# SIMULATION AND ANALYTICAL PERFORMANCE STUDIES OF GENERIC ATM SWITCH FABRICS

Nigel John Burton, B.Sc.

A thesis submitted in partial fulfilment of the requirements of Bournemouth University for the degree of Doctor of Philosophy.

April 1993

Bournemouth University
In collaboration with

GPT Limited,
Sopers Lane,
Poole,
Dorset

# ABSTRACT

## Simulation and Analytical Performance Studies of Generic ATM Switch Fabrics

N. J. Burton

At the present time numerous telecommunications networks exist such as the voice network, telex network and packet switched network. Each network has been individually designed specifically to support the limited range of services it carries and as such there is no resource sharing between them.

As technology improves exciting new services such as video phone become possible and economically viable but their deployment is hampered by the inability of the present networks to carry them. The long term vision is to have a single network able to carry all present and future services.

Asynchronous Transfer Mode, ATM, is the versatile new packet–based switching and multiplexing technique proposed for the single network. Interest in ATM is currently high as both industrial and academic institutions strive to understand more about the technique.

Using both simulation and analysis, this research has investigated how the performance of ATM switches is affected by architectural variations in the switch fabric design and how the stochastic nature of ATM affects the timing of constant bit rate services. As a result the research has contributed new ATM switch performance data, a general purpose ATM switch simulator and analytic models that further research may utilise and has uncovered a significant timing problem of the ATM technique.

The thesis will also be of interest and assistance to anyone planning on using simulation as a research tool to model an ATM switch.

## Acknowledgements

This work has been funded by the Telecommunications Systems Group of GPT and the author gratefully acknowledges the support of Mr Mahir Ozdamar and Mr John Kitchener in this matter.

My sincere thanks for continued support and encouragement go to Mr Alan Potton, Director of Studies, and Mr John Reynolds, Project Supervisor, both of Bournemouth University particularly for their assistance in the presentation and content of the thesis.

I am grateful to Dr John Cosmas of Queen Mary and Westfield College, University of London for his help in the early stages of the program.

I am indebted to my friend and colleague Dr John Titchener of the Mathematics and Teletraffic Engineering Group of GPT at Poole for the time he has spent reviewing, discussing, questioning and suggesting new directions for the work contained in this thesis.

Finally, my greatest thanks go to my parents, my wife Alaina and daughter Anna for their faith and encouragement throughout.

# Chapter 1 Introduction and the path to ATM

Asynchronous Transfer Mode, or ATM, is a term applied to a flexible method of information transfer across telecommunications networks. The information to be transferred is split into fixed length packets, now commonly referred to as cells, at the entry to the ATM network. The cells are switched individually across the network and the information is reassembled in its original form at its departure point from the network.

Cell switching is fundamentally different from existing and well understood packet switching and it is these differences that have recently introduced the entirely new technique of ATM switching. This research is concerned with investigating and further understanding the performance of the most important components of these ATM switches.

## 1.1 Summary

This thesis presents the results of studies on performance modelling ATM switch fabrics. Firstly the techniques of performance modelling are developed and proved. This includes the development of a general purpose simulation tool and of analytic techniques. Subsequently the simulation and analytic tools are used in specific investigations of the performance of various architectural configurations of ATM switch fabric. Many of the results, particularly those with relation to the timing problems of ATM, will add significantly to the ongoing ATM debate.

In chapter 1, the path to the present situation in telecommunications networks is plotted. This includes a brief history from telegraphy to manually then automatically switched telephony, through multiplexing, PCM and digital techniques concluding

with a discussion of present day synchronous circuit switching techniques and the voice network. The prospects for the near term future, a short description of ATM including a functional model of an ATM switch and the difference between conventional packet switching and ATM are given. The aims of the research are given at the end of this chapter.

Chapter 2 presents a literature review. The ATM arena is wide and therefore the review concentrates on those areas pertinent to the research. Specifically included are the major ATM projects world–wide, fundamental switch architectures and theory, source policing, source models, queueing theory, a history of simulation techniques, the current state of performance modelling ATM switch fabrics and finally some of the problems of the ATM technique including delay and network synchronisation.

The design, development and validation of the general purpose simulation tool during this project represents a large proportion of the total work. Chapter 3 is devoted to this aspect. In this chapter simulation is introduced as a technique with an important discussion on its limitations. A full description of the philosophy behind the model, the validation techniques employed and the results of the validation are given. The problem of specification of queue lengths in an ATM switch was encountered during the validation stage. The problem along with the solution derived is detailed in this chapter. Not all of the functionality of the simulator designed in at the outset was subsequently exploited in the investigations.

It is widely believed that the performance of ATM switches will depend largely on the arrival characteristics of the cells. As part of the design philosophy of the simulation tool various source models were included to reproduce these arrival characteristics. Chapter 4 presents the three source models investigated including the problem of implementing the continuous negative exponential function in a discrete time model.

The difference in performance (in terms of effect on the ATM switch block performance) between the binomial and the negative exponential models is investigated.

In chapter 5 the performance parameter cell–loss probability is studied for various input loads and buffer size. The chapter presents simulation results for cell loss probabilities down to $10^{-5}$ and introduces a new model giving cell loss probability as a function of input load and the number of queue places. The model is used to predict cell loss probabilities down to $10^{-12}$ and the extrapolated results thus obtained compared with those derived from a generalised analytic model. The comparison shows good correlation between the 2 sets of results.

Chapter 6 presents an important study into some of the timing problems of ATM. Differential delay is defined and a potential method of removing it is discussed. The chapter introduces and defines "re–timing" delay and with the assistance of a hypothetical reference network model goes on to estimate values for both STM and ATM networks. A modification to the re–timing method to reduce the delay in the ATM case is proposed and studied. In this chapter the phase distortion suffered when a 2Mbit/s PCM signal is carried across an ATM network in circuit emulation mode is investigated.

## 1.2 The History of Telecommunications Networks to the present day

This brief historical section is included to illustrate the way that telecommunications networks have developed and to put into perspective the particularly rapid progress of the last decade.

To tell the story fully and well would take many volumes and the few words here cannot hope to do justice to the foresight, ingenuity and persistence of those who were part of it.

### 1.2.1 The Early Days of Telegraphy

One of the first attempts at communication using the properties of magnetic induction found in the early part of the seventeenth century was made circa 1800 by Ronalds with a telegraph consisting of a transmitter, transmission line and a receiver. The operator would manually generate a voltage on the line using a friction machine. The charge thus generated on the line would rotate a face–plate over a disk containing (most of) the letters of the alphabet such that one letter showed through at a time. This would happen simultaneously at both the transmitting and receiving ends. After initial manual synchronisation of the face–plates to a pre–assigned letter, a message could be sent letter by letter. The Ronalds telegraph contains all the essential elements of a communications system, a compatible receiver and transmitter, a transmission link between the two and a protocol to allow the system to be synchronised and signal the start of the message.

The next notable point in the history of communications followed the discovery of the electromagnetic induction effect and resulted in very similar, but incompatible, developments occurring on both sides of the Atlantic. In England, Gauss and Weber developed a telegraph that utilised a magnet within a coil of wire as a transmitter generating a bipolar current which in turn deflected a needle in the receiver either to the left or to the right. Characters were encoded according to a predetermined pattern using up to 4 deflections for each. At the same time as the needle telegraph was under development in England, the Morse telegraph started to come into use in America.

The Morse system used 2 states on a transmission line such as the presence or absence of current to encode characters and was the fore–runner of the technique still used today primarily for radio telegraphy over long distances.

There is a lesson to be learnt in studying the reaction of both notable scientists and a sceptical general public to new ideas such as the telegraph. It is reported that H.G.Dyar, inventor of an electric telegraph system in America some years before Morse, was forced to flee town after being accused of the crime of "conspiracy to commit illegal communication between cities". Further such examples including Lord Kelvin's famous telegraph to the designers of the Niagara Falls power station which read "Trust you avoid gigantic mistake of alternating current" may be found in [1].

### 1.2.2 Transmission and Switching Principles

At this point in the development of telecommunications, experimental systems consisted of 2 terminals with a single connection between them. In uni–directional (simplex) systems, allowing transmission in one direction only, one terminal would be a transmitter and the other a receiver. In bi–directional systems, allowing transmission in both directions, each terminal would consist of a transmitter plus a receiver.

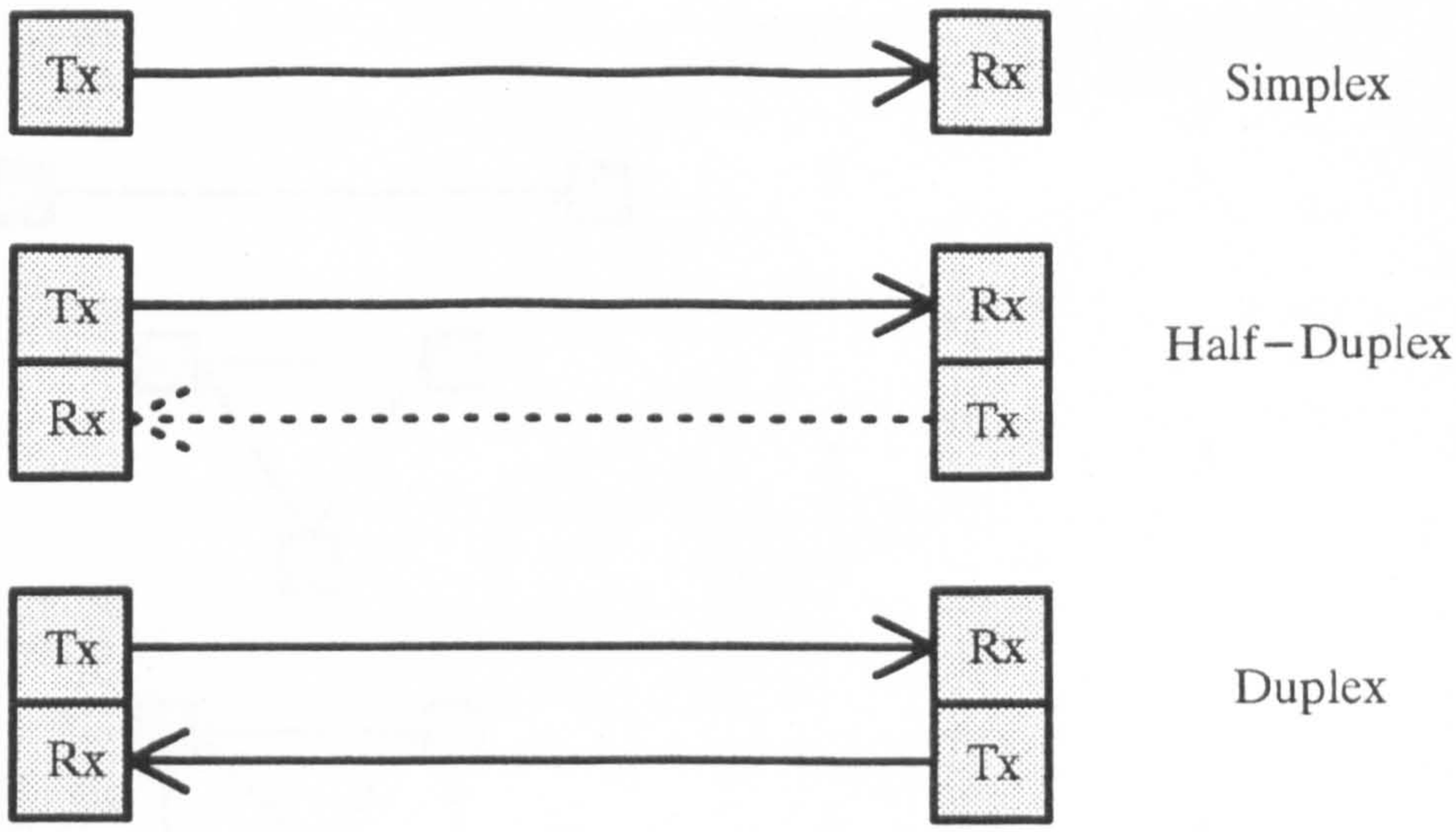


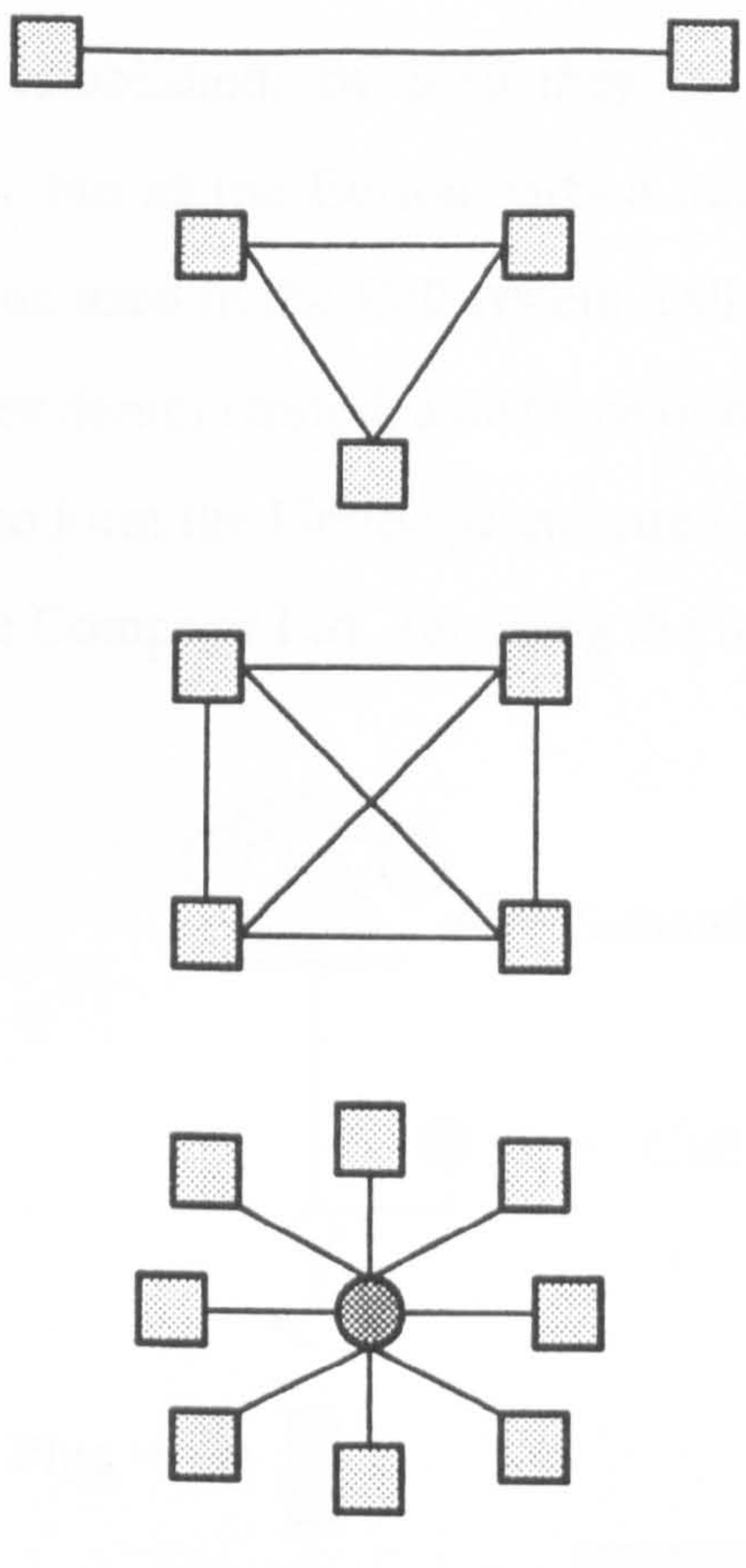

Figure 1.1: Simplex, Half−Duplex and Duplex Transmission

Two possibilities for bi−directional transmission exist, one way at once (half or semi−duplex) and both ways at once (duplex or full−duplex). This is illustrated in figure 1.1.

