flow analysis (as in [7]) has been carried out in [50] to determine the survivor function of buffer occupancy.

Markov modulated AR process

As seen before, an AR process captures short term correlations quite accurately . In [51], [52], [53] an AR process with time varying parameters is proposed as a model to characterize the bit rate process from a motion adaptive video codec (one that adapts the encoding scheme to the motion in the scene picturized). The time dependence of the parameters of the AR process captures long term correlations. According to this model the no. of bits in a frame is given by a first order Gaussian AR process whose parameters are determined by the state of a Markov chain(Figure 3.18). Thus each

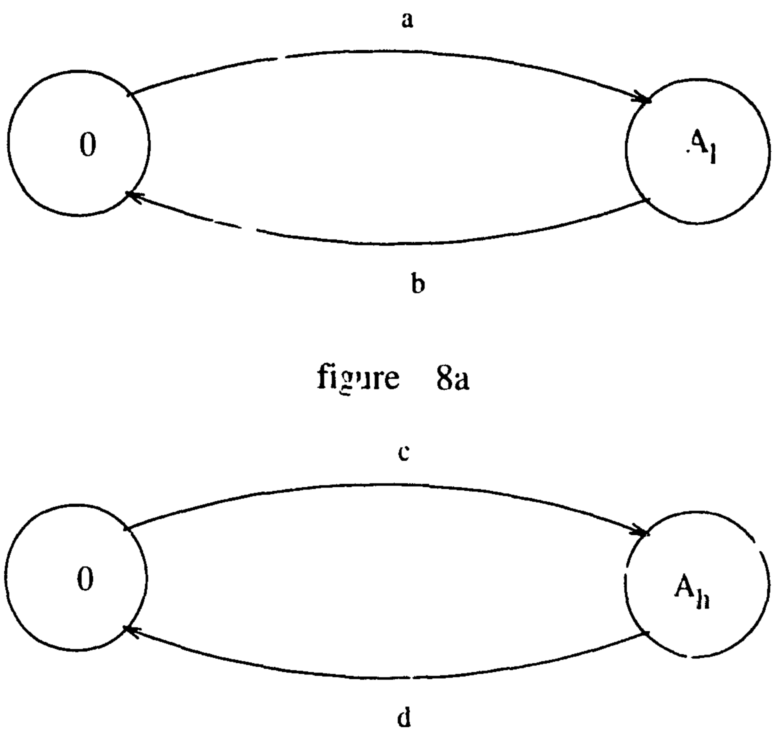


figure 8a

figure 8b

Figure 3.17: Mini-process models

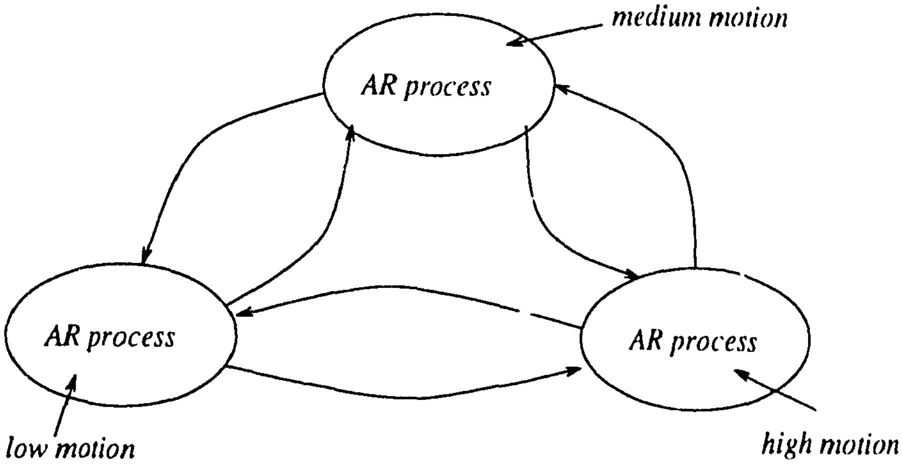


Figure 3.18: Markov modulated AR process

state of the Markov chain, with its own set of parameters, represents the various classes of motion. A Gaussian density was used because it was found from the study [51, 52, 53] that the bit rate distribution of the VBR coded full motion video can be represented by a composite Gaussian PDF.

In this model, the range of bit rates are separated into N adjacent intervals demarked by thresholds γ_i, $i = 1, 2, \ldots N-1$ and $0 \leq \gamma_1 \leq \gamma_2 \leq \gamma_{N-1}$. These bit rate intervals form the state space of the Markov chain. i.e., state 1 corresponds to the range $0 \leq \lambda_n \leq \gamma_1$ and state i corresponds to range $\gamma_{i-1} \leq \lambda_n \leq \gamma_i$, where λ_n represents the number of bits in frame n. If S_n denotes the state of the process at frame n, then the model can be represented mathematically as

$$\lambda_n = \begin{cases} a(i)\lambda_{n-1} + G(\mu(i).\sigma^2(i)) & \text{if } S_n = S_{n-1} = i \\ G(\eta(i), v(i)) & \text{if } S_n \neq S_{n-1};\ S_n = i \end{cases} \tag{3.56}$$

where G(.) denotes a Gaussian random variable with specified mean and variance. $\eta(i)$ and $v(i)$ denote the mean and variance of λ_n conditioned on state i. i.e., $\eta(i) = E[\lambda_n | S_n = i]$; $v(i) = var(\lambda_n | S_n = i)$. $a(i)$ is the correlation coefficient between the bit rates of two successive frames when the Markov chain is in state i.

Increasing the number of states N, results in an accurate model at the expense of increasing its complexity. The number of parameters of the model depends on the number of states, as a set of parameters characterize the AR process in each state. The parameters of the AR process in various states are obtained by matching the following statistics with the measured values, for each state:

$$\text{(a) mean bit rate in state } i, \eta(i) = \frac{\mu(i)}{1 - a(i)} \tag{3.57}$$

$$\text{(b) variance of bit rate in state } i, v(i) = \frac{\sigma^2(i)}{1 - a^2(i)} \tag{3.58}$$

$$\text{(c) measure of correlation of bit rates, } D^2(i) = \frac{2\,\sigma^2(i)}{1 + a(i)} \tag{3.59}$$

where $D^2(i)$ is the measure of correlation of bit rates between two successive frames. i.e.,

$$D^2(i) = E[(\lambda_n - \lambda n - 1)^2 | S_n = S_{n-1} = i]$$

Hence the parameters of the AR process in each state, namely, $\mu(i)$, $\sigma(i)$, and $a(i)$ are obtained from Equations 3.57 to 3.59.

The duration of a state is geometrically distributed with mean $\frac{1}{\theta_i}$,as given by the following p.d.f.,

$$F_i(k) = \frac{\theta_i}{1-\theta_i}(1-\theta_i)^k \qquad (k = 1,2,\ldots)$$

where k is in terms of the number of frames. The quantity $\frac{1}{\theta_i}$ and $\pi_{i,j}$ (probability the next state is j given that the present state is i) can be obtained from measurements directly. Then the transition probability matrix P can be obtained as

$$P = \begin{bmatrix} 1-\theta_1 & \theta_1\pi_{12} & \theta_1\pi_{13} & \ldots & \ldots \\ \theta_2\pi_{21} & 1-\theta_2 & \theta_2\pi_{23} & \ldots & \ldots \\ \theta_3\pi_{31} & \theta_3\pi_{32} & 1-\theta_3 & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix} \qquad (3.60)$$

If the vector $\mathbf{p} = [p_1, p_2, \ldots p_N]$ denotes the steady state probability (obtained by solving $\mathbf{p} = \mathbf{p}$ P and $\sum p_i = 1$) then the number of bits generated according to this model has the following PDF

$$f(x) = \sum_{i=1}^{N} p_i \, G(\eta(i), v(i)) \qquad (3.61)$$

This model can also be used to characterize an ..ggregate of N sources. As before, though this model is a good candidate for simulation, it does not provide a suitable framework for a queueing analysis.

Model of Indices

A novel method of video traffic characterization, that does not depend on the variable bit rate coding algorithm employed is discussed in [54]. Here, a set of simple parameters called the indices that sufficiently characterize the video sequence are identified by working on the uncoded video sequence. The bit rate process from any coder is then predicted from a linear combination of the corresponding indices.

The parameters developed are grouped into 3 classes. One is derived from the histogram of the pixel information. The second is derived from the spatial correlation of the pixel values in a frame and the third set of indices are derived from the temporal correlations of the pixel values along the time axis.

The first class of parameters are derived from histogram of the pixel values of a single frame. Three indices are considered under this category, namely the average index (which gives a measure of the brightness in a frame), variance index(which gives a measure of the variability of the pixel values in a frame) and the entropy index (which represents the best possible compression performance for the codes that use first order statistics of the pixel values).

For the second class of parameters based on spatial correlation, the indices chosen were vertical entropy (which is entropy of difference in intensity between adjacent rows in a pixel array of a frame) and the horizontal entropy index (entropy of the difference in intensity between adjacent columns in a pixel array of a frame)

For the third class of parameters, based on temporal correlation, the indices chosen were difference index (reveals the difference in the amplitude of pixels between consecutive frames), motion index(the magnitude of the displacement vector corresponding to the pixels within the frame) temporal entropy index (entropy of the temporal difference in the vertical consecutive frames).

Study of the several coding schemes in [54] revealed that the output bit rate process was strongly correlated with some of the indices. Hence the output bit rate

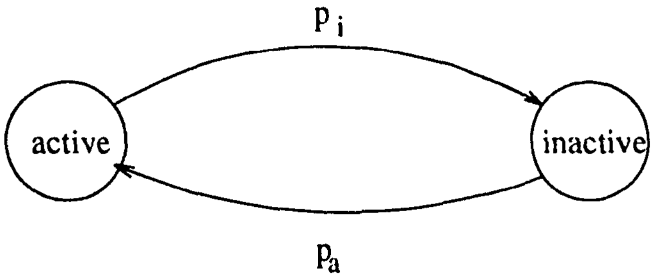


Figure 3.19: active/inactive process model

was predicted using a linear predictor model, a model fitting algorithm was then used to reduce the number of parameters according to linear regression measures of fit.

Though the model averts the necessity of modeling the bit rate process from different encoders separately, it cannot be used for an analytical evaluation.

Switched fractal source

In [55] and [56] a switched fractal source has been proposed as a model to characterize video source of less than 5 Mbps, using a highly compressed encoding scheme. Here the cell generation process is modeled directly in order to reproduce the bursty characteristics of the VBR traffic. In the encoding characterized, the original image is divideu into smaller sub-blocks, each block is transferred into another domain and the blocks of transform coefficients are scanned, coded and packaged into ATM cells. Due to the highly compressed nature of the coding scheme, the number of ATM cells produced after processing each block is small, either zero or one. Hence the cell generation process at the sub-block level can be modeled by a simple active/inactive process as shown in Figure 3.19. It was found that the transition probabilities p_a (inactive to active) and p_i (active to inactive) were dependent on the time spent in their present state. In particular the relationship is of the form $p(t) \propto t^D$ where D

is known as the fractal dimension and the model is called a fractal model. Therefore

$$p_a(t) = A_a\, t^{D_a} \tag{3.62}$$

$$p_i(t) = A_i\, t^{D_i} \tag{3.63}$$

where A_a, A_i are proportionality constants and D_a, D_i are fractal dimensions. The parameters are obtained from a logarithmic plot of experimentally measured active and inactive time periods. Such a fractal model accurately represents the cell traffic characteristics of uniformly active images.

In order to represent the traffic statistics of varying activity levels, a model that switches between multiple fractal sources is proposed. [56] discusses a five-mode fractal source model. The five cell generation modes correspond to average bit rates of 1, 2, 3, 4 and 5 Mbps. These five fractal sources were obtained by monitoring the traffic produced by five artificially constructed images, in which each of the image sub-blocks (when coded) produced average bit rates of 1-5 Mbit/sec, respectively. Logarithmic plots of the experimentally measured, active and inactive time periods were plotted. Parameters A and D of Equations 3.62 and 3.63 are given by the y-axis intercept and straight line gradient of these plots, respectively.

To simplify the switching process, each row of sub-blocks in an image (thirty two 16 x 16 sub-blocks per row for 512 x 512 images) can be divided into image 'sections', each containing N ($N = 8$) sub-blocks. Switching between fractal sources is permitted only at the beginning of one of these sections. The switching scheme used is given by

$$L_n = \begin{cases} |L_0| & n = 0 \quad 1 \leq L_N \leq 5 \\ L_{n-1} + |\delta_n|, & n > 0 \end{cases} \tag{3.64}$$

where L_n is the activity level for image section n, L_0 is the 'starting' or 'average' activity level for the image (directly proportional to the average complexity of the image), δ is a normally distributed random variable with zero mean and standard deviation σ. The results of mean delay obtained by a queueing simulation using this model indicates that this model behaves similar to the 'real' traffic for network utilization levels up to 90%.

3.6 Selection of a video traffic model

The various models of the video traffic were presented in the previous section. However, from the standpoint of developing our traffic generator we are only interested in a model that can capture enough correlations, so that when packet arrivals from such a model fed into a queue produce the same queueing behaviour as the original arrival process. From this point of view the MMPP is perhaps the simplest model that can model the correlations in the arrival process accurately. Also with the MMPP we model the aggregate video traffic arrival process as against arrivals from the individual sources as in the case of Autoregressive or ARMA processes. Also the MMPP successfully captures correlations over short durations as seen earlier.

A recent study [57] of VBR video have revealed that they exhibit the phenomenon of statistical self similarity. In [57], the results of detailed statistical analysis of a 2-hour long empirical sample of VBR video are discussed. The samples were obtained by applying a simple intraframe compression code to an action movie. The study showed that the autocorrelation function of the VBR video sequence decays hyperbolically, (a manifestation of long range dependence). However another recent study [58] of the same traffic traces has confirmed that from the point of view of queueing results, long-range dependence does not affect buffer occupancy when the busy periods are not large. In [58] the video trace is modeled by a Markov chain (a short range dependent model) and various operating characteristics are obtained. It was found that these characteristics closely matched those obtained from the actual trace. Hence, long-range dependence is not a crucial property in determining the buffer behaviour of VBR sources. Thus a short-range dependent model such as the MMPP would suffice to charcterize the correlations in the video traffic arrival process. We choose the 2-state MMPP to characterize aggregate video traffic in our traffic generator.

Chapter 4

Data Traffic Models

Data traffic is highly bursty. Unlike real time traffic (voice or video), data traffic is delay or jitter tolerant, while being sensitive to losses. The statistical characteristics of data traffic are complex and application dependent. In this chapter we discuss the characteristics of data traffic and present the various modeling methodologies that are used to model data traffic. We also present the new long-range dependent traffic model called the PMPP and discuss its suitability in characterizing data traffic. The statistical characteristics of this model are studied by means of simulation and analysis.

4.1 Nature of data traffic

Modeling of data traffic is of fundamental importance in the performance evaluation and traffic engineering of packet (or BISDN) networks. Data traffic is highly bursty. The characteristics of data traffic are complex and application dependent like FTP, Telnet, TCP, etc.

Recent studies of packet data traffic [8, 9, 10] in local area networks have thrown more light on the characteristics of data traffic. The studies revealed that data traffic exhibits long range dependence and *statistical self-similarity*, i.e, the traffic exhibits "burstiness" across a wide range of time scales, ranging from milliseconds to minutes to hours (as shown by Figure 2.2 in Chapter 2.) In [8] high quality high time resolution LAN traffic was collected from the Bellcore's Ethernet and analyzed statistically. It was found that the data traffic exhibited the following features: slowly decaying variances, long term dependency (i.e., hyperbolically decaying autocorrelation function), 1/f noise. Also, the analysis revealed that the generally accepted argument for the "Poisson like" nature of aggregate traffic, namely that aggregate traffic becomes smoother (less bursty) as the number of traffic sources increases had very little to do with reality. In fact, the burstiness (degree of self-similarity) of LAN traffic was found to intensify as the number of active sources increased. As seen in Chapter 2 all these features of data traffic clearly suggest that the data traffic possesses self-similarity and long-range dependence. Following this study the subsequent studies of Common Channel Signaling Network data in [59] and Wide Area network traffic in [60] also showed that the data traffic is self-similar.

These findings change the traditional view of modeling data traffic and has serious implications on issues related to the design, control and performance analysis of high speed networks. Some of the implication of self-similarity in data traffic are:

- The degree of self similarity measured in terms of the Hurst parameter H, provides a satisfactory measure of burstiness (burstier the traffic, higher the value of H). Other commonly used measures of burstiness such as index of dispersion (for counts), peak to mean ratio or coefficient of variation are meaningless, since for fractal traffic these measures can assume any value depending on the length of the interval over which these measurements are made.

- The presence of low frequencies in the spectral density (or equivalently the slowly decaying autocorrelation and variances) causes heavy losses and long delays during long time frame bursts. Hence nature of network congestion produced by fractal traffic differs drastically from that predicted by conventional traffic models.

- For fractal traffic the overall packet loss decreases very slowly with increasing buffer size.

- Source models for individual sources are expected to show extreme variability in terms of the inter arrival times of packets; the inter-arrival times between packets have a "heavy-tailed" distribution.

- Aggregation of bursty traffic streams does not produce smooth "Poisson-like," superposition process as previously assumed. Hence new traffic models that capture long range dependence and fractal properties are required.

Thus modeling the self-similarity is of fundamental importance in modeling data traffic. Traditional models for data traffic such as the MMPP, batch Poisson, etc., are short range dependent models and do not capture these characteristics. This stresses the need for more accurate models to capture these aspects in data traffic.

Another important characteristic of data traffic is that the data traffic is bi-modally distributed; i.e., it has two predominant rates at which the packets arrive. Earlier measurements of data traffic [61] indicate that the message length distribution of data traffic is bimodal. Since a burst of packets are produced for each message, this also suggests that the burst of data packets may be bimodally distributed. As noted in [62], if a source generates a long burst of data like file transfer among short bursts which may correspond to commands, the source traffic essentially consists of short and long bursts. Hence the net data traffic from many such data sources is also likely to be bi-modal. This bi-modal nature of data traffic may also be verified from the recent measurements of data traffic at the Bellcore's Ethernet LAN. Figure 4.1 shows the distribution of packet lengths from the bellcore Ethernet traffic data from October 1989. (The traffic file is available via anonymous

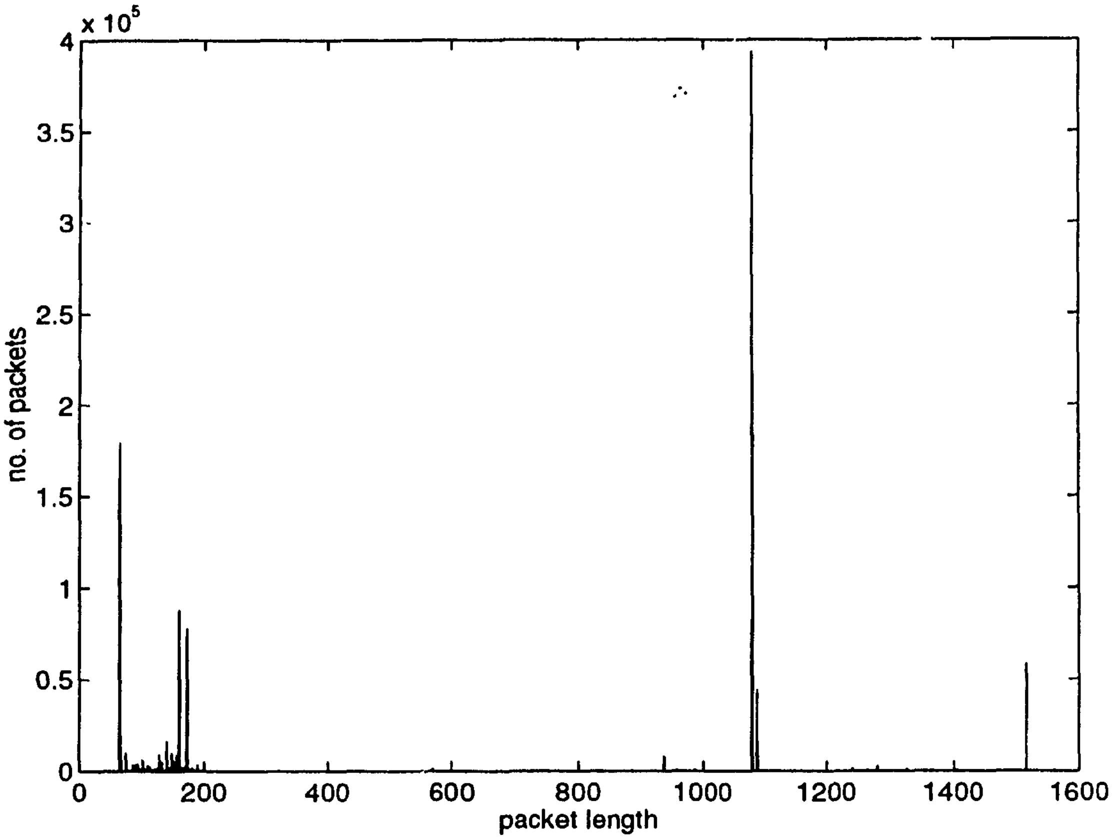


Figure 4.1: Packet length distribution from Bellcore's Ethernet data

FTP from *flash.bellcore.com.*) Ethernet is a physical layer protocol and has variable packet size ranging from 64 bytes to 1518 bytes. Figure 4.1 shows two peaks; one at packet size of 64 bytes and one at 1082 bytes, illustrating the fact that data traffic is bi-modal.

Thus we see that there are two pre-dominant burst rates prevalent in the data traffic and the bursts extend for large periods of time.

4.2 Modeling aggregate data traffic

There have been many models proposed in the literature for characterizing individual data traffic sources or a superposition of multiple sources. The conventional models like fluid flow, batch Poisson, MMPP and HAP incorporate some form of Markovian structure, either in the way the way the arrival processes are modulated or in the arrival process themselves, for reasons of mathematical tractability. Thus these models are good candidates for the analytical performance evaluation of packet data networks. However all these models are short range dependent models and do not capture the long-term dependence and self-similarity in the dat traffic.

Hence, new models that can represent self-similar (or fractal) characteristics have been proposed [3, 2, 4]. Thus the modeling approaches for data traffic may broadly be classified as shown in Figure 4.2.

The fractal models that have been proposed in the literature account for the self-similar phenomena exhibited in data traffic. However all the fractal or self-similar models proposed do not lead to tractable analytic solutions. On the other hand, the conventional traffic models that are blessed with a wealth of analytical tools, fail to capture the long term correlations and fractal properties of packet traffic. The models currently considered in literature (like Markov model, MMPP, ARIMA, etc.), may be used to capture fractal properties. However the process of modeling long range dependence with the help of short-range dependent processes is

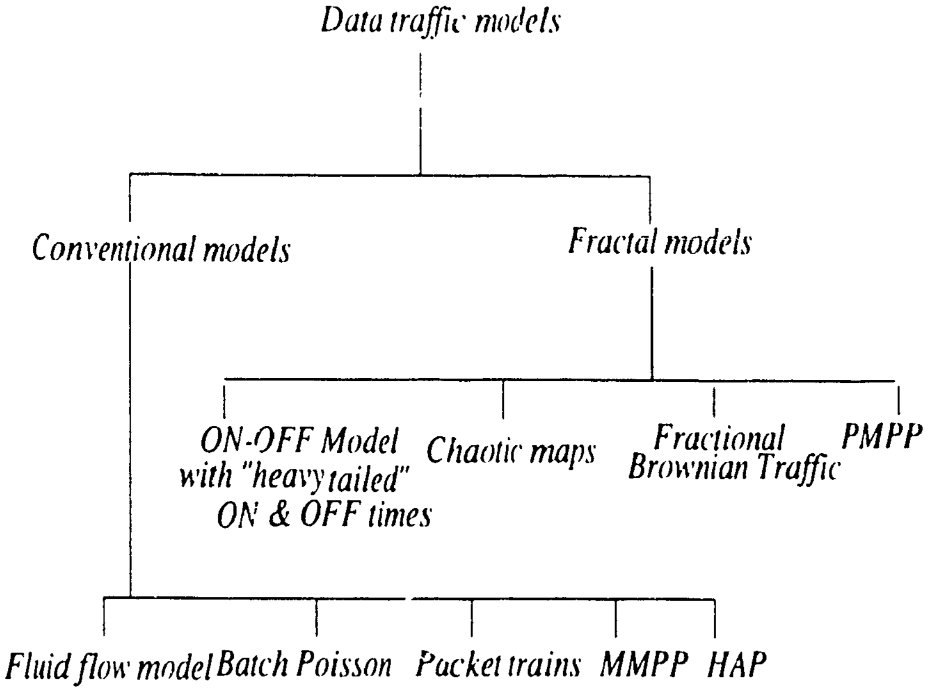


Figure 4.2: Models for data traffic

equivalent to approximating a hyperbolically decaying autocorrelation function by a sum of exponentials and hence requires a large number of parameters. Parsimonious modeling of fractal properties by conventional models can be achieved by resorting to some approximations.

