Rochester Institute of Technology

RIT Scholar Works

Theses

6-1-1999

Self-Similarity in a multi-stage queueing ATM switch fabric

Adam Lange-Pearson

Follow this and additional works at: <https://scholarworks.rit.edu/theses>

Recommended Citation

Lange-Pearson, Adam, "Self-Similarity in a multi-stage queueing ATM switch fabric" (1999). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.

Self-Similarity in a Multi-Stage Queueing ATM Switch Fabric

by

Adam Lange-Pearson

A Thesis Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
in
Computer Engineering

Approved by:

Principle Advisor _____
Dr. Muhammad Shaaban, Assistant Professor

Committee member _____
Dr. Wendy Chang, Assistant Professor

Committee member _____
Dr. Seshavadhani Kumar, Associate Professor

Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
June 1999

RELEASE PERMISSION FORM

Rochester Institute of Technology

Self-Similarity in a Multi-Stage Queueing ATM
Switch Fabric

I, Adam Lange-Pearson, hereby grant permission to any individual or organization to reproduce this thesis
in whole or in part for non-commercial and non-profit purposes only.

Adam Lange-Pearson

6-25-99
Date

Abstract

Recent studies of digital network traffic have shown that arrival processes in such an environment are more accurately modeled as a statistically self-similar process, rather than as a Poisson-based one. We present a simulation of a combination shared-output queueing ATM switch fabric, sourced by two models of self-similar input. The effect of self-similarity on the average queue length and cell loss probability for this multi-stage queue is examined for varying load, buffer size, and internal speedup.

The results using two self-similar input models, Pareto-distributed interarrival times and a Poisson-Zeta ON-OFF model, are compared with each other and with results using Poisson interarrival times and an ON-OFF bursty traffic source with Geometrically distributed burst lengths. The results show that at a high utilization and at a high degree of self-similarity, switch performance improves slowly with increasing buffer size and speedup, as compared to the improvement using Poisson-based traffic.

Contents

List of Figures	iii
List of Tables	v
Glossary	vii
1 Introduction	1
2 The ATM Standard	3
2.1 Introduction	3
2.2 ATM Protocol Reference Model	5
2.2.1 Physical Layer	5
2.2.2 ATM Layer	6
2.2.3 ATM Adaptation Layer	8
3 Switch Fabrics	11
3.1 Interconnection Networks	12
3.1.1 Routing Networks	14
3.1.2 Sorting Networks	16
3.1.3 Nonblocking and Rearrangeable Networks	16
3.2 Queueing Methods	19
4 Self Similarity in Data Traffic and Queueing Models	23
4.1 Self-Similar Stochastic Processes	23
4.1.1 Discrete-time Definition	23
4.1.2 Long-range Dependence	25
4.1.3 Heavy-tailed Distributions	25

4.2	Self-Similar Behavior in Data Traffic	27
4.2.1	Determining Self-Similarity in Empirical Data	27
4.2.2	Ethernet	27
4.2.3	Variable Bit Rate Video	28
4.2.4	World Wide Web Traffic	28
4.3	Methods of Generating Data Traffic	28
4.3.1	Self-Similar Traffic	29
4.3.2	Poisson-Based Traffic	32
4.4	Two Analytical Queueing Models	34
4.4.1	Single ATM Output Queue: Pareto Interarrivals	34
4.4.2	ATM Output Queue: Poisson-Zeta ON-OFF Source	39
5	Description of the Simulation	41
5.1	Problem Statement	42
5.1.1	Stage 1: Shared Buffer	43
5.1.2	Stage 2: Output Buffers	44
5.2	General Structure	44
5.3	Event Functions	46
5.3.1	arrivalEvent()	46
5.3.2	sqDepartureEvent()	47
5.3.3	oqDepartureEvent()	47
5.3.4	slotEvent()	47
5.3.5	Methods Used for Statistics Accounting	47
5.4	Input Sources and Queueing Models Used	48
5.4.1	Single Output ATM Queue with Pareto Arrivals	49
5.4.2	ATM Output Queue: Poisson-Zeta ON-OFF Source	49
5.4.3	Fractional Gaussian Noise	50
5.4.4	Poisson Arrivals	50
5.4.5	Bursty (Geometric) ON-OFF	50
6	Simulation Results	51
6.1	Single Buffer	52
6.1.1	Average Queue Length	52
6.1.2	Cell Loss Probability	55
6.2	4x4 Switch	56

6.2.1	Shared Queue Buffer Size	56
6.2.2	Speedup	58
Conclusion		60
6.3	Summary	60
6.4	Future Work	61
Bibliography		62

List of Figures

2.1	Synchronous transfer mode versus ATM	4
2.2	ATM protocol architecture	5
2.3	Communication between layers of two end users	6
2.4	Virtual channels and virtual paths	6
2.5	An ATM cell	8
3.1	General architecture of an ATM switch fabric	12
3.2	Routing between switches	13
3.3	Crossbar network	14
3.4	The n-cube routing network, in a blocking state.	15
3.5	Switching element states	15
3.6	A Batcher sorting network. Each M_i stands for a merge-sorting switch that accepts i inputs.	17
3.7	A rearrangeable network. Incoming cells are rearranged in the sorting stage, so that the routing stage will be free of internal conflicts	18
3.8	General structure of a K -rearrangeable network	19
3.9	Switch buffer architectures	21
4.1	Graphical inverse-transform method for discrete distributions	31
4.2	A 2-state ON-OFF bursty process	33
4.3	Pictorial representation of elements from the transition probability ma- trix.	37
4.4	Bursts in an input with aggregated ON-OFF sources	39
4.5	Poisson-Zeta ON-OFF model of an ATM switch	39
5.1	A two-stage, N-queue, N-server system (servers are part of the queue, for simplicity).	42

5.2	Service time for a cell entering an empty server.	42
5.3	Flow of control for the queueing simulation	45
6.1	Average queue length for a single buffer: Pareto interarrival source. .	53
6.2	Average queue length for a single buffer: Poisson Zeta ON-OFF source	53
6.3	Average queue length for a single buffer: Bursty and Poisson sources	54
6.4	Cell loss probability for a single queue configuration ($\rho = 0.8$).	55
6.5	Cell loss probability in a 4x4 switch configuration	57
6.6	Cell loss probability in a 4x4 switch configuration, with uncorrelated bursty process	57
6.7	Cell loss with a change in speedup ($\rho = 0.8$)	59

List of Tables

2.1	Service classification for AAL	9
6.1	Summary of simulation results (QL = Average Queue Length vs. utilization, CLP = Cell Loss Probability vs. buffer size)	52

Glossary

AAL (page 8) ATM Adaptation Layer. An ATM layer which interfaces the ATM layer with non-ATM protocols and transport methods.

ATM (page 3) Asynchronous Transfer Mode. A connection-oriented scheme which uses the switching of uniformly-sized cells to achieve high speed, a variety of services, and quality of service (QoS) guarantees.

ATM Layer (page 6) An ATM protocol stack layer which is responsible for transmitting the cells.

Banyon Network (page 14) A multistage routing network.

Batcher network (page 16) A multistage sorting network.

Bincount (page 27) In network traffic analysis, the number of packets passing through a given point in the network in a fixed amount of time.

Blocking (page 13) A connection between an IPC and an OPC cannot be made, due to another connection blocking its path internally.

c.d.f. (page 28) Cumulative Distribution Function.

HOL (page 20) Head-Of-Line. Refers to a cell at the front of a queue.

IN (page 12) Interconnection Network. The part of an ATM switch fabric which performs an ordered routing from an IPC to an OPC.

IPC (page 11) Input Controller. The part of an ATM switch fabric which processes the cell headers, and determines which OPC each cell should be addressed to.

Nonblocking (page 13) Any inlet can connect to any outlet in the ATM switch fabric.

OPC (page 11) Output Controller. The part of an ATM switch fabric which accepts up to K cells in a given cycle, and provides the cell headers with the appropriate VP/VC identifiers based on lookup table information.

p.d.f. (page 30) Probability Density Function.

PDU (page 5) Protocol Data Unit. An ATM layer-specific unit of data, which consists of header and trailer information that is added and stripped off as the cell passes through the various layers.

Rearrangeable nonblocking (page 14) A network in which the set of existing connections can be rearranged to accommodate a new connection.

SDU (page 5) Service Data Unit. A PDU which passes control information throughout an ATM system.

SE (page 15) Switching Elements. A small-dimension crossbar switch, used as a building block for a multistage interconnection network.

Slot time (page 18) The time to process one cell, one switch cycle.