As these early experimental systems consisted of only 2 terminals, "switching" between terminals was not necessary. However as n (the number of terminals) increases, the number of interconnections also increases:

$$\text{Number of interconnections} = \sum_{i=1}^{n-1} i = \frac{1}{2}n(n-1)$$

and therefore in order to provide a fully available network, i.e. one in which every subscriber can be connected to every other subscriber, the technique of "switching" the subscribers together at a central point was employed. This is illustrated in figure 1.2



| Terminals | Interconnections |
|---|---|
| 2 | 1 |
| 3 | 3 |
| 4 | 6 |
| 8 | 8 |

Figure 1.2: Fundamental "Switching" Requirement

The switches, along with other ancillary functions such as power line feeding and supervisory equipment, were located at a central point which became known as an "exchange".

### 1.2.3 First Steps in Telephony

The telephone was first demonstrated by Alexander Graham Bell, a Scotsman, in America in 1876. By 1878, The Telephone Company Ltd. were offering connections between any 2 points not more than a few miles apart, and before the end of 1879, three exchanges had been opened serving about 200 customers switched by the plug

and cord method as shown in figure 1.3. Following the invention of the carbon microphone by Thomas Edison in 1877, the Edison Telephone Company of London Ltd. was established. By 1880 they too had three exchanges serving about 170 customers, but as the Edison carbon microphone was far more sensitive than the microphone used in the Bell system, calls between London and Norwich had been successfully demonstrated, a distance of over 100 miles. The 2 companies were soon to merge to form the United Telephone Company and later joined with the National Telephone Company Ltd., retaining the latter's name.

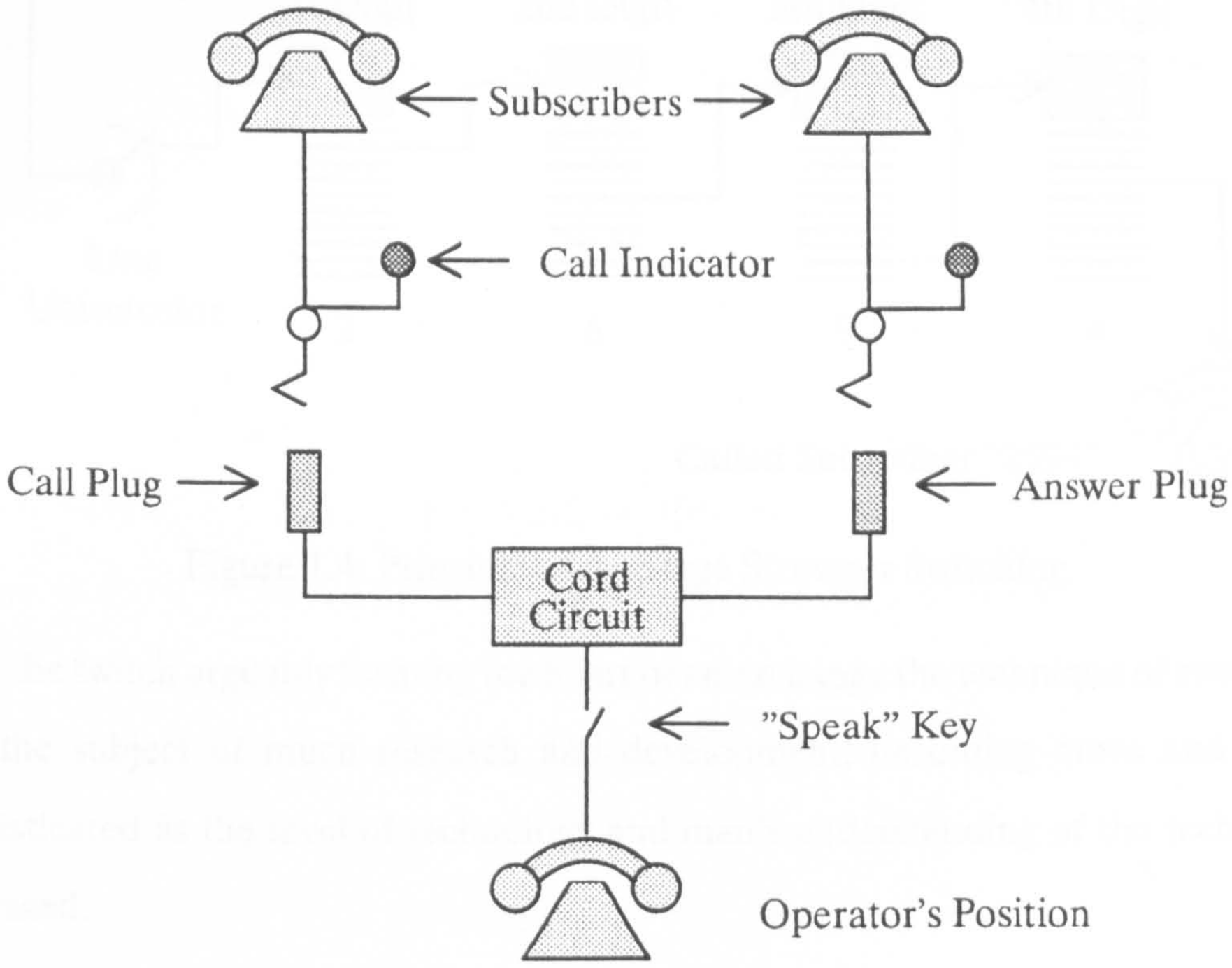



Figure 1.3: Arrangement for Manually–Operated Plug and Cord Switching

### 1.2.4 The Evolution to Automated Switching

These early exchanges utilised manually operated switchboards, or cord boards, to facilitate the connection but by 1880, probably as a result of the rapid expansion in the

use of the basic telephone system, the first patents for automatic switching were filed. However, it was Strowger who provided the first practical electromechanical automatic switching technique with the step–by–step method and the first automatic exchange was opened in 1912. The principle is shown in figure 1.4 in which the calling subscriber has dialled 2594, the 4–digit number of the called subscriber. It is fitting tribute to the sophistication of this method that it is still in widespread use today.

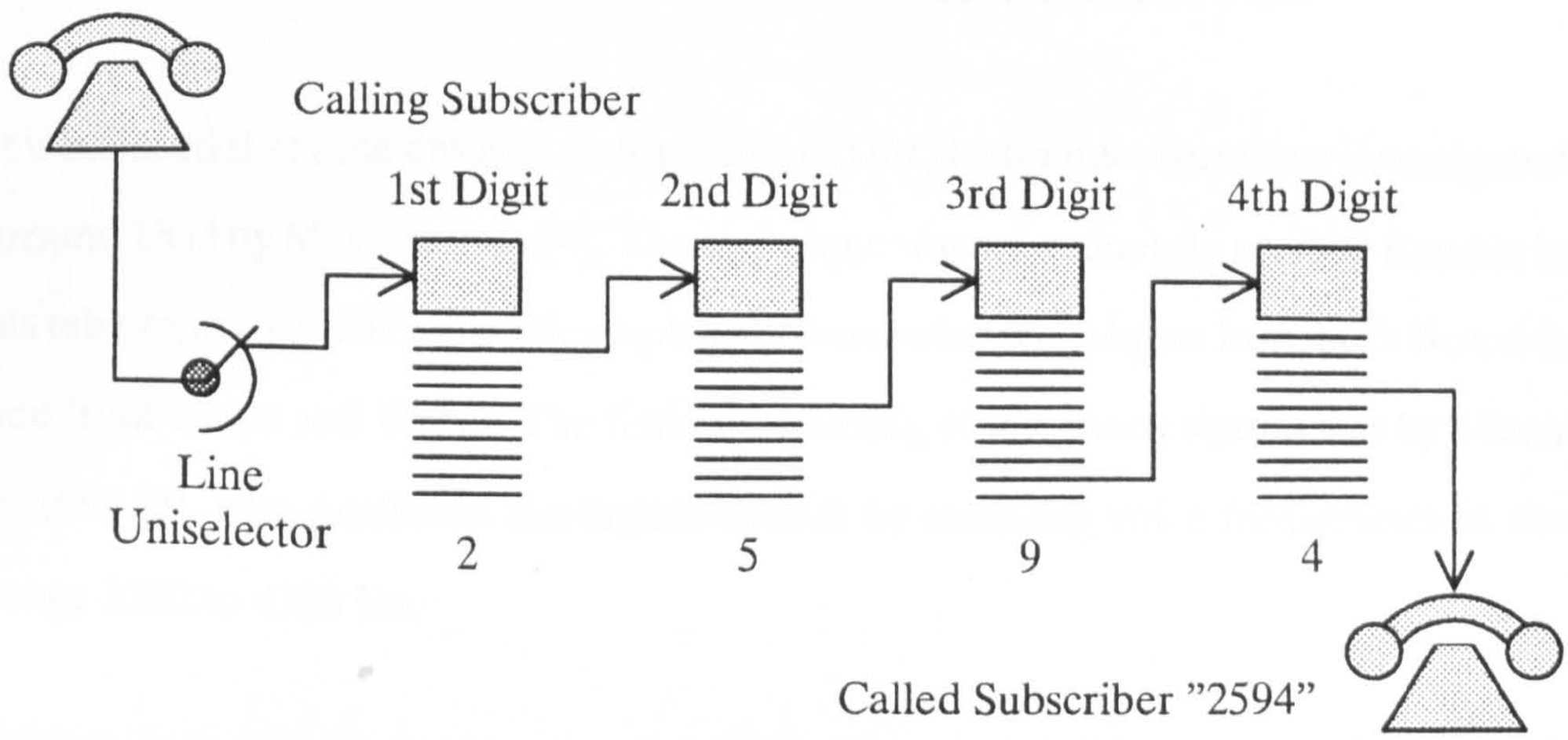



Figure 1.4: Principle of 4–Stage Strowger Switching

With the switch arguably forming the heart of an exchange the technique of switching was the subject of much research and development, becoming more and more sophisticated as the level of technology and man's understanding of the technique increased.

The early development of a new type of switch called a cross–bar switch [2] culminated in 1938 with the first operational version of Bell System's No.1 cross–bar exchange. In this type of switch, subscribers were terminated in groups of 10 on to a primary 10x10 crosspoint matrix. Each output of the primary matrix was in turn connected via an intermediate link–interconnection stage to a secondary 10x10 crosspoint matrix. This three–stage structure allowed each subscriber access to one

of 100 output circuits, and the three–stage architecture could be cascaded to produce wider connectivity. This architecture later formed the basis for the first electronic circuit switches which used reed relays with common control, the TXE–2 (Plessey) and TXE–4 (STC) systems. (An excellent history of the telecommunications industry in Britain up to 1948 is given by J. H. Robertson in The Story of the Telephone. [3])

### 1.2.5 The Introduction of Multiplexing, Digital Techniques and PCM

It is believed that time division multiplexing (TDM) techniques were first investigated around 1853 by M. B. Farmer [4]. The technique was subsequently used by Baudot in his telegraph circa 1875, the telegraph itself borrowing techniques from both Ronalds and from Gauss and Weber. The first multiplexing of telephony signals was by Miner in 1903 [5], who produced intelligible speech by sampling voice frequencies in the range 3500 to 4300 Hz.

Digital techniques clearly were used in the very earliest forms of telegraph communication. By the mid–nineteen thirties, the theory of speech sampling and TDM techniques was known, albeit in a rudimentary form. The work of A. H. Reeves towards the end of this period resulted in the first patents on PCM in France, Britain and America [6]. This work was motivated by the desire to exploit the immunity from interference property of the telegraph system and apply that to telephony signals. At that time, the specification had 32 quantisation levels as against 256 today. It was not until after the second world war that the PCM technique was finally demonstrated, and it took until 1962 to be introduced by the AT&T Corporation into the American network in the form of a 24 channel system.