4.2.1 Conventional models

This section gives a brief overview of each of the conventional models. As already mentioned, these models do not capture the long term correlations and self-similar properties of data traffic.

Fluid flow model

The fluid flow model [63] (also referred to as Uniform Arrival and Service model (UAS)) assumes that the information flow in and out of the buffer (at the multiplexer) is uniform and continuous rather than in discrete packets. In this model the source generates information to the transmitter at the rate of one unit of information per unit time and the server removes information from the buffer at a uniform rate not to exceed C units of informatior per unit time. With these assumptions the equilibrium queue distribution is described by a set of differential equations, which together with a set of boundary conditions can be solved to yield the equilibrium queue distribution. The method is outlined in the section on voice traffic models.

Though this modeling methodology leads to a tractable analysis, its largest drawback is that it cannot model the short-term queue increases that occur when two or more packets arrive almost simultaneously.

Batch Poisson model

The batch Poisson model [64] is an extension of the Poisson model. Here the arrivals occur in batches. The batch arrival is Poisson. The batch size b_i can be random. The b_i's are independent and identically distributed and the total number of arrivals in an interval of duration t is

$$A(t) = \sum_{i=1}^{N(t)} b_i$$

where $N(t)$ is the number of original Poisson arrivals. The IDC of a batch Poisson process may be given by [11]

$$I(t) = \frac{Var(b_i)}{E(b_i)} + E(b_i) \tag{4.1}$$

When the distribution of batch arrivals b_i is geometric, i.e.,

$$Pr(b_i = k) = (1 - p)p^{k-1}$$

with $0 < p < 1$, the IDC of the batch Poisson process becomes

$$I(t) = 1 + \frac{p}{1 - p}$$

As seen above the IDC of the batch Poisson process is constant. This is so because, a batch Poisson process is a regenerative process with independent increments. Thus a batch Poisson process, though simple is not suited for capturing the correlations present in the packet arrival process

Some level of correlations can be modeled by the batch Poisson process if the batch size distribution of successive batch arrivals were chosen according to a Markov chain. The batch Poisson model is a special case of the general Batch Markovian Arrival Process (BMAP) for which extensive analytical (transient and steady state) results exist [65]. Hence this model provides an efficient means for analysis.

Packet trains model

In [66] a new model called the packet trains model is proposed to characterize the data traffic in a token passing ring LAN. The model is based on the observation that data traffic exhibits *source locality* (i.e., given a packet going from node A to B, there is a high probability that the next packet will be going from node A to B or from B to A. The traffic on the network (here a token passing ring) is divided into a number of packet streams between various pairs of nodes of the network. Each node-pair stream consists of a number of trains. Each train consists of a number of packets (or cars) going in either direction (from node A to B or node B to A), as shown in Figure 4.3. The intercar time is smaller than a (user) specified maximum

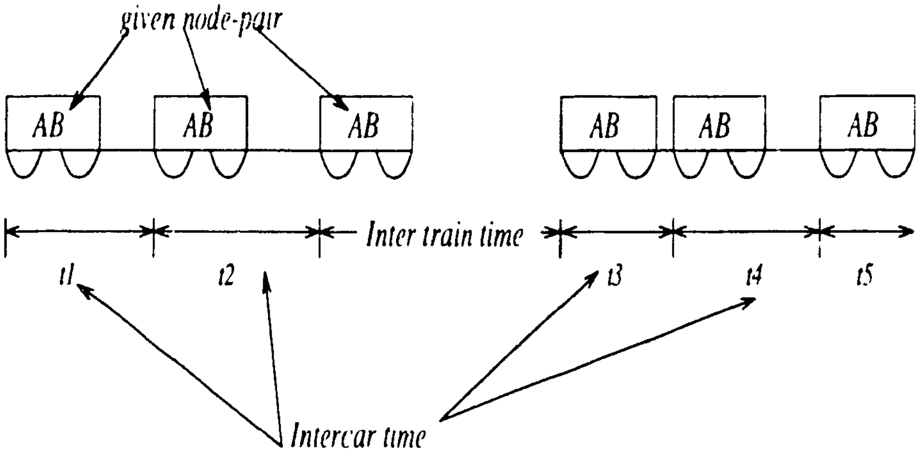


Figure 4.3: Packet train model

called maximum allowed intercar gap (MAIG). The inter-train time is larger than MAIG. Hence the inter-train time is a user parameter, while the inter-car interval is a system parameter. Partitioning of the network into streams based on node-pair processes as explained above helps increase the predictability of data traffic, since they make use of the property of source locality inherent in data traffic. Hence this model is good for simulation purposes.

MMPP models

Markov modulated Poisson Process (MMPP) is a nonrenewal, doubly stochastic Poisson process where the rate process is determined by the state of a continuous time Markov chain. In other words underlying is a continuous state Markov chain, where the sojourn time for state j is exponentially distributed with mean r_j^{-1}. When in state j, cells are generated according to a Poisson process with rate λ_j. [21] uses a two state MMPP and approximates the traffic of multiple data and voice sources. However, as seen in an earlier chapter, the 2-state MMPP for example, captures the correlation only over certain durations. Increasing the number of states of the

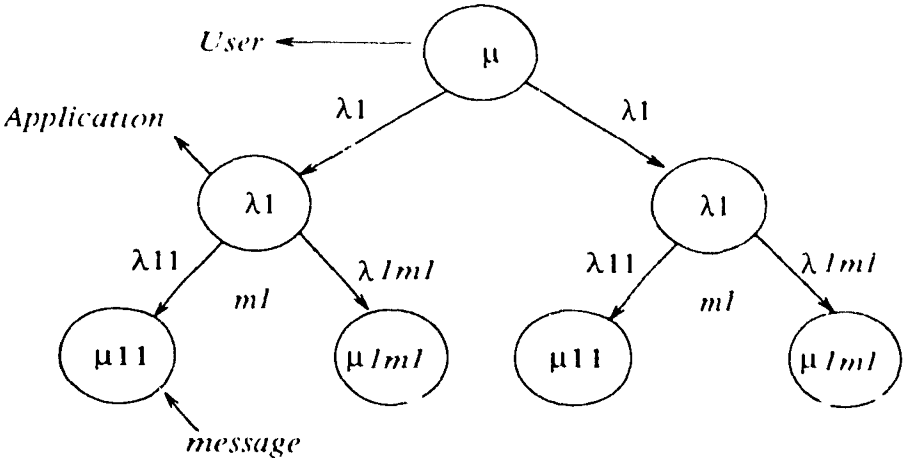


Figure 4.4: HAP model

MMPP will help capture correlations over longer intervals, but this would increase the complexity of the model.

HAP models

The HAP (Hierarchical Arrival Process) model is based on the fact that there are many processes modulating a single packet arrival stream. For example the long term correlation depends on the user and application behaviour, while the short-term correlation depends primarily on the network hardware and software. HAP [67] models both the short-term and long-term correlations by modeling the arrival process at 3 levels - ***user, application and message*** (Figure 4.4). A set of parameters describe the arrival and departure processes at each level. As shown, users arrive in the system according to an interarrival time distribution (with mean λ) and stay in the system according to a service distribution (with mean μ). The user may invoke applications according to an interarrival time distribution (with mean λ_i) which may remain active according to a specified distribution (with mean μ_i). During the active interval, the application generates several types of messages with different rates and with different message size distributions. The HAP can be mapped into a MMPP [67] and analysis can be carried out with the resultant MMPP.

The HAP model captures the correlation at different levels. It also lends itself to analysis easily. However, the HAP, models the arrival process only at a message level and not at a packet level.

4.2.2 Fractal models

This section briefly discusses the models that can capture the fractal (or self-similar) properties of packet traffic.

ON - OFF model with "heavy tailed" ON and OFF times.

Mandelbrot originally suggested [68] that the superposition of many sources which exhibit the "Noah effect" (or infinite variance syndrome) results in a self-similar stream. In [69] [11] Leland *et. al.* employ this method to provide an explanation for the observed self-similarity of the traffic in terms of the nature of the traffic generated by an individual source. They suggest that each of the individual sources contributing to the self-similar traffic stream can be represented by the familiar on-off abstraction. However, these on-off sources exhibit the "Noah effect" in that they have a highly variable on and off periods (sojourn times). i.e., the sojourn times of the on-off sources are characterized by "heavy-tailed" distributions. Similar conclusions were also made in [10] based on studies on individual ISDN data traffic sources. Hence the sojourn times of individual sources can aptly be characterized by a heavy tailed distribution like the stable Pareto distribution. This distribution has a survivor function of the form:

$$P\{X \geq x\} = x^{-\alpha} \qquad \alpha > 0, x > 1 \tag{4.2}$$

The density function is given by

$$p_X(x) = \frac{\alpha}{x^{\alpha+1}} \qquad \alpha > 0, x > 1 \tag{4.3}$$

The parameter α denotes the thickness of the tail of the distribution. If $1 < \alpha < 2$, then the Pareto distribution possesses an infinite variance but a finite mean as given by

$$E(X) = \frac{\alpha}{\alpha - 1} \tag{4.4}$$

Thus for $1 < \alpha < 2$, the Pareto distribution exhibits the infinite variance syndrome. The tail of the stable Pareto distribution decays far more slowly (by a power law) than an exponential distribution. A Pareto distributed random variable takes a larger value with a higher probability than an exponentially distributed random variable. Higher the value of α, thicker the tail of the distribution.

Now, superposition of many on-off sources whose sojourn whose ON and OFF sojourn times are described by a Pareto distribution will result in a self-similar traffic stream. It has also been proved in [70] that if $1 < \alpha < 2$ for the sojourn times of the constituent on-off processes then the Hurst parameter H of the resultant self-similar stream is given by

$$H = \frac{(3 - \alpha)}{2} \qquad (1 < \alpha < 2) \tag{4.5}$$

Thus self-similar streams with any Hurst parameter H, $(0.5 < H < 1)$ can be generated by varying the parameter α of the sojourn times of the constituent ON - OFF processes.

Chaotic Maps

[3] uses deterministic chaotic maps to model fractal properties in packet traffic Chaotic maps are low dimensional non linear systems whose time evolution is described by a knowledge of an initial state and a set of dynamical laws. The trajectory of chaotic system are very often fractal in nature. Hence by adjusting the parameters of the chaotic maps it is possible to capture the fractal nature of packet traffic.

Consider a one-dimensional map in which the state variable x_n evolves over time according to the non linear map:

$$x_{n+1} = f_1(x_n) \quad y_n = 0 \quad (0 < x_n \leq d)$$

$$x_{n+1} = f_2(x_n) \quad y_n = 1 \quad (d < x_n < 1)$$

The packet generation process is modeled as follows:

- The source alternates between a passive and active state.
- When $y_n = 0$ $(0 < x_n \leq d)$ the source is in passive state and when $y_n = 1$ $(d < x_n < 1)$ the source is in active state (Figure 4.5).
- Every iteration of the map in the active state is taken to generate a packet (or batch of packets).
- suitable $f_1(.)$ and $f_2(.)$ should be chosen so that properties of y(n) match those of actual packet traffic.

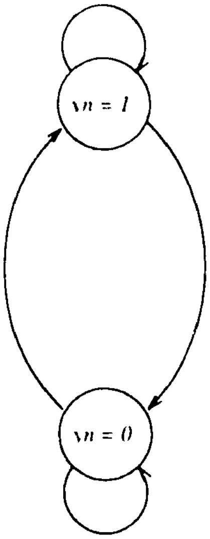


Figure 4.5: Basic source model (Chaotic Map)

The *Intermittency Map* with $f_1(.)$ and $f_2(.)$ as given below captures fractal properties of data traffic well [3]

$$x_{n+1} = \begin{cases} \epsilon + x_n + c x_n^m & 0 < x_n \leq d \\ \frac{x_n - d}{1-d} & d < x_n < 1 \end{cases}$$

where $c = \frac{1-\epsilon-d}{d^m}$ (Figure 4.6)

While chaotic maps is effective in characterizing much of the fractal properties of data traffic like 1/f noise, "thick-tail" behaviour of interarrival time densities, etc., using very few parameters, there are considerable analytical difficulties in their application.

Fractional Brownian Motion model

The fractional brownian motion is a self-similar process. i.e., if $Z(t)$ is a brownian motion process then $Z(\alpha t)$ is identical in distribution to $\alpha^H Z(t)$, where $(1/2 < H < 1)$ is the self-similarity parameter. In [2] a model based on Fractional Brownian

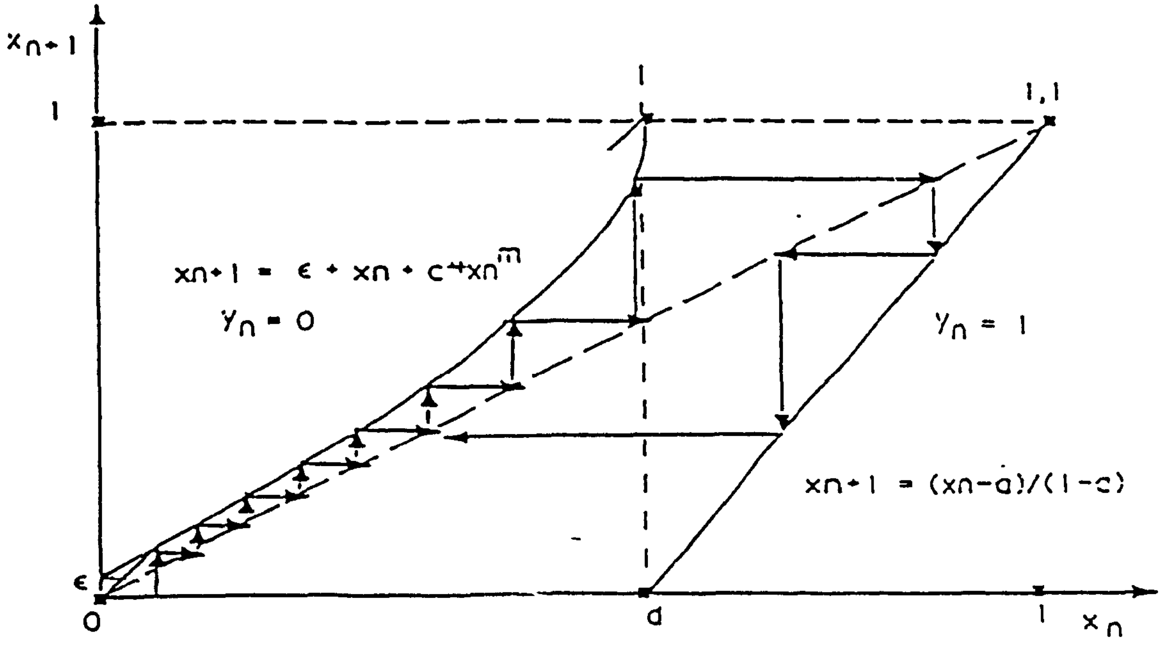


Figure 4.6: Intermittency map

Motion is proposed to characterize the self-similar properties of packet traffic. The following model is studied :

$$A(t) = mt + \sqrt{am}Z(t) \qquad (1.6)$$

where $A(t)$ is the number of cell arrivals to the multiplexer in the time interval $(0,t]$, m is the arrival rate of a Poisson process and $Z(t)$ is a fractional brownian motion with self-similarity parameter H. The above model is based on a diffusion approximation of the number of arrivals from a Poisson process. The parameters of the model are H, m and a. The above model could also be used to characterize the superposition of N independent and identically distributed cumulative traffic processes. Hence now $A(t) = \sum_{i=1}^{N} A_i(t)$. Now, the parameters H and a characterize the type of the traffic mix while m gives its amount. In [2] an analysis is done by using a storage model based on $A(t)$ as the input process.

PMPP model

From the point of view of synthetic traffic generation, the ON-OFF model with heavy tailed sojourn times, Chaotic maps and the FBM models are complex. For instance the chaotic map [3] and the ON-OFF model approach model the characteristics of an individual data traffic source. This approach may give an intuitive explanation and understanding of the source characteristics that contribute to the long-range dependence and self-similarity of the aggregate data traffic: the chaotic map produces individual on-off sources whose sojourn times are characterized by "heavy-tailed" distributions. However this approach is not suited to synthetic generation of the aggregate data traffic: the output from several of these chaotic map/on-off process have to be combined to produce a data traffic stream with a given Hurst parameter. This adds to the simulation complexity. The fractal brownian motion on the other hand produces the aggregate data traffic with the given Hurst parameter. The approach is however not suited to synthetic traffic generation, because the approach requires that the fractal brownian stream $Z(t)$ be produced before the self-similar stream of $A(t)$ may be produced (see Eqn. 4.6).

Our emphasis in this thesis is to develop a traffic generator that is efficient in characterizing the correlations and variability present in the multi-media traffic so that a synthetic traffic stream may be generated using the same. However, for long-range dependent data traffic we are faced with a situation where none of the existing approaches may be used to produce this stream. We do not intend to characterize the individual sources that contribute to the aggregate traffic stream. We are interested in characterizing the aggregate traffic stream itself. We are looking for some simple process that could be easily simulated. The Doubly Stochastic Poisson Process (DSPP) model was examined as a candidate.

DSPP is a class of Poisson process, whose rate of events varies according to a stochastic process itself. The class of DSPP models is a wide one covering both stationary and non-stationary point processes and processes which are generated by continuous rate processes or non-continuous rate processes (such as when the rate changes its value only at particular instants). Thus a doubly stochastic Poisson process is called so because it inherits its stochastic properties both due to the random variation of the Poisson rate and the usual Poisson process variability. The DSPP was first introduced by D.R. Cox [71]. As given in [72], the Index of dispersion of counts (IDC) of a DSPP whose intensity follows a stationary stochastic process $\Lambda(t)$ in continuous time is given by

$$I(t) = 1 + \frac{2\sigma_\Lambda^2}{\lambda t}\int_0^t (t-u)\rho(u)du \tag{4.7}$$

where

σ_Λ^2 = variance of the function $\Lambda(t)$.

$\rho(u)$ = autocorrelation function of $\Lambda(t)$.

As seen from Eqn 1.7, the count is overdispersed than a Poisson process (As discussed in Chapter 2, the IDC of a Poisson process is 1.) Now, consider the asymptotic behaviour of Eqn 1.7 as t tends to infinity [72]. Two cases arise depending on the autocorrelation function $\rho(u)$ of $\Lambda(t)$.

- *a)* If $\rho(u)$ dies down exponentially fast (which is the case with most stochastic processes) and if $\int_0^t u\rho(u)du = o(x)$ and $\int_0^\infty \rho(u)du$ is convergent to a non zero value, A_ρ, then
$$\int_0^t (t-u)\rho(u)du \sim t\int_0^\infty \rho(u)du = tA_\rho$$
Hence as t tends to infinity the IDC converges to
$$I(t) = 1 + \frac{2A_\rho\sigma_\Lambda^2}{\lambda} \qquad (1.8)$$

- *b)* If $\rho(u)$ does not die down exponentially fast but dies down as a power law, i.e., $\rho(u) = o(u^{-\beta})$ where $0 < \beta < 1$, then
$$I(t) = Kt^{1-\beta} \qquad (1.9)$$
Such processes exhibit long range dependence and self-similarity, as was discussed in Chapter 2.

Hence doubly stochastic Poisson processes are very versatile in characterizing both short term dependent and long range dependent (self-similar) processes [73]. By choosing suitable stochastic processes for $\Lambda(t)$ one may model various scenarios.

As a special case of DSPP, consider a Poisson process whose rate alternates between two levels λ_1 and λ_2. The two levels endure for times forming an alternating renewal process with interval probability density functions (p.d.f) $f_1(x)$ and $f_2(x)$, respectively. Since it is supposed that the change over instants of the levels are not observable, the resulting process becomes an interesting point process. Depending upon the distributions of $f_1(x)$ and $f_2(x)$, the process may yield interesting results. If for instance, $f_1(x)$ and $f_2(x)$ are exponentially distributed, then we have

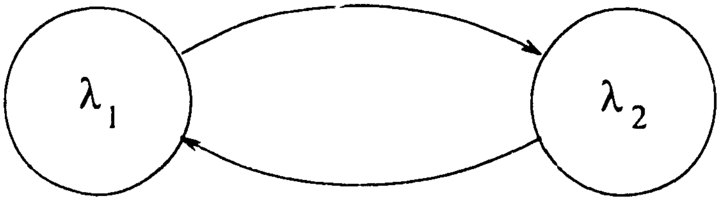


Figure 4.7: PMPP model

a Markov Modulated Poisson process (MMPP). The MMPP (Markov modulated Poisson process), has previously been successfully used to model the arrival process from a set of voice sources [21], [22] and a set of video sources [41], [42], [43], [1]. The MMPP is itself a correlated non-renewal stream. In these methods the MMPP models can accurately characterize the aggregate arrival process (either from a set of voice sources or from a set of video sources as the case may be) because a large number of statistics can be matched and the correlations among the arrival process accurately captured.

In this thesis we propose a new class of DSPP. The model consists of a Poisson process switching between 2 rates λ_1 and λ_2. The sojourn time in these 2 states are independent and identically distributed with a Pareto distribution with parameter α. This model exhibits long term dependence and self-similarity. This model is also very simple and ideally suited for synthetic traffic generation. Hence this model may be used to model the self-similar data traffic. The two states of this switched Poisson process would correspond to the long and short burst rates of the data traffic. The sojourn time distribution is chosen to be a thick tailed one in order to capture the long term dependencies in the net arrival process.Since the Poisson process is switched between two rates by the underlying Pareto distribution, we call this model a Pareto modulated Poisson process (PMPP) [4].

In the next section we shall explore in detail the statistical characteristics of this model and prove that the model exhibits long-range dependence and self-similarity.