Speedup (page 18) A given outlet can accept a cell from more than one input during a single slot time.

Strict-sense nonblocking (page 13) All paths through the switch fabric are independent — that is, any connection can always be made between an idle IPC and an idle OPC without conflict.

VC (page 6) Virtual Channel: A generic, single transmission path within the ATM standard.

VP (page 6) Virtual Path. A bundle of Virtual Channels (VCs).

Chapter 1

Introduction

The goal of this thesis is to study the effect of self-similar traffic on cell loss in an Asynchronous Transfer Mode (ATM) switch, by simulating a combined shared and output buffered architecture using self-similar and Poisson-based input traffic.

Switch fabrics which support the ATM standard have traditionally been analyzed and simulated using Poisson or short-range dependent bursty traffic sources. In [12] it was shown that traffic from a digital source (in this case, Ethernet) is statistically self-similar. In other words, bursts that occur over short time periods are likely to be accompanied by swells of heavy traffic over larger periods of time. Furthermore, if the traffic is very bursty during a given time period, it is likely that the traffic will be bursty in the future. This is in contrast to Poisson-based traffic, which evens out over large periods of time.

In addition to Ethernet traffic, self-similarity has been found in World Wide Web traffic, TCP, FTP, and variable bit rate video streams. One explanation given for this stems from the observation that the degree of self-similarity increases with an increasing aggregation of traffic sources. This implies that when the load on the network is high, the degree of self-similarity is likely to be large as well.

While several analytical studies of an ATM output queue with self-similar input can be found in the literature [18][2], this thesis investigates a more complex queueing model, in which a set of input ports address logical queues in a shared buffer stage, which in turn feeds into an output buffer stage. This model is different from a single output stage switch fabric in that a given logical buffer in the shared queue can grow to accomodate a heavy load at the expense of the buffer space in the remaining logical buffers. Furthermore, the model may feature internal speedup between the shared and output queue stages, which allows more than one cell in a slot time to leave a logical buffer in the shared queue stage.

Because of the level of complexity in this model, performance results are obtained by simulation rather than by an analytical solution. This has several advantages, one of which is that several different kinds of input traffic can be applied. For this study, Pareto distributed interarrival times and a Poisson-Zeta ON-OFF model provide two self-similar processes, while Poisson interarrivals and a Geometrically distributed ON-OFF model are used for comparison. Additionally, the two self-similar processes are compared with each other in terms of average queue length and cell loss probability for a single buffer system.

The simulation results are verified using analytical results from the literature. Then, the simulation is configured as a 4-input, 4-output system and studied for its cell loss characteristics for varying buffer sizes and degrees of speedup.

This investigation furthers the effort to assess the impact of self-similar traffic on ATM systems. Investigating the phenomenon in a complex queueing model provides useful data toward this end.

Chapter 2

The ATM Standard

2.1 Introduction

The evolution of the telephone service into a high-bandwidth network offering a large variety of services has led to the development of next-generation infrastructure technologies. Out of this research has emerged a wide area service called B-ISDN (Broadband Integrated Services Digital Network), which promises to support video on demand, high-resolution music, high-speed LAN interconnection, and other modern digital data technologies. ATM networks underlie the implementation of B-ISDN. It is a connection-oriented scheme which uses the switching of uniformly-sized cells to achieve high speed, a variety of services, and quality of service (QoS) guarantees.

The ATM standards are defined by two organizations, the ITU-T and the ATM Forum. Whereas the former is concerned with ATM as it applies to B-ISDN, the latter produces standards regarding a wide range of ATM applications, including QoS, performance, and testing [5, 6, 7, 8, 9]. Unless stated otherwise, the description of the ATM standard used here comes from [17] and [14], and the above-mentioned documents.

An ATM session essentially consists of a call setup period, followed by transmission of (usually many) 53-byte cells along a predefined path. Cell delivery is not guaranteed, but the sequential ordering of cells is strictly maintained.

The “Asynchronous” in ATM refers to the fact that this system does not use traditional multiplexing techniques. Instead, a cell from a particular source may enter the switch at any free time slot. Figure 2.1 shows that a synchronous technique preallocates a user to a particular place in a frame, while ATM is unframed (no preallocation). Of course, ATM is synchronized in that it switches a cell in regular time slots.

Figure 2.1: Synchronous transfer mode versus ATM

The rest of this chapter gives an overview of the ATM standard, in order to provide context for the ATM switch analysis. A description of the ATM protocol reference model is given, including a brief coverage of the structure of an ATM cell and an explanation of logical connections in ATM networks.

2.2 ATM Protocol Reference Model

The ATM standards are based on an architecture illustrated in Figure 2.2. It consists of three general planes of operation: the user plane provides the transmission of the user's data, the control plane handles call and connection control, and the layer and plane management planes provide control to the particular layer (physical, ATM, etc.) and to the interactions between layers.

The layers of two end users communicate as in Figure 2.3. A cell is encapsulated within a layer-specific protocol data unit (PDU), which consists of header and trailer information that is added and stripped off as the cell passes through the various layers. Similarly, a service data unit (SDU) passes control information throughout the system.

Figure 2.2: ATM protocol architecture

2.2.1 Physical Layer

This layer is responsible for transporting bits from node to node, and packaging them into units suitable for the ATM layer. The main operations performed are framing, cell delineation, and header error correction.

Figure 2.3: Communication between layers of two end users

The network is defined to operate at a speed of 155.52 Mbps (symmetrical), which is compatible with SONET. This means that an ATM switch processes one cell every $2.7 \mu s$. Enhanced versions of ATM networks run at 622.08 Mbps, or four times 155Mbps, and can be either symmetrical or asymmetrical.

2.2.2 ATM Layer

The ATM layer is responsible for transmitting the cells. As this is a connection-oriented system, an end to end transmission path is determined at call time and remains unchanged for the duration of the session. A transmission path within the ATM standard is described in terms of a generic, single connection, known as a virtual channel (VC), and a virtual path (VP, which is a bundle of VC's with a common destination (Figure 2.4). Thus, when an end to end connection is made at

Figure 2.4: Virtual channels and virtual paths

call time, a set of VC's (connected by switches) are found, each of which is associated with a VP. If for some reason a certain VP does not exist for this user, one is created

and a new VC is associated with it. In general, a VC is used to carry data and control signaling specific to those users, while a VP is used for operations that handle routing and communication on a larger scale. The total end to end connection is called a virtual channel connection (VCC).

There are several advantages to the abstraction of a virtual path. Most importantly, the setup time for a VC is incurred only when a VP with the desired destination does not exist. Other advantages include the separation of functions between VC's and VP's and increased network reliability as a result of handling the less numerous VP's.

Properties which help define the nature of ATM transmission for both a VC and a VP are:

Quality of Service. This includes cell loss ratio (number of cells lost divided by the number sent) and cell delay variation.

Nature of Connection. A switched connection type is determined on the fly, but a semipermanent one is of long duration and is set up by special configuration.

Ordering. Cells are guaranteed to be received in the same order in which they are sent.

Traffic Parameters. These include arrival rate, peak rate, burstiness, and peak duration.

The first and the last of the above attributes are determined by contract between the user and the network, essentially within the ATM Adaptation Layer.

The protocol data unit of the ATM Layer is the ATM cell itself. As shown in Figure 2.5, a cell consists of a 5 byte header and a 48 byte payload. The minimal

Figure 2.5: An ATM cell

information needed to satisfy the connection characteristics is contained in the header that specifies:

VP/VC identifiers. When a cell arrives at a switch, a lookup table is used to replace the cell's VP and VC identifiers with new ones for the next link. These identifiers are associated with internal switch addresses and QoS information.

Payload type. This identifies the cell for network or user use, with possible congestion information.

Cell Loss Priority bit. This gives a hint at how important this cell is, when several cells must be discarded within a switch.

Header error control. An 8-bit polynomial-based single-error correction which is calculated using the remaining 32 header bits.

This information is found in cells used for both user data and network control. The user data cells have a smaller VP identifier field in order to fit in a "generic flow control" field. This field is not well-defined, but its proposed uses include multiple priority levels which are specified according to the service desired.