In addition to the significant benefits of the PCM system of its immunity from the effects of noise, its ability to be regenerated and to carry accurate timing information

across the network, the PCM system allows a number of simultaneous conversations to be carried on the same transmission line. This is achieved by dividing the available *time* on the transmission system into a number of channels giving rise to the term *time–division multiplexing*, or TDM. The principle is shown in figure 1.5

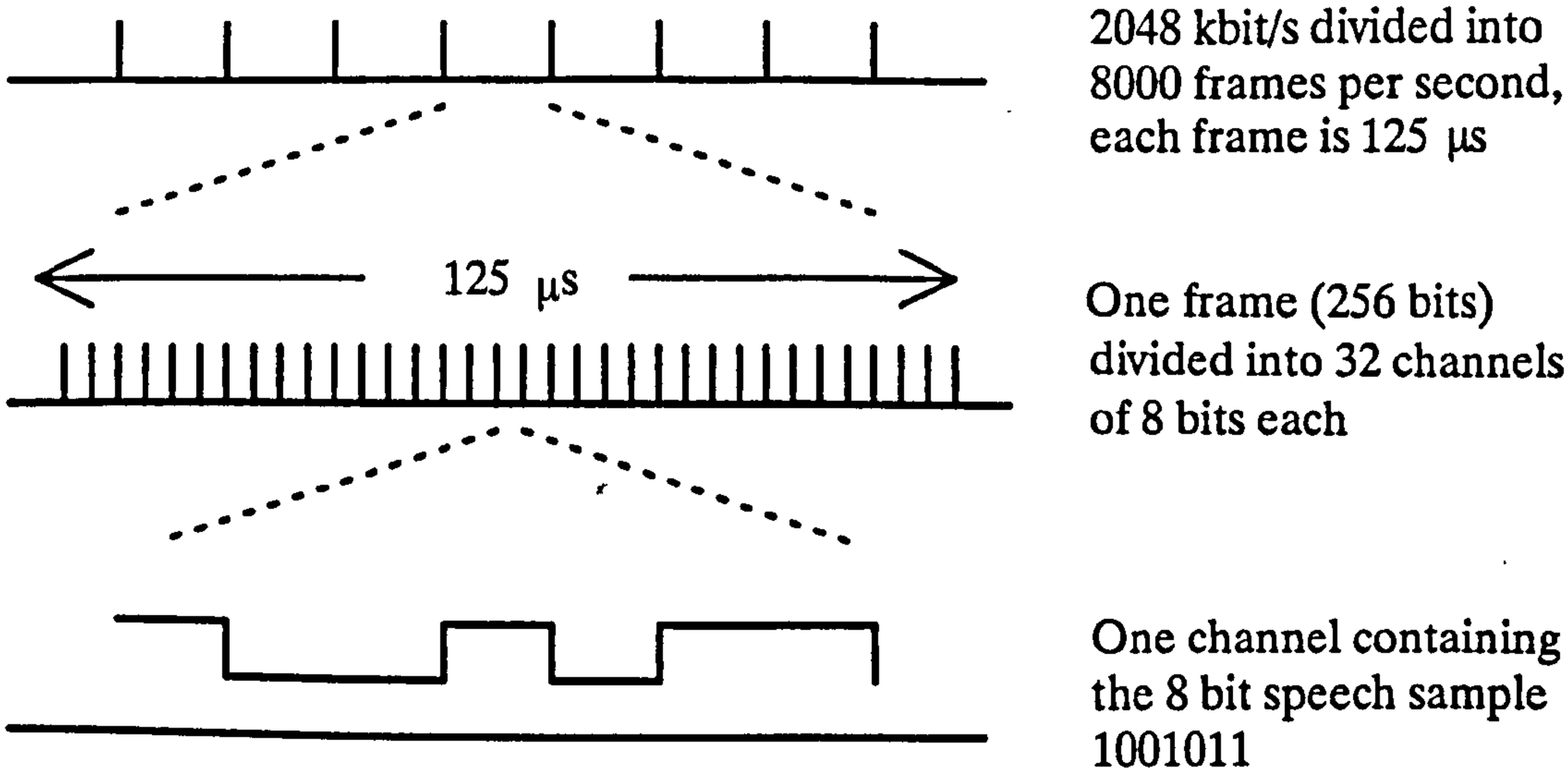



Figure 1.5: Time Division Multiplex Principles

Firstly, the 2.048 Mbit/s raw data signal is divided into frames, each 256 bits long. The frame repetition rate is therefore 8000 $x^{-s}$. Secondly, a frame is divided into 32 channels, each of 8 bits. The first channel, referred to as timeslot zero, contains a repetitive pattern to allow the receiver to identify and locate the frame structure. The remaining channels, each providing 8 x 8000 bits/s, may then carry 31 simultaneous conversations in digital form along the wire. (Note: it is usual for timeslot number 16 to carry signalling information.) An important point to note is that the source of the speech in a channel is determined solely by the position of that channel in time, i.e. its position with respect to timeslot zero.

The PCM system is important in that its structure determines the requirements, in terms of switching capability in its broadest sense, of the electronic digital switching systems that followed the electronic crossbar variety. This is no small claim as all of

the present day systems, for example Ericsson's AXE10, GPT's System X and Siemens' EWSD, have at their heart a digital switch interchanging timeslots from a standard 30 channel PCM transmission system as defined in [7] (Note1).

### 1.2.6 The Voice Network and Synchronous Digital Circuit–Switching

By far the most extensive network today is the voice network. Carrying digitally encoded speech in PCM systems, the United Kingdom network alone has 18.7 million residential subscribers, 5.2 million business subscribers and 379,000 miles of optical fibre (1990 figures, source: [8]). As discussed in section 1.2.5, the switches for the voice network are designed specifically for handling channels extracted from PCM transmission systems, and these channels are required to carry speech in the frequency range 300 to 3400Hz quantised into 256 levels therefore requiring a bandwidth of 64kbit/s.

Basic synchronous circuit switch designs are optimised around this single bandwidth parameter. Switch architectures for local and trunk exchanges have been devised to switch a number of 64kbit/s channels. The optimum number of such channels for these switches lies in the range 8000 to 64000. Concentrator switches of the same family switch from 200 to 8000 channels. The switches thus designed are extremely good at the job required, but if 8, 16 or 32kbit/s signals are required to be switched for example, a whole 64kbit/s channel is required resulting in inefficient use of the resources and additional expense. It is fundamentally possible to switch higher bandwidths than 64kbit/s by associating a number of single channels together but this

(Note1). Some confusion in terminology can exist here. The American standard PCM system has 24 timeslots containing 24 speech channels with embedded bit–robbed and 193rd bit signalling. The European standard PCM system has 32 timeslots containing 30 speech channels, 1 synchronisation and alarm timeslot and 1 signalling timeslot. The European system is usually referred to as 30 channel, and the term channel usually refers to a timeslot containing speech.

introduces more problems in terms of management and the consistency of the data, and is restricted to a maximum of just under 2Mbit/s.

The early nineteen eighties brought a use of associated channels in the form of narrow band ISDN (N−ISDN) in which two 64kbit/s channels and a 16kbit/s signalling channel are provided to a customer over a standard copper pair, the access being digital. The two channels could be used for either voice or data and the elegance of the solution for data communication is that the modems and codecs in the subscriber's loop can be dispensed with. The analogue to digital conversion for speech is performed by a speech convertor at the subscriber's premises. One can only speculate why there are only approximately 12000 N−ISDN subscribers in the United Kingdom today [9].

### 1.2.7 Other Present−Day Telecommunications Networks

The voice network is by no means the only type of telecommunications network. Numerous others exist including the telex and the packet switched network, and all are characterised by a single word, specialised. Each network is optimised to carry its particular service, and each is managed individually. Other networks include the various mobile systems, broadcast television and radio over the air, and television from both satellite and cabled systems. Each network's performance characteristics, for example in terms of delay, data loss and simple bandwidth, match the requirements of the service it carries, and each has been through its own specification, design, manufacturing, installation and maintenance phases.

The important characteristics of some sample services are shown in table 1.1 below:

| Service | Data Rate | Average Call Holding Time | Affected by delay? | Affected by jitter? | Affected by corruption? |
|---|---|---|---|---|---|
| Voice | 64kbit/s | 60s op. assist<br>120s ordinary | Yes | Severely | No |
| Telex | 0.1kbit/s | 300s | No | No | Yes |
| PSS | 48kbit/s | Variable | No | No | Yes |
| Credit card verification | 9.6kbit/s | 20s | No | No | Yes |
| Video | 2Mbit/s | 100s (viewphone) | Yes | Yes | Moderately |
| Broadcast TV | 140Mbit/s + | 3000s + | Yes | Yes | Yes |

Table 1.1: Characteristics of Various Communications Services

An important thing to note about all of the independent networks is that there is no resource sharing between them, and therefore resources that are free in one network may not be used in another.

In concluding the discussion on present day networks, it can be seen from table 1.1 that each service has very different requirements on its network, and each network is tailored to the individual requirements of the service it is carrying, and therefore when viewed together, they are specialised, inflexible and inefficient. One vision of the future that emerged in the early nineteen eighties was that of a single, all–purpose network allowing all services to use, and share, the same resources. Clearly new transmission and switching techniques would be required for the single network

philosophy and these would need to be versatile enough to support the wide variety of present and future services.

## 1.3 The Near Future and Longer Term Aims and Prospects

The previous sections have described how multiple networks have been developed to serve their own specialised service. Responding to diverse needs by application specific development is no longer a viable approach in an age of increasing and rapidly changing demands. In the early nineteen eighties, Kulzer and Montgomery [10] addressed the problem of providing a single network for all telecommunications services and speculated that a statistical switching technique could be employed for the purpose. Turner, addressing the same single network problem in 1986 [11], proposed a unified network called an *information transport network* . It was envisaged that this would provide connections with a wide range of bandwidths from a few bits per second to 100 Mbit/s or more, transport for both bursty (sporadic) and constant bit rate sources and would be implemented using fast–packet switching techniques.

Such a network, referred to as a *multi–service network*, was proposed by Gallagher of British Telecom in 1986 [12] and would "accommodate the uncertain requirements of future public communications" by providing a "network with a high degree of flexibility". Again, packet switching this time in the form of a ring structure (the Orwell ring) was proposed as an implementation.

Both of the previous terms for the unified network, 'information transport network' and 'multi–service network' have since fallen out of use, to be replaced by the term

Broadband Integrated Services Digital Network, or B–ISDN (Note1).

The B–ISDN is seen as providing a long–term, flexible and future proof network able to cope with a wide range of service requirements by exploiting the current rapid advancement of technology and much research has been done on the evolution towards this single network [13][14][15].

It has recently been agreed within CCITT [16] that ATM, asynchronous transfer mode, will be the target switching and transmission technique (carried in an SDH structure) for the B–ISDN and it is an appropriate point to introduce ATM in more detail.

## 1.4 ATM the Solution

ATM, previously also referred to as ATD (asynchronous time–division multiplexing) and fast–packet switching, is the universal transfer mode proposed for the B–ISDN.

In the ATM technique the information to be transferred is split up at the entrance to the network into fixed length blocks. A header is added to form the complete cell, and this is then launched into the network. As the header contains the identity of the source of the data, a *time structure* such as that shown in figure 1.5 is no longer needed and a user may send data in the form of cells into the network at any rate from zero up to the maximum bandwidth limit. It is probably this feature, called *bandwidth on demand*, that has done more than any other to sell the ATM technique to the world telecommunications industry.

The ATM network would need the capacity to switch an enormous number of cells to allow the principle to work. This in turn implies fast transmission systems and large cell switches.

(Note1). The synonymous term IBCN, Integrated Broadband Communications Network is also used to refer to the B–ISDN.

ATM integrates synchronous (constant bit rate) and asynchronous (variable bit rate) traffic by partitioning the information into small, fixed length blocks called ATM cells. By combining this principle with fibre optic transmission, more efficient source coding techniques and VLSI hardware technology, a wide range of services with bandwidths from a few kbits/s to several hundred Mbits/s can be switched reliably and economically using only simple protocols within the network. By using packet switching techniques, ATM can provide the customer with various "quality of service" classes allowing greater choice particularly in the area of tariffing. In addition, many new services could utilise the asymmetric flow of information capability (simplex paths, section 1.2.2 above).

### 1.4.1 ATM Cell Structure and Header Functionality

The ATM cell as currently agreed [17] is a fixed length, 53 bytes long (424 bits) comprising 48 information bytes and 5 header bytes. As a general rule, the user may fill the information bytes in whatever format they choose providing this conforms to the appropriate adaptation layer specification. The format of the cell header at the UNI (User–Network Interface) is shown in figure 1.6. It is noticeable that the header has only limited functionality, largely due to the "connection–orientated" nature of an ATM network.

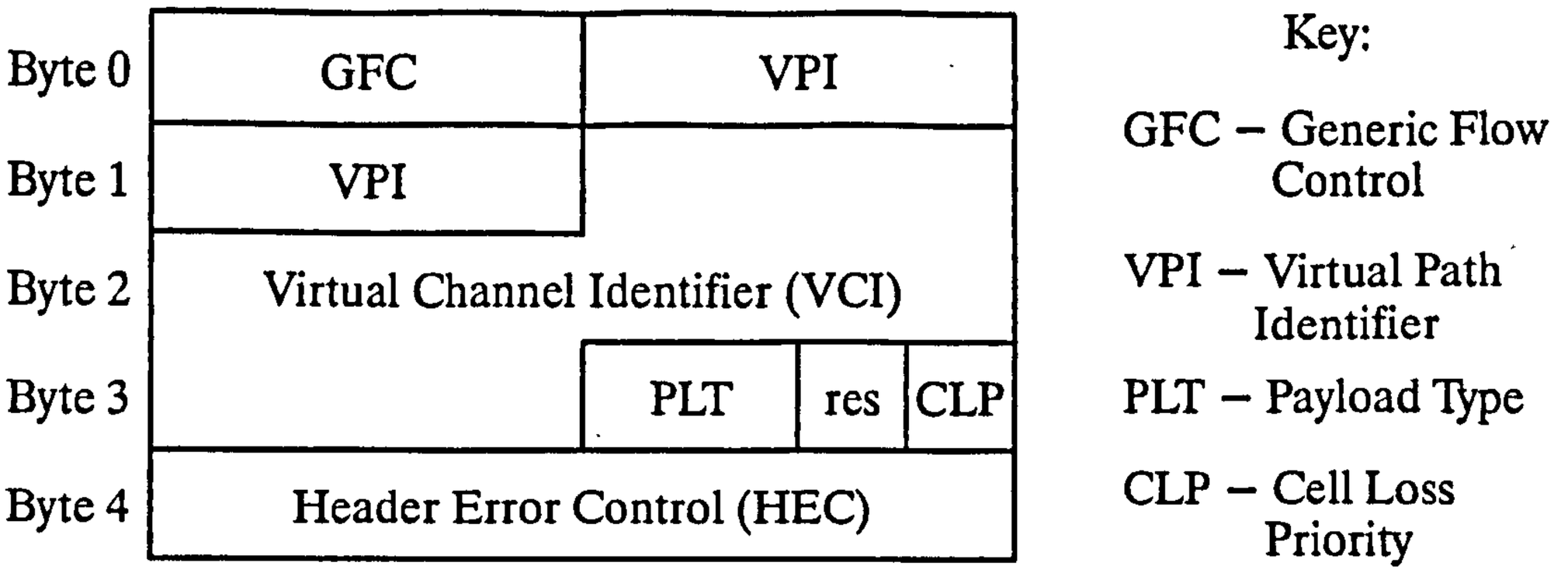



Figure 1.6: ATM Cell Header Format at UNI

The format of the cell header at the NNI (Network Node Interface) is slightly different to that at the UNI in that at the NNI the GFC (Generic Flow Control) field does not exist and the bits are used to extend the VPI (Virtual Path Identifier) field. The 48 user bytes follow the header and are unchanged by the network.