4.3 Statistical characteristics of the PMPP model

In order to determine if the model captures the long term correlations, we look at the IDC (Index of dispersion of Counts) and the Variance time plots of the model. For a given time interval of length t, the Index of dispersion of counts is given by the ratio of the variance of the no. of arrivals during the interval to the mean of the number of arrivals in the same interval. If we divide the time axis into equal intervals called frames and if $(X_1, X_2, X_3, \ldots)$ are the number of packets generated by the process in successive frames, then IDC is defined as follows.

$$IDC(t) = Var(X_1 + X_2 + X_3 + \ldots\ldots X_t)/nX_{avg} \tag{4.10}$$

where X_{avg} is the average number of packets generated in a frame. IDC of a process is indicative of the burstiness of the process. Pure Poisson process has a IDC of 1. A process having IDC greater than one is overdispersed while that having IDC below one is underdispersed. For a self-similar stream of Hurst parameter H, IDC increases monotonically and is proportional to t^{2H-1}. Hence such an IDC when plotted in a log-log plot produces a straight line appearance. The value of the Hurst parameter, H, of the stream may then be calculated from the slope m of the IDC curve, in log-log plot. i.e.,

$$H = (m+1)/2 \tag{4.11}$$

The PMPP model considered is akin to the random hazard doubly stochastic Poisson process considered in [29] and [30]. Generally stated such processes alternate between two levels λ_1 and λ_2, with the sojourn times in each state forming an alternating renewal process with interval p.d.f.s $f_1(x)$ and $f_2(x)$ respectively. If ν_1 (ν_2) is the average sojourn time in state 1 (state 2), $f_1^*(s)$ ($f_2^*(s)$) is the Laplace transform of the p.d.f of the sojourn time in state 1 (state 2) and if $R_1^*(s)$ ($R_2^*(s)$) is the Laplace transform of the survivor function in state 1 (state 2), then the Laplace transform of the probability generating function $\phi^*(z,s)$, of the number of arrivals

$N(t)$ in time t is given by [30]

$$\begin{aligned}\phi^*(z,s) &= \frac{1}{\nu_1+\nu_2}\left(\frac{\nu_1}{s+\lambda_1(1-z)}+\frac{\nu_2}{s+\lambda_2(1-z)}\right)\\ &-\frac{(\lambda_1-\lambda_2)^2}{\nu_1+\nu_2}\left(\frac{(1-z)^2}{(s+\lambda_1(1-z))(s+\lambda_2(1-z))}\right)\\ &\times\left(\frac{R_1^*(s+\lambda_1(1-z))R_2^*(s+\lambda_2(1-z))}{(1-f_1^*(s+\lambda_1(1-z)f_2^*(s+\lambda_2(1-z)))}\right)\end{aligned}$$

(4.12)

The mean of the counting process can be obtained from the probability generating function, as in Chapter 2, Eqn. 3.5 as

$$E\{[N(t)]\}=\left(\frac{\lambda_1\nu_1+\lambda_2\nu_2}{\nu_1+\nu_2}\right)t \tag{4.13}$$

The Laplace transform of variance of $N(t)$ can be obtained by differentiating Eqn. 3.4 twice and setting z = 1.

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1\nu_1+\lambda_2\nu_2}{(\nu_1+\nu_2)s^2}+\frac{2(\lambda_1-\lambda_2)^2}{(\nu_1+\nu_2)^2}\frac{\nu_1\nu_2}{s^2}\\ &\times\left[\frac{1}{s}-\left(\frac{\nu_1+\nu_2}{\nu_1\nu_2}\right)\left(\frac{R_1*(s)R_2^*(s)}{1-f_1^*(s)f_2^*(s)}\right)\right]\end{aligned} \tag{4.14}$$

An explicit equation for the variance may be obtained by inverting the above equation depending on the sojourn time densities $f_1(t)$ and $f_2(t)$.

Now, for PMPP

$$\begin{aligned}&f_1(t)=f_2(t)=\alpha t^{-(\alpha+1)} && 1<\alpha<2 \quad t\geq 1\\ &R_1(t)=R_2(t)=t^{-\alpha} && 1<\alpha<2 \quad t\geq 1\\ &F_1(t)=F_2(t)=1-t^{-\alpha} && 1<\alpha<2 \quad t\geq 1\\ &\nu_1=\nu_2=\tfrac{\alpha}{\alpha-1}\end{aligned}$$

Since here, $t\geq 1$ the Laplace transforms of these functions have to be computed from definition as follows. Let

$$R_1^*(s)=R_2^*(s)=\int_1^\infty \exp(-st)t^{-\alpha}dt=g(s,\alpha) \tag{4.15}$$

Integrating by parts, we have the following recursion,

$$g(s,\alpha) = \frac{1}{s}\left[\exp(-s) - \alpha g(s,\alpha+1)\right] \qquad (4.16)$$

Extending the recursion,

$$g(s,\alpha) = \frac{\exp(-s)}{s}\left[1 + \sum_{i=1}^{\infty}(-1)^i \frac{\prod_{k=0}^{i-1}(\alpha+k)}{s^i}\right] \qquad (4.17)$$

Now,

$$f_1^*(s) = f_2^*(s) = g(s,\alpha+1)$$

Substituting the above functions in Eqn. 3.6 and making use of recursion 4.16 we have

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1+\lambda_2}{2s^2} + \\ &\frac{(\lambda_1-\lambda_2)^2}{2s^3}\left[1 + \frac{2(\alpha-1)}{\alpha s}\frac{\left[\exp(-s) - \alpha g(s,\alpha+1)\right]^2}{\left[1-\alpha^2 g^2(s,\alpha+1)\right]}\right]\end{aligned}$$

Now, for small s, i.e., as $s \to 0$, $\exp(-s) \approx 1$, then

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1+\lambda_2}{2s^2} + \\ &\frac{(\lambda_1-\lambda_2)^2}{2s^3}\left[1 + \frac{2(\alpha-1)}{\alpha s}\frac{\left[1 - \alpha g(s,\alpha+1)\right]}{\left[1+\alpha g(s,\alpha+1)\right]}\right]\end{aligned}$$

Now from recursion 4.16, $1 + \alpha g(s,\alpha+1) = 2 - sg(s,\alpha)$ and $1 - \alpha g(s,\alpha+1) = sg(s,\alpha+1)$, hence

$$\begin{aligned}\mathcal{L}\{Var[N(t)]\} &= \frac{\lambda_1+\lambda_2}{2s^2} + \\ &\frac{(\lambda_1-\lambda_2)^2}{2s^3}\left[1 + \frac{2(\alpha-1)}{\alpha}\frac{g(s,\alpha+1)}{2-sg(s,\alpha)}\right]\end{aligned}$$

Now, for small s, $sg(s,\alpha) \approx 1$, (from Eqn. 4.17). Also for small s, the first term inside the braces may be neglected. Inverting, the final equation we have

$$Var[N(t)] = \frac{(\lambda_1+\lambda_2)}{2}t + \frac{(\lambda_1-\lambda_2)^2}{2}\left(\frac{\alpha-1}{\alpha}\right)t^{3-\alpha} \qquad (4.18)$$

Also, mean is given by,

$$E[N(t)] = \frac{\lambda_1 + \lambda_2}{2} t \tag{4.19}$$

Hence the IDC may be obtained by dividing the variance by the mean and is given by

$$IDC(t) = 1 + \frac{(\lambda_1 - \lambda_2)^2}{\lambda_1 + \lambda_2} \left(\frac{\alpha - 1}{\alpha}\right) t^{2-\alpha} \tag{4.20}$$

As seen from the above expression, IDC increases as a fractional power of the interval under consideration. Such is the characteristics of a long range dependent self-similar process. When plotted in a log-log scale the IDC has a slope m equal to $2 - \alpha$. From (6) the Hurst parameter, H may be derived from the slope m as follows

$$H = \frac{3 - \alpha}{2} \tag{4.21}$$

We arrive at the same relation as in [70]. Hence as we vary the parameter α of the Pareto distribution, the Hurst parameter of the packet stream generated varies.

The PMPP model was simulated on OPNET and the IDC was computed. Figures 4.8 and 4.9 compare the IDC curves obtained from Eqn. 4.20 against simulation for values of $\lambda_1 = 100$, $\lambda_2 = 120$ and $\alpha = 1.3$ and 1.5 respectively. As seen from these curves, the results obtained from simulation agree fairly with the theoretical results. The IDC plot for $\lambda_1 = 100$ and $\lambda_2 = 120$ and for various values of α of the Pareto distributed sojourn times is shown in Figure 4.10. As seen, the linear characteristics of IDC in a log-log plot suggest that the model exhibits self-similar characteristics. Also, given in the figure are the Hurst parameter H estimated from the slope of the IDC. It is seen that the Hurst parameter so obtained satisfies the relation 4.21, quite fairly.

The Variance time curves for the PMPP were also obtained from simulation. The variance time curve is obtained by computing the variance of the arithmetic mean of the count process. i.e., if as before $X = (X_1, X_2, X_3, \ldots)$ denote the number of packets generated by the process in successive frames, let $X^{(m)} = (X_k^{(m)}\ ; k =$

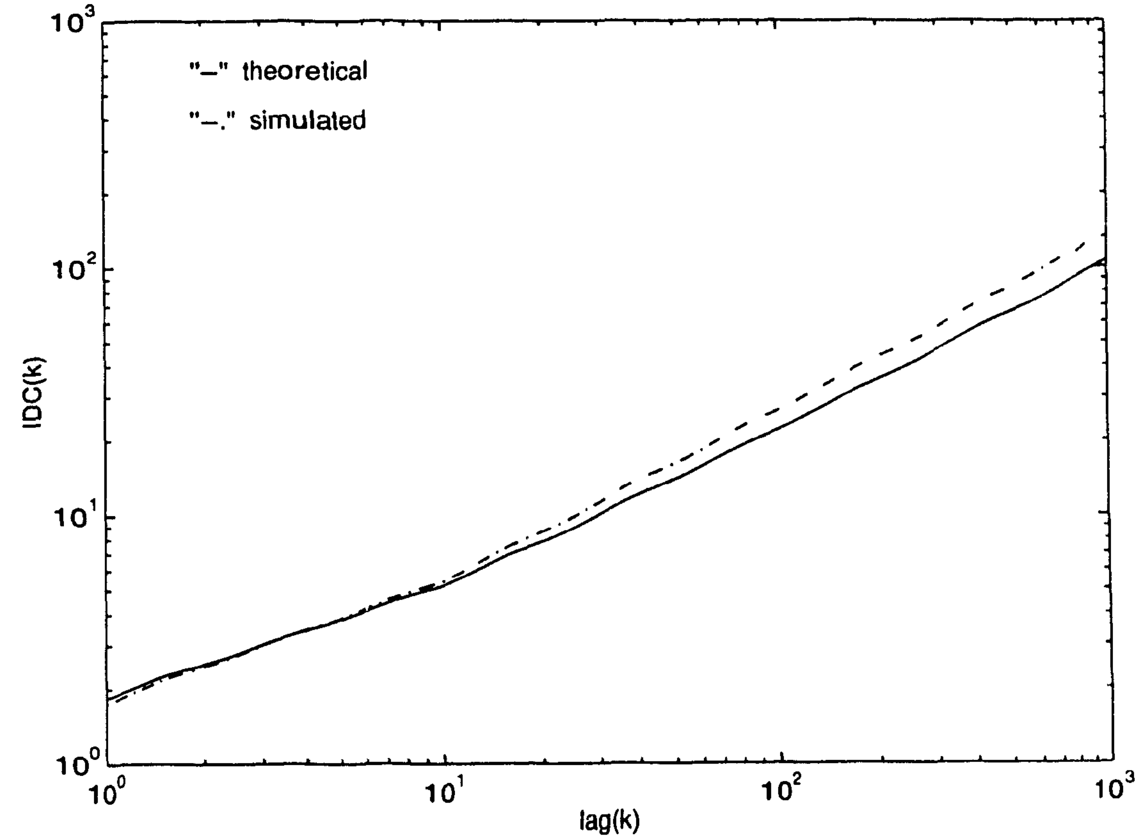


Figure 4.8: Simulated and theoretical curves of IDC, for PMPP, with $\lambda_1 = 100$, $\lambda_2 = 120$ and $\alpha = 1.3$

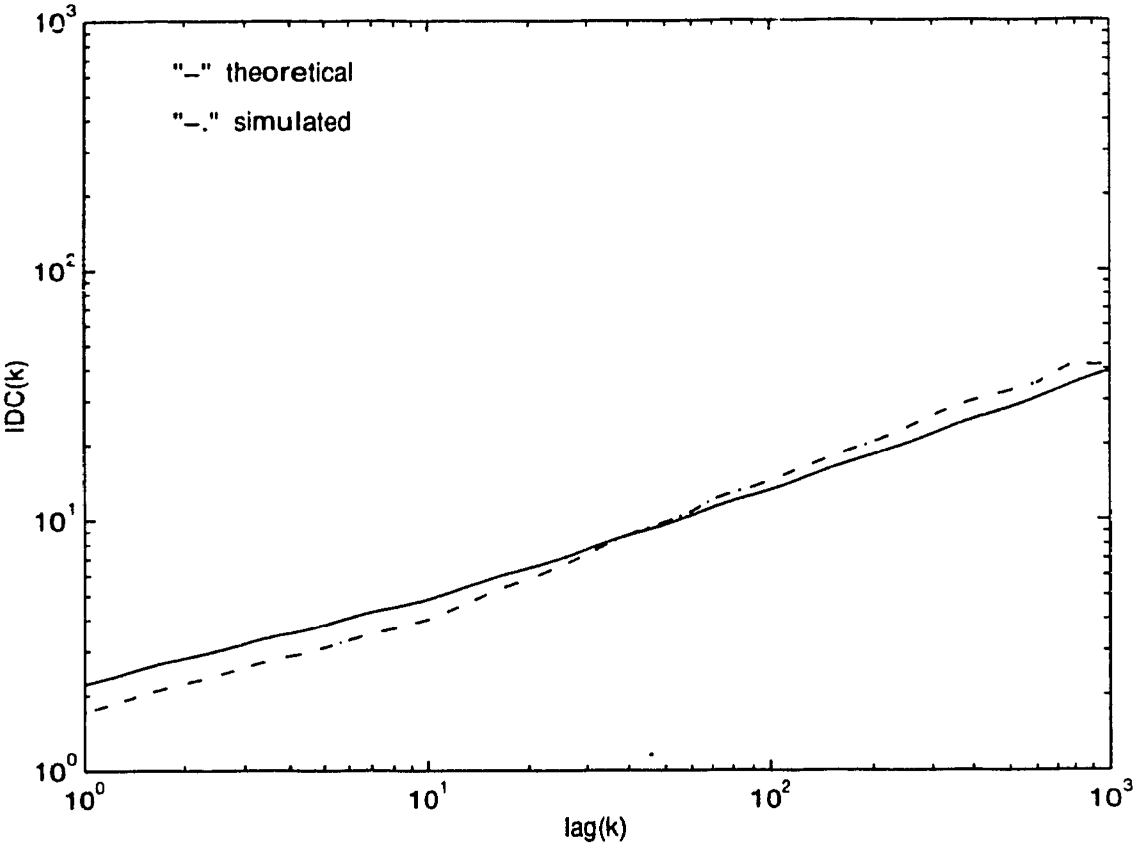


Figure 4.9: Simulated and theoretical curves of IDC, for PMPP, with λ_1 = 100,λ_2 = 120 and α = 1.5

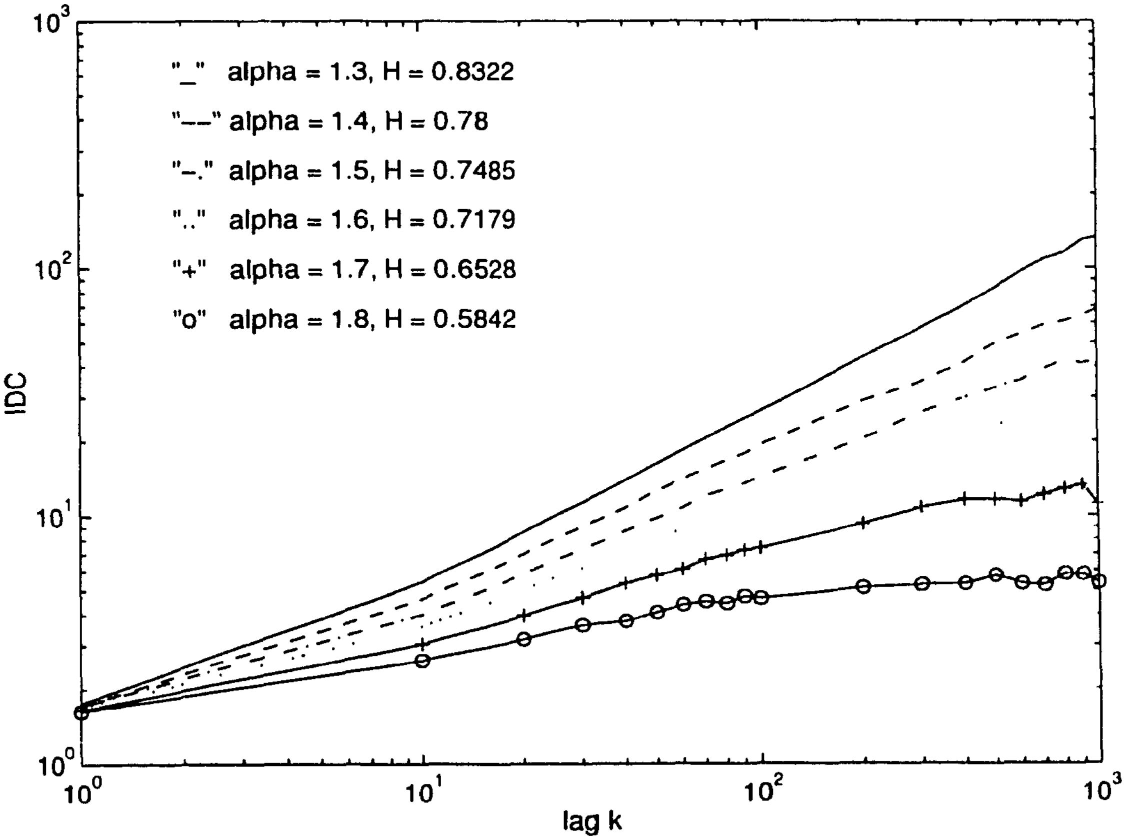


Figure 4.10: IDC curves for various values of α, with $\lambda_1 = 100$, $\lambda_2 = 120$

1, 2, 3, ...) denote a new (aggregated) time series obtained by averaging the original series X over non-overlapping blocks of size m.i.e. for each $m = 1, 2, 3, \ldots$, $X^{(m)}$ is given by $X_k^{(m)} = 1/m(X_{m(k-1)} + \ldots + X_{km})$. Then plotting Variance($X^{(m)}$ against various values of m gives the variance time plots. While for conventional models the variance of the sample mean is inversely proportional to the sample size , for long-range dependent processes, it decreases as a fractional power of sample size (i.e., it decreases more slowly than the reciprocal of the sample size). Hence in the case of long-range dependent processes

$$\mathrm{Var}(X^{(m)}) = a_1 m^{-\beta} \quad \text{with} \quad 0 < \beta < 1$$

where a_1 is a constant. When the variance time curve is plotted in a log-log scale, the slope β is related to the Hurst parameter, H, by the relation

$$H = 1 - |\beta|/2 \tag{4.22}$$

Figure 4.11 shows the variance time plot for various values of α with $\lambda_1 = 100$ and $\lambda_2 = 120$) obtained from simulation. The linear behaviour of Variance time curve in a log-log plot shows the presence of slowly decaying variances. The Hurst parameter estimated from these graphs also indicate that the relation 4.21 holds well.

Hence the PMPP is efficient in characterizing the fractal nature of the data traffic. Also the proposed model captures the presence of the long and short bursts inherent in data traffic. This model is easy to simulate when compared to other methods for generation of self-similar traffic. Hence this method may be used to generate a self-similar traffic stream with $H = (3 - \alpha)/2$. The other 2 parameters of the model namely λ_1 and λ_2 are to be matched with that of the aggregate traffic stream by a suitable matching technique (an illustration of which is given below).

The PMPP model was used to match an actual traffic trace from Bellcore Ethernet traffic data from October 1989. The traffic file is available via anonymous FTP from *flash.bellcore.com*. The Hurst parameter was estimated from a log-log plot of the IDC of the trace, to be 0.8202. From the Hurst parameter H of the

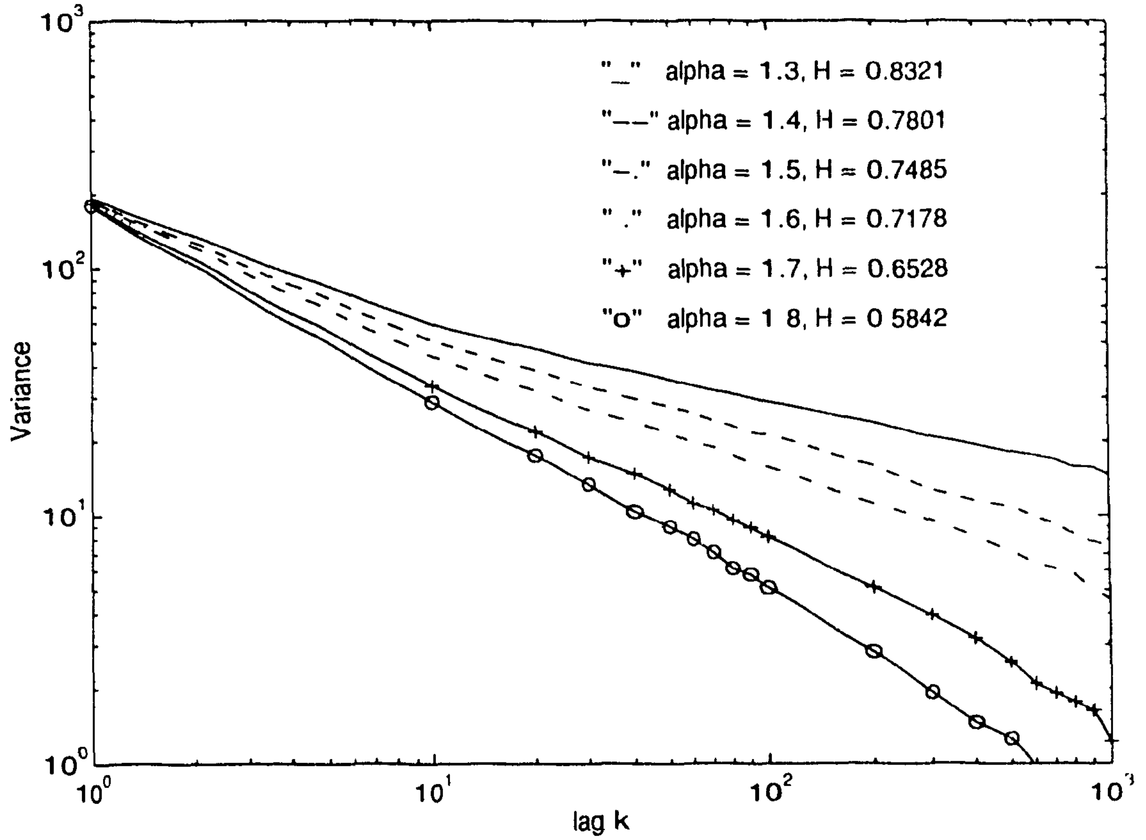


Figure 4.11: Variance time curves for various values of α, with $\lambda_1 = 100$, $\lambda_2 = 120$

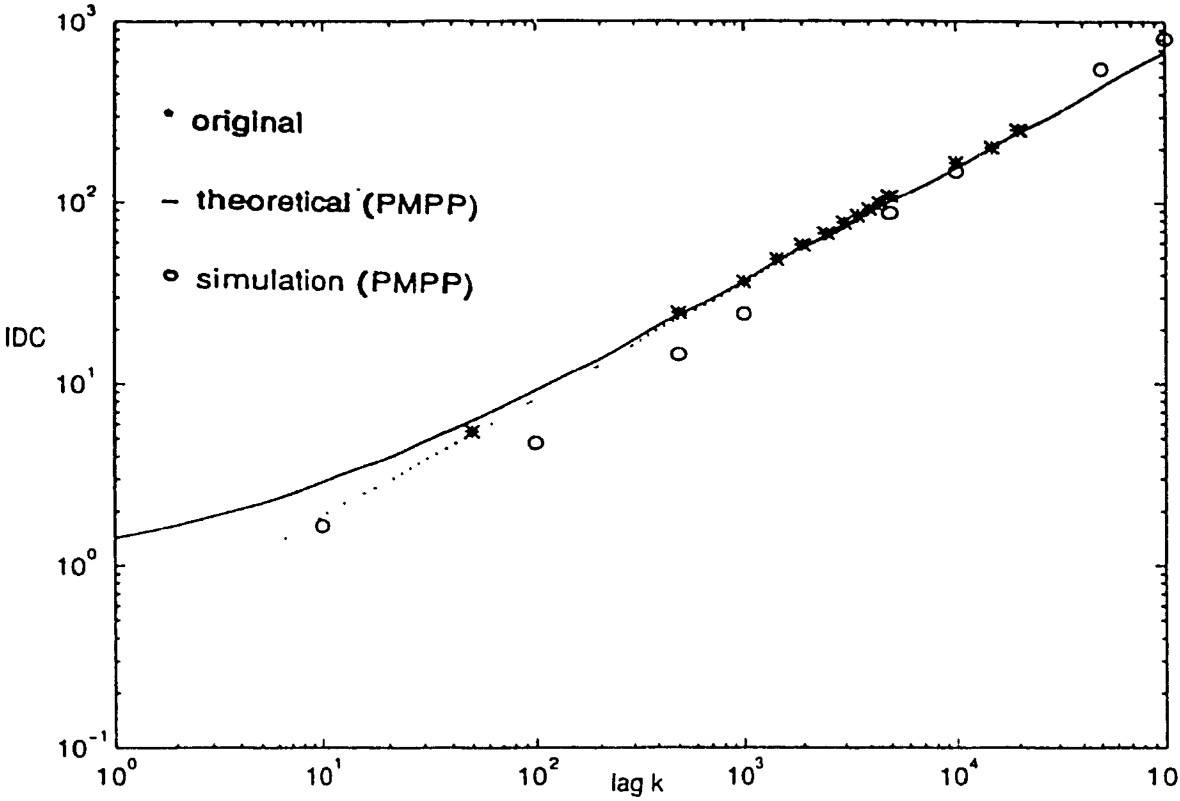


Figure 4.12: IDC plots of the traffic trace obtained from Bellcore traffic data compared against the plots obtained from the simulation of PMPP model.

trace the parameter α was determined to be α = 1.3596 using relation 4.21. The parameters λ_1 and λ_2 were obtained by matching the average number of packets generated from the traffic data and IDC(1) (i.e. IDC at lag 1) of the data with equations 4.19 and 4.20 respectively. The estimated values of λ_1 and λ_2 are λ_1 = 2.8565 packets/ $10ms$ and λ_2 = 6.8235 packets/$10ms$. The PMPP model was simulated using these values for the parameters λ_1. λ_2 and α and the IDC was plotted. The IDC plot obtained from simulations (circled plot) is compared against the original plot (starred plot) in Figure 4.12. Also shown in the figure is the plot of IDC (bold line) obtained by using the Eqn. 4.20. As can be observed from the figure the plots obtained from simulation closely follow the IDC plot obtained from the experimental data.