2.2.3 ATM Adaptation Layer

The ATM Adaptation Layer (AAL) interfaces the ATM layer with non-ATM protocols and transport methods. An example of this is simple bulk data transfer, in which the amount of data is much larger than the 48 byte payload of the ATM cell. In this case

	Class A	Class B	Class C	Class D
Timing relation between source and destination	Required		Not Required	
Bit rate	Constant	Variable		
Connection mode	Connection oriented			Connectionless
AAL Protocol	Type 1	Type 2	Type 3/4	
			Type 5	

Table 2.1: Service classification for AAL

the AAL breaks up the data and assembles it into multiple cells. To meet the variety of requirements resulting from the different higher-level protocols, the AAL provides the appropriate functionality:

- Handling of transmission errors
- Segmentation and reassembly of large data blocks into ATM cells
- Flow control and timing control

The ITU-T has defined four classes of transmission services which ATM can support. Table 2.1 shows how these are defined by the timing relationship between source and destination, the bit rate variability, and connection mode.

This table also shows the five types of protocols that are defined to support the service classes. The differences between these types lie in the kind of information kept in the AAL protocol data unit, and how this information is used. Each protocol type has its own intended use:

Type 1 Constant bit rate video and voice signals; circuit emulation

Type 2 Realtime audio and video (e.g. videoconferencing)

Type 3/4 Generic data transfer, connection-oriented

Type 5 Generic data transfer, connectionless

The type of service is agreed to between the user and network at call time (or during configuration in the case of a semipermanent connection). A connection of a given type is allowed by the network only if the network determines that the requirements can be met, given the current and expected traffic characteristics.

Chapter 3

Switch Fabrics

In this chapter, an overview of switching theory is provided to describe the context in which any ATM queueing model exists, namely an interconnect network with input and output ports. Such a queueing model may be used to study the performance of a switch architecture.

Central to the ATM layer is a switch fabric, whose task is to route incoming cells to the appropriate outlet, based on the VP/VC address. The most general architecture of an $N \times N$ switch fabric is shown in Figure 3.1, which includes N input controllers (IPC), N output controllers (OPC), and an interconnection network (IN). If more than one cell can be sent out of an IPC or into an OPC, then K links are attached to that IPC or OPC.

An input controller processes the cell headers, and determines which OPC each cell should be addressed to. This is done using a lookup table relating VP/VC identifiers with output addresses. The output controller accepts up to K cells in a given cycle, and provides the cell headers with the appropriate VP/VC identifiers based on lookup table information. The inlets, outlets, and interconnect may employ buffering strategies in order to reduce cell loss.

Figure 3.1: General architecture of an ATM switch fabric

An example of routing from switch to switch (at the ATM layer level) is shown in Figure 3.2. Here, two cells with VP/VC identifiers C and A enter an IPC in switch I. The IPC determines the next VP/VC identifier pair for each cell, as well as the outlet address for this switch. For example, the cell with VP/VC A is sent to outlet c and given VP/VC identifier pair F. This identifier pair is found in a lookup table in switch J, which maps the cell to outlet c and VP/VC E.

Section 3.1 describes the model given in Figure 3.1 in more detail. With a closer look at interconnection networks, Section 3.2 covers of queueing methods within the switch architecture.

A useful reference on switching theory is [14]. Unless otherwise noted, the information found in this chapter is taken from this source.

3.1 Interconnection Networks

The purpose of the interconnection network (IN) is to perform an ordered routing from an IPC to an OPC. This can be done using either a multistage network or a

Figure 3.2: Routing between switches

bus, but the scalability of a bus is dependent on its bandwidth. An implementation for ATM must allow future designs to grow in dimension without necessarily changing the line speed, so multistage networks are generally chosen.

An architecture (or fabric) is said to be *blocking* when a connection between an IPC and an OPC cannot be made, due to another connection blocking its path internally. This state is shown in Figure 3.4. Conversely, in a *nonblocking* state any inlet can connect to any outlet. Obviously, a switch should be nonblocking. This state is divided into three categories:

Strict-sense nonblocking (SNB) All paths through the fabric are independent — that is, any connection can always be made between an idle IPC and an idle OPC without conflict.

Wide-sense nonblocking Blocking is prevented through a judicious connection allocation scheme.

Rearrangeable nonblocking (RNB) The set of existing connections can be rearranged to accommodate a new connection.

Of these, strict-sense and rearrangeable strategies have the least amount of computational overhead, and make the most sense for use in a switch.

A distinction is made between two different types of conflicts that may arise in a switch fabric. An *external conflict* occurs when two inputs try to connect to the same output. In an *internal conflict*, two inputs to an internal SE contend for the same SE output. A nonblocking architecture is free of internal conflicts. However, external conflicts will result in cell loss in the absence of a buffering scheme.

3.1.1 Routing Networks

A routing network provides a way from an arbitrary inlet to an outlet. The simplest type of routing network, the crossbar, gives a direct connection from every inlet to every outlet (Figure 3.3). This method is nonblocking and fast, but is costly for large switch dimensions. For an $N \times N$ switch, the number of crosspoints is N^2 .

Figure 3.3: Crossbar network

A more scalable way is a multistage routing network, or banyon network, such as the one found in Figure 3.4. Such a network is a set of *stages*, each of which

Figure 3.4: The n-cube routing network, in a blocking state.

is comprised of a series of small crossbar switches, called switching elements (SE). The dimension of an SE is typically 2×2 , and can be in one of four states given in Figure 3.5. The SE uses a bit in the output address to determine which outlet of the

Figure 3.5: Switching element states

SE to send an incoming cell. The broadcast states are not used in the basic operation of an ATM switch fabric (though the complexity of broadcasting is supported in many current commercial and research ATM switches).

Referring back to Figure 3.4, the packet addressed to outlet 0110 (the 13th inlet)

is routed through four SE's, each using a different bit in the output address. This is called "self-routing", and it is an important property in multistage networks, due to the resulting distributed nature of the routing process. The other two packets (addressed to 1010 and 1000) have their most significant bit set to 1. Unfortunately, their inlets are connected to the same SE, which causes a blocking state.

There are several classes of interconnects between stages of SE's, one being that found in Figure 3.4. The special properties and design of such interconnects is beyond the scope of this study, but it can be noted that all of the classes can be made functionally equivalent by appending or prepending an additional stage, with a carefully chosen interconnect type.

3.1.2 Sorting Networks

As was seen in 3.1.1, a banyon network can be in a blocking state if the cells happen to arrive in certain inlet positions. A sorting network can be used to arrange these cells at the inlets of the banyon network such that blocking does not occur. These designs are made of binary comparators in which the outlets 0 and 1 are assigned the input cells in the order min-max or max-min. As shown in the batcher sorter in Figure 3.6, the entire 4-bit output address is used to change the initial arbitrary arrangement to a predictable one.

3.1.3 Nonblocking and Rearrangeable Networks

As mentioned above, strict-sense nonblocking (SNB) and rearrangeable nonblocking (RNB) networks can function with minimal overhead.

A SNB is a type of banyon network which is designed to have more than one path between any given inlet/outlet pair. Enough paths exist so that at least one is free

Figure 3.7: A rearrangeable network. Incoming cells are rearranged in the sorting stage, so that the routing stage will be free of internal conflicts

regardless of the other connections made in the fabric, and the computation required in this case is to find a free path for an incoming cell. A sorting network is not needed.

Several design strategies provide multiple paths. One way is *link dilation*, which essentially provides redundant links between stages. Another possibility is the replication of an entire routing network. A third option is to cascade multiple stages connected by a pattern called an “extended general shuffle”. Each strategy involves a tradeoff in terms of cost and design complexity.

An alternative to SNB designs is a rearrangeable network. When cells enter a RNB, they are first sorted and then routed in a nonblocking fashion. Figure 3.7 shows this for a switch with 8 inputs.

A RNB switch fabric can be modified so that a given outlet can accept a cell from more than one input during a single slot time (time to process one cell, one switch cycle). This is known as *output multiplexing*, or *output speedup*. An outlet has a speedup of K , if it can support up to K connections at a given time. A RNB

Figure 3.8: General structure of a K -rearrangeable network

network with this characteristic is referred to as a rearrangeable K -nonblocking network ($K = 1$ for designs without this feature). The general structure of this kind of architecture is shown in Figure 3.8.

When this speedup exists in the switch fabric, some type of buffering must be used at the outlets, because the cell output rate is related to a fixed standard. At minimum, K packets must be stored.

3.2 Queueing Methods

In all of the interconnect designs mentioned in 3.1, up to K external conflicts can be tolerated. As was mentioned above, in an $N \times N$ switch, cell loss occurs when more than $K (< N)$ inputs need to connect to a given output. Since the ATM standard provides guarantees regarding its services, the loss of data is intolerable past a certain point (packet loss probability more than 10^{-8} , currently).

This problem is addressed by buffering at certain stages of the switch fabric. However, this does not solve the problem, because the buffers can overflow when the

arrival rate is greater than the service rate for a long enough time.