In synchronous systems, the source of the information is determined by the position of the information in the time domain (time–division multiplexing channel structure). However in the ATM system, the source of a cell is uniquely identified by information carried in the cell, the VPI and VCI (Virtual Circuit Identifier) fields in the 5 byte header. The VPI identifies unique paths, or conceptually ATM pipes, through the network. These paths, representing semi–permanent connections between two end points, are not set up on a dynamic basis but may be updated online. The VCI uniquely identifies the channel dynamically allocated to that connection for the duration of the call, and it is worth noting that two different components of the same service, voice and video for example in a video–phone connection, could use two different VCIs. This would enable different priority treatment (time or semantic) to be applied to the different components, and also open–up the possibility to add

or remove components during the call, for example changing from voice, to voice plus video.

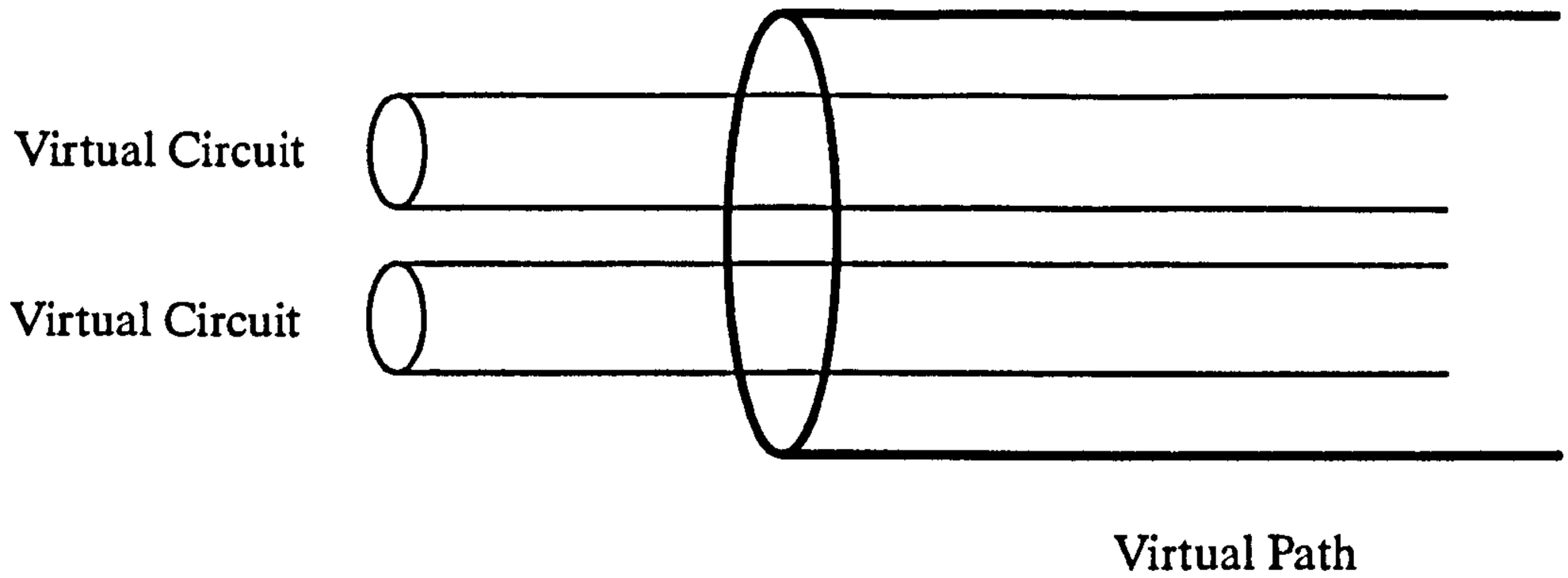



Figure 1.7: Virtual Paths and Circuits Concept

Figure 1.7 shows how virtual circuits are carried in virtual paths where a large number of virtual paths may be carried on one, high speed, transmission system.

The header error control covers the whole header, and provides some protection for the cell header against bit errors which could cause mis−routing and thereby corrupt other VCs. PLT (Payload Type) and CLP (Cell Loss Priority) fields provide the network with essential information to allow efficient cell handling. Priorities other than that specified by the CLP field may be negotiated at call set−up time on an individual VPI/VCI basis. This has the advantage that no fixed definition of priorities has to be made by the network designer. Two possible priorities include time and semantic. Time priority would be used on a service where delay must be kept to a minimum, semantic priority where information loss or corruption must be kept to a minimum.

### 1.4.2 Basic Functional Model of an ATM Switch

Whilst the implementation of a large ATM switch may be highly complex, the basic functional model is relatively simple, as shown in figure 1.8.

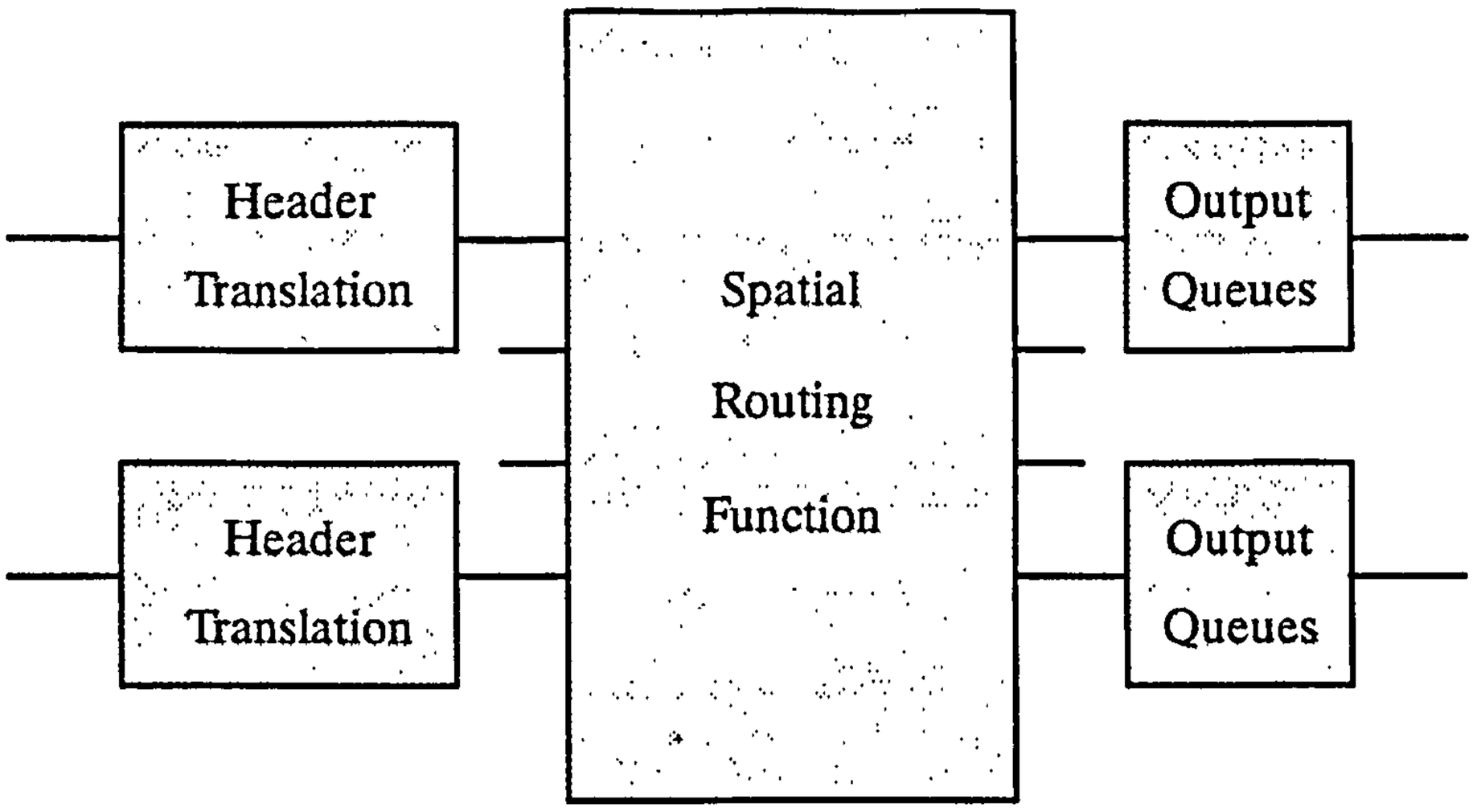



Figure 1.8: Basic ATM Switch Functionality

In a synchronous switch, the two fundamental operations are time–switching and space–switching. By contrast, in an ATM switch they are address–switching and space–switching. In addition, due to the non–predictive nature of arrivals of cells, some queueing is required to cope with peaks in the offered traffic.

The address–switching, or header translation (Note1) function is the connection–oriented part of the switch, exchanging a valid incoming VCI/VPI combination for the outgoing values, preset in tables at call set–up time. The function of cell–policing is also performed at this point, preventing cells from unknown VCI/VPI combinations entering the switch and preventing a single source exceeding its allocated bandwidth. (The policing function is described in more detail in section 2.4.) If an internal header is to be added to the cell to facilitate routing across the switch or to contain additional error check data, this is also done here.

The spatial routing function performs the space–switching which is the ability to transfer a cell from any incoming multiplex to any outgoing multiplex. Also called

(Note1). This function has other names including header decode and address decode.

port–switching, this function has many subtle constraints such as incurring as small a delay as possible, keeping the delay from cell to cell as constant as possible, keeping the sequence of cells intact and to lose as few cells (due to hardware failure, lack of resources, cell header corruption etc) as possible.

The output queueing function resolves the contention between cells for the same resource. The contention is caused by the occasional inability of the outgoing multiplex to transmit cells as fast as it is receiving them from the spatial routing function. This is achieved by the temporary storage of the waiting cells in output queues often referred to as buffers.

### 1.4.3 Virtual Path/Virtual Circuit Switching Concepts

Section 1.4.2 introduced the concept of address switching in ATM networks. The exchanging of VCI/VPI combinations allows flexible switching functions to be applied to cells under software control.

If a number of circuits are to be "held" together within their path it is only necessary to write a new VPI value into cells within the path. This is referred to as virtual path switching and may be employed to provide ATM pipes through the network.

Figure 1.9 shows the difference between virtual path switching and virtual path plus virtual circuit switching.

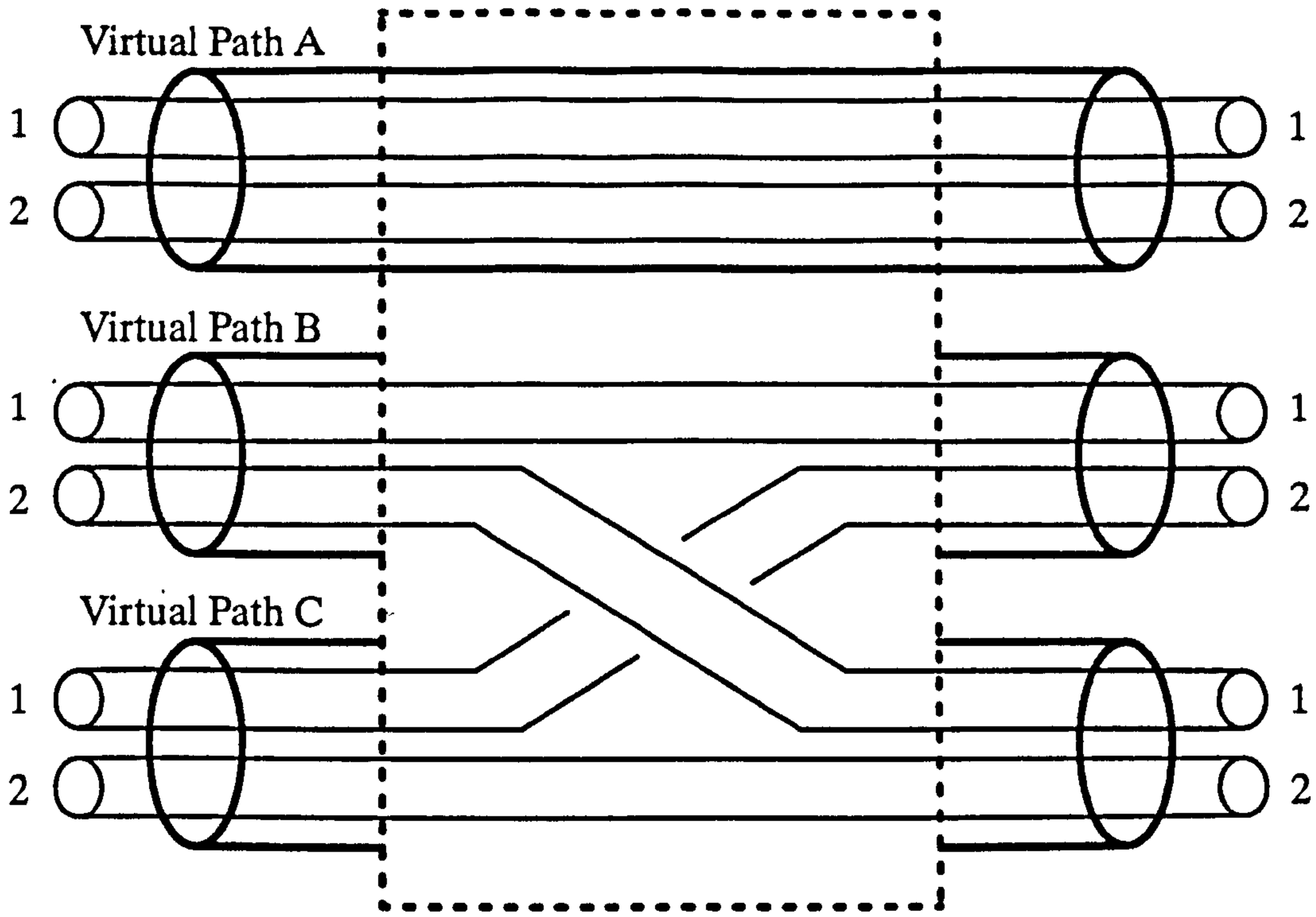



Figure 1.9: Virtual Path and Virtual Circuit Switching

Virtual path A is virtual–path switched to the output with all of its circuits intact. Circuits within virtual paths B and C however are circuit switched, that is some cells have both their VCI and VPI values in the header exchanged for new values.