Next we compare the characteristics of the PMPP model with that of the Fractional Brownian traffic [2]. As outlined before the fractional Brownian traffic model [2] characterizes the aggregate traffic stream, with a given Hurst parameter. The FBM model characterizes the number of packet arrivals $A(t)$ in the time interval $(0,t]$ by

$$A(t) = mt + \sqrt{ma}Z(t)$$

$Z(t)$ is a normalized fractional Brownian motion. The process has 3 parameters m, a and H. $m > 0$ is the mean input rate, $a > 0$ is the variance coefficient, and $\frac{1}{2} < H < 1$ is the Hurst parameter of $Z(t)$. Thus with this model for a given Hurst parameter H, traffic can be generated using the relation given above. These parameters of the fractional Brownian traffic model can be matched with the parameters λ_1, λ_2, and α of the PMPP by matching the variance and mean of both the processes.

The mean input rate m, of the fractional Brownian traffic model can be equated with that of the PMPP model

$$m = \frac{\lambda_1 + \lambda_2}{2} \tag{4.23}$$

From [74] the variance of the fractional Brownian traffic is

$$Variance\{A(t)\} = mat^{2H} \tag{4.24}$$

Equating the above variance with that of PMPP, Eqn. 4.18 and using the relation 4.21 between Hurst parameter H and α,

$$\begin{aligned}
mat^{3-\alpha} &= \frac{(\lambda_1 + \lambda_2)}{2}t + \frac{(\lambda_1 - \lambda_2)^2}{2}\left(\frac{\alpha - 1}{\alpha}\right)t^{3-\alpha} \\
\frac{\lambda_1 + \lambda_2}{2}at^{3-\alpha} &= \frac{(\lambda_1 + \lambda_2)}{2}t + \frac{(\lambda_1 - \lambda_2)^2}{2}\left(\frac{\alpha - 1}{\alpha}\right)t^{3-\alpha} \\
a &= t^{\alpha-2} + \frac{(\lambda_1 - \lambda_2)^2}{\lambda_1 + \lambda_2}\left(\frac{\alpha - 1}{\alpha}\right)
\end{aligned}$$

As $t \rightarrow \infty$, $t^{\alpha-2} \rightarrow 0$ for $1 < \alpha < 2$

$$a = \frac{(\lambda_1 - \lambda_2)^2}{\lambda_1 + \lambda_2}\left(\frac{\alpha - 1}{\alpha}\right) \tag{4.25}$$

Thus equations 4.23, 4.25 give the relation between the parameters of fractional Brownian traffic and that of PMPP.

In [74], the parameters m, a and H of the fractional Brownian traffic model were estimated by linear regression for the Bellcore Ethernet traffic data from October 1989 (available with anonymous FTP from *flash.bellcore.com*). The estimated parameters of the fractional Brownian traffic for this traffic trace are

$$\begin{aligned} m &= 2279 kbit/sec \\ a &= 262.8 kbit.sec \\ H &= 0.78 \end{aligned}$$

We match this fractional Brownian traffic model with the PMPP model. Using a packet size of 53 bytes, and converting the values of fractional Brownian traffic model from Kbit to packets, we have

$$\begin{aligned} m &= 5379 packet/sec \\ a &= 619.81 packet.sec \\ H &= 0.78 \end{aligned}$$

The parameters of the equivalent PMPP model, were obtained from relations 4.23 and 4.25 as

$$\begin{aligned} \lambda_1 &= 3040 packet/sec \\ \lambda_2 &= 7709 packet/sec \\ \alpha &= 1.44 \end{aligned}$$

Figure 4.13 shows the IDC obtained from the fractional Brownian traffic model and PMPP model for the Bellcore Ethernet traffic data. As seen from the figure, both the curves have the same slope and they follow each other closely.

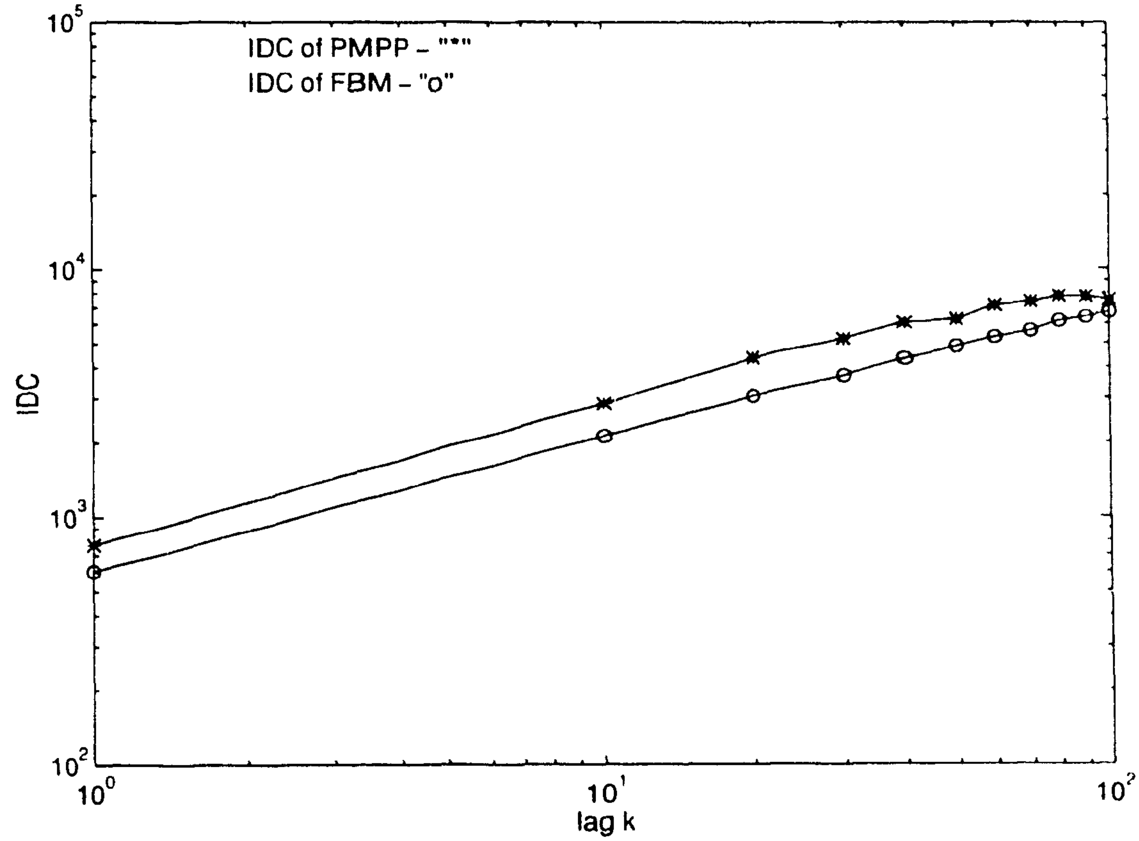


Figure 4.13: IDC curves of FBM model and PMPP model for the Bellcore Ethernet traffic data

Chapter 5

Traffic generator for multi-media services

Earlier chapters outlined in detail the characteristics of the various constituents of the aggregate multi-media traffic. As seen there, multi-media traffic exhibits a diverse mixture of traffic characteristics. In this chapter we present the traffic generator developed in this study. The traffic generator has a selection of traffic models that may be used to characterize the constituents of multi-media traffic. The constituent models of the traffic generator are presented. Finally we use the traffic generator developed to study a G/D/1 queueing system.

5.1 Requisites of the traffic generator

Our goal in this thesis has been to build a traffic generator that aptly characterizes the variability and statistical correlations in the packet arrival process. As seen in the earlier chapters, multi-media traffic exhibits a wide spectrum of traffic characteristics. Thus the traffic generator should be versatile in capturing these statistical characteristics. The developed traffic generator is to be used for network performance evaluation or evaluation of multiple access schemes or for evaluating/devising connection admission control and source policing algorithms. In other words, our main intent here is not to model the actual arrival process at the source level, but to develop a model that can capture enough statistical characteristics of the aggregated (multiplexed) arrival process, so that the arrival process when fed to a queue produces the same queueing characteristics as that produced by the actual multiplexed traffic.

Recent studies of LAN data traffic indicate that such traffic exhibits long range dependence and self-similar (or fractal) characteristics, i.e., the traffic exhibits "burstiness" across a wide range of time scales ranging from milliseconds to hours. Hence, in a multi-media environment fractal traffic co-exists with non-fractal traffic. Characterizing such a mix of traffic by an unique model poses a great challenge to the modeler. The model proposed should be versatile in the sense that it should be able to capture the long term and short term correlations of the multiplex. Hence in our aggregate traffic generator we model the individual components of the multi-media traffic, namely voice, video and data traffic and then superpose them to produce the aggregate traffic stream.

The following aspects have been considered in selecting an appropriate traffic model for the traffic generator.

- *accuracy:* The models chosen to represent the individual components of multimedia traffic in the traffic generator must capture the statistical characteristics of the traffic they characterize.

- *parsimonious models:* The models should be parsimonious in the number of parameters, lest the parameters lose their physical significance. The requirement of parsimonious models (i.e., models with a few parameters) is very important from the point of view of synthetic traffic generation.

- *flexibility:* The models should be flexible in the sense that they should be able to represent varied statistical characteristics by tuning the parameters of the model.

- *simplicity:* The models must be extremely simple from the point of view of synthetic traffic generation. This is a very important requirement because some models may be very accurate but it may be extremely complex to synthetically generate traffic using these models.

The traffic generator built for the generation of multi-media traffic consists of suitable models chosen to characterize aggregate voice, video and data traffic respectively.

5.2 The traffic generator

The block diagram of the traffic generator is given in Figure 5.1. The traffic generator offers a wide choice of models that could be used to capture the varied statistical characteristics in the aggregate traffic. As shown in the figure voice traffic can be characterized by employing many individual on-off sources (one for every voice source), characterized by their output rate $\lambda^{(vc)}$, transition rate from ON state $\alpha^{(vc)}$ and transition rate from OFF state $\beta^{(vc)}$. The ON-OFF sources have exponentially distributed ON and OFF times. Instead the aggregate traffic from the voice sources may be characterized by a MMPP with parameters $\lambda_1^{(vc)}$, $\lambda_2^{(vc)}$, $alpha_1^{(vc)}$ and $\alpha_2^{(vc)}$. The matching technique by which the 4 parameters of the MMPP are determined from the original superposition of the voice processes may be found in [21] and [22].

For video traffic, the parameter $N_{mini-sources}$, specifies the number of mini ON-OFF sources (with exponentially distributed On and OFF times) to be used to simulate the output from one video source. Accumulating many sets (one set for every video source) of such sources characterizes the aggregate traffic from video sources. If the characterization of aggregate video traffic is directly sought, a MMPP with suitable parameters $\lambda_3^{(vd)}$, $\lambda_4^{(vd)}$, $\alpha_3^{(vc)}$ and $\alpha_4^{(vd)}$ may be used instead of the ON-OFF source characterization. The matching techniques by which the 4 parameters

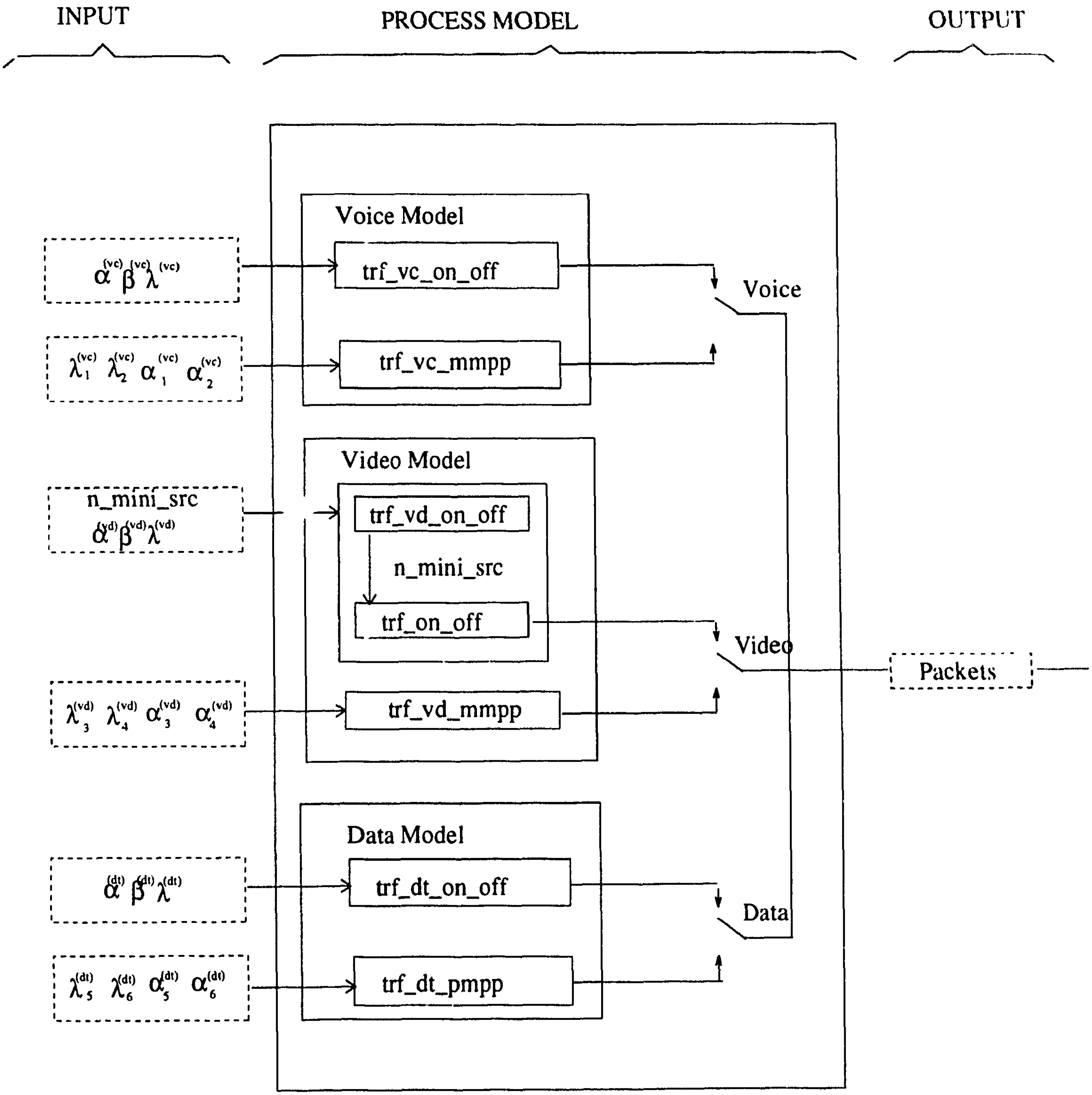

INPUT
PROCESS MODEL
OUTPUT
$\alpha^{(vc)}\beta^{(vc)}\lambda^{(vc)}$
$\lambda_1^{(vc)}\ \lambda_2^{(vc)}\ \alpha_1^{(vc)}\ \alpha_2^{(vc)}$
n_mini_src
$\alpha^{(vd)}\beta^{(vd)}\lambda^{(vd)}$
$\lambda_3^{(vd)}\ \lambda_4^{(vd)}\ \alpha_3^{(vd)}\ \alpha_4^{(vd)}$
$\alpha^{(dt)}\beta^{(dt)}\lambda^{(dt)}$
$\lambda_5^{(dt)}\ \lambda_6^{(dt)}\ \alpha_5^{(dt)}\ \alpha_6^{(dt)}$
Voice Model
trf_vc_on_off
trf_vc_mmpp
Video Model
trf_vd_on_off
n_mini_src
trf_on_off
trf_vd_mmpp
Data Model
trf_dt_on_off
trf_dt_pmpp
Voice
Video
Data
Packets

Figure 5.1: Block diagram of the traffic generator

of the MMPP are determined, may be found in [41] [42] [43] [1].

The data traffic in the traffic generator may either be characterized by employing many ON-OFF sources, with the ON and OFF periods characterized by a Pareto distribution with parameter $\alpha_O N$ and $\alpha_O FF$ respectively. Alternatively the aggregate traffic may directly characterized by employing a PMPP model with parameters $\lambda_5^{(dt)}$, $\lambda_6^{(dt)}$ and $\alpha^{(dt)}$.

In the traffic generator employing individual sources to generate the aggregate traffic increases the computing complexity. It is ideal to obtain an aggregate model for voice, video and data directly and use it to generate the traffic. The aggregate traffic model for multi-media traffic employing the appropriate traffic model for aggregate voice, video and data traffic that has been proposed here is as shown in Figure 5.2. This method is simple and is also versatile in capturing the statistical characteristics of the multiplex. The resulting model is the superposition of three 2-state switched Poisson processes, giving rise to an eight state switched Poisson process as shown in Figure 5.3. The model is simple and easy to simulate.

If the data traffic were approximated by a 2 state MMPP, then by the property that the superposition of MMPP is again an MMPP, we obtain an eight state MMPP. This may simplify the mathematical analysis of the model. However, this model is not accurate in the sense that it does not capture the long term correlations of data. On the other hand, if the PMPP is selected for data traffic, we obtain an eight state switched Poisson process, which may not simplify into a simple form as in the case of MMPP.

In the next section we use the developed traffic generator to study the performance of a G/D/1 queue by simulation.

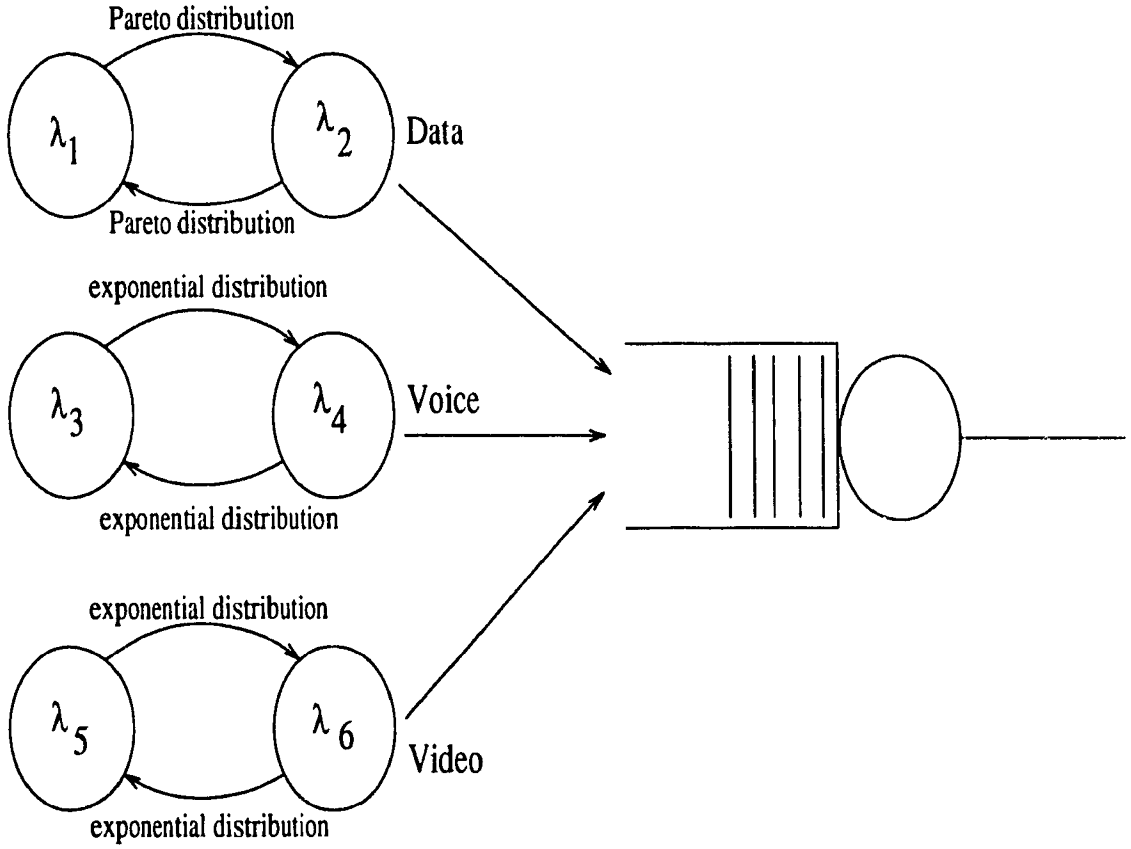


Figure 5.2: Aggregate traffic model

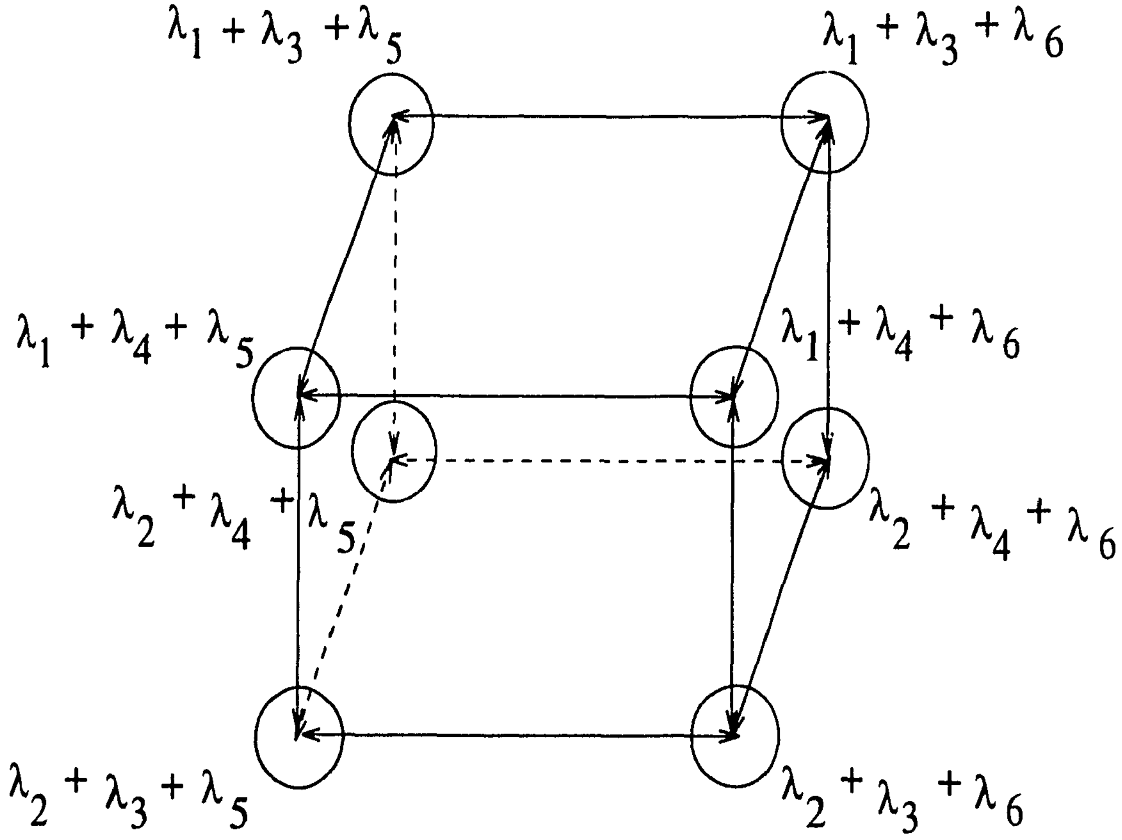

$\lambda_1 + \lambda_3 + \lambda_5$
$\lambda_1 + \lambda_3 + \lambda_6$
$\lambda_1 + \lambda_4 + \lambda_5$
$\lambda_1 + \lambda_4 + \lambda_6$
$\lambda_2 + \lambda_4 + \lambda_5$
$\lambda_2 + \lambda_4 + \lambda_6$
$\lambda_2 + \lambda_3 + \lambda_5$
$\lambda_2 + \lambda_3 + \lambda_6$

Figure 5.3: Superposed process

5.3 Performance evaluation of a G/D/1 queue

This section presents the simulation results for queueing performance of a G/D/1 queue fed by the arrivals from the traffic generator. The main contribution of this section is to add to the current efforts in gaining a better understanding of queueing performance when the input to the queue is not given by a traditional traffic model but instead by a long range dependent model. We also compare the queueing performance obtained from more conventional models such as the MMPP with that of the long range dependent model such as the PMPP. Next, we investigate the queueing performance of the aggregate multi-media traffic. Herein, we try to find out the effect of long range dependent traffic such as the data traffic on the aggregation. We present the simulation results for the queueing performance of aggregate traffic for various composition of the constituent long range dependent and short range dependent traffic.