There are two general ways to deal with buffer overflow:

Backpressure A buffered stage communicates with upstream stages, to insure that it is sent only as many packets as can safely be handled.

Cell loss No communication exists between stages, and the cell is discarded when it arrives at a full buffer.

When using backpressure, only the first buffered stage may lose packets. In the cell loss case, cells may be lost at any buffered stage.

The remainder of this chapter briefly covers the three main types of buffering strategies, divided by which stage “owns” the buffer space.

A queueing strategy aims to optimize throughput, cell loss probability, and delay time in the system. Additionally, the design must be amenable to VLSI implementation. The three general buffer architectures are shown in Figure 3.9.

An input queueing scheme holds a generic cell at its input port until no external conflict exists with its intended outlet. In its simplest form, the queues are FIFO buffers, which means that if the head-of-line (HOL) cell is blocked due to another input addressing the same outlet, the cells behind it must also wait. The result is that such a switch, with equal input and output throughput, and independent, randomly addressed traffic saturates at about 60% of its capacity ([10], in terms of number of links). Under bursty traffic, this type of system has been shown to saturate at about 25% of link capacity.

Non-FIFO input buffering is a better option, but this requires a much more complex scheduling scheme to be implemented in hardware.

An output queueing scheme can accept a cell from all of the inputs in the same slot time, for a speedup of N in an $N \times N$ switch. However, this may also be

Figure 3.9: Switch buffer architectures

implemented with a speedup of $K < N$. This design minimizes link utilization, but its implementation incurs a cost because of the wide multiplexers and extra interconnect required.

A pure shared buffer design is so named because it consists of a single buffer which is shared among all the inputs and outputs. The buffer has a capacity NB_s , where N is the number of inlets and B_s is the buffer space per inlet. This queue is said to contain N *logical queues*, one for every outlet, each containing the packets destined for a given outlet. Of course, the total size of these logical buffers cannot exceed NB_s . When no other buffering is present, one cell for every outlet may exit from the queue during one slot time.

A shared buffer is known to be the best performer among the queueing schemes discussed here in terms of link and buffer utilization (see [4], [10], and [14]). However, the tradeoff is that the ability to accept and send multiple cells in a slot time requires a high buffer throughput.

In addition to the three schemes discussed here, combinations of these form input-output queueing, shared-output queueing, and input-shared queueing architectures. They can perform better, but at the expense of increased design complexity and larger buffer space. A simulation of shared-output queueing is the focus of Chapter 5.

Chapter 4

Self Similarity in Data Traffic and Queueing Models

4.1 Self-Similar Stochastic Processes

4.1.1 Discrete-time Definition

To look at the behavior of a stationary time series X over different time scales, the m -aggregated time series $X^{(m)} = \{X_k^{(m)}, k = 0, 1, 2, \dots\}$ is defined by averaging X over non-overlapping blocks of size m . This can be expressed as

$$X_k^{(m)} = \frac{1}{m} \sum_{i=km-(m-1)}^{km} X_i$$

One can use this to view X at different time resolutions. For example, $X^{(1)}$ represents the finest possible resolution, and $X^{(3)}$ is the same series whose magnification is reduced by a factor of 3. If the process has the same statistical properties at all values of m (all aggregations), then that process is self-similar.

Self-similarity for a process is defined in terms of its variance $Var[X(t)]$ and autocorrelation $R(t_1, t_2)$:

$$Var[X(t)] = E[X^2(t)] - \mu^2(t)$$

$$R(t_1, t_2) = E[X(t_1)X(t_2)]$$

A process X is *exactly self-similar* with parameter β ($0 < \beta < 1$) if for all $m = 1, 2, \dots$,

$$Var(X^{(m)}) = \frac{Var(X)}{m^\beta}$$

$$R_{X^{(m)}}(k) = R(t_1, t_1 + k) = R_X(k)$$

In many cases, a weaker definition is needed: A process X is *asymptotically self-similar* with parameter β ($0 < \beta < 1$) if for all k large enough,

$$Var(X^{(m)}) = \frac{Var(X)}{m^\beta} \tag{4.1}$$

$$R_{X^{(m)}}(k) \rightarrow R_X(k) \text{ as } m \rightarrow \infty \tag{4.2}$$

The variance of a self-similar process decreases slowly (proportional to $\frac{1}{m^\beta}$) as m approaches infinity. This means that significant deviations from the mean can occur over large time scales, with more likelihood than for typical data packet models. These latter processes have a variance that decreases proportional to $\frac{1}{m}$ [17].

Equation 4.2 shows that the autocorrelation of the aggregated process has the same form as the original one, which suggests that the degree of variability is the same at all time resolutions.

The variable $H = 1 - \frac{\beta}{2}$, $0 < \beta < 1$, is known as the Hurst parameter, and gives

the degree of self-similarity of a process. When $H = 0.5$, self-similarity does not exist, and the degree of self-similarity increases as H approaches one.

4.1.2 Long-range Dependence

An important aspect of a self-similar process is that it is long-range dependent, that is,

$$R(k) \sim k^{-D} L_1(k), \text{ as } k \rightarrow \infty, \quad (4.3)$$

where $0 < D < 1$, and L_1 is a slowly varying function. This shows that the autocorrelation function decays hyperbolically as the time distance increases, and it implies $\sum_k R(k) = \infty$. This non-summability means that small, high-lag correlations have a significant effect on the behavior of the process. This is in contrast to a short-range dependent process, whose autocorrelation function decays exponentially.

An equivalent way of describing long-range dependence is through the frequency domain. This approach is useful for establishing self-similarity in empirical data.

4.1.3 Heavy-tailed Distributions

The formulations in Equations 4.1, 4.2, and 4.3 describe self-similarity in terms of aggregated time series and long-range dependence. To develop queueing models with self-similar input, it is useful to have an interarrival time probability distribution which is self-similar. For this, one can use a heavy-tailed distribution to characterize probability densities relating to interarrival times and burst lengths.

The distribution of a random variable X is said to be heavy-tailed if

$$1 - F(x) = Pr[X > x] \sim \frac{1}{x^\theta} \text{ as } x \rightarrow \infty, \theta > 0$$

The Pareto distribution is cited in the literature as the simplest heavy-tailed distribution. Its density function and distribution functions — $f(x)$ and $F(x)$, respectively — have parameters ϵ and θ ($\epsilon, \theta > 0$), and are given by:

$$f(x) = \begin{cases} \frac{\theta}{\epsilon} \left(\frac{\epsilon}{x}\right)^{\theta+1} & x > \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (4.4)$$

$$F(x) = \begin{cases} 1 - \left(\frac{\epsilon}{x}\right)^\theta & x > \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (4.5)$$

The parameter ϵ is the smallest time value that can be assigned to X . The parameter θ determines the mean and variance of X . For $1 \leq \theta \leq 2$, the distribution has an infinite variance and a finite mean.

It is shown [2] that as the autocorrelation and variance-time curve for a heavy-tailed process with finite mean interarrival time are asymptotically hyperbolic for large times, and hence this distribution is asymptotically self-similar with $\beta = \theta - 1$ according to Equations 4.1 and 4.2.

4.2 Self-Similar Behavior in Data Traffic

4.2.1 Determining Self-Similarity in Empirical Data

Three main methods of characterizing self-similarity in empirical data can be found in the literature. A brief survey of these methods is given in [17], [1], [12], [15], and [16]. Two of these procedures are graphical to identify if the data has self-similar characteristics. The first method is a variance-time plot which exploits the definition given in 4.1. The second method graphs the rescaled range of X over various time intervals. The third procedure is a periodogram, and analyzes the frequency-domain definition of long-range dependence. It assumes that the data is self-similar, and can be used to estimate the Hurst parameter, H .

4.2.2 Ethernet

In a landmark paper [12] on this subject, a large amount of Ethernet traffic over the course of four years was studied. Each packet passing through a monitor in a particular location of a LAN was timestamped with a resolution of up to $20\ \mu s$.

The data was described in terms of bincounts, that is, the number of packets passing through the monitoring equipment in a given amount of time. It was found that the bincount-time plot of this data is similar at different levels of aggregation with Hurst parameters ranging from 0.7 to 0.9, depending on the utilization, for example at time units of $0.01\ sec$ and $1\ sec$ (high utilization was accompanied by a high degree of self-similarity). Furthermore, the researchers found higher H -values as the number of Ethernet users increased. This is in contrast to conventional traffic modeling, in which the aggregate traffic becomes more smooth as the number of users increases.