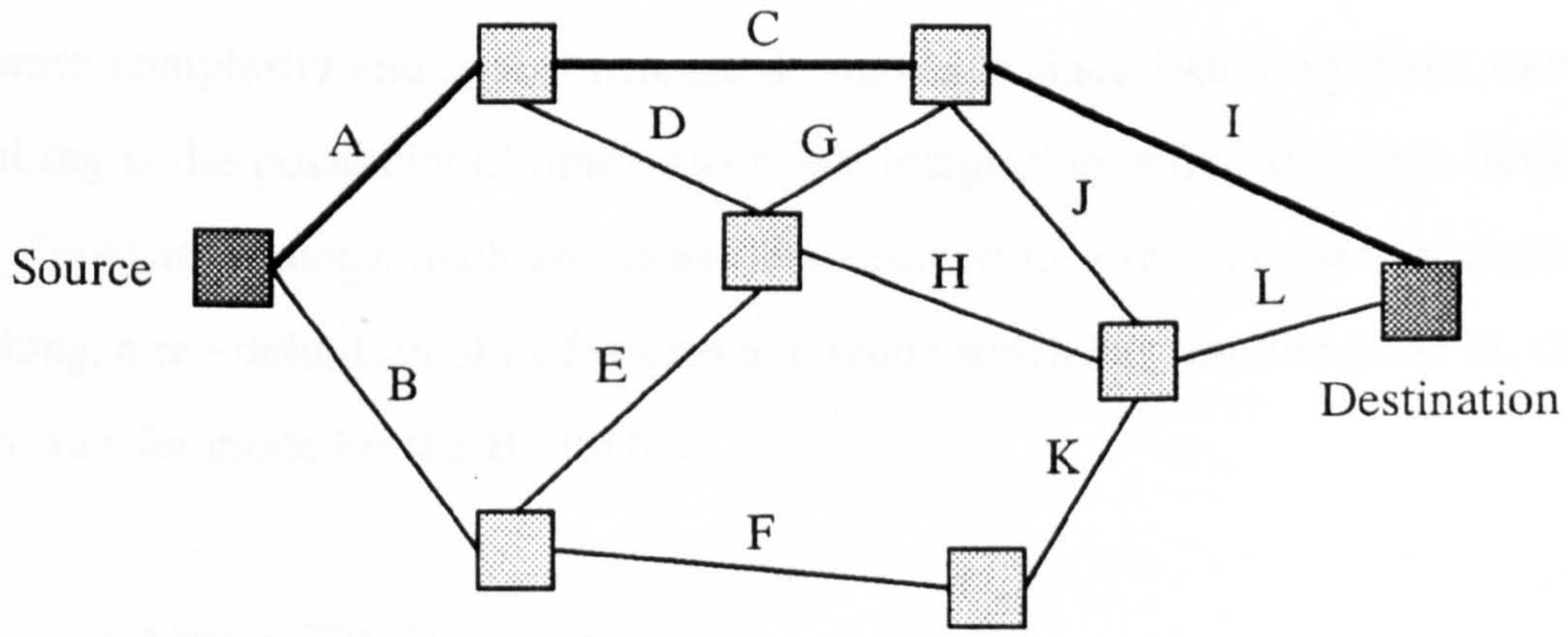
Because the path and circuit switching functions are defined by software parameters and not by hard–wiring transmission systems as at present, this "dual–level" switching technique allows for very flexible networks to be configured and changed as services and requirements for bandwidth evolve.

### 1.4.4 The Difference Between Packet Switching and ATM

Conventional packet switching as we understand it today was first devised in the early nineteen sixties by the RAND Corporation as a spin off from their work on military

communications systems. The concept, based on the requirements for the military systems, was that a network would still be secure and able to carry traffic even in the event of partial failure. To achieve this, the network architecture formed a mesh structure as shown in figure 1.10 in which many alternative routes between 2 points existed. The information, in this case digitised speech, was transmitted in short, separate bursts called packets.

The technology available at the time largely governed the performance and functionality of the PSN (packet switched network). Relatively high bit error rates (by today's standards) of $10^{-6}$ implied error control on each link in the network to achieve satisfactory end−to−end performance. The limited amount and slow operation of memory available for storage in the switching nodes meant the PSN had to be able to control the traffic flow entering the network with a flow control protocol. To guarantee dynamic alternative routing in the event of failure without having data about each call in every switching node in the network, complete source and destination address information had to be contained in each packet.




1st choice route: A−C−I

Many alternatives exist in event of 1) Link Failure
2) Node Failure

Figure 1.10: Mesh Topology of a Packet Switched Network showing Alternative Routing

These 3 characteristics, link by link error control, flow control and complete address information, plus the additional feature of variable length packets, implied further restrictions on the services using the PSN. The complex protocols increase the per packet processing requirements and therefore also the delay incurred per packet. The delay is further increased by the buffer control required to handle variable length packets which due to complexity had to be implemented in software. The individual and independent nature of the address information carried in each packet implied that packets of the same channel could get out of sequence between source and destination by taking different routes through the network. These restrictions meant that while the PSN was very good at carrying low speed data, it was very poor at carrying real time services such as speech.

The constraints imposed on the PSN would remain until technology provided high speed processing and high speed, reliable transmission links. Recent advances in optical technology have seen major improvements in the 4 key transmission parameters bit rate, distance, quality and reliability [18] while advances in silicon bi−polar and GaAs electronic devices with sub−micron geometries have seen hardware complexity and speed increase dramatically since 1980 [19]. Even more promising is the possibility of opto−electronic integration within the same device using GaAs technology. Such advances have resulted in a re−appraisal of packet switching, a re−definition of its features and scope which has resulted in ATM, the target transfer mode for the B−ISDN.

## 1.5 The Aims of This Research

In the last 5 years or so, the B−ISDN concept has been refined and technological advances have made fast−packet switching, or ATM, theoretically capable of fulfilling the requirements of the transfer mode proposed for the B−ISDN.

Thus, ATM has proved a large and fruitful area for research in universities, telecommunications equipment manufacturers and especially within the RACE (Research into Advanced Communications in Europe) projects funded by the EEC.

As a result, there are currently many concept architectures capable of switching ATM cells as described in section 1.4.2 above. Indeed there are far more prototypes and demonstrators of ATM switches in research laboratories than could ever be hoped to finally emerge as large–scale production switches.

The broad aim of this research was to develop, assess and if possible utilise tools and methods to allow ATM switch fabrics to be modelled and therefore to further understand through a unified approach using simulation and analytic techniques how variations in the architecture of the switch block used for ATM affect its performance.

More specifically the questions posed at the commencement of this research were:

1. Is it possible to create a general purpose simulator capable of abstracting the features and properties of many different architectural solutions to the ATM switch fabric problem?

2. Do any areas of the switch block play a more important role in its performance and are therefore critical and more important to investigate?

3. How do variations in the arrival patterns of cells affect the switch block performance?

4. Is it possible to create metrics on the performance of switch fabrics that have a general applicability for both ATM switch and network designers, perhaps leading to standards for switch and network design?

5. Is it possible to create generalised analytic solutions applicable to many different architectural configurations?

6. Are there currently any unforeseen problems in an ATM network that might affect the final implementation?

7. Does the anticipated performance of ATM switches in a network actually meet the requirements of the services it would carry?

At the outset of the project definitive answers to these questions were not available and the research presented in this thesis contributes to the knowledge of these areas.

This chapter has mapped out the historical path to ATM and described ATM as a technique. The next chapter will cover in more depth those aspects of ATM relevant to the subject of this research by analysing both historically significant and recently published material from the literature.

# Chapter 2 Critical Literature Review

ATM is currently an enormous and still expanding area of research and therefore this literature review is restricted to topics closely related to the original research presented in this thesis. In particular, the following aspects of ATM have not been included; signalling, load control, call acceptance, charging policies, MANs (Metropolitan Area Networks), and the management of ATM networks.

## 2.1 The Definition of B−ISDN

As mentioned briefly in section 1.3, the B−ISDN is seen as the future telecommunications network with ATM as the transfer mode and SDH as the transmission system for that network. This distinction between B−ISDN and ATM is important and the relationship is shown in figure 2.1.

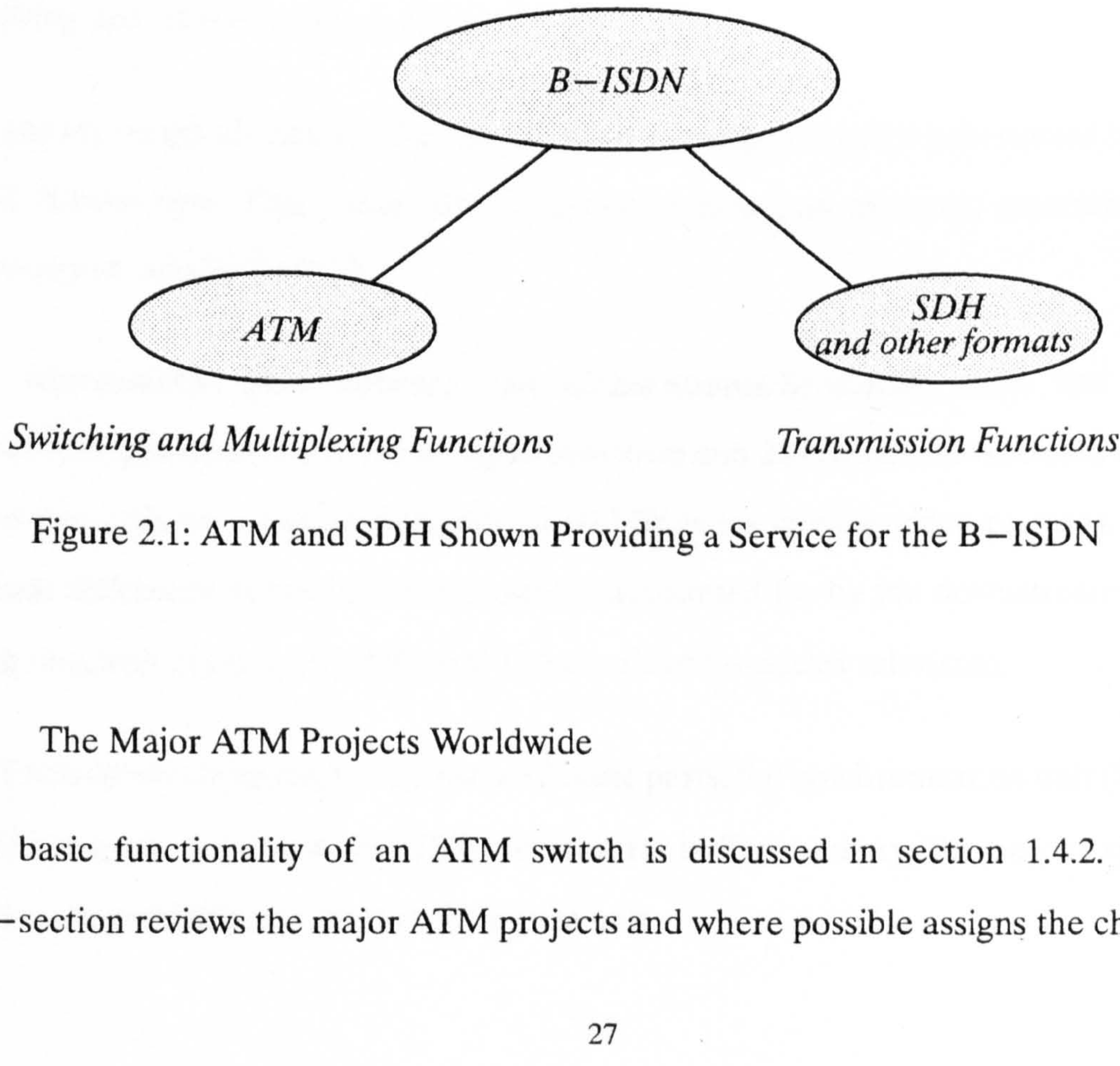



Figure 2.1: ATM and SDH Shown Providing a Service for the B−ISDN

## 2.2 The Major ATM Projects Worldwide

The basic functionality of an ATM switch is discussed in section 1.4.2. This sub−section reviews the major ATM projects and where possible assigns the choice

of switch block architecture to 1 of 3 classes depending on where the queueing or port/link contention is performed. The 3 different classes are:

1. Switches with input queues.
2. Switches with output queues.
3. Switches with central queues.

A more rigorous description of these 3 classes may be found in "Asynchronous Transfer Mode, Solution for Broadband ISDN" by Professor M. De Prycker [20].