The general queueing system that we consider for our study is as shown in Figure 5.4. The server serves a fixed number of packets per second, as is the case in an ATM multiplexer. The arrival process that we consider to the queue is a PMPP/MMPP or the aggregate traffic model proposed. The figure of merit chosen for the queueing system is the survivor function of the queue length (which is the complementary function of the probability distribution of queue length).

5.3.1 Performance of a PMPP/D/1 queue

Now, we consider the case when the queue of constant service time is fed by a PMPP model. The OPNET model of the PMPP/D/1 is simulated to obtain the complementary distribution of queue length. The parameters of the PMPP model considered are $\lambda_1 = 200$ pkts/unit time and $\lambda_2 = 250$ pkts/unit time. The loading

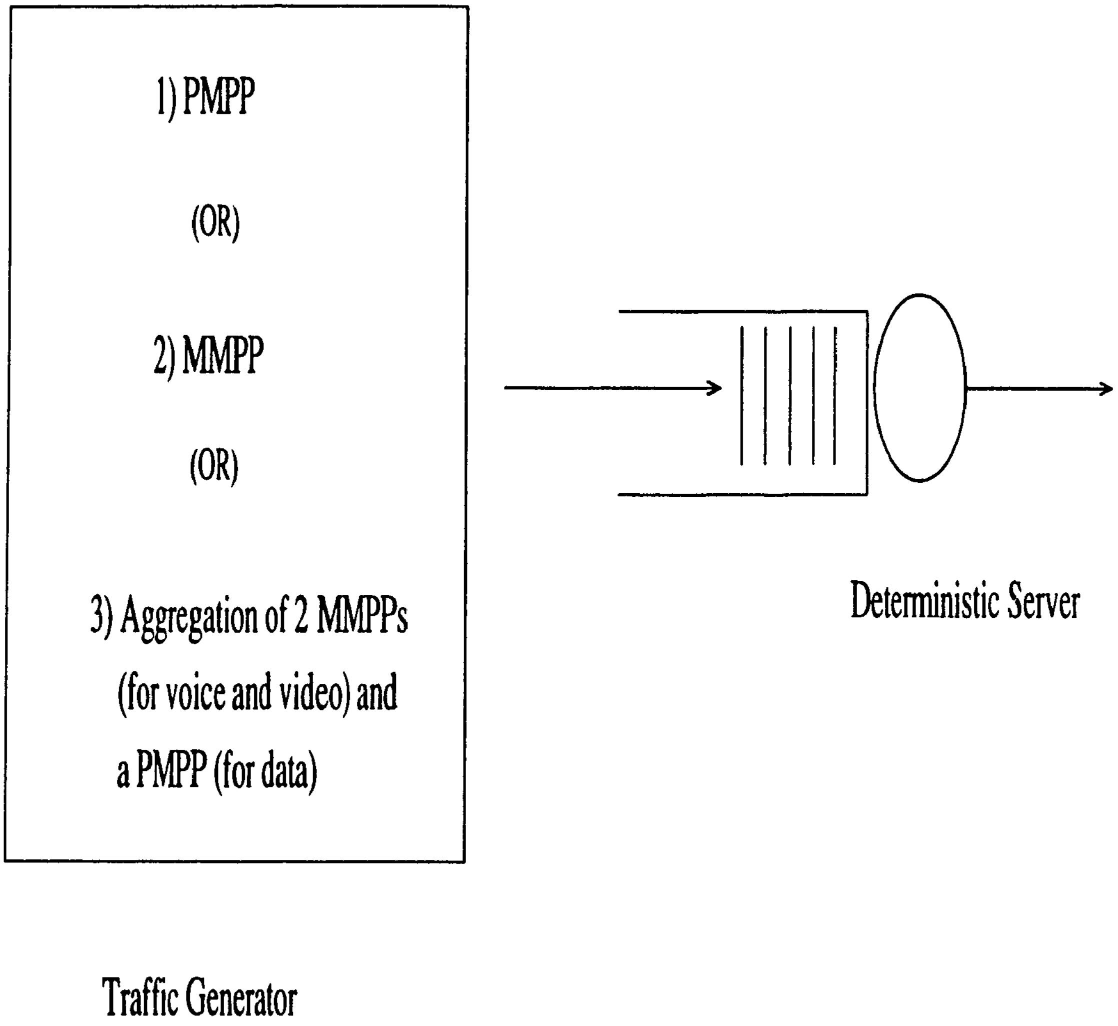


Figure 5.4: Block diagram of the G/D/1 queue

of the queue considered is 0.9. The parameter α of the PMPP is varied to obtain various Hurst parameters of the input stream. The logarithm of the complementary distribution $\log[P(X > x)]$ is plotted against x. It can be observed from Figure 5.5 that the tails of queue length distributions are not linear and are more heavy tailed. Such behaviour was found in the queueing simulations done with the actual data traces of long range dependent traffic in Bellcore [75]. In [75] queueing simulation experiments were performed with actual traces of Ethernet LAN traffic. In order

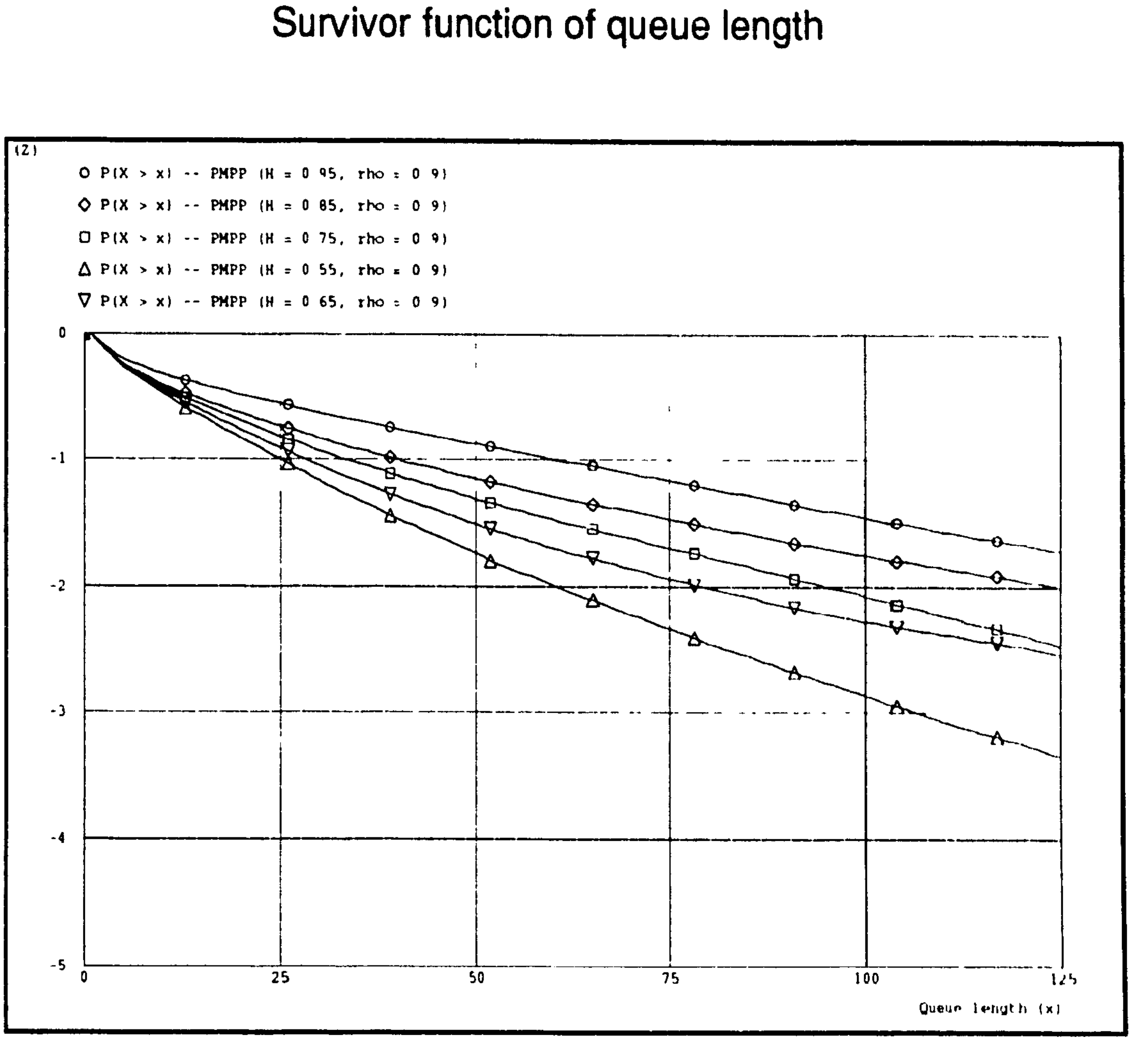


Figure 5.5: Survivor function of Queue length of PMPP arrival in a semi-log scale

to study the effect of long range dependence on the queueing behaviour, simulations were also conducted with "shuffled" versions of the traffic traces. Two kinds of shuffling were done with traces; one that preserves short term correlations only and one that preserves long term correlations only. It was observed that the simulations conducted with the shuffled traces that preserved the long term correlations only, produced the same thick tailed queueing behaviour produced by the original trace. The trace that preserves only short term correlations resulted in a queueing behaviour as would have been produced by more conventional models like MMPP or QNA approximations . Hence, the heavy tailed behaviour is attributable to the long range dependent correlations and the PMPP model captures this behaviour.

In [74], the G/D/1 queue fed by FBM (Fractional Brownian Motion) traffic was also shown to result in a Weibull distribution of the form

$$P(X > x) \approx \exp(-\gamma x^{(2-2H)}) \tag{5.1}$$

Also, similar results were obtained in [76] by aggregating many ON/OFF sources with heavy tailed sojourn times. Hence, the queue length decreases at a slower rate with increase in buffer space, than would be expected from the results obtained from a Poisson or MMPP model. Figure 5.5 also illustrates the fact that higher the Hurst parameter of the traffic stream more heavy tailed the queue length distribution is. This again .ndicates that a long range dependent stream with a high Hurst parameter may suffer more loss. This is also demonstrated by our simulation results. Figure 5.6 plots the probability of loss against the Hurst parameter, for a finite buffer size of K = 150, and with the same parameters of the PMPP model as before. As can be seen from the figure, higher the Hurst parameter of the input strec m, higher the loss.

In order to compare the performance of a MMPP in the same scenario, we also obtain the performance characteristic of the equivalent MMPP for the PMPP model under consideration; i.e., we choose λ_1 and λ_2 to be the same as before (λ_1 = 200 pkts/unit time and λ_2 = 250 pkts/unit time). Also, as in the case of PMPP, the sojourn times in both the states of the MMPP are identical but exponentially distributed. The average sojourn time in each state of the MMPP is equated to the average sojourn time in the corresponding state of the PMPP. This gives us a fair basis of comparison of the two models. Figure 5.7 shows the queueing behaviour of a PMPP with H = 0.95 and its equivalent MMPP plotted together. As seen from the figure, for the equivalent MMPP model, the logarithmic plot is asymptotically linear, indicating that the complementary distribution is exponential. This is markedly different from the queueing behaviour obtained from a PMPP.

The PMPP model captures long range dependence by modeling consecutive

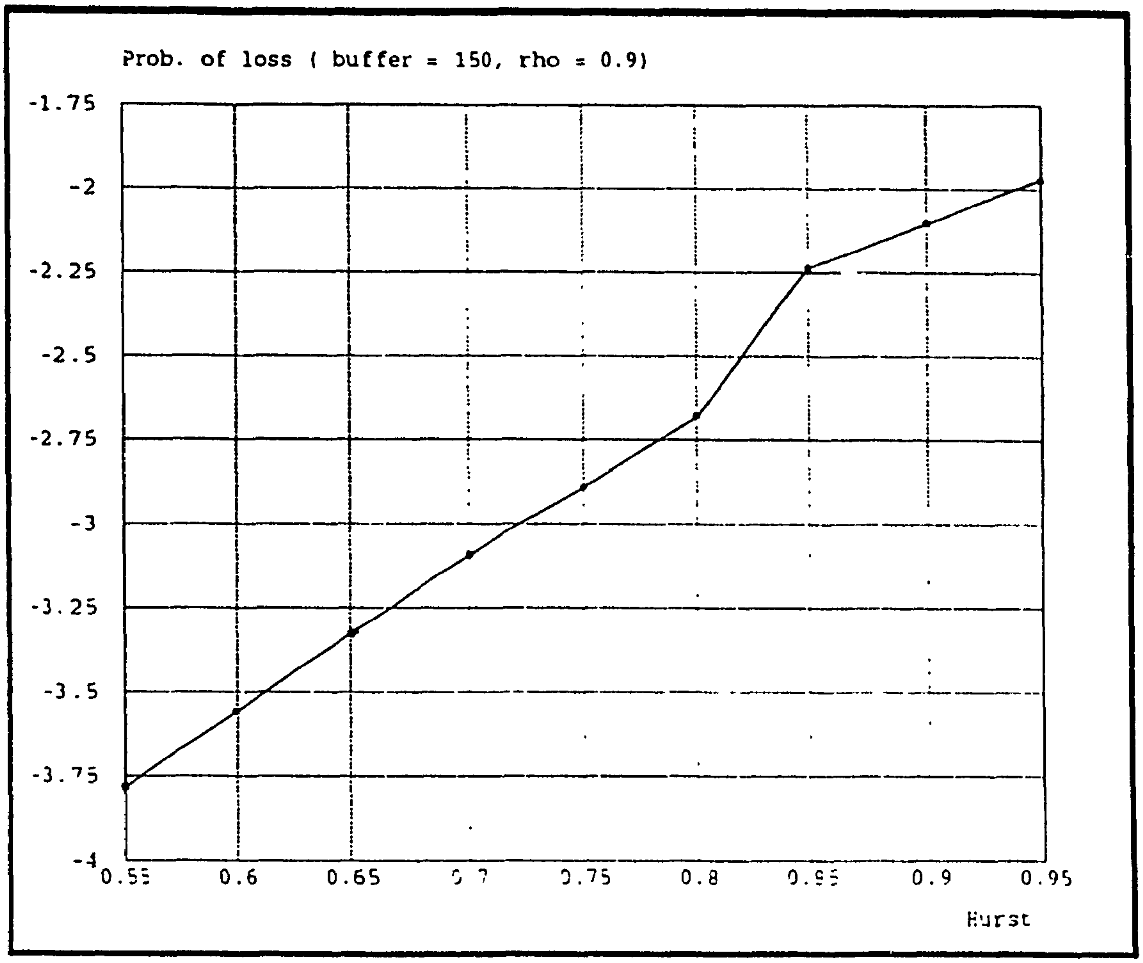


Figure 5.6: Probability of loss as a function of Hurst parameter in a semi-log scale

long and short burst periods that persist for a long time. These two rates λ_1 and λ_2 may correspond to the long and short burst rates inherent in data traffic. The difference between these two rates λ_1 and λ_2. $\delta\lambda$ may intuitively be thought as representing the burstiness of the traffic stream. The previous set of results with λ_1 = 200 and λ_2 = 250 had a $\delta\lambda$ of 50. Figure 5.8 plots the queue length for the case when $\delta\lambda$ = 100. The load is 0.9 as before, however now the λ_1 = 175 and λ_2 = 275. As seen from the figure the residual function of queue length is more heavy tailed than in the case of $\delta\lambda$ = 50. Hence with an increase in $\delta\lambda$. we see a more burstier

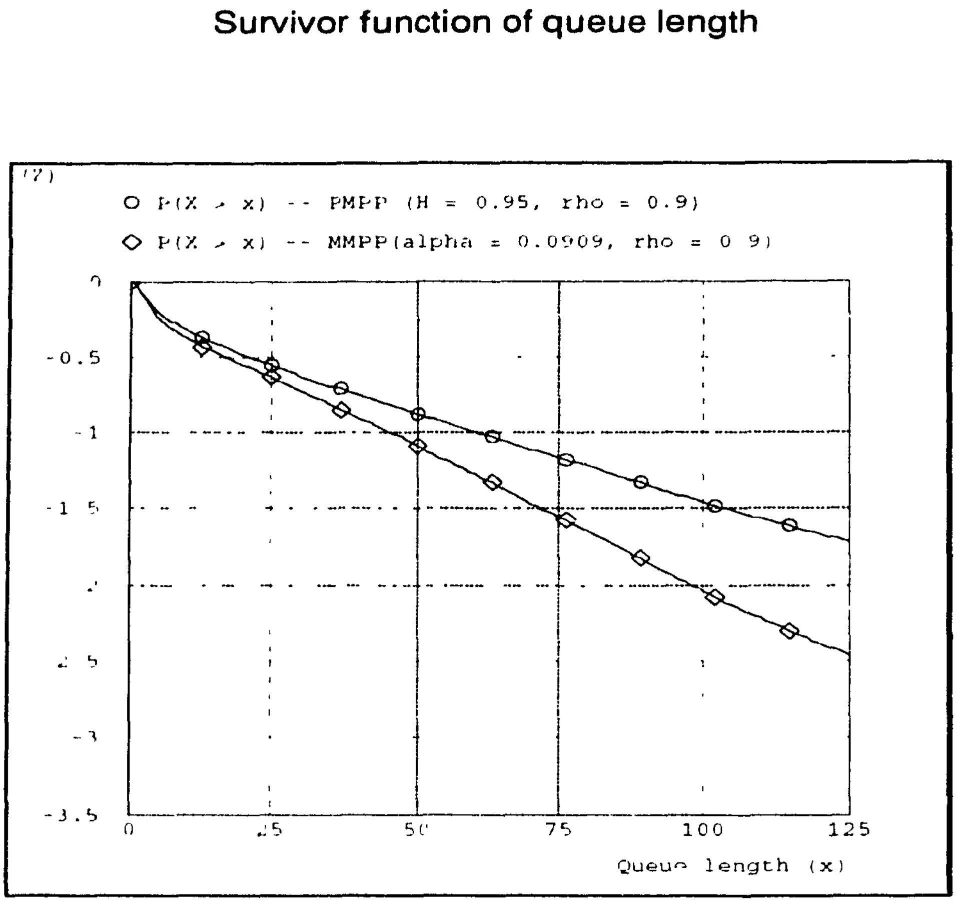


Figure 5.7: Survivor function of Queue length of PMPP and MMPP arrival in a semi-log scale

traffic stream.