4.2.3 Variable Bit Rate Video

Several studies of digitized video have shown self-similar characteristics. One such study [1] analyzed 20 video sequences for their statistical properties, and found H -values over the entire meaningful range ($0.5 < H < 1.0$), dependent on the level of activity in the clip. It was also speculated that the long-range dependence characteristic is related to the compression method, that is, to the aggregation of pixel contributions to the compressed output.

4.2.4 World Wide Web Traffic

More examples of self-similarity in data traffic are discussed in [17]. One study of World Wide Web traffic showed self-similarity like that found in the Ethernet study in Section 4.2.2. Here, the researchers found that the data fit a Pareto distribution, with θ ranging from 1.16 to 1.5 ($0.75 < H < 0.92$).

Other studies looked at processes such as control signaling, TCP, FTP, and TELNET traffic.

4.3 Methods of Generating Data Traffic

One of the significant tasks of simulating the ATM switch queueing model is the generation of interarrival times from a self-similar random process. The challenge is two-fold: One is to use an appropriate interarrival distribution that accurately captures the self-similar behavior of the input and the second is to have a distribution from which it is relatively easy to generate random variates. In this section we describe some of the processes used in the simulation in this work.

For those processes where the cumulative distribution function (c.d.f.) is invert-

ible, we use the inverse transform method [11] to generate the corresponding random variate. If x is a random variable with c.d.f. $F(x) = P(X \leq x)$ the method requires the following two steps:

1. Generate U from a uniform probability distribution on the interval $[0, 1]$.
2. Set $x = F^{-1}(U)$

4.3.1 Self-Similar Traffic

Pareto Distribution

The Pareto distribution has a c.d.f. given by

$$F(x) = \begin{cases} 1 - \left(\frac{\epsilon}{x}\right)^\theta & x > \epsilon \\ 0 & otherwise \end{cases} \quad (4.6)$$

The inverse transform method gives a variate of this distribution as

$$t = \frac{\epsilon}{\sqrt[\theta]{U}} \quad (4.7)$$

The mean of this Pareto distribution above is

$$E[T] = \int_{\epsilon}^{\infty} t f(t) dt \quad (4.8)$$

$$= \int_{\epsilon}^{\infty} t (\theta \epsilon^\theta t^{-\theta-1}) dt \quad (4.9)$$

$$= \frac{\epsilon \theta}{\theta - 1} \quad (4.10)$$

where $f(t)$ is the p.d.f. of the Pareto distribution, so the arrival rate λ is

$$\lambda = \frac{\theta - 1}{\epsilon\theta} \quad (4.11)$$

This may be used to find ϵ for a given λ as

$$\epsilon = \frac{\theta - 1}{\lambda\theta} \quad (4.12)$$

Using the expressions in Equations 4.7 and 4.12, interarrival times based on the Pareto function may be generated using the input parameters λ (the arrival rate) and θ (the degree of self-similarity).

Zeta Distribution

Some traffic models are based on a so-called ON-OFF source, which generates one cell per time unit during the ON, or active period, and no cells during the OFF, or idle period. Bursty traffic can be modeled in this way, where the length of a burst is the length of the ON period. Heavy-tailed traffic may be generated by making the distribution of the burst length self-similar [3].

Since the length of a burst is discrete, the zeta distribution is employed to give the heavy-tailed characteristic to the output. The zeta distribution is the discrete counterpart to the pareto distribution:

$$g(L) = KL^{-(1+\rho)} \quad (4.13)$$

where $(L = 1, 2, \dots)$ is the length of the burst, the parameter $\rho(1 < \rho < 2)$ is related to the Hurst parameter $(\rho = 3 - 2H)$, and K is the normalizing constant.

Since the c.d.f. of the random variable is not available, and iterative method is

used to generate variates of this type. If $G(N) = P(X \leq N)$ then we generate the variate with $G(N - 1) < U \leq G(N)$ as shown in figure 4.1. This results in a table look-up scheme. For example, any value of U in the range for l_5 generates the discrete value l_5 . To generate a discrete burst length, the random value $U \sim U(0, 1)$ is used

Figure 4.1: Graphical inverse-transform method for discrete distributions

to look up a value for l . For example, any probability U in the range for l_5 generates the discrete value l_5 .

Fractional Gaussian Noise

It was pointed out in Section 4.2.2 that self-similar ethernet data was described in terms of bincounts, or the number of arrivals in a fixed interval of time. A fast and accurate method of synthesizing self-similar bincounts is to generate a sequence of complex numbers corresponding to the power spectrum of fractional Gaussian

noise [16]. The inverse discrete Fourier transform is then used to obtain a set of time-domain values, which has been shown to have self-similar characteristics.

It is beyond the scope of this report to describe the algorithm used to efficiently implement this process. Using the resulting bincounts is straightforward: A method is chosen to derive interarrival times from the bincounts, the simplest being a uniform distribution of cells within the fixed bin time, and a more sophisticated method being the use of a bursty process withing each bin time. This latter method has been used [4], and the general form of the bursty process is described in Section 4.3.2.

4.3.2 Poisson-Based Traffic

Traditionally, source traffic modeled using various Poisson-based functions. Two simple implementations of this are given below, as they are used in this study for comparison.

Poisson Distribution

The use of the Poisson distribution to generate interarrival times lacks both the long-range dependence of self-similar traffic and the short-range dependence of correlated bursty traffic (described in Section 4.3.2), but the fact that the use of this distribution can greatly simplify many queueing problems makes it a function of choice for many analytical studies [14], [17].

Poisson trace generation is done using the process described in Section 4.3.1. If the arrival process is Poisson with rate λ , then the interarrival times are exponentially distributed with the c.d.f. $F(x) = 1 - e^{-\lambda x}$ [11]. Using the inverse transform method,

we generate an interarrival time as:

$$t_i - t_{i-1} = -\left(\frac{1}{\lambda} \ln U\right) \quad (4.14)$$

Bursty ON-OFF Method

Short-range dependence is found in a bursty process. Whereas a self-similar process guarantees a dependence among interarrival times over a long period of time, the presence of arrival bursts implies a dependence for short periods of time. This makes sense for data traffic, since many transport algorithms involve sending information to a single destination in bursts.

A simple way to model this type of traffic is to use an ON-OFF process as described in Section 4.3.1 that has been described in the literature [4]. This process can be viewed as in Figure 4.2, a state machine with an idle state and an active state. The lengths of the active and idle periods are independent geometrically distributed

Figure 4.2: A 2-state ON-OFF bursty process

random variables, with parameters t_{10} and t_{01} , respectively:

$$t_{10} = \frac{1}{L} \quad (4.15)$$

$$t_{01} = \frac{\lambda}{L(1 - \lambda)} \quad (4.16)$$

where L is the average burst length. The parameter t_{10} is the probability of changing

from an active period to an idle period, and t_{01} is the probability of a transition from idle to active. Hence, the pmf functions are, for $k \geq 1$:

$$P\{L_{active} = k\} = t_{10}(1 - t_{10})^{k-1} \quad (4.17)$$

$$P\{L_{idle} = k\} = t_{01}(1 - t_{01})^{k-1} \quad (4.18)$$

were L_{active} and L_{idle} are the random active and idle periods, respectively. Using the inverse transform method, the active and idle periods are generated as:

$$K_{active} = \left\lceil \frac{\ln U}{\ln(1 - t_{10})} \right\rceil \quad (4.19)$$

$$K_{idle} = \left\lceil \frac{\ln U}{\ln(1 - t_{01})} \right\rceil \quad (4.20)$$

Cell arrivals can be correlated by making each arrival in a burst address the same output.

4.4 Two Analytical Queueing Models

Few analytical studies are found in the literature which examine the characteristics of an ATM output queue with self-similar traffic. The methods of two such studies, each incorporating a heavy-tailed distribution in the input source, are discussed below.

4.4.1 Single ATM Output Queue: Pareto Interarrivals

An ATM switch output queue is modeled in [2] as a queue with deterministic service times and a self-similar discrete Pareto input.

This study considers a single buffer of an ATM switch, which is fed by a finite number of s input ports. Slot time T is partitioned uniformly into s units, and one

input port may inject a cell during a given $\frac{T}{s}$ units of time. The overall effect is that during a given slot time, the output buffer may receive up to s cells and releases up to one cell.

The system is described by the researchers as a Markov chain embedded at the arrival epochs on the state space $0 \cup \{(i, j) | i = 0, 1, \dots; j = 1, \dots, s\}$. The state 0 represents an empty queue and idle server, and (i, j) represents i cells in the queue and j units of service remaining for the cell in service (one unit is of time $\frac{T}{s}$).