### 2.2.1 The French Prelude Network

Prelude, as described in [21] and [22], is an experimental network of the Centre National d'Etudes des Telecommunications (CNET) in France made as a broadband ISDN using ATM techniques. The experimental network consists of components that represent all parts of a real network including subscriber terminals, multiplexors, switching and transmission equipment.

The packet length chosen for Prelude is 16 bytes, composed of 15 information bytes and 1 header byte. This packet size is quoted as giving an optimum transmission efficiency of close to 94% [23].

The transmission links between the Concentrator/Broadcast units and the subscribers' premises are 70 Mbit/s upstream links and 280 Mbit/s downstream links based primarily on a required bit rate of 60 Mbit/s for moving video pictures. The unusual difference between the two rates is accounted for by the downstream link being required to carry an additional 3 channels of broadcast television.

The Prelude switching matrix consists of 5 basic parts, the synchronisation unit (SU), the input space division stage (ISS), the central buffer memory, the output space division stage (OSS) and the switch control unit.

The SU provides the interface between the 16 incoming 280 Mbit/s multiplexes and the switch connection matrix. It is responsible for the alignment of incoming packets to the local clock (called "framing") and for compensating for the plesiochronous nature of the incoming multiplexes by clock–adapting to the local synchronisation source. Framing is achieved by holding up cells until the cell–restart signal, clock–adaptation is achieved by inserting blank packets into nominally slow streams and deleting empty packets from nominally quick ones.

The ISS, also called a super–multiplexor, converts packets (incoming from the SU) from a serial format to a parallel/diagonal and interleaved format. This is called the *paragonal* format. Each outgoing link *n* from the ISS carries octets numbered *n* from all incoming links. Octets of a particular cell appear in sequential octet cycles on their appropriate link. From this it can be seen that the SCU only needs to interface with the link carrying the header byte, the other links being processed depending on the content of the header. The switching delay of this mechanism is longer than it need be. The delay would be reduced if all octets after the header were processed in the same clock cycle instead of over the 16 octet clock cycles this mechanism demands.

The central buffer memory temporarily holds the cells, by now with translated headers, until the OSS determines that they are required for transmission on the outgoing multiplexes. The output port contention is therefore performed in this switching stage and therefore Prelude falls into class 3, switches with central queues.

### 2.2.2 The Belgian Broadband Experiment

The Belgian Telecommunications Administration (RTT Belgium) in the form of the Bell Telephone Manufacturing Company Belgium (BTM) have been conducting an experiment on ATM techniques along similar lines to the CNET experiment. As with Prelude, the experiment encompasses all components of a real network.

As described in [24] and [25], the packet length in the experiment is fixed at 16 bytes composed of 14 information bytes and 2 header bytes. The header is kept small to keep its overhead down and processing speed up. The switching nodes are interconnected with ATM trunks having an effective data rate of 565 Mbit/s. Subscriber Group Equipment (SGE) interfaces the local distribution network with the switching nodes which allows efficient transportation of common information such as broadcast television to a point close to the subscribers' premises. One SGE can serve 16 broadband subscribers, would be located within 1 km of them and is connected to the switching node by 1 unidirectional 565 Mbit/s link carrying unswitched distributed television and radio channels, and 1 bi–directional 565 Mbit/s link carrying the switched information. The lack of security of this configuration, i.e. relying on a single link, gives a total loss of service in the event of link failure.

Between the SGE and the subscriber there is a 565 Mbit/s downstream link and a 70 Mbit/s upstream link. The bandwidth of the downstream link is over twice that of the Prelude network.

The switch fabric, called the broadband switching network (BSN), is composed of broadband switching elements (BSEs) and is a self–routing switch achieved by the addition of a routing tag to the packet. The BSN is a multistage network using a number of BSEs in several stages. An important point is that the BSEs are bi–directional implying that a route from A to B has a fixed path in the reverse direction from B to A. This is wasteful of resources where unidirectional, or broadcast, traffic predominates.

It is necessary to conceptually *unfold* the bi–directional switch, giving a unidirectional switch of twice the size, to understand how it works. The unidirectional switch is divided into 2 halves around what is called the mirror plane. The routing tag

added to the cell at the input is divided into 2 parts, one part to specify the route from the input to the mirror plane, the other to specify the route from the mirror plane to the output. Setting up a switch path is achieved by selecting and setting up a random path through the first half of the switch (called the *randomisation network*) then setting up a fixed path from the mirror plane to the appropriate output through the second half of the switch (called the *routing network*). There are many routes through the randomisation network which helps spread the load across all BSEs and allows the switch control to try a new path when congestion is encountered. The provision of queues within the BSEs gives a non−zero probability of cell−loss due to queue overflow.

### 2.2.3 The ATT Bell Labs Knockout Switch

The Knockout ATM switch as proposed in [26] is a self−routing, low−latency (low delay) non−blocking high speed packet switch falling into the category of switches with output queues. The non−blocking property is given by the fully interconnected nature of the switch fabric, i.e. each input has a path to each output. This is achieved by broadcast buses in which each bus is driven by one input port. Each output port is connected with each bus via a bus interface. As several cells may arrive at the bus interface simultaneously, it contains concentration stages and cell queues. The name "Knockout" comes from the concentration stages in which cells compete in a knockout−tournament fashion for the output port. In this structure however, cells "losing" are given a further chance in subsequent rounds. This introduces a certain amount of cell−loss. At 40 queue places, with an input load of 85% the cell−loss is quoted as $10^{-8}$.

### 2.2.4 Siemens AG Scalable ATM Switch

An interesting ATM switch design has been proposed by Dr Fischer, Hr Fundneider and Dr Göldner [27]. The concept includes full details of the integration with Siemens synchronous switch, EWSD. The ATM switching part is based around an ATM Switching Element (SE) which switches cells between the 16 input and 8 output ports. The port speed is 155 Mbits/s plus an allowance for the internal message header added to the front of each ATM cell for routing purposes. A functional block diagram of the SE is shown in figure 2.2.

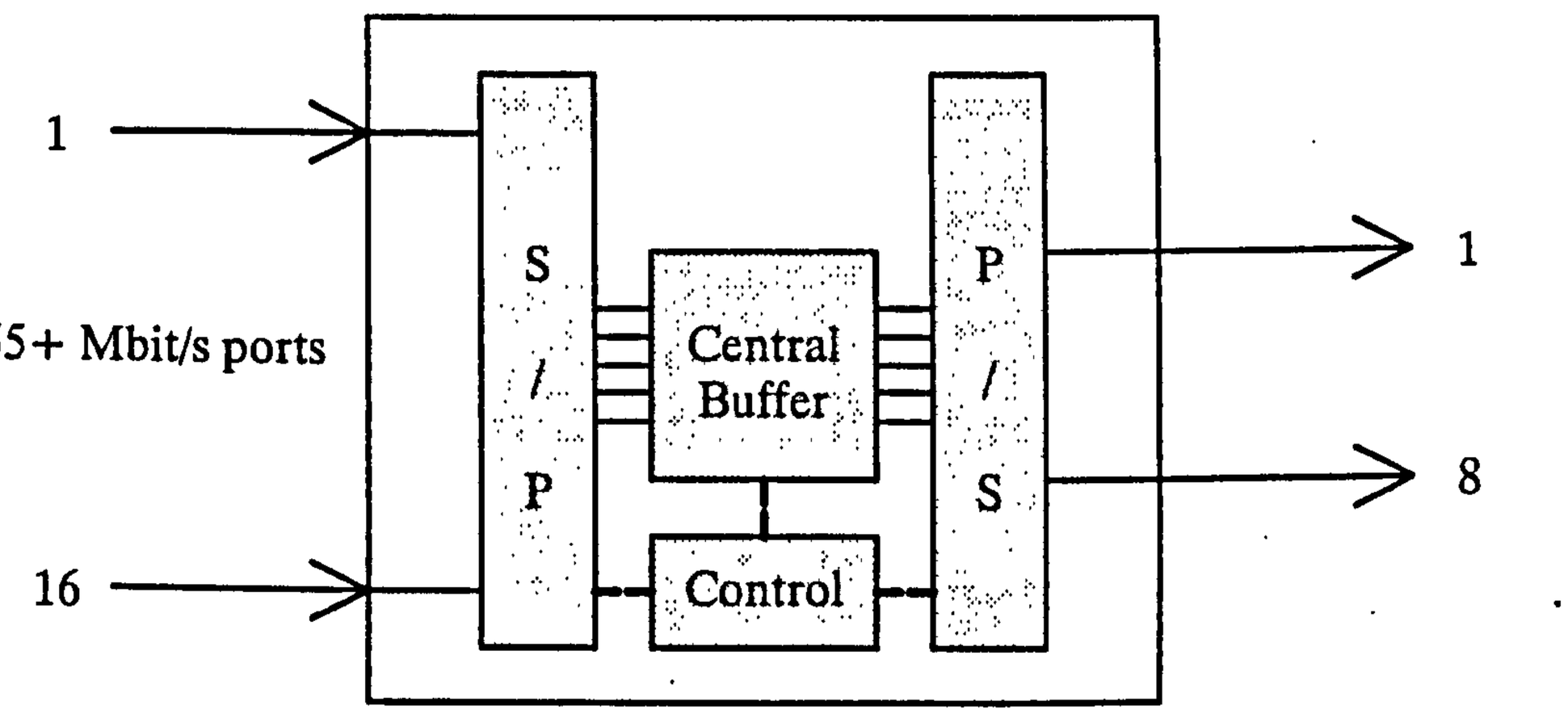



Figure 2.2: Functional Block Diagram of Siemens' 16 x 8 ATM Switching Element.

The SE is implemented on a single CMOS chip of unspecified size and geometry. The 16 incoming ports are converted from serial to parallel format due to speed constraints of the technology. The first stage of the switching is performed by writing cells into the central buffer memory. The control function holds logical queues in the form of cell addresses within the central buffer. The switching is completed by reading the cell from the central buffer at the appropriate point in time and writing it out to its destination output. The cell format is changed back to serial before output. This

concept mixes output queueing with central shared buffers and is therefore claimed to give optimum performance.

Twelve 16 x 8 SEs as described above are connected together to form an ATM Switching Module. The ATM switching module has 32 off 155+ Mbit/s inputs and 32 outputs of the same speed. A functional block diagram containing one quarter of the SE functionality is shown in figure 2.3.

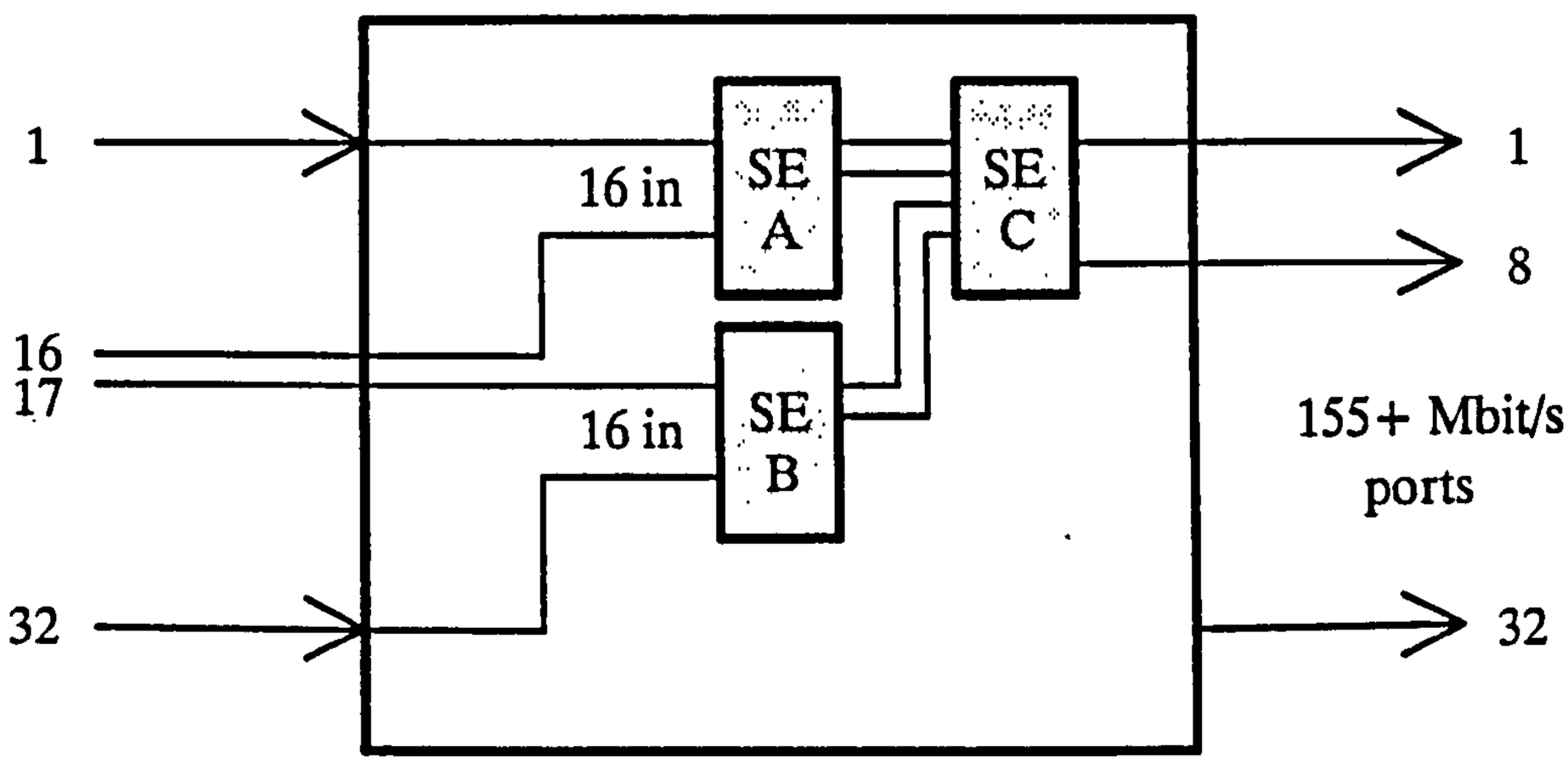



Figure 2.3: Functional Block Diagram of Siemens' 32 x 32 ATM Switching Module

The 12 SEs are grouped into 4 groups of 3 connected in a funnel arrangement as shown. SEs A and B each receive 16 of the 32 inputs. They each output 8 links to SE C giving it 16 inputs in total. SE C in turn outputs 8 of the 32 links. To provide full interconnectivity each input is additionally fed to the 3 other groups of 3 SEs in parallel. The SEs in the first stage of the funnel only accept cells with an appropriate header value. The 32 x 32 ATM Switching Module can be cascaded with others into multi–stage arrangements to produce a wide range of switch sizes.