5.3.2 Performance of aggregate traffic model

Now we investigate the G/D/1 queue by feeding it with the aggregate traffic consisting of voice, video and data. As discussed in the previous section, voice and video traffic are modeled by a MMPP, while data is modeled by a PMPP. To illustrate

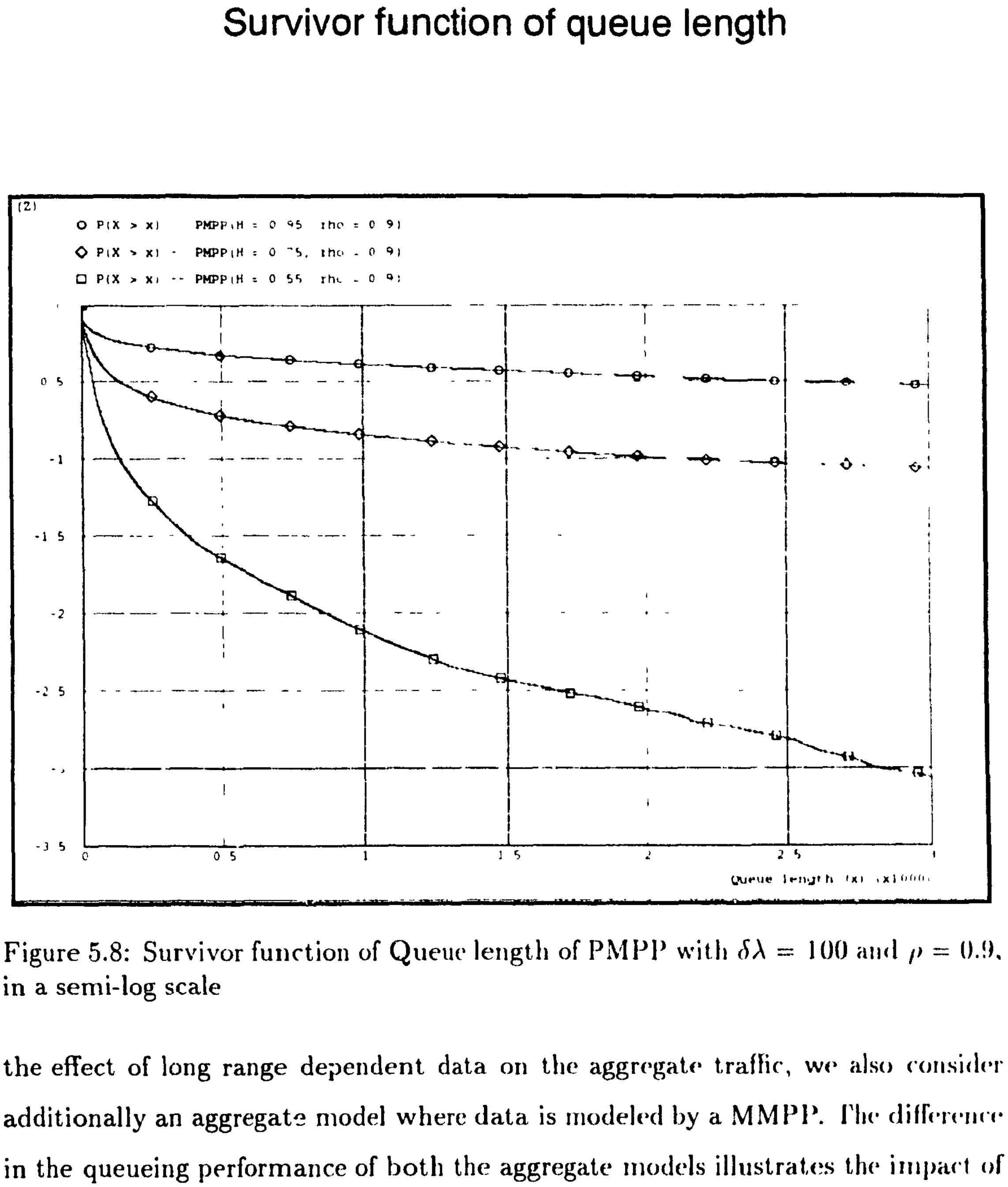


Figure 5.8: Survivor function of Queue length of PMPP with $\delta\lambda$ = 100 and $\rho = 0.9$, in a semi-log scale

the effect of long range dependent data on the aggregate traffic, we also consider additionally an aggregate model where data is modeled by a MMPP. The difference in the queueing performance of both the aggregate models illustrates the impact of long range dependent correlations. The parameter values used are as follows:

- for voice (MMPP): $\lambda_1 = 28$ pkts/ms ; $\lambda_2 = 41$ pkts/ms; $\alpha_1 = 0.000956\ ms^{-1}$; $\alpha_2 = 0.0250\ ms^{-1}$

- for video (MMPP): $\lambda_3 = 24$ pkts/ms ; $\lambda_4 = 39$ pkts/ms; $\alpha_3 = 0.0087\ ms^{-1}$; $\alpha_4 = 0.0483\ ms^{-1}$
- for data (PMPP): $\lambda_5 = 10$ pkts/ms; $\lambda_6 = 38$ pkts/ms; $\alpha_5 = 1.4$; $\alpha_6 = 1.4$.

where for the MMPPs the α stand for the transition rate from one state to another and for the PMPP the α is the parameter of the Pareto distribution. The above traffic composition has a ratio of 1:1:1. i.e., together voice and video are twice that of the data traffic. The server serves at the rate of 100 pkts/ms, thus giving a ρ of 0.8.

Figure 5.9 also show . the survivor function of queue length, when the PMPP model is replaced by a MMPP model in the aggregate model. The parameters of the MMPP models for voice and video are chosen as before. The parameters of the MMPP model chosen for data are$\lambda_5 = 10$ pkts/ms; $\lambda_6 = 38$ pkts/ms; $\alpha_5 = \alpha_6 = 0.2857$. As seen from figure 5.9 both the curves follow each other closely. This is due to the fact that the volume of long range dependent traffic is less when compared with the others. Hence the queueing behaviour of the aggregate traffic is dictated by the dominant short range dependent traffic.

Next, we obtain the queueing performance of an aggregate traffic consisting of twice as much long range dependent data as voice and video. The parameters for the MMPP models of voice and video are chosen as before; the PMPP model parameters for data traffic are as given below: $\lambda_5 = 60$ pkts/ms; $\lambda_6 = 150$ pkts/ms and $\alpha_5 = \alpha_6 = 1.4$. The service rate is increased to 200 pkts/ms to yield a ρ of 0.8. Again, in order to study the effect of long range dependence, an aggregate model using MMPP model for data traffic was used, for this traffic mix. The following parameters were used for the MMPP model used for data traffic, while the parameters for voice and video are chosen as before: $\lambda_5 = 60$ pkts/ms; $\lambda_6 = 150$ pkts/ms; $\alpha_5 = \alpha_6 = 0.2857$. This gives a combination of 2:1 for data *vs* voice and video.

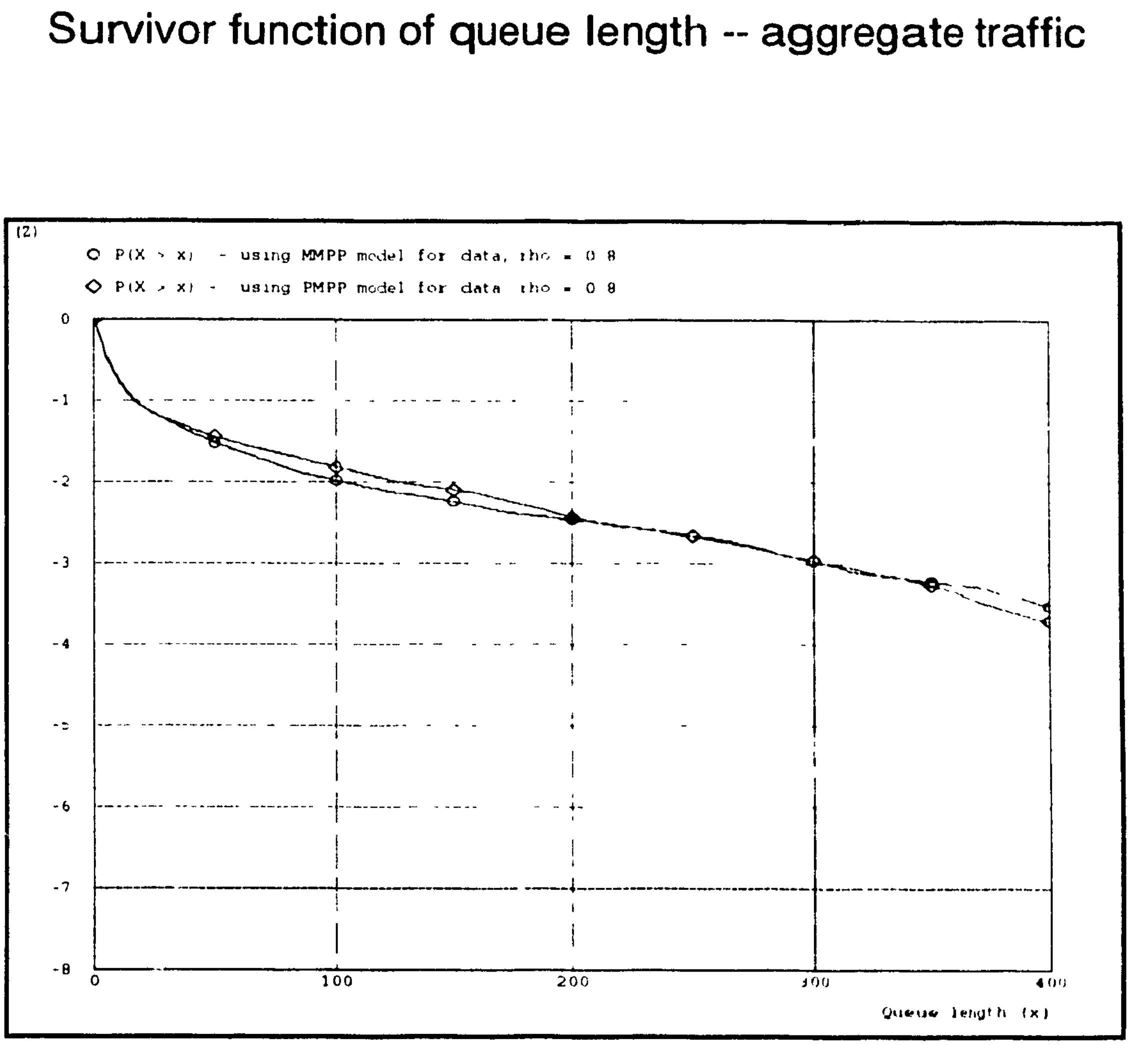


Figure 5.9: Survivor function of Queue length for aggregate traffic with 1:1:1 composition

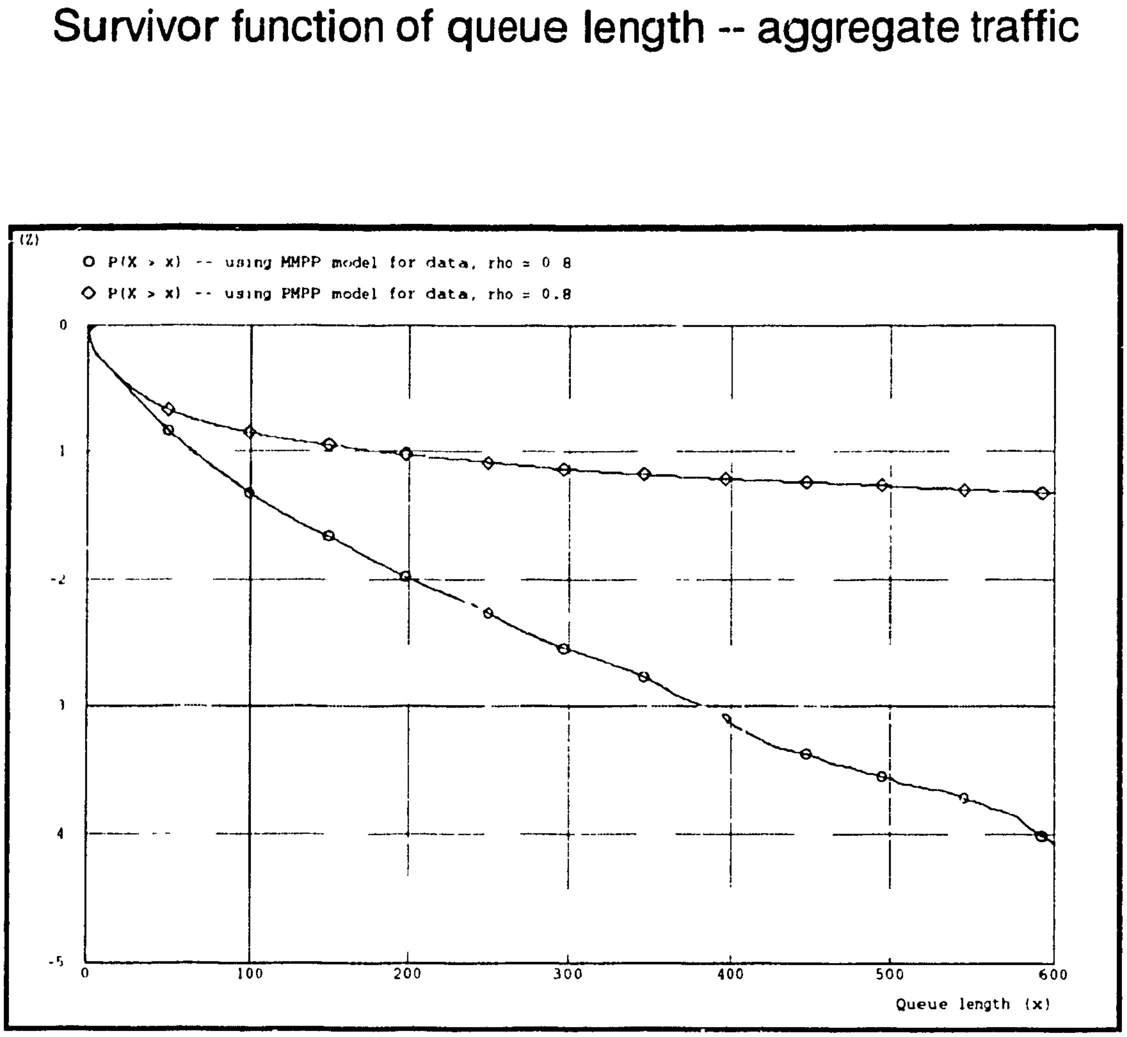


Figure 5.10: Survivor function of Queue length for aggregate traffic with 2:1 (data vs. voice + video) composition

Figure 5.10 shows the queue length distributions for the above traffic mix. It can be clearly observed that the model using PMPP model for data has a heavy tailed distribution than the other model for aggregate traffic, that uses MMPP for data traffic. Hence the traffic composition plays a role in the behaviour of net traffic.

Chapter 6

Conclusion

The wide spectrum of traffic sources in a multi-media network exhibit a diverse mixture of traffic characteristics. This thesis has addressed the salient issues in modeling the traffic in such an environment.

Following aspects have been addressed by this thesis

- The salient issues in the characterization of packet traffic have been emphasized.
- A new model for the long-range dependent traffic was proposed. This model was simulated on OPNET and studied.
- A traffic generator for generating synthetic traffic for multi-media networks was built.
- The queueing performance of aggregate multi-media traffic was studied by using the developed traffic generator.

The thesis outlined the salient issues in the modeling of packet traffic. It also provided a study of traffic models proposed in the literature and classified them.

The new long-range dependent model PMPP proposed in this research is very simple and yet versatile in characterizing the long-range dependence and self-similar characteristics. This model is ideally suited to synthetic traffic generation due to its simplicity. The traffic generator developed in this research is offers a range of models to characterize the varied correlations and dependence in multi-media packet traffic. This generator can be used to generate traffic with any given statistical characteristics, by fine tuning the input parameters of the generator.

The queueing performance of the PMPP and the aggregate traffic model were obtained using the developed traffic generator. The results of the performance study indicates that the PMPP has a queueing behaviour similar to that of the long-range dependent models proposed in the literature, i.e., the survivor function of queue length has a "stretched exponential" behaviour and is very different from the queueing performance of MMPP. The performance study of the aggregate traffic model indicates that the queueing behaviour is affected by the ratio of the long-range dependent traffic in the aggregate traffic mix. Thus implying that the composition of aggregate traffic is also important in engineering the buffers at the statistical multiplexers of multi-media traffic.

The findings of this project opens up new avenues of research. As a topic for future research, a queueing analysis of the aggregate model may be attempted at. This in itself is a complex task, given the non-Markovian nature of the processes involved.

Bibliography

[1] E. D. Sykas, K. M. Vlakos, and N. G. Anerousis, "Performance evaluation of statistical multiplexing scheme in ATM networks," *Computer Networks*, vol. 14, pp. 273–286, June 1991.

[2] I. Norros, "A storage model with self-similar input," *Queueing Systems*, no. 16, pp. 387–396, 1994.

[3] A. Erramilli, R. P. Singh, and P. Pruthi, "Chaotic maps as models of packet traffic," in *The fundamental role of teletraffic in the evolution of Telecommunications networks (Proc. 14th ITC)*, (Antibes Juan-les-Pins), pp. 329–338, 1994.

[4] S. N. Subramanian and T. Le-Ngoc, "Traffic modeling in a multi-media environment," in *Proc. Canadian Conference on Electrical and Computer Engineering*, (Montreal, Canada), pp. 838-841, Sept. 1995.

[5] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data," *IEEE J. Select. Areas Commun.*, pp. 833–846, Sept. 1986.

[6] K. Sriram and W. Whitt, "Characterizing superposition arrival processes and the performance of multiplexers for voice and data," in *Proc. IEEE Globecom '85*, Dec. 1985.

[7] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Trans. Commun.*, vol. 36, pp. 834–843, July 1988.

[8] W. E. Leland and D. V. Wilson, "High time-resolution measurement and analysis of LAN traffic: implications for LAN interconnection," in *Proc. IEEE Infocom '91*, (Bal Harbour, FL), pp. 1360–1366, 1991.

[9] H. J. Fowler and W. E. Leland, "Local area network traffic characteristics, with implications for broadband network congestion management," *IEEE J. Select. Areas Commun.*, pp. 1139–1149, September 1991.

[10] K. S. Meier-Hellstern, P. E. Wirth, Y. L. Yan, and D. A. Hoeflin, "Traffic models for ISDN data users: office automation application," in *Teletraffic and data traffic in a period of change (Proc. 13th ITC)*, (Copenhagen, Denmark), pp. 167–172, A.Jensen, V.B.Iversen (Eds.), North Holland, 1991.

[11] R. Gusella, "Characterizing the variability of arrival processes with index of dispersion," *IEEE J. Select. Areas Commun.*, pp. 203–211, Feb. 1991.

[12] D. R. Cox and V. Isham, *Point processes.* New York: Chapman and Hall, 1980.

[13] D. R. Cox, "Long-range dependence: A review," in *Statistics an appraisal*, pp. 55–74, Iowa State Statistical Laboratory, 1984.

[14] W. E. Leland, S. M. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, pp. 1–14, February 1994.

[15] H. E. Hurst, "Long term storage capacity of reservoirs," *iTrans. Amer. Soc. Civil Engineers*, pp. 770–799, 1951.

[16] P. T. Brady, "A statistical analysis of on-off patterns in 16 conversations," *Bell Syst. Tech. J.*, pp. 73–91, Jan 1968.

[17] S. J. Campanella, "Digital speech interpolation," *Comsat Technical Review*, vol. 6, pp. 127–158, Spring 1976.

[18] P. T. Brady, "A model for generating on-off speech patterns in two way conversation," *Bell Syst. Tech. J.*, pp. 2445–2472, Sep 1969.

[19] I. Ide, "Superposition of Interrupted Poisson Processes and its application to packetized voice multiplexers," in *Proc. International Teletraffic Congress - 12*, pp. 1399–1405, 1989.

[20] Y. C. Jenq, "Approximations for packetized voice traffic in statistical multiplexer," in *Proc. IEEE Infocom '84*, pp. 256–259, 1984.

[21] H. Heffes and D. M. Lucantoni, "A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance," *IEEE J. Select. Areas Commun.*, vol. SAC-4, no. 6, pp. 856–868, 1986.

[22] R. Nagarajan, F. Kurose, and D. Towsley, "Approximation techniques for computing packet loss in infinite buffered voice multiplexers," *IEEE J. Select. Areas Commun.*, pp. 368–377, April 1991.

[23] T. E. Stern, "A queueing analysis of packet voice," in *Proc. IEEE Globecom '83*, pp. 71–76, 1983.

[24] J. N. Daigle and J. D. Langford, "Models for analysis of packet voice communication systems," *IEEE J. Select. Areas Commun.*, pp. 847–855, Feb. 1986.

[25] J. N. Daigle and J. D. Langford, "Queueing analysis of a packet voice communicating system," in *Proc. IEEE Infocom '85*, pp. 18–26, 1985.

[26] R. C. F. Tucker, "Accurate method for analysis of a packet speech multiplexer with limited delay," *IEEE Trans. Communications*, pp. 479–483, April 1988.

[27] C. Yuan and J. A. Sylvester, "Queueing analysis of delay constrained voice traffic in packet switching system," *IEEE J. Select. Areas Commun.*, pp. 729-739, 1989.

[28] S. Q. Li, "Study of packet loss in a packet switched voice system," in *Proc. ICC '88*, pp. 1519–1526, 1988.

[29] D. P. Gaver, "Random hazard in reliability problems," *Technometrics*, vol. 5, pp. 211–226, May 1963.

[30] A.J.Lawrence, "Some models for stationary series of univariate events," *Stochastic Point processes : Statistical Analysis Theory and Applications*, pp. 199–256, 1972.

[31] M. F. Neuts, *Matrix-geometric solutions in stochastic models: An algorithmic approach.* The John Hopkins University press, 1981.

[32] W. Whitt, "Approximating a point process by a renewal process: Two basic methods," *Operations Research*, Jan-Feb 1982.

[33] W. Whitt, "The queueing network analyzer," *Bell Syst. Tech. J.*, pp. 2779-2813, Nov 1983.

[34] S. L. Albin, "Approximating a point process by a renewal process, superposition arrival process to queues.," *Operations Res* pp. 1133–1162, Sept 1984.

[35] N. Ohta, *Packet video.* Artech House, Boston, MA, 1994.

[36] M. Nomura, T. Fujii, and N. Ohta, "Basic characteristics of variable rate video coding in ATM environment," *IEEE J. Select. Areas Commun.*, pp. 752–760, June 1989.

[37] A. LaCorte, S. Lombardo, S. Palazzo, and S. Zinna, "Modeling activity in VBR video sources," *Signal processing: Image Communication*, pp. 167–178, June 1991.

[38] T. Fujii, M. Nomura, and N. Ohta, "Characterization of variable rate interframe video coding for ATM-based networks," in *Proc. IEEE Globecom '88*, pp. 1063–1067, Dec 1988.

[39] W. Verbiest, L. Pinnoo, and B. Voeten, "The impact of the ATM concept on video coding," *IEEE J. Select. Areas Commun.*, pp. 1623–1632, Dec 1988.

[40] B. Maglaris, B. Anastassiou, G. K. P. Sen, and J. D. Robbins, "Performance analysis of statistical multiplexing for packet video sources," in *Proc. IEEE Globecom '87*, pp. 1890–1899, 1987.

[41] A. Baiocchi, N. B. Melazzi, M. Listanti, A. Roveri, and R. Winkler, "Loss performance analysis of an ATM multiplexer loaded with on-off sources," *IEEE J. Select. Areas Commun.*, vol. SAC-9, pp. 388–393, April 1991.

[42] S. B. Kim, M. Y. Lee, and M. J. Kim, "Σ-Matching technique for MMPP modeling of heterogeneous on-off sources," in *Proc. IEEE Globecom '94*, (San Fransisco, CA), pp. 1090–1094, 1994.

[43] J. W. Lee and B. G. Lee, "Performance analysis of ATM cell multiplexer with MMPP input," *IEICE Trans. Commun.*, vol. E75-B, pp. 709–714, August 1992.

[44] R. Grunenfelder, J. P. Cosmas, S. Manthorpe, and A. Odinma-Okafor, "Characterization of video codecs as autoregressive moving average processes and related queueing system performance," *IEEE J. Select. Areas Commun.*, pp. 284–293, April 1991.

[45] B. Melamed, D.Raychaudri, B. Sengupta, and J. Zdepski, "TES-based traffic modeling for performance evaluation of integrated networks," in *Proc. IEEE Infocom '92*, pp. 75–84, 1992.

[46] B. Melamed, D.Raychaudri, B. Sengupta, and J. Zdepski, "TES-based video source modeling for performance evaluation of integrated networks," *IEEE Trans. Commun.*, pp. 2773–2777, October 1994.

[47] B. Melamed, "The TES methodology: modeling temporal dependence in empirical time series," in *Proc. MASCOTS '93. International workshop on modeling, analysis and simulation of computer and telecommunication systems*, pp. 11–16, 17-20 Jan 1993.

[48] P. Bratley, B. L. Fox, and L. E. Schrage, *A Guide to Simulation.* Springer Verlag, New York, NY, 1987.

[49] D.Geist and B.Melamed, "TEStool: An environment for visual interactive modeling of autocorrelated traffic," in *Proc. ICC '92*, (Chicago, IL), pp. 1285–1289, 1992.

[50] P. Sen, M. Maglaris, N. Rikli, and D. Anastassiou, "Models for packet switching of variable bit rate video sources," *IEEE J. Select. Areas Commun.*, pp. 865–869, June 1989.

[51] F. Yegenoglu, B. Jabbari, and Y. Q. Zhang, "Modeling of motion classified VBR video sources," in *Proc. IEEE Infocom '92*, pp. 105–109, 1992.