The transition probability matrix is given in partitioned form in [2] as

$$P = \begin{bmatrix} B_{00} & B_{01} & & \\ B_0 & A_1 & A_0 & \\ B_1 & A_2 & A_1 & A_0 \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}$$

where

$$\begin{aligned}
 A_0 &= \begin{bmatrix} 0 & \cdots & & & \\ f_1 & 0 & \cdots & & \\ f_2 & f_1 & 0 & \cdots & \\ \vdots & \ddots & \ddots & \ddots & \\ f_{s-1} & \cdots & f_2 & f_1 & 0 \end{bmatrix} \\
 A_i &= \begin{bmatrix} f_{is} & f_{is-1} & \cdots & f_{(i-1)s+1} \\ f_{is+1} & f_{is} & \cdots & f_{(i-1)s+2} \\ \vdots & & \ddots & \\ f_{(i+1)s-1} & \cdots & \cdots & f_{is} \end{bmatrix} \\
 B_i &= \begin{bmatrix} F_{(i+1)s+1} \\ F_{(i+1)s+2} \\ \vdots \\ F_{(i+1)s+s} \end{bmatrix} \quad B_{01} = \begin{bmatrix} f_{s-1} & f_{s-2} & \cdots & f_1 & 0 \end{bmatrix}
 \end{aligned} \tag{4.21}$$

$$B_{00} = F_s \quad F_i = \sum_{j=i}^{\infty} f_j \tag{4.22}$$

The values f_i are the discretized probability mass function values obtained from the survival function given as

$$S(t) = \begin{cases} \left(\frac{\epsilon}{t}\right)^\theta & t > \epsilon \\ 1 & otherwise \end{cases} \tag{4.23}$$

The interpretation of the elements may be understood by referring to Figure 4.3. Each of the four figures describes one of the probabilities (f_i) from a partition. The boxes represent cells in the system, and they are numbered according in order of

Figure 4.3: Pictorial representation of elements from the transition probability matrix.

arrival. Cells arrive at the queue one at a time, as shown by the single-ended arrows, and they accumulate into columns of cells, which represent the queue length. The number above the top cell in each column denotes the amount of service time remaining for the head-of-line cell.

The elements in the matrix correspond to the probabilities of the process moving from one state to another at arrival epochs, indicated by double-ended arrows in Figure 4.3. For example, the element B_{00} is a scalar, and is the probability that more than s units of service time exist between two arrivals. If cell number 1 arrives at an empty system, so does cell number 2 if all of cell 1's service time (s units) has elapsed.

The array B_{01} holds probabilities for the case in which the system is empty at the first arrival epoch, and that cell remains when the next cell arrives. For the queue to increase, less than s units of service time passes between arrivals.

Similarly, A_0 represents an increase in queue length by one, for the cases in which one or more cells are in the system at the first arrival. The A_i matrices ($i \geq 1$) represent the cases in which the queue loses cells at the second arrival, and are indexed by the number of cells that depart during the interarrival time.

When the Markov chain is ergodic, the stationary vector $\underline{x}$ is the solution of the system $\underline{x}P = \underline{x}$, $\underline{x}e = 1$ (here, e is defined as a column of ones). The vector $\underline{x}$ is partitioned similar to the matrix P . In [13] it has been shown that under these conditions the stationary distribution $\underline{x}$ has a matrix geometric form and efficient algorithms may be devised to compute $\underline{x}$ iteratively. Once $\underline{x}$ is known, performance measures of the system may be computed.

In [2], it was found that with a self-similar interarrival time distribution that has a Hurst parameter of 0.9, the average queue length grows to over 100 cells even for queues with such small utilization as 0.6 showing the impact of self-similarity vis-a-vis

4.4.2 ATM Output Queue: Poisson-Zeta ON-OFF Source

Another way to model an ATM output buffer with self-similar traffic is to focus on aggregations of ON-OFF sources [3]. In this study, the number of bursts arriving at a discrete point in time is an independent random variable with a Poisson distribution, and the lengths of the bursts have independent identical Zeta distributions. The general form of the Zeta distribution is given in Equation 4.13.

The behavior of one input source is shown in Figure 4.4, as layers of bursts. All of the cells in each burst addresses the same output port, according to some probability (a uniform distribution in this case). As shown in Figure 4.5, an ATM switch is modeled by a set of input ports, each input containing an aggregation of bursts which address the output buffers.

Figure 4.4: Bursts in an input with aggregated ON-OFF sources

Figure 4.5: Poisson-Zeta ON-OFF model of an ATM switch

This traffic model was used by the researchers in [3] to find upper and lower

bounds to the cell loss probability, and showed that cell loss is many orders of magnitude higher for heavy-tailed burst lengths than when the burst lengths are geometrically distributed. Furthermore, cell loss under self-similar traffic decreases non-exponentially with increasing buffer size. Traditional Markovian models exhibit an exponential decrease in cell loss in this case.

Chapter 5

Description of the Simulation

The primary goal of this thesis is to examine the behavior of a multi-stage queuing ATM switch fabric, under several types of self-similar input traffic. The two queueing stages used are a shared queue and a set of output queues, introduced in Section 3.2.

A simulation was chosen to analyze the performance of this switch, mainly because the correlation between logical buffers in the shared queue would make an analytical solution very difficult to find. The correlation happens when one logical buffer in the shared queue fills up and takes space away from neighboring logical buffers. This changes the size of the logical buffers, which means that the sizes of the logical buffers are interrelated and are not fixed. Two important simplifications used in an analytical study of cell loss are that the buffers are of finite size and their statistics are independent of one another.

This chapter describes the simulation in detail, beginning with the problem statement, followed by the flow structure of the simulation. Finally, the implementation of the various traffic models supported by the simulation is described.

5.1 Problem Statement

We consider a two-stage, multiple queue/multiple server queueing system (see Figure 5.1), for which the interarrival times are independent, identically distributed random variables. Time is partitioned into slot times, which in an ATM switch is the

Figure 5.1: A two-stage, N -queue, N -server system (servers are part of the queue, for simplicity).

time in between cell departures. If a cell arriving from one of the N input ports finds its destination server idle (shared queue), it must wait in that stage until the next slot boundary, and then wait for a full slot time to pass. This is shown in Figure 5.2.

Figure 5.2: Service time for a cell entering an empty server.

A cell arriving at a busy queue must wait until the other cells in the queue have exited, before departing. Depending on the parameters given to the system, one or more cells may depart from one first-stage queue at the end of each slot time (this is internal speedup, and is described in the next section).

Upon departing from the first queueing stage, a cell must wait in the second stage, with the same definition of service time. The second stage releases 0 or 1 cells at the end of each slot time.

The system begins with no arrivals and no cells in the system. The simulation ends when a given number of traces have arrived at the system, and no cells are found in either queueing stage.

The interconnect linking the inputs and the two queueing stages is assumed to be fully nonblocking, and exhibit zero delay. How an ATM switch achieves nonblocking behavior is the focus of Chapter 3;

The following sections describe the two queueing stages in detail.

5.1.1 Stage 1: Shared Buffer

The first stage is a single buffer which is “shared” by all of the inlets. The buffer is uniformly partitioned into logical output queues, one for each output in the second stage, as shown by the bold lines in Figure 5.1. The size of these logical buffers can change, so that if one buffer is experiencing a higher load than the others, it can expand in order to prevent cell loss while one or more of the other logical buffers temporarily shrinks. Thus, the individual logical buffer sizes are dynamic, but the sum of all logical buffer sizes remains constant.

This stage may have an output speedup K , that is, at each clock time it can eject up to K cells. In an ATM switch this implies that the shared queue can process cells

K times faster than the switch's clock speed.

5.1.2 Stage 2: Output Buffers

The second stage of this system is a set of output queues, each of which may accept up to K cells during a slot time, and which emits 0 or 1 cells at the end of each slot time. Several ways of achieving speedup are found in Section 3.1.3.

5.2 General Structure

The queueing system is implemented as an event simulation, where the state changes at arrivals to and departures from each queueing stage, and at the end of every slot time. The program design, depicted in Figure 5.3, is based on a model found in the literature [11], and is meant to accomodate any type of input traffic. This can be done because the origin of an incoming cell is important only at the time of arrival, and does not affect the cell's subsequent movement through the system.

The software is written in C, in order to obtain the flexibility needed to input different forms of input traffic and output addressing. Also, C is a procedural language, which means that program operation is oriented around operations (rather than objects, for instance, in object oriented designs). Designing the simulation from this perspective allows the use of proven methods, and works well for the straightforward interaction between functions and data.