The principal limitations with this design are the lack of point to multipoint (broadcast) capability and the lack of a "delay priority" facility to allow cells of a

delay–sensitive service to be promoted to the head of the output queues. Additionally the multistage structures as presented would appear to block once the basic switch size of 32 x 32 155 Mbit/s ports is exceeded.

## 2.3 Switch Architectures and Switching Theory

In this sub–section various classic switch architectures are reviewed and their merits and drawbacks when applied to ATM switching are discussed. Firstly, Hr Clos's 1953 paper on non–blocking switching networks is examined. Following this, Benes, Banyan (and derivatives) and cross–bar networks are reviewed. Their application in switching technology today is highlighted.

### 2.3.1 Non–blocking Networks

A major milestone in the studies of switch architectures was the publication of a paper by Charles Clos in 1953 on his investigations into non–blocking switching networks [28]. The paper presented the derivation of generalised formulæ to find the minimum number of crosspoints required for the interconnection of N inputs to M outputs with no blocking and the architecture required to do this. The motivation was to keep the number of cross points required to a minimum due to their relative expense compared to the common control equipment.

"Blocking" used in this context is defined as the inability of the switch fabric to interconnect two otherwise free ports due to no mutual, free links between the two ports being available.

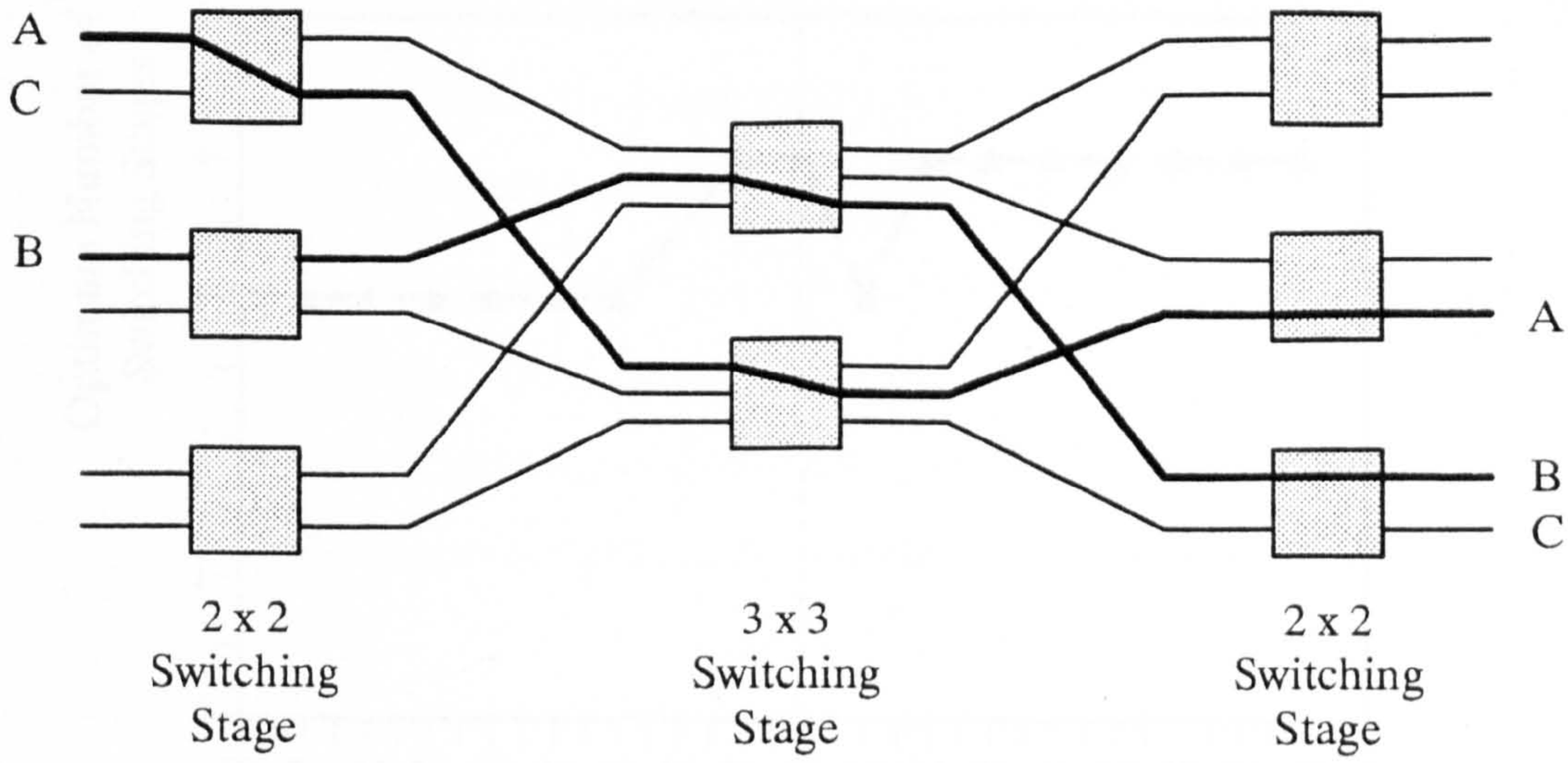
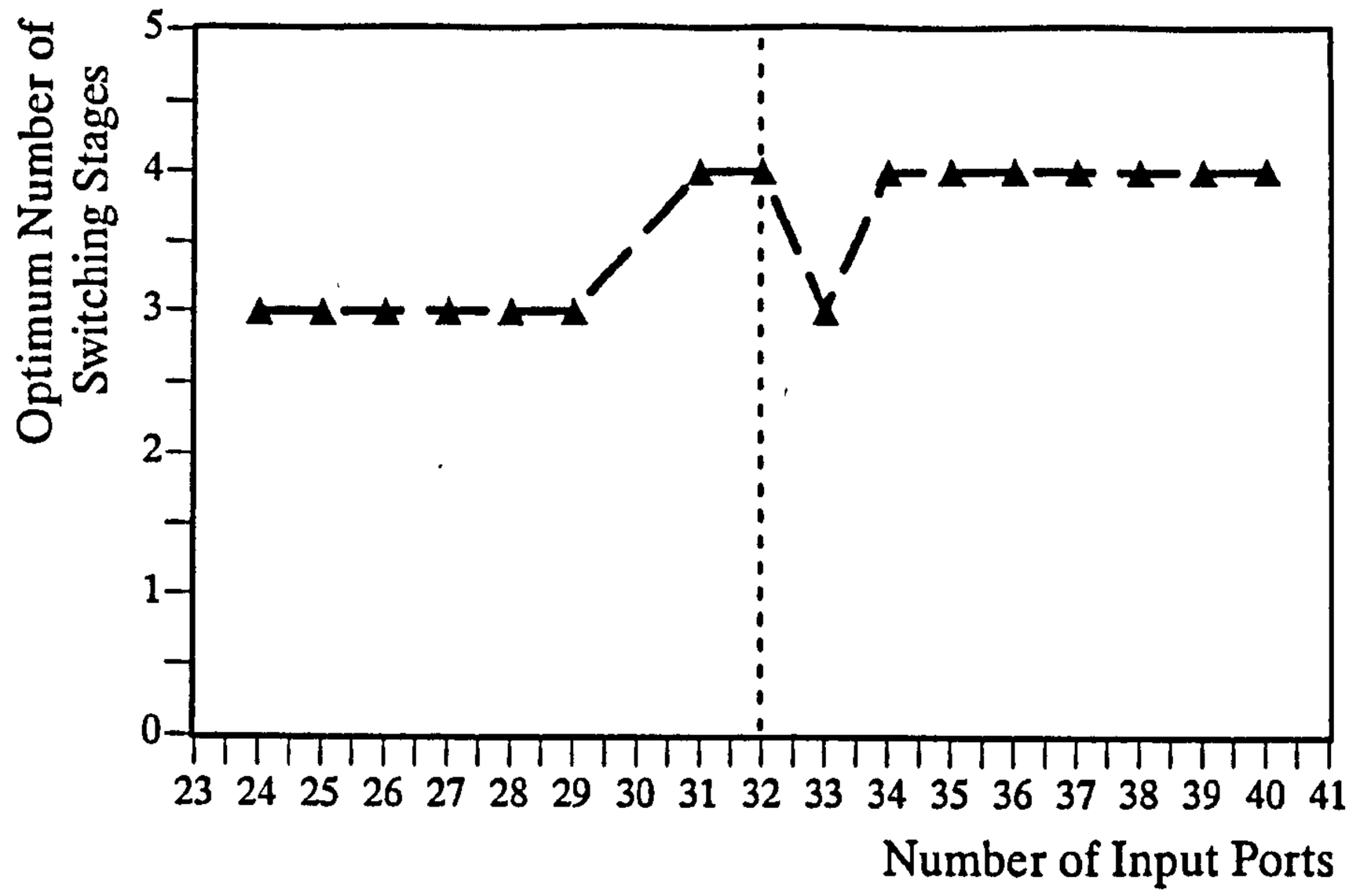



Figure 2.4: Blocking in a 3−stage Switch Architecture

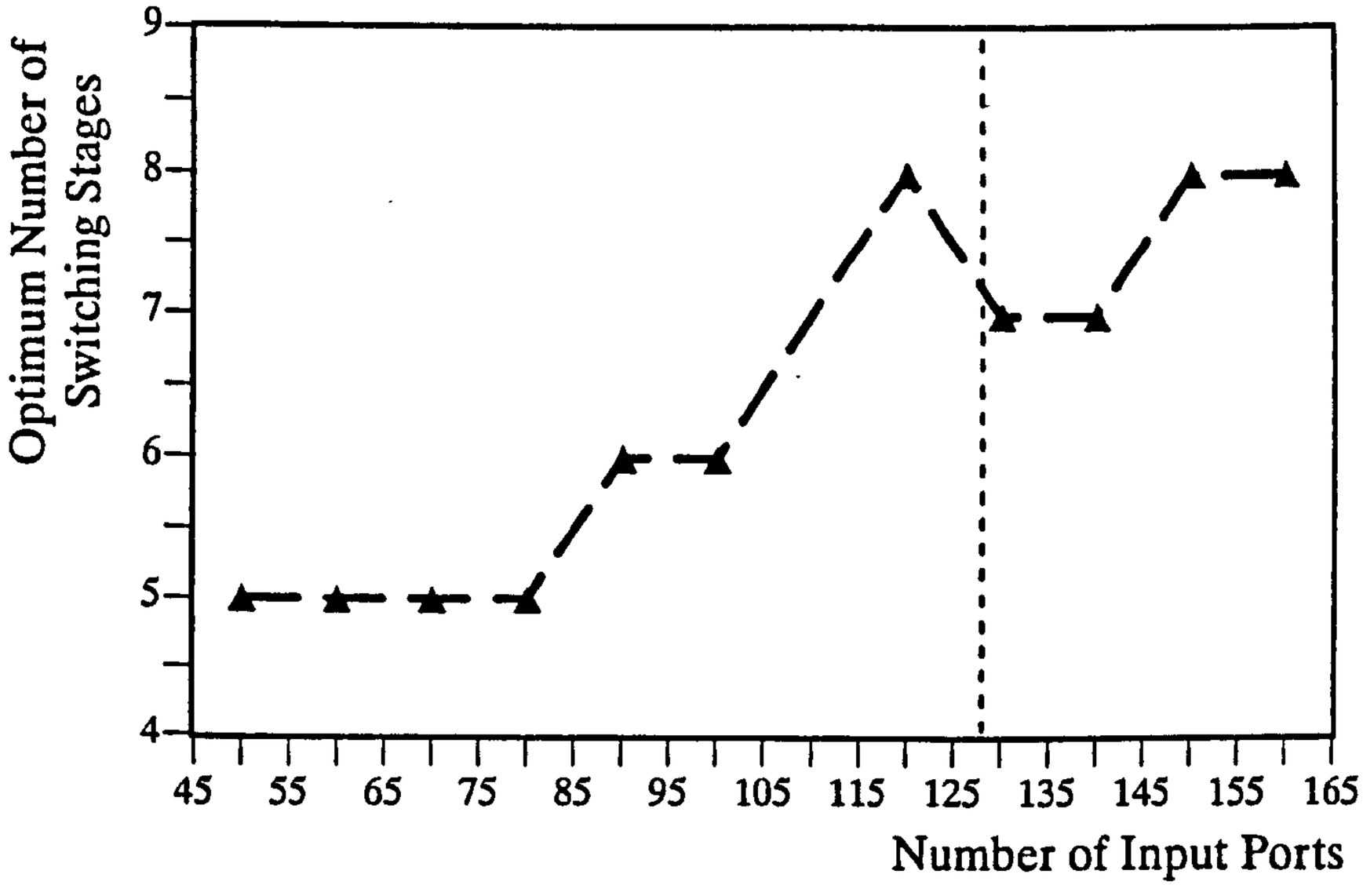
This is shown in figure 2.4. Six central routes are provided, 3 per central switching stage. With only 2 out of the 6 paths across the switch busy, A to A and B to B, it is not possible to connect C to C and therefore the connection in this case is *blocked*.

The method of derivation of the minimum number of crosspoints formulæ was to analyse the worst case conditions for a 3−stage architecture, and dimension a system to cope with those, assuming that the number of 1st stage switches was $N^{1/2}$ where N = the number of inputs. Then, a 5−stage architecture was dimensioned to worst case conditions based on a 3−stage architecture in the centre, with 2 more stages added, one on the input side and one on the output side. From this followed derivations of formulæ for 7 and 9 stage architectures, and then generalised polynomial forms for $s$−stages where $s$ is any odd integer.

The formulæ were first derived for the specific case where the number of first stage switches was equal to the $(s+1)/2$ th root of N and then more generally to provide a lower minimum−number of cross−points.




Graph 2.1: Optimum Number of Switching Stages for 23 to 41 Input Ports




Graph 2.2: Optimum Number of Switching Stages for 45 to 165 Input Ports

Graphs 2.1 and 2.2 above show the optimum number of stages in a switch architecture giving the minimum number of cross points for two ranges of the parameter, number of input ports. The points are derived from the paper, and it is interesting to note that the two dips in the graphs occur at 33 and 130 input ports. These dips are not discussed

in the paper, but it may or may not be significant that they occur just after the two "powers of 2" values, 32 and 128. The same dip does not occur however at 64 inputs. Most switch designs today use powers of 2 values as their basic "magic" numbers due to ease of implementation in hardware and software, and thus designers recognise these as both targets and limits for the number of input ports. It is also interesting to note that Hr Clos chose initially to divide his inputs into square, cube, etc. roots to derive his general formulæ rather than into powers of two as would probably be done today.

The paper was written in the days of analogue switching, and thus no delay (apart from electrical propagation delay) was introduced with each added stage. With digital switching, the time domain must also be considered, and it is recognised that adding more stages into a switch architecture, even if these are simple space–switching stages, adds to the overall delay. This factor is not included in the calculations. Also, one of the major limitations in switch architecture design today is getting the far higher bandwidth around the switch network, and in particular on and off a single SIU (slide–in–unit). The number of physical inter–connections, due to the space they occupy in addition their cost, becomes significant.

This paper was of major significance being the first to yield a mathematical derivation of the minimum number of cross–points required given any number of input and output ports. The significance is somewhat reduced today in that while the formulæ still hold true, and are often referred to, factors other than minimising the number of cross points must be taken into account when undertaking switch design.

### 2.3.2 Benes Switching Networks

Benes networks, as proposed by V.E.Benes in [29] were intended to solve the problem of providing a switching network capable of connecting any input port to any output

port with minimum cost in terms of cross points per inlet port. Benes networks are composed of square switches of the cross bar type. The main disadvantage of Benes networks is that they fall into the class of rearrangeable networks, i.e. the non−blocking property can only be achieved by rearranging the calls in progress. The rationale behind this rests on the premise that small, rearrangeably non−blocking networks are an improvement on both small, blocking networks and on large, non−blocking networks of the Clos type.

Even with this disadvantage, Benes networks still have a place in the forefront of technology today. In [30], Duthie and Wale describe a state of the art (1991) single chip optical switch in lithium niobate providing 16 x 16 switching capability using a Benes switch architecture.

In [31], Benes presents a mathematical proof that the number of central stage 2 x 2 switches required to fully interconnect $n$ input ports may be reduced from $2n-1$ to $3n/2$ in his 3−stage rearrangeable architecture. In [29] Benes does acknowledge that factors other than a minimum number of cross points per inlet port must be taken into account when considering various switch architectures, e.g. the placing and cost of any common control equipment required.

To date, with the exception of [30] in which ATM cells could be carried in the optical domain (although not proposed to be used in that way) Benes networks have not yet found an application in ATM switching and interest in them remains only due to their historical significance.

### 2.3.3 Banyan Switching Networks

Banyan networks however have been proposed for use in ATM switch architectures. A Banyan network is a self−routing network having the property of exactly one route

through the switch architecture from any input to any output. First defined by Goke and Lipovski in [32] they may be made up from $n$ x $n$ identical switching elements such that the destination identity also serves to route the information through the multi−stage network. Delta networks form a sub−class of Banyan networks in which a rectangular array of M x M input and output ports are interconnected using identical $n$ x $n$ switching elements in $k$ stages such that:

$$M = n^k$$

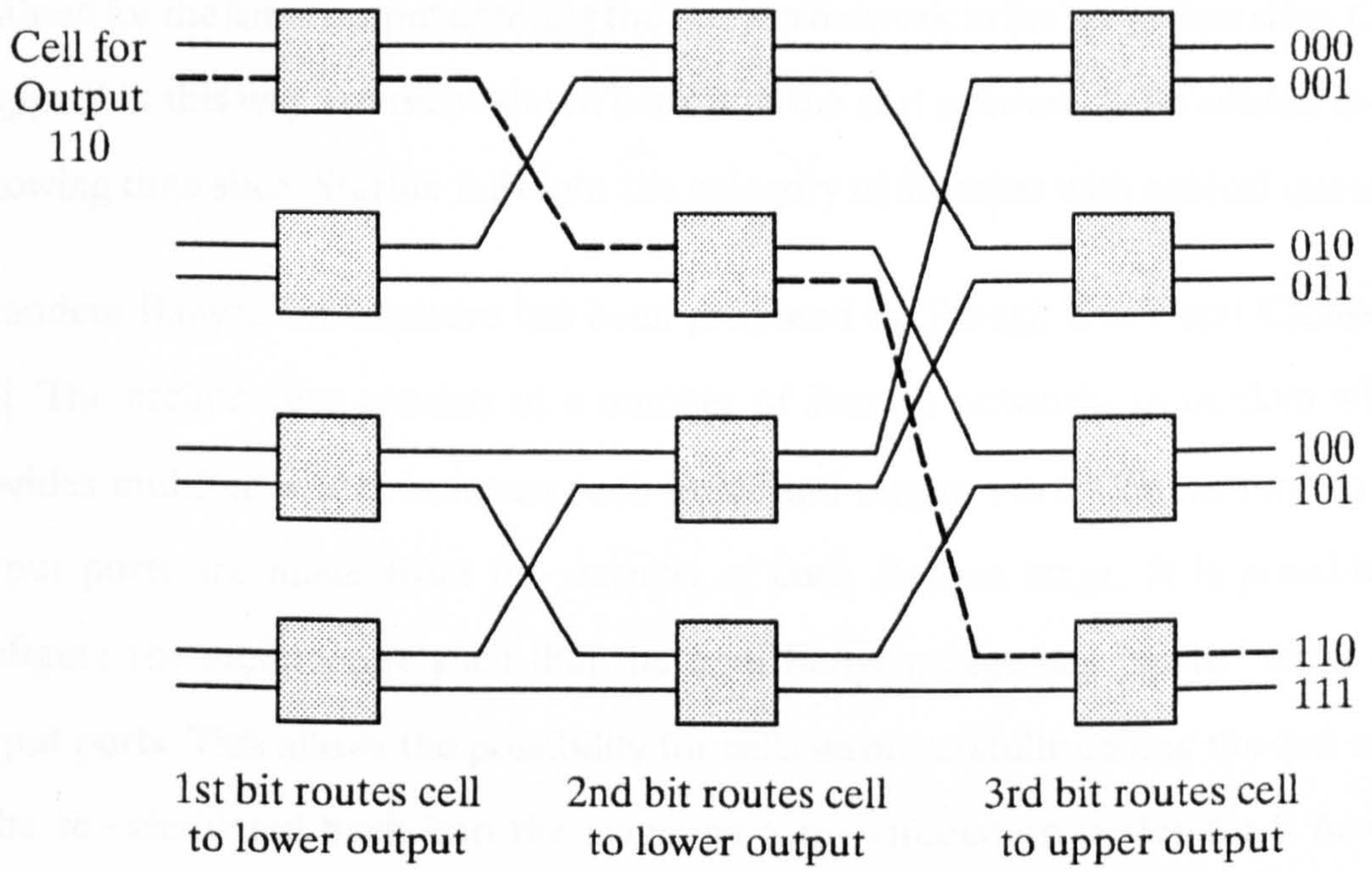



Figure 2.5: An 8 x 8 Delta−2 Network of the Banyan Class

Figure 2.5 shows an example of an 8 x 8 Delta−2 network with the routing for cell destined for output 110 highlighted.

When applied in an ATM switching application the principal problem with the Banyan class of switch, apart from the common performance constraint of output port contention, is *internal link blocking*. The problem is caused by the single−route property and is solved by providing temporary storage in the intermediate stage switching elements as well as in the final stage switching elements. This type of switch

is known as a buffered–Banyan and leads to increased memory requirements, delay and potential for further cell–loss.

The performance of the Banyan class of networks for ATM applications can be improved by preceding them with a sorting network called a Batcher network. These were first defined by Batcher in [33]. The sort–Banyan, or Batcher–Banyan, switch architecture forms the basis of the Starlite ATM switch [34]. This switch employs an additional trap network at the exit of the sort (Batcher) network to prevent 2 cells destined for the same output entering the Banyan network in the same time slice. Cells 'trapped' in this way are recirculated back into the sort network to be retried in the following time slice. Starlite falls into the category of switches with central queues.

A tandem Banyan architecture has been proposed by Tobagi, Kwok and Chiussi in [35]. The architecture consists of a number of Banyan networks in tandem which provides multiple routes between each input and output port. Connections to the output ports are made from the outputs of each Banyan stage. It is possible to configure the architecture such that the final Banyan stage has "spare" input and output ports. This allows the possibility for cells unsuccessfully exiting the 3rd stage to be re–circulated back into the stage via a re–circulation buffer for a further switching opportunity.

### 2.3.4 Cross–bar Switching Networks

This type of switching network is briefly described in section 1.2.4. The principal advantages of the cross–bar architecture are its non–blocking nature, its relatively simple hardware implementation and simple control requirements. The draw back of the cross–bar architecture is that the switch fabric size grows with the square of the number of input ports and is therefore generally only used for small systems or as a modular component of a larger, multi–stage system.

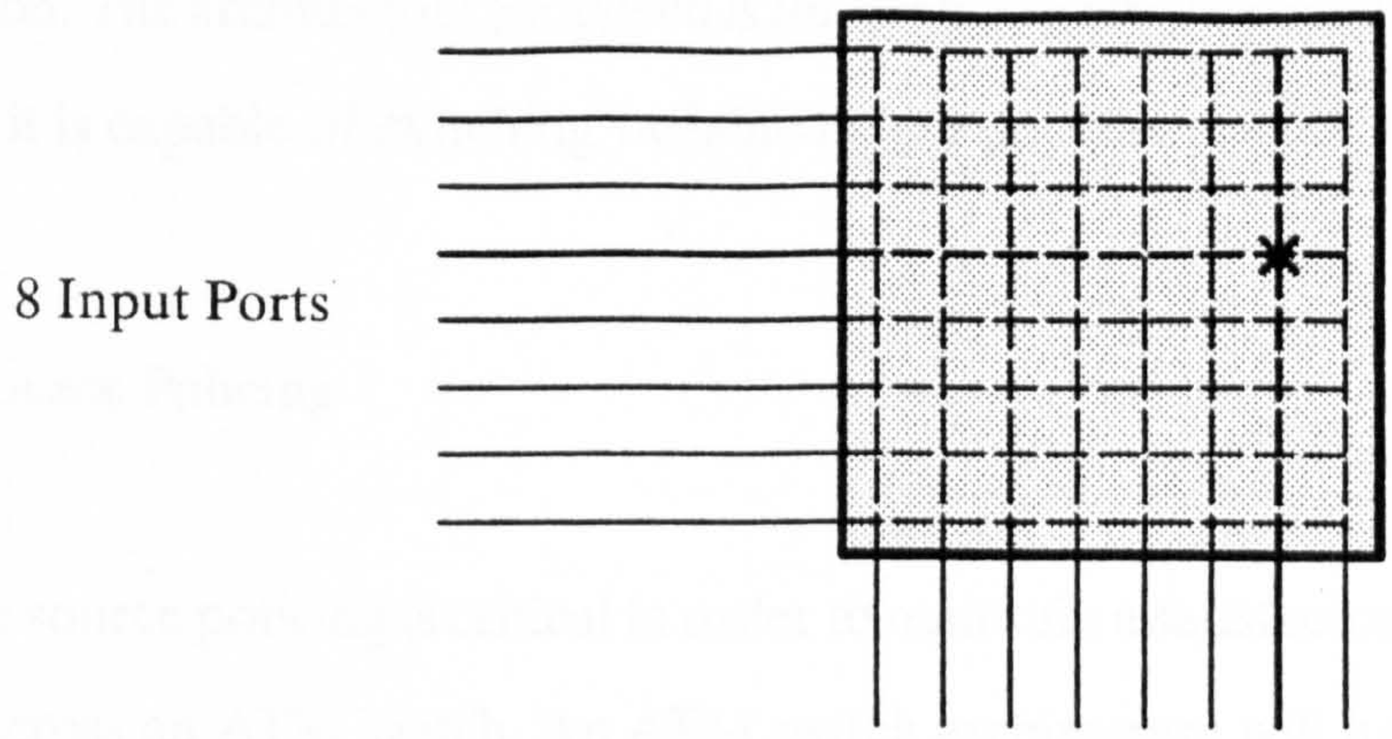



Figure 2.6: 8 x 8 Cross−bar Switching Network

Figure 2.6 shows the principle with an 8 x 8 network having 64 cross points. The connection between input 4 and output 7 has been made. The inefficient use of resources is clear from the figure. Fourteen cross points have now become unusable, 7 on the horizontal input port and 7 on the vertical output port. However, point to multipoint connections with full availability are a feature of this architecture.

This old−fashioned, basic architecture has been proposed as a basis for ATM switching. While it is non−blocking the problem of output port contention still exists. This may be solved in one of three ways, queues at the input ports, at the matrix cross−points or at the output ports. An excellent summary of the three possibilities, along with details of and references to many other configurations of switch architecture is given in [36]

In [37] Lizcano, Chas and Jimenez quite correctly comment that the new, novel shared buffer type architectures are not entirely suitable for some applications including voice due to their sensitivity to bursty traffic. The solution presented is a cross bar architecture called a Matricial Packet Switch. Output buffers are provided to resolve

contention. The architecture proposed is for general packet switching, not specifically ATM as it is capable of switching variable length