[52] F. Yegenoglu, B. Jabbari, and Y. Q. Zhang, "Motion-classified autoregressive modeling of variable bit rate video," *IEEE Trans. on Circuits and Syst. for Video Technolgy*, pp. 42–53, Feb 1993.

[53] B. Jabbari, F. Yegenoglu, S. Z. Y. Kuo, and Y. Q. Zhang, "Statistical characterization and block-based modeling of motion-adaptive coded video," *IEEE Trans. on Circuits and Syst. for Video Technolgy*, pp. 199–207, June 1993.

[54] R. M. Rodriguez-Dagnino, M. R. K.Khansari, and A. Leon-Garcia, "Prediction of bit rate sequences of encoded video signals," *IEEE J. Select. Areas Commun.*, pp. 305–314, April 1991.

[55] D. L. McLaren and D. T. Nguyen, "Modeling low bit rate video traffic as switched-fractal source," *Electronics Letters*, pp. 745–747, April 1991.

[56] D. L. McLaren and D. T. Nguyen, "Variable bit-rate source modeling of ATM-based video services," *Signal processing: Image Communication*, pp. 233–244, June 1992.

[57] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," in *Proc. ACM SIGCOMM '94. Conference on communication architecture, protocols and applications*, pp. 269–281, Aug. 31 - Sep. 2 1994.

[58] D. P. Heyman and T. V. Lakshman, "What are the implications of long-range dependence for vbr - video traffic engineering," *IEEE/ACM Trans. on Networking*, vol. 4, pp. 301–317, June 1996.

[59] D. E. Duffy, A. A. McIntosh, M. Rosenstein, and W. Willinger, "Statistical analysis of ccsn/ss7 traffic data from working ccs7 subnetworks," *IEEE J. Select. Areas Commun.*, no. 12, pp. 544–551, 1994.

[60] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," in *Proc.ACM SIGCOMM '94*, pp. 257–268, 1994.

[61] R. Gusella, "A measurement study of diskless workstation traffic on an ethernet," *IEEE Trans. Communications*, vol. 38, no. 9, pp. 1557–1568, 1990.

[62] H. Saito, *Teletraffic Technologies in ATM Networks.* Artech House, 1993.

[63] S. Anick, D. Mitra, and M. M. Sondhi, "Stochastic theory of a data-handling system with multiple source," *Bell Syst. Tech. J.*, vol. 36, pp. 834-843, July 1988.

[64] B. Meister. "Waiting time in a preemptive resume system with compound-Poisson input," *Comput.*, no. 1, pp. 17-28, 1980.

[65] D. M. Lucantoni, "New results for the single server queue with a batch Markovian arrival process," *Stoch. Models*, pp. 1-46, 1991.

[66] R. Jain and S. A. Routhier, "Packet trains - meaurements and a new model for computer network traffic," *IEEE J. Select. Areas Commun.*, pp. 986-995, September 1986.

[67] Y. D. J. Lin, T. C. Tsai, S. C. Huang, and M. Gerla, "HAP: A new model for packet arrivals," in *Proc.ACM SIGCOMM '93*, pp. 212-223, September 1993.

[68] B. B. Mandelbrot, "Long-run linearity, locally Gaussian processes, H-spectra and infinite variances," *Technometrics*, vol. 10, pp. 82-113, 1969.

[69] W. E. Leland, S. M. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," in *Proc. ACM SIGCOMM '93*, (Sanfransisco, CA), pp. 183-193, September 1993.

[70] M. S. Taqqu and J. B. Levy, "Using renewal processes to generate long range dependence and high variabilty," in *Dependence in Probabilty and Statistics*, pp. 73-89, E.Eberlein and M.S.Taqqu (Eds.), Progress in Prob. and Stat. Vol. 11, Birkhauser, Boston, 1986.

[71] D. R. Cox, "Some statistical methods connected with series of events," *Journal of the Royal Statistical Society, Series B*, vol. 17, no. 2, 1955.

[72] D. R. Cox, *The Statistical Analysis of Series of Events.* Metheun, London, 1966.

[73] B. S. Slimane and T. Le-Ngoc, "A doubly stochastic Poisson model for self-similar traffic," in *Proc. ICC '95*, (Seattle, Washington), pp. 456–460, 1995.

[74] I. Norros, "On the use of Fractional Brownian motion in the theory of connectionless networks," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 953–962, August 1995.

[75] A. Erramilli, O. Narayan, and W. Willinger, "Experimental queueing analysis with long-range dependent packet traffic," *IEEE/ACM Trans. on Networking*, vol. 4, pp. 209–223, April 1996.

[76] P. Pruthi and A. Erramilli, "Heavy tailed on/off source behaviour and self-similar traffic," in *Proc. ICC '95*, (Seattle, Washington), pp. 445–450, 1995.

Appendix A

Program Listing

The traffic generator was developed on a simulation package called OPNET (OPtimized Network Engineering Tools). OPNET provides a easy to use graphical user interface. The simulation is built with independent building blocks called process models. The operation of these process models are specified by finite state machines, translated into C code. The process models for the folowing models of the traffic generator are given here.

(i) *trf_generator* - Traffic generator process model.

(ii) *trf_vc_on_off* - ON - OFF process model.

(iii) *trf_vd_on_off* - Mini-sources process model.

(iv) *trf_dt_on_off* - "Heavy tailed" ON - OFF process model.

(v) *trf_vc_mmpp* - MMPP process model.

(vi) *trf_dt_pmpp* - PMPP process model.

(vii) *fifo_pk_q_pdf* - FIFO queue process mod l.

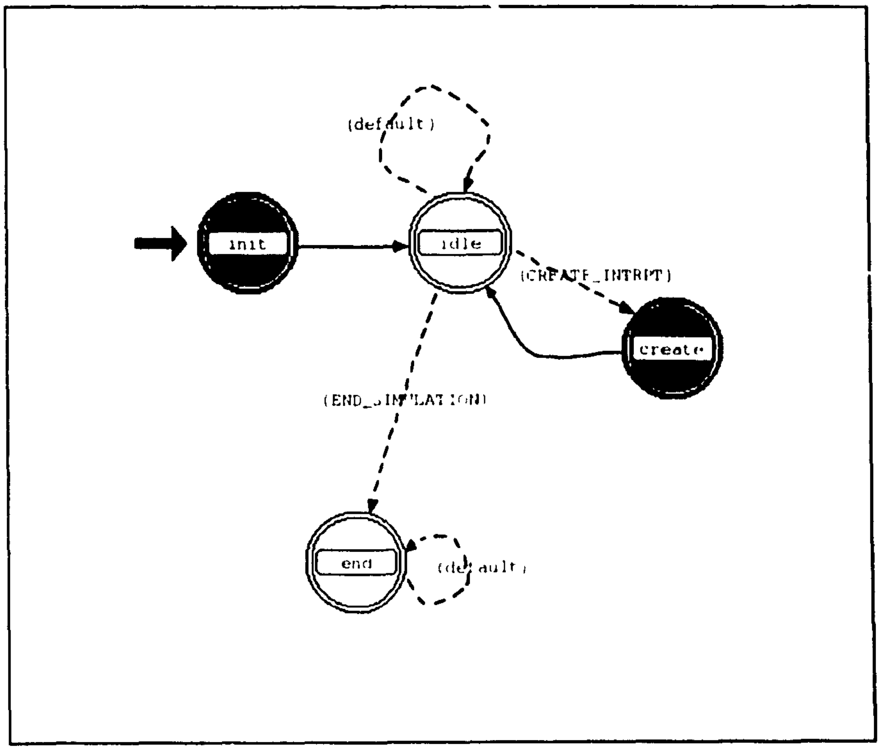


Figure A.1: Traffic generator process model

Process Model Report: **trf_generator**	Sat Aug 24 15:57:44 1996	Page 1 of 3
Traffic Generator ...		

Process Model Attributes			
attribute	*value*	*type*	*default value*
vc_model	promoted	string	trf_vc_on_off
dt_model	promoted	string	trf_dt_on_off
vd_model	promoted	string	trf_vd_on_off
N_voice	promoted	integer	1 (src)
N_data	promoted	integer	1 (src)
N_video	promoted	integer	1 (src)

Header Block

```
/* This process model creates child processes of the models selected for voice, video
  and data traffic The number of child processes created for each traffic type is
  determined by the input numbers "N_voice", "N_video", and "N_data"  */

#define         CREATE_INTRPT_CODE        1
#define         CREATE_INTRPT             op_intrpt_type() == OPC_INTRPT_SELF && \
                                          op_intrpt_code() == CREATE_INTRPT_CODE
#define         END_SIMULATION            op_intrpt_type() == OPC_INTRPT_ENDSIM
```

State Variable Block

```
char          \vc_model[40];
char          \dt_model[40];
char          \vd_model[40];
int           \N_video;
int           \N_data;
int           \N_voice;
int           \prc_id;
int           \i_gen;
```

Temporary Variable Block

forced state **Init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **Init**

```
prc_id = op_id_self();

/* get parameters for  number of voice,video and data  sources
  and also the models to be used  for voice, video and data. */
op_ima_obj_attr_get(prc_id,"vc_model ",vc_model);
op_ima_obj_attr_get(prc_id,"vd_model ",vd_model);
op_ima_obj_attr_get(prc_id,"dt_model",dt_model);
op_ima_obj_attr_get(prc_id,"N_video",&N_video);
op_ima_obj_attr_get(prc_id,"N_voice",&N_voice);
op_ima_obj_attr_get(prc_id,"N_data", &N_data );

/* schedule to interrupt to create the above child processes */
op_intrpt_schedule_self(op_sim_time(),CREATE_INTRPT_CODE);
```

unforced state **idle**

attribute	value	type	default value
name	idle	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **idle**

forced state **create**

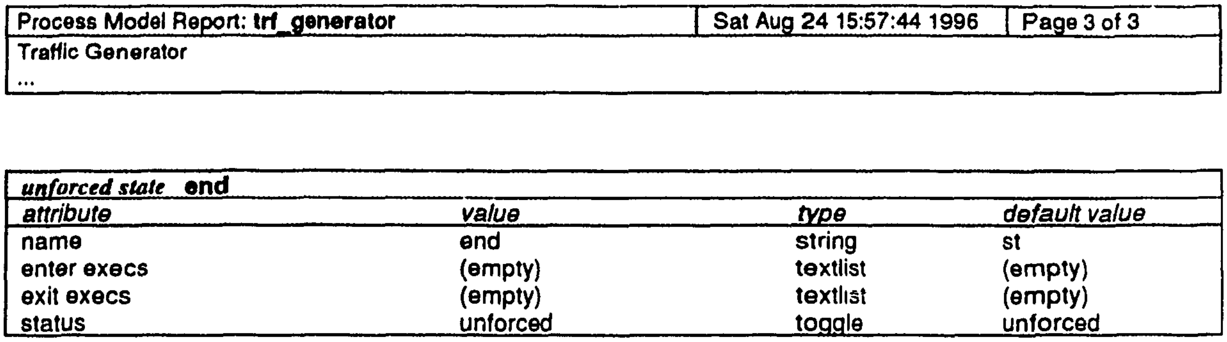
attribute	value	type	default value
name	create	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **create**

```
/* generate N video sources */
for( i_gen=0; i_gen<N_video; i_gen++)
{
  op_pro_invoke(op_pro_create(vd_model,OPC_NIL),OPC_NIL);
};

/* generate N data sources */
for( i_gen=0; i_gen<N_data; i_gen++)
{
  op_pro_invoke(op_pro_create(dt_model,OPC_NIL),OPC_NIL);
};

/* generate N voices sources */
for( i_gen=0; i_gen<N_voice; i_gen++)
{
  op_pro_invoke(op_pro_create(vc_model,OPC_NIL),OPC_NIL);
};
```



Process Model Report: **trf_generator**	Sat Aug 24 15:57:44 1996	Page 3 of 3
Traffic Generator ...		

unforced state **end**			
attribute	*value*	*type*	*default value*
name	end	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

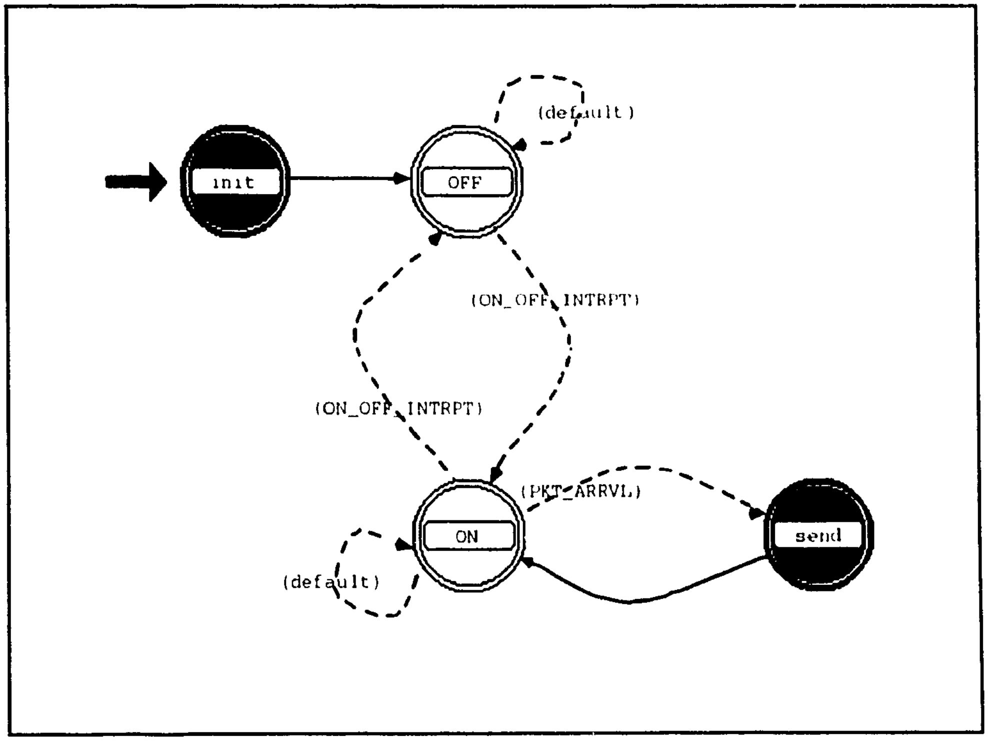


Figure A.2: ON-OFF process model

Process Model Report: **trf_vc_on_off**	Sat Aug 24 16:09:05 1996	Page 1 of 3
ON-OFF model for voice ...		

Process Model Attributes			
attribute	*value*	*type*	*default value*
alpha	promoted	double	2.84 (1/sec)
beta	promoted	double	1.54 (1/sec)
lamda	promoted	double	62 5 (pkts/sec)

Header Block

```
/* This process model generates the traffic from one from one voice source
   in accordance with the on-off process model */

#include        <math.h>
#include        <stdio.h>
#include        <sys/time.h>

#define         PKT_ARRVL_CODE       0
#define         ON_OFF_CODE          1

#define         PKT_ARRVL            op_intrpt_type() == OPC_INTRPT_SELF &&\
                                     op_intrpt_code() == PKT_ARRVL_CODE

#define         ON_OFF_INTRPT        op_intrpt_type() == OPC_INTRPT_SELF &&\
                                     op_intrpt_code() == ON_OFF_CODE
```

State Variable Block

```
Distribution*       \alpha_dist;
Distribution*       \beta_dist;
int                 \id;
int                 \ON;
double              \alpha;
double              \beta;
double              \lambda;
```

Temporary Variable Block

```
Packet*         pkptr;
```

forced state **init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs init

```
id = op_id_self();
```

```
/* Get parameter */
op_ima_obj_attr_get(id,"alpha",&alpha);
op_ima_obj_attr_get(id,"beta",&beta);
op_ima_obj_attr_get(id,"lamda",&lambda);

/* Initialize variable */
ON = 0;

/* Get exponential distribution */
alpha_dist = op_dist_load("exponential",1/alpha,0.0);
beta_dist  = op_dist_load("exponential",1/beta,0.0);
```

unforced state **OFF**

attribute	*value*	*type*	*default value*
name	OFF	string	st
enter execs	(See below)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **OFF**

```
/* schedule for ON state */
if (ON==0)
{
 op_intrpt_schedule_self(op_sim_time()+op_dist_outcome(beta_dist),ON_OFF_CODE);
 ON = 1;
};
```

unforced state **ON**

attribute	*value*	*type*	*default value*
name	ON	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(See below.)	textlist	(See below)
status	unforced	toggle	unforced

enter execs **ON**

```
/* schedule for OFF state and send state */
if (ON == 1)
{
 op_intrpt_schedule_self(op_sim_time()+op_dist_outcome(alpha_dist),ON_OFF_CODE);
 op_intrpt_schedule_self(op_sim_time()+1/lambda,PKT_ARRVL_CODE);
 ON = 0;
};
```

exit execs **ON**

forced state **send**

attribute	*value*	*type*	*default value*
name	send	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **send**

```

/* Send unformat pac ket */
pkptr = op_pk_create(0).
op_pk_send(pkptr,0);

/* Schedule for next packet */
op_intrpt_schedule_self(op_sim_time()+1/lambda,PKT_ARRVL_CODE);
```

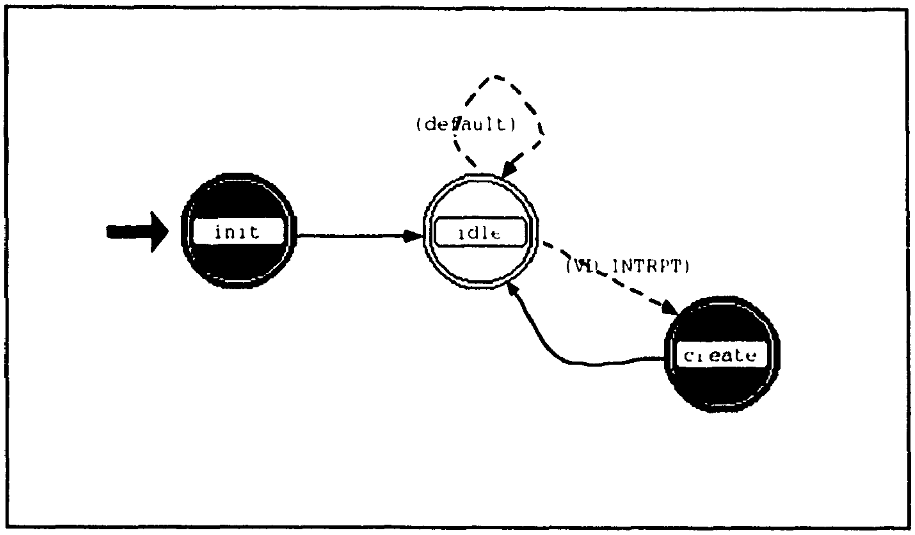


Figure A.3: Mini-sources process model

Mini-source model for video
...

Process Model Attributes			
attribute	*value*	*type*	*default value*
n_mini_src	promoted	integer	20 (src)

Header Block

```
/* This process takes "n_mini_sources" as its input and invokes as many
   child "on_off" processes to generate the traffic from one video source.
   The "on_off" source is similar to the one used for voice  */


#define        VD_INTRPT_CODE    10
#define        VD_INTRPT         op_intrpt_type() == OPC_INTRPT_SELF && \
                                 op_intrpt_code() == VD_INTRPT_CODE
```

State Variable Block

```
int        \n_mini_src;
int        \prc_vd_id;
int        \i_vd;
```

Temporary Variable Block

forced state **init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs init

```
/* Get parameters */
prc_vd_id = op_id_self();
op_ima_obj_attr_get(prc_vd_id,"n_mini_src",&n_mini_src);
op_intrpt_schedule_self(op_sim_time(),VD_INTRPT_CODE);
```

unforced state **idle**			
attribute	*value*	*type*	*default value*
name	idle	string	st

enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **idle**

forced state **create**

attribute	*value*	*type*	*default value*
name	create	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **create**

```

for( i_vd=0; i_vd<n_mini_src; i_vd++)
{
  op_pro_invoke(op_pro_create("trf_on_off",OPC_NIL),OPC_NIL);
};
```

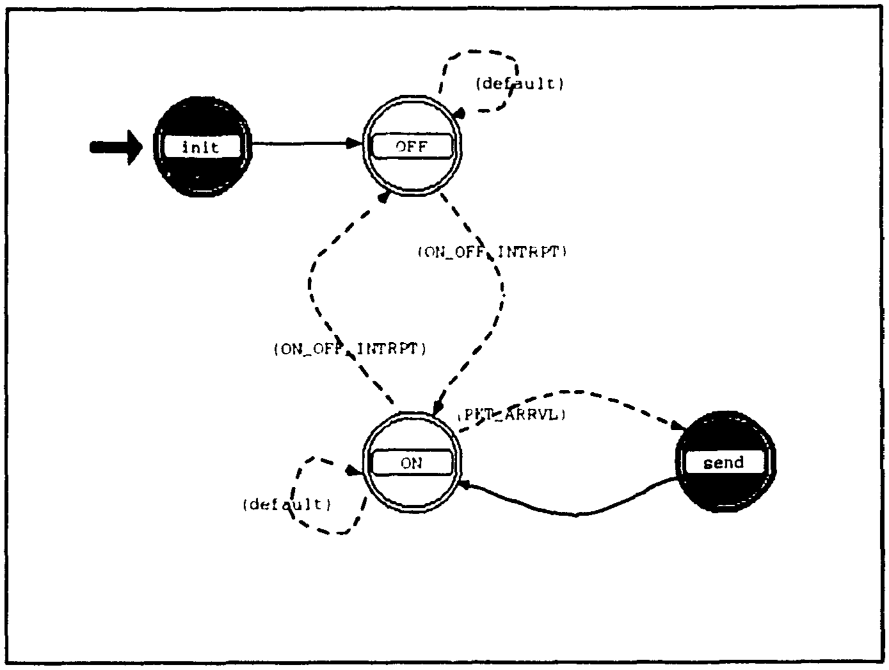


Figure A.4: "Heavy tailed" ON-OFF process model

Process Model Report: **trf_dt_on_off**	Sat Aug 24 16:17:27 1996	Page 1 of 3
ON-OFF model for data ...		