The simulation begins with initialization in `main()` and in an initialization routine (`initState()`), where all variables are set to appropriate values and buffers are allocated memory. The program then goes into a loop that terminates on the stopping condition given in Section 5.1.

The loop consists of a timing routine and one of several event routines. The

Figure 5.3: Flow of control for the queueing simulation

purpose of the timing routine is to determine which event should occur next, based on the system state, and to advance the system clock to the time of that event. The events modify the state of the system according to the type of event it is, as well as scheduling the occurrence of future events (such as a cell arrival or departure). The events are described in detail below.

Each event makes use of special utility functions which generate random numbers from the various distributions needed for this simulation.

After reaching the stopping condition of the specified number of traces having gone through the system, the simulation generates a report and exits.

5.3 Event Functions

The simulation operation is based on a set of counters which represent the size of the various queues in the system, as well as other aspects of the state of these queues (number entered, number lost, number of cells ready to depart, etc.). Each event function updates a subset of these counters, depending on which part of the system it is controlling.

If several events are scheduled for the same time, they are called in the order presented below.

5.3.1 `arrivalEvent()`

The arrival of a cell to the shared queue constitutes an arrival event. When this function is called, an output address for the incoming cell is determined, and counters pertaining to the state of the shared queue are updated. Additionally, another cell arrival is scheduled from the input sources.

5.3.2 sqDepartureEvent()

This event is called when a cell in the shared queue has completed its service time and is at the front of the queue (in the server). This event processes both the departure of the cell from the shared queue and the arrival to the addressed output queue, so the state pertaining to both queues are updated.

5.3.3 oqDepartureEvent()

This event is called when a cell in an output queue has completed its service time and is at the front of the queue. In this case, the state pertaining to the output queues is updated.

5.3.4 slotEvent()

This event is needed to carry out operations that must occur every slot time, after all arrivals and departures have been processed. The state changes done in this function mainly pertain to parameters that the `timing()` function uses to determine the next event.

5.3.5 Methods Used for Statistics Accounting

Statistics are kept for each queue in both of the two stages.

Arrival rate

The arrival rate, λ , is an input parameter to the simulation, and is compared with the measured arrival rate for each queue in the two stages as determined by dividing the number of cells offered to the queue by the simulation run time, in number of slot times. This is used for maintaining internal accuracy.

Cell Loss Probability

If an arriving cell (in either stage) is addressed to a full buffer, that cell is lost, and counter for this occurrence is incremented. In `report()`, the probability of cell loss for a queue is calculated as the number of cells lost at that queue divided by the total number offered to that queue.

Average Queue Length

The average queue length $\hat{q}(t)$ can be computed by accumulating the number of cells in the queue at time t , over the total run time T of the simulation, then dividing the result by T [11]:

$$\hat{q}(t) = \frac{\int_0^T Q(t) dt}{T} \quad (5.1)$$

The integral is computed for a given queue by accumulating the number of cells in that queue multiplied by the time since the last event for that queue. Since the number in the queue is constant over this time, the result is an area under the graph of $Q(t)$ and is an accurate value for the integral.

5.4 Input Sources and Queueing Models Used

For the purpose of comparison, this simulation supports several input sources. The first two mentioned in this section, the Pareto arrival process and the Poisson-Zeta ON-OFF sources, are based on analytical models described in Section 4.4, and can be used in this simulation for studying the system as well as for verification.

5.4.1 Single Output ATM Queue with Pareto Arrivals

The major difference between the simulation and the model described in Section 4.4.1 is that the simulation involves both a shared queue and multiple output queues, whereas the latter specifies a single output queue. Both account for multiple input ports.

To make the simulation consistent with the analytical model, the simulation is given parameters to set it up for a single output queue and hence a single logical buffer in the shared buffer. Internal speedup is set to one, so that the single logical buffer in the shared queue emits one cell per slot time. Furthermore, the buffer sizes are set to a sufficiently high number that cell loss does not occur.

Given these parameters, the shared queue functions the same as the output queue in the matrix geometric model. The remaining dissimilarity is that the waiting time in the simulation is not the same as in the matrix geometric model, but the resulting difference is insignificant.

5.4.2 ATM Output Queue: Poisson-Zeta ON-OFF Source

This model, described in Section 4.4.2, requires two significant changes to the simulation.

First, recall that in the analytical model, a single input port consists of an aggregation of bursts, each emitting a cell on consecutive slot times. In the simulation model presented, and in a physical ATM switch, an input port may inject just one cell per slot time. If all ports could source more than that, the input rate could be consistently higher than the overall output rate, causing chronic buffer overflow.

To accomodate this, the simulation considers a single “virtual” input port, which can take on new bursts according to a Poisson distribution. Each new burst is assigned

to a unique input port in the simulation, from a large pool of inputs initialized at the outset. The parameters to the Poisson and Zeta functions can then be scaled to be consistent with the analytical model.

This model also calls for correlated output addressing, which means that all the cells in a given burst must address the same output. This is done in the simulation by keeping track of the output the burst is addressing, and generating a new address for a cell only when it is the first one in a burst.

5.4.3 Fractional Gaussian Noise

The self-similar traces from this method are generated as bincounts in software provided by the researcher who developed it [16] (see Section 4.3.1). To use these traces, the simulation simply reads interarrival times from a unique file for each input port.

One issue is how to convert the bincounts into arrival times. The method chosen here is the most simplistic one, which is to uniformly distribute cell arrivals throughout the fixed bin time.

5.4.4 Poisson Arrivals

Pure Poisson arrival times are generated as discussed in Section 4.3.2, and as this is an input source taken directly from a distribution, no changes to the simulation are needed.

5.4.5 Bursty (Geometric) ON-OFF

Section 4.3.2 describes in detail how this source is generated. Similar to the Poisson-Zeta ON-OFF model, the simulation accomodates bursts and correlated output addressing.

Chapter 6

Simulation Results

Data was gathered from the simulator described in Chapter 5 to satisfy two inquiries: A comparison of the Pareto and the Poisson-Zeta self-similar traffic models, and an investigation of the internal characteristics of the 2-stage buffering ATM switch. The traffic model comparison was achieved by configuring the simulator as a single output buffer with four input sources. The simulation of the entire switch was done by configuring the system as a 4 input, 4 output fabric. This small number kept the simulation runtimes within practical limits.

The four different input source traffic models were fit to the simulation using the methods discussed in Section 5.4.

Each data point in this study is the average of four simulation results with different random number seeds, and each simulation was run for 10 million cells (40 million cells total for each data point). This means that results in the range of 10^{-6} or below may contain non-negligible error.

Table 6.1 gives a summary of the results discussed in this chapter.

		Pareto	Poisson-Zeta	Bursty	Poisson
Single Queue	QL	Exponential growth at high utilization and self-similarity	Exponential growth at high utilization and self-similarity	Exponential growth, lower magnitude than for self-similar processes	Linear with small slope
	CLP	Non-exponential decrease (H=0.8)	Non-exponential decrease (H = 0.8)	Non-exponential decrease, but faster decrease than Poisson-Zeta process	Exponential decrease
4x4 Switch	CLP	Non-exponential decrease (H=0.8)	Non-exponential decrease (H=0.8)	Non-exponential decrease, but faster decrease than Poisson-Zeta process	Exponential decrease

Table 6.1: Summary of simulation results (QL = Average Queue Length vs. utilization, CLP = Cell Loss Probability vs. buffer size)

6.1 Single Buffer

The results of the two analytical studies described in this thesis, the Pareto-distributed interarrival model (described in Section 4.4.1) and the Poisson-Zeta ON-OFF model (Section 4.4.2), can be compared by looking at the switch in a single queue configuration. Since the analytical models can be related to a single queue cse, the results in this section can be used to verify the simulation by comparison with the analytical results.

6.1.1 Average Queue Length

The effect of increasing self-similarity is seen in Figure 6.1, where at high utilizations and a high degree of self-similarity (large Hurst value for the Pareto function) the average queue length increases exponentially. The results obtained for this data

Figure 6.1: Average queue length for a single buffer: Pareto interarrival source.

Figure 6.2: Average queue length for a single buffer: Poisson Zeta ON-OFF source

Figure 6.3: Average queue length for a single buffer: Bursty and Poisson sources

agrees with the numerical results obtained in [2] from the analytical model described in Section 4.4.1.

Comparing Figures 6.1 and 6.2 show that while the Pareto and Poisson-Zeta ON-OFF processes are comparable for $H = 0.9$, the Poisson-Zeta traffic is more aggressive than the Pareto for lower values of H . Two possible explanations for the larger queue lengths exhibited by the Poisson-Zeta process are that this process may have a higher variance, or that the ON-OFF bursts in the process affects this statistic.

The queue length results from the Geometrically distributed bursty model in Figure 6.3 is similar to the contour and magnitude of the ($H = 0.8$) curve of the Pareto source in Figure 6.1, while the Poisson source is relatively flat for all utilizations. The Poisson behavior is similar to the self-similar curves at $H = 0.6$, because as the Hurst parameter approaches 0.5, the process becomes less self-similar and more Poisson in nature.

Figure 6.4: Cell loss probability for a single queue configuration ($\rho = 0.8$).

6.1.2 Cell Loss Probability

Figure 6.4 shows a comparison of the cell loss probability characteristics for the four types of input traffic used. The Poisson-Zeta process at $H = 0.8$ has the most detrimental effect on the single queue, as it decreases non-exponentially with increasing buffer size. Cell loss associated with the bursty process declines slower than does the Pareto process, possibly because the two processes differ in their variance, or because the bursty arrivals occur in strings of ON periods, unlike the Pareto process.

The Pareto process with $H = 0.6$ and the Poisson source each result in an exponential drop in cell loss consistent with the behavior of a queue with Poisson arrivals. Again, these sources give similar results, because a self-similar process with a Hurst parameter near 0.5 behaves as a Poisson process.

6.2 4x4 Switch

An important use of a simulation such as this is to aid in the process of choosing appropriate buffer sizes. In this queueing model, it was found that for even small buffer sizes (> 8) in the shared and output queues, a speedup greater than 1 caused the cell loss probabilities to drop below the measurable range. Furthermore, just one of the two queueing stages exhibited loss for a given simulation run in all of the practical cases (buffer sizes above 10).

In certain cases, however, the results are interesting and unique to the type of shared buffer in this model, and these are discussed in Sections 6.2.1 and 6.2.2.

6.2.1 Shared Queue Buffer Size

It is seen in Figure 6.5 that when the complete switch is simulated, the cell loss probability characteristics associated with the four input processes are similar to those found in the single buffer case shown in Figure 6.4. With the exception of the Poisson-Zeta process, the flexibility of the shared buffer results in smaller cell loss probabilities over the range of buffer sizes. The Poisson-Zeta process, however, performs worse for the 4x4 case than in the single queue case. This may be due to the variable number of bursty sources increasing above 4.

Figure 6.6 shows a comparison between the Pareto and Poisson interarrival processes with an uncorrelated addressing bursty process. Here, the effect of correlating the addressing for each ON period in the bursty processes has a detrimental effect on performance. Clearly, introducing output addressing correlation for each burst causes the bursty process to perform worse than the highly self-similar Pareto process. Since the Poisson-Zeta process is an ON-OFF model, correlation could likewise affect the results from this model.

Figure 6.5: Cell loss probability in a 4x4 switch configuration

Figure 6.6: Cell loss probability in a 4x4 switch configuration, with uncorrelated bursty process

However, by comparing the correlated processes (Poisson-Zeta and bursty) and the uncorrelated processes (Pareto and Poisson) separately in Figure 6.5 reveals a more important result, which is that the self-similar processes have a more detrimental effect on the cell loss probability characteristic than their Poisson-based counterparts.

In choosing a buffer size for the shared queue in an actual switch, the designer must first determine the nature of the expected input traffic. If the traffic is most accurately modeled as one of the self-similar processes used in this study, then based on these results, the size of the logical buffer can be as low as 50 (Pareto), or it should be well over 400 (Poisson-Zeta). It should be noted that modern ATM switch fabrics do not necessarily need to include this much buffer space. This is partly because fabrics may approach the problem of buffer overflow in different ways. The model used in this study used queue loss to deal with a saturated buffer, but other systems can use more sophisticated schemes, possibly involving some form of backpressure or priority scheme. One example of this is the Atlas switch [10], which uses flow control and a credit-based scheme in coordination with a shared buffer to deal with buffer overflow. The total shared buffer size in this case is 256 cells.

6.2.2 Speedup

The plots in Figure 6.2.2 show the effect on cell loss of changing speedup for small buffer sizes and fixed utilization, ρ (recall that a logical shared buffer size of 2 means that the total shared buffer size is $4 \times 2 = 8$). For each traffic type, the cell loss probability predictably declines when the speedup increases from 1 to 2. The loss probability levels off after that, because the reduced loss in the shared buffer results in an increased loss for the output buffers. The probabilities are quite high in this case, because of the very small buffer sizes.

Figure 6.7: Cell loss with a change in speedup ($\rho = 0.8$)

Conclusion

6.3 Summary

In this thesis we have examined the effect of self-similar traffic on an ATM switch fabric. It was found that for processes with either correlated or uncorrelated addressing schemes, the combination of a high degree of self-similarity in the input process and high utilization degrades the cell loss performance of the system significantly more than for a comparable Poisson-based traffic source.

It was also found that the two self-similar processes used in this study did not have the same effect on the queueing model. While both the Pareto and the Poisson-Zeta processes resulted in similar queue length behavior, the Poisson-Zeta source produced values nearly twice the magnitude of the corresponding Pareto results, for high utilization and Hurst parameter values below 0.9. The cell loss probabilities resulting from the Poisson-Zeta process were likewise higher and more heavy tailed than for the Pareto interarrival process.

Clearly, these results show that if the actual traffic in a physical system is self-similar, then simulations of this system should not ignore the effects of this long-range dependence. Furthermore, the process used to model the self-similar traffic should be carefully chosen, as it may have an impact on the accuracy of the simulation results.

6.4 Future Work

Several results from this study revealed work that remains to be done in this area. First, the effect of correlated output addressing on self-similar processes was significant enough in this study to warrant further work in this area.

Second, a closer analysis needs to be done of the relative merits of different self-similar processes when used as simulation input processes. To accurately simulate a more complex ATM switch fabric than the one simulated here, not only the self-similar nature of the input traffic should be examined, but the method used to model the self-similar process should be considered as well.

Bibliography

- [1] J. Beran, R. Sherman, et al. Long-range dependence in variable-bit-rate video traffic. *IEEE Transactions on Communications*, 43:1566–1579, 1995.
- [2] J.E. Diamond and A.S. Alfa. Matrix analytical model of an ATM output buffer with self-similar traffic. *Performance Evaluation*, 31(3–4):201–210, January 1998.
- [3] Yanhe Fan and Nicolas D. Georganas. Performance analysis of ATM switches with self-similar input traffic. *Computer Systems Science and Engineering*, 1997.
- [4] Simon Fong and Samar Singh. On the relative importance of arrival statistics and output addressing for shared-buffer ATM switches. In *Proceedings of the 20th Australasian Computer Science Conference*, February 1997.
- [5] ATM Forum. User-network interface (uni) specification version 3.1, September 1994.
- [6] ATM Forum. Traffic management specification version 4.0, April 1996.
- [7] ITU-T Recommendation I.321. B-ISDN protocol reference model and its application, 1991.
- [8] ITU-T Recommendation I.361. B-ISDN ATM layer specification, 1995.
- [9] ITU-T Recommendation I.362. B-ISDN ATM adaptation layer (AAL) functional description, 1993.
- [10] Manolis Katevenis, Panagiota Vatsolaki, and Aristides Efthymiou. Pipelined memory shared buffer for VLSI switches. In *ACM SIGCOMM'95 Conference Proceedings*, pages 39–48. ACM, August 1995.

- [11] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Inc., 1991.
- [12] W. Leland, M. Taqqu, et al. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2:1–15, February 1994.
- [13] Marcel F. Neuts. *Matrix-Geometric Solutions in Stochastic Models*. Dover Publications, inc., 1994.
- [14] Achille Pattavina. *Switching Theory: Architecture and Performance in Broadband ATM Networks*. John Wiley and Sons, 1998.
- [15] V. Paxson, R. Sherman, et al. Wide area traffic: The failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3:226–244, 1995.
- [16] Vern Paxson. Fast, approximate synthesis of fractional gaussian noise for generating self-similar network traffic. *Computer Communication Review*, 1997.
- [17] William Stallings. *High-Speed Networks: TCP/IP and ATM Design Principles*. Prentice-Hall, Inc., 1998.
- [18] B. Tsybakov and N. Georganas. Overflow probability in an ATM queue with self-similar input traffic. In *1997 IEEE International Conference on Communications*, volume 2, pages 822–826, Montreal, Quebec, June 1997. IEEE.