Process Model Attributes

attribute	*value*	*type*	*default value*
alpha	promoted	double	1.4 (1/sec)
beta	promoted	double	0.2 (1/sec)
lamda	promoted	double	170 (pkts/sec)

Header Block

```
/* This process model generates the traffic from a single data sources by using
  the on-off model (with heavy-tailed sojourn time) abstraction */

#include        <math.h>
#include        <stdio.h>
#include        <sys/time h>

#define         PKT_ARRVL_CODE       0
#define         ON_OFF_CODE          1

#define         PKT_ARRVL            op_intrpt_type() == OPC_INTRPT_SELF && \
                                     op_intrpt_code() == PKT_ARRVL_CODE
#define         ON_OFF_INTRPT        op_intrpt_type() == OPC_INTRPT_SELF && \
                                     op_intrpt_code() == ON_OFF_CODE

double          pareto();
```

State Variable Block

```
Distribution*       \alpha_dist;
Distribution*       \beta_dist;
int                 \id;
int                 \ON;
double              \alpha.
double              \beta;
double              \lambda;
```

Temporary Variable Block

```
Packet*         pkptr;
```

Function Block

```
/* Function to distribute a Pareto distributed random variate */
double          pareto(dist_ptr)
Distribution* dist_ptr,
{
     double y;

     y = op_dist_outcome(dist_ptr);
     return(exp(y));
/* If y has an exponential distribution,
      with parameter alpha, then exponential(y)
      is pareto generated with alpha. */

}
```

forced state **Init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **Init**

```

id = op_id_self(),

/* Get parameter */
op_ima_obj_attr_get(i l,"alpha",&alpha);
op_ima_obj_attr_get(id,"beta",&beta);
op_ima_obj_attr_get(id,"lamda",&lambda);

/* Initialize variable */
ON = 0;

/* Get exponential distribution */
alpha_dist = op_dist_load("exponential",1/alpha,0.0);
beta_dist  = op_dist_load("exponential",1/beta,0 0);
```

unforced state **OFF**			
attribute	*value*	*type*	*default value*
name	OFF	string	st
enter execs	(See below)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **OFF**

```
/* schedule for ON state */
if (ON==0)
{
 op_intrpt_schedule_self(op_sim_time()+op_dist_outcome(beta_dist),ON_OFF_CODE);
 ON = 1;
};
```

Process Model Report: **trf_dt_on_off**	Sat Aug 24 16:17:27 1996	Page 3 of 3
ON-OFF model for data ...		

unforced state **ON**			
attribute	*value*	*type*	*default value*
name	ON	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(See below.)	textlist	(See below)
status	unforced	toggle	unforced

enter execs **ON**

```
/* schedule for OFF state and new packet send */
if (ON == 1)
{
 op_intrpt_schedule_self(op_sim_time()+pareto(alpha_dist),ON_OFF_CODE);
 op_intrpt_schedule_self(op_sim_time()+1/lambda,PKT_ARRVL_CODE);
 ON = 0;
};
```

exit execs **ON**

forced state **send**			
attribute	*value*	*type*	*default value*
name	send	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **senJ**

```

/* send packet */
pkptr = op_pk_create(0);
op_pk_send(pkptr,0);

/* schedule for new packet */
op_intrpt_schedule_self(op_sim_time()+1/lambda,PKT_ARRVL_CODE);
```

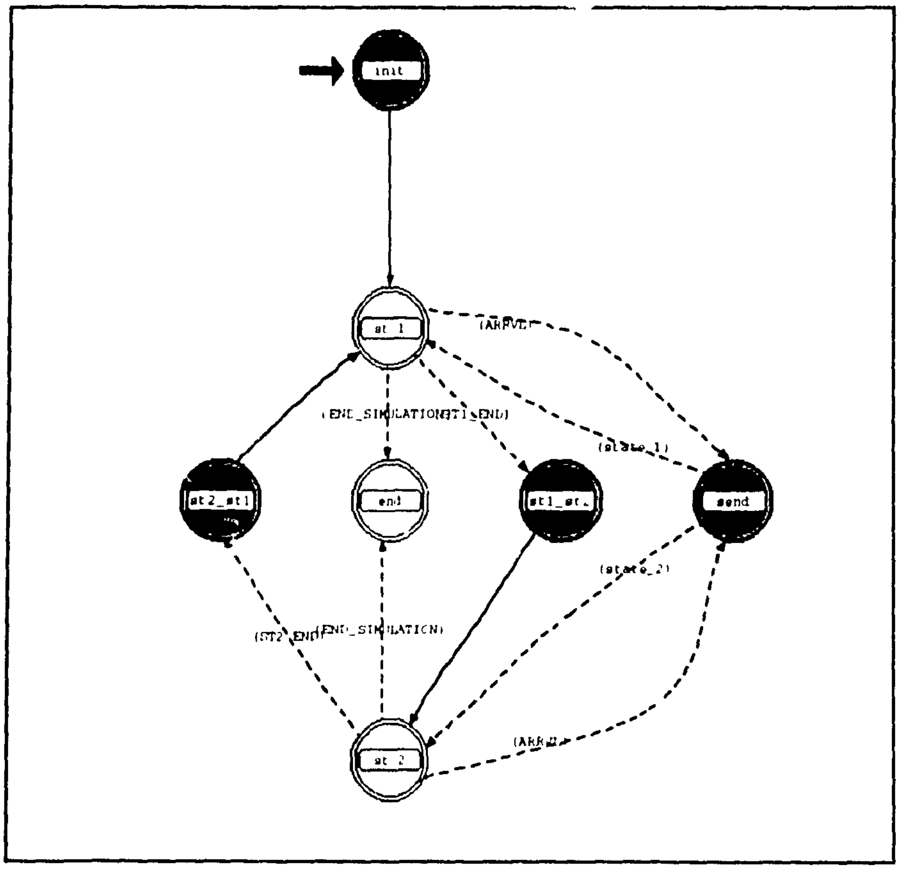


Figure A.5: MMPP process model

Process Model Report: **trf_vc_mmpp**	Sat Aug 24 16:22:49 1996	Page 1 of 4
MMPP model ...		

Process Model Attributes

attribute	*value*	*type*	*default value*
lambda_1	promoted	double	1.0 (pkts/ms)
lambda_2	promoted	double	1 0 (pkts/ms)
alpha_1	promoted	double	1 0 (1/ms)
alpha_2	promoted	double	1 0 (1/ms)

Header Block

```
/* This process model generates packet according to the MMPP model, for voice */

#include        <math.h>
#include        <stdio.h>
#include        <sys/time.h>

#define         ST1_CODE         1
#define         ST2_CODE         2
#define         ARRV_CODE        3


#define         ST1_END          op_intrpt_type() == OPC_INTRPT_SELF && \
                                 op_intrpt_code() == ST1_CODE

#define         ST2_END          op_intrpt_type() == OPC_INTRPT_SELF && \
                                 op_intrpt_code() == ST2_CODE

#define         ARRVL            op_intrpt_type() == OPC_INTRPT_SELF && \
                                 op_intrpt_code() == ARRV_CODE


#define         END_SIMULATION   op_intrpt_type()==OPC_INTRPT_ENDSIM
```

State Variable Block

```
Distribution*        \alpha1_ptr,
Distribution*        \alpha2_ptr;
Distribution*        \lambda1_ptr,
Distribution*        \lambda2_ptr,
int                  \state_1,\state_2;
double               \st_time;
int                  \id;
```

Temporary Variable Block

```
Packet*         pkptr;
double          pkt_arrv_time,
double          alpha_1;
double          alpha_2;
double          lambda_1;
double          lambda_2;
```

Function Block

forced state **init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **init**

```
id=op_id_self();

/* Obtain parameter values */
op_ima_obj_attr_get(id,"lambda_1", &lambda_1);
op_ima_obj_attr_get(id,"lambda_2", &lambda_2);
op_ima_obj_attr_get(id,"alpha_1", &alpha_1);
op_ima_obj_attr_get(id,"alpha_2", &alpha_2);

/* Load distribution */
alpha1_ptr=op_dist_load("exponential",1/alpha_1,0.0);
alpha2_ptr=op_dist_load("exponential",1/alpha_2,0.0);
lambda1_ptr=op_dist_load("exponential",1/lambda_1,0.0).
lambda2_ptr=op_dist_load("exponential",1/lambda_2,0 0);

/* Initialize variables */
state_1=1;
state_2=0;

/* Schedule transition interrupt from state 1 */
st_time = op_sim_time() + op_dist_outcome(alpha1_ptr);
op_intrpt_schedule_self(st_time,ST1_CODE);

/* Schedule a packet if the arrival time does
  not exceed state 1 sojourn time */
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE).
```

unforced state **st_1**			
attribute	*value*	*type*	*default value*
name	st_1	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **st_1**

unforced state **st_2**

attribute	*value*	*type*	*default value*
name	st_2	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

forced state **st1_st2**

attribute	*value*	*type*	*default value*
name	st1_st2	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **st1_st2**

```
state_2=1;
state_1=0;

/* Schedule transition interrupt from state 2 */
st_time = op_sim_time() + op_dist_outcome(alpha2_ptr).
op_intrpt_schedule_self(st_time,ST2_CODE);

/* Schedule a packet if the arrival time does
  not exceed state 1 sojourn time */
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda2_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

forced state **st2_st1**

attribute	*value*	*type*	*default value*
name	st2_st1	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **st2_st1**

```
state_1=1;
state_2=0.

/* Schedule transition interrupt from state 1 */
```

```
st_time = op_sim_time() + op_dist_outcome(alpha1_ptr);
op_intrpt_schedule_self(st_time,ST1_CODE);

/* Schedule a packet if the arrival time does
   not exceed state 1 sojourn time*/
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

forced state **send**

attribute	*value*	*type*	*default value*
name	send	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **send**

```
/* send packet */
pkptr = op_pk_create(0);
op_pk_send(pkptr,0);

if (state_1)
      pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
else
      pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda2_ptr);

/* Schedule a packet if the arrival time does
   not exceed state 1 sojourn time */
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

unforced state **end**

attribute	*value*	*type*	*default value*
name	end	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

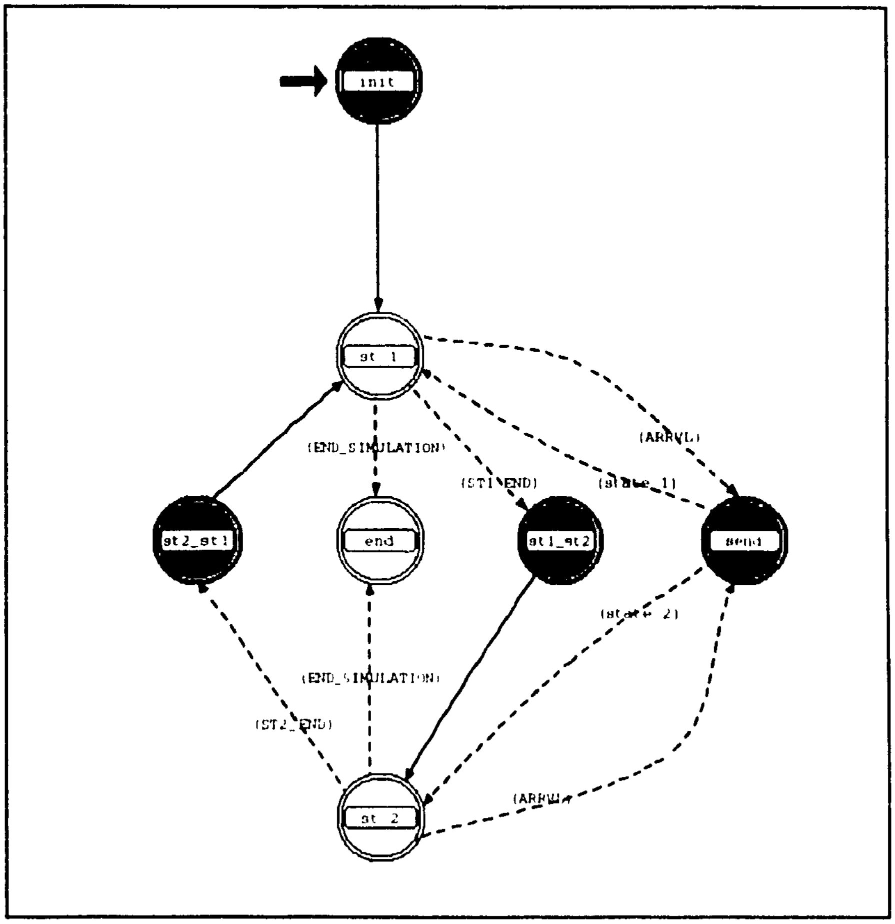


Figure A.6: PMPP process model

PMPP model

...

Process Model Attributes

attribute	value	type	default value
lambda_1	promoted	double	1.0 (pkts/ms)
lambda_2	promoted	double	1.0 (pkts/ms)
alpha_1	promoted	double	1.0 (1/ms)
alpha_2	promoted	double	1.0 (1/ms)

Header Block

```
/* This process model generates packets according to the PMPP model, for data */

#include        <math.h>
#include        <stdio.h>
#include        <sys/time.h>

#define         ST1_CODE        1
#define         ST2_CODE        2
#define         ARRV_CODE       3


#define         ST1_END         op_intrpt_type() == OPC_INTRPT_SELF && \
                                op_intrpt_code() == ST1_CODE

#define         ST2_END         op_intrpt_type() == OPC_INTRPT_SELF && \
                                op_intrpt_code() == ST2_CODE

#define         ARRVL           op_intrpt_type() == OPC_INTRPT_SELF && \
                                op_intrpt_code() == ARRV_CODE


#define         END_SIMULATION op_intrpt_type()==OPC_INTRPT_ENDSIM

double          pareto();
```

State Variable Block

```
Distribution*           \alpha1_ptr;
Distribution*           \alpha2_ptr;
Distribution*           \lambda1_ptr;
Distribution*           \lambda2_ptr;
int                     \state_1,\state_2;
double                  \st_time;
int                     \id;
```

Temporary Variable Block

```
Packet*         pkptr;
double          lambda_1, lambda_2;
double          alpha_1, alpha_2;
double          pkt_arrv_time;
```

Function Block

```
/*Function to generate a pareto distributed random variate*/

double pareto(dist_ptr)
Distribution* dist_ptr;
{
      double y;
      y = op_dist_outcome(dist_ptr);

      return(exp(y));
      /* If y is exponentially distributed with
         alpha then exponential(y) is pareto
       distributed with alpha*/
}
```

forced state init

attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs init

```
id=op_id_self();

/* Obtain parameter values */
op_ima_obj_attr_get(id,"lambda_1", &lambda_1);
op_ima_obj_attr_get(id,"lambda_2", &lambda_2);
op_ima_obj_attr_get(id,"alpha_1", &alpha_1),
op_ima_obj_attr_get(id,"alpha_2", &alpha_2);

/* Load distribution. */
alpha1_ptr=op_dist_load("exponential",1/alpha_1,0.0);
alpha2_ptr=op_dist_load("exponential",1/alpha_2,0.0);
lambda1_ptr=op_dist_load("exponential",1/lambda_1,0.0);
lambda2_ptr=op_dist_load("exponential",1/lambda_2,0.0);

state_1=1;
state_2=0;

 * Schedule transition interrupt from state 1 */
st_time = op_sim_time() + pareto(alpha1_ptr);
op_intrpt_schedule_self(st_time,ST1_CODE);

/* Schedule a packet if the arrival time does
   not exceed state 1 sojourn time*/
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

unforced state **st 1**			
attribute	*value*	*type*	*default value*
name	st_1	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs **st 1**

unforced state **st 2**			
attribute	*value*	*type*	*default value*
name	st_2	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

forced state **st1 st2**			
attribute	*value*	*type*	*default value*
name	st1_st2	string	st
enter execs	(See below)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **st1 st2**

```

state_2=1;
state_1=0;

/* Schedule transition interrupt from state 2 */
st_time = op_sim_time() + pareto(alpha2_ptr);
op_intrpt_schedule_self(st_time,ST2_CODE);

/* Schedule a packet if the arrival time does
  not exceed state 1 sojourn time. */
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda2_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

forced state **st2_st1**			
attribute	*value*	*type*	*default value*
name	st2_st1	string	st
enter execs	(See below)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **st2_st1**

```
state_1=1;
state_2=0;

/* Schedule transition interrupt from state 1 */
st_time = op_sim_time() + pareto(alpha1_ptr);
op_intrpt_schedule_self(st_time,ST1_CODE);

/* Schedule a packet if the arrival time does
  not exceed state 1 sojourn time */
pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
if (pkt_arrv_time < st_time)
   op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

forced state **send**			
attribute	*value*	*type*	*default value*
name	send	string	st
enter execs	(See below)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **send**

```
/* send packet */
pkptr = op_pk_create(0);
op_pk_send(pkptr,0);

if (state_1)
     pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda1_ptr);
else
     pkt_arrv_time = op_sim_time() + op_dist_outcome(lambda2_ptr);


/* Schedule a packet if the arrival time does
  not exceed state 1 sojourn time  */
if (pkt_arrv_time < st_time)
  op_intrpt_schedule_self(pkt_arrv_time, ARRV_CODE);
```

unforced state **end**

attribute	*value*	*type*	*default value*
name	end	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

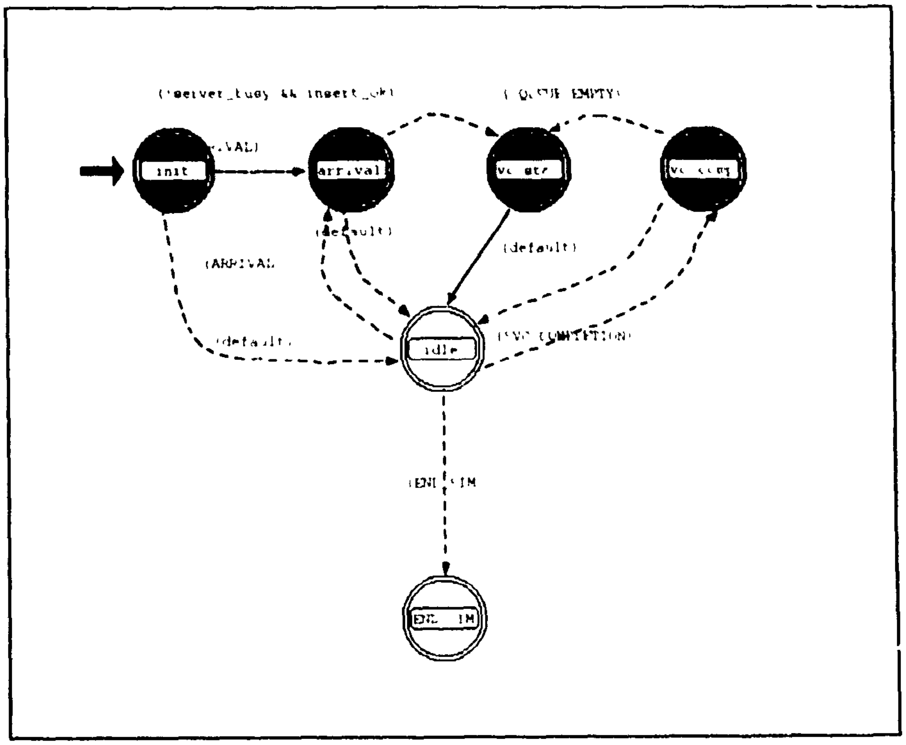


Figure A.7: FIFO queue process model

Process Model Report: fifo_pk_q_pdf	Sat Aug 24 16:34:59 1996	Page 1 of 5
Deterministic queue ...		

Process Model Attributes			
attribute	*value*	*type*	*default value*
service rate	promoted	double	1.0 (pkts/sec)

Header Block

```
#define        QUEUE_EMPTY        (op_q_empty ())
#define        SVC_COMPLETION     op_intrpt_type () == OPC_INTRPT_SELF
#define        ARRIVAL            op_intrpt_type () == OPC_INTRPT_STRM
#define        END_SIM            op_intrpt_type() == OPC_INTRPT_ENDSIM

double*        ary_ptr;
extern FILE *  fptr;
char           opt_file[40];
```

State Variable Block

```
int         \server_busy;
double      \service_rate;
Objid       \own_id;
int         \ary_end_index;
```

Temporary Variable Block

```
Packet*      pkptr;
int        insert_ok;
int              i;
double           total;
double           q_length;
double       pk_svc_time;
```

forced state **init**			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **init**

```
/* initially the server is idle */
server_busy = 0;

/* get queue module's own object id */
own_id = op_id_self ();

/* get assigned value of server processing rate */
op_ima_obj_attr_get (own_id, "service_rate", &service_rate);
```

```

/*Allocate memory for the array that stores the frequency
of queue lengths. Start with a single element, this will be
expanded dynamically */
ary_ptr = (double *)malloc(sizeof(double));
ary_ptr[0] = 0;

/* Indicates the max  array index of the dynamically
allocated array, currently 0 */
ary_end_index = 0;

```

forced state **arrival**			
attribute	*value*	*type*	*default value*
name	arrival	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **arrival**

```

/* acquire the arriving packet */
/* multiple arriving streams are supported. */
pkptr = op_pk_get (op_intrpt_strm ());


/* attempt to enqueue the packet at tail of subqueue 0 */
if (op_subq_pk_insert (0, pkptr, OPC_QPOS_TAIL) != OPC_QINS_OK)
   {
   /* the insertion failed (due to a full queue) */
   /* deallocate the packet */
   op_pk_destroy (pkptr);

      /* set flag indicating insertion fail */
      /* this flag is used to determine transition */
      /* out of this state */
   insert_ok = 0;
   }
else{
      /* insertion was successful */
   insert_ok = 1;
   }

/* Sample the queue length after this arrival  */
q_length = op_q_stat(OPC_QSTAT_PKSIZE);

if(ary_end_index >= q_length)
      /* If current array index is greater than or equal
         to the queue length, increment the corresponding
         element in the array  */
      {
         ary_ptr[(int)q_length]++;
      }
```

```
    /* Else reallocate memory and dynamically increase
       array size */
else{
       ary_ptr =(double *) realloc(ary_ptr, ((int)q_length + 1)* sizeof(double));
    /* If reallocation fails, quit */
     if (ary_ptr == NULL)
       printf("Can't reallocate memory\n");


    /* Initialize the newly allocated memory locations */
     for (i=ary_end_index+1; i<=(int)q_length; i++)
            {
                 /* Code for testing
                 printf("Initializing %d time\n", i-ary_end_index); */

         ary_ptr[i] = 0;
            }

    /* Increment the element corresponding to the queue
       length in the array */
     ary_ptr[(int)q_length]++;

  /* Update the Max. array index */
     ary_end_index = (int) q_length;
    }
```

unforced state **idle**

attribute	*value*	*type*	*default value*
name	idle	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

forced state **svc_start**

attribute	*value*	*type*	*default value*
name	svc_start	string	st
enter execs	(See below.)	textlist	(See below)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **svc_start**

```
/* determine the time required to complete */
/* service of the packet */
pk_svc_time = 1.0 / service_rate;

/* schedule an interrupt for this process */
 * at the time where service ends. */
op_intrpt_schedule_self (op_sim_time () + pk_svc_time, 0);

/* the server is now busy. */
```

```
server_busy = 1;
```

forced state **svc_compl**			
attribute	*value*	*type*	*default value*
name	svc_compl	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **svc_compl**

```
/* extract packet at head of queue */
/* this is the packet just finishing service */
pkptr = op_subq_pk_remove (0, OPC_QPOS_HEAD);

/* forward the packet on stream 0, */
/* causing an immediate interrupt at dest */
op_pk_send_forced (pkptr, 0);

/* server is idle again. */
server_busy = 0;

/* Sample the queue length after this arrival */
q_length = op_q_stat(OPC_QSTAT_PKSIZE);

if(ary_end_index >= q_length)
      /* If current array index is greater than or equal
        to the queue length, increment the corresponding
        element in the array */
      {
        ary_ptr[(int)q_length]++;
      }
      /* Else reallocate memory and dynamically increase
        array size */
else{
        ary_ptr =(double *) realloc(ary_ptr, ((int)q_length + 1)* sizeof(double)),
      /* If reallocation fails, quit */
        if (ary_ptr == NULL)
         printf("Can't reallocate memory\n");


      /* Initialize the newly allocated memory locations */
        for (i=ary_end_index+1; i<=(int)q_length; i++)
                 {
           ary_ptr[i] = 0;
                 }

      /* Increment the element corresponding to the queue
        length in the array */
        ary_ptr[(int)q_length]++;

    /* Update the Max array index */
```

Deterministic queue

...

```
    ary_end_index = (int) q_length;
   }
```

unforced state END SIM

attribute	*value*	*type*	*default value*
name	END SIM	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs END SIM

```
/* Calculate the total number of observations */
total = 0;
for (i=0; i <= ary_end_index; i++)
     total = total + ary_ptr[i];

/* Calculate the probability mass (or) frequency
   corresponding to each queue length */
for (i=0; i <= ary_end_index; i++)
    ary_ptr[i] = ary_ptr[i]/total ;

/* Prepare the output ASCII file in the OPNET
   required format */
fprintf(fptr,"trace_count = 1\n");
fprintf(fptr, "abscissa: Queue length\n");
fprintf (fptr, "ordinate: Frequency of queue length\n");
fprintf (fptr, "length = %d\n", ary_end_index + 1),
fprintf (fptr, "number of values = %d\n \n", ary_end_index +2);

for (i=0; i<=ary_end_index; i++)
     fprintf(fptr, "%3d -> %g\n",i, ary_ptr[i]);

fprintf(fptr, "%3d -> end\n", ary_end_index + 1);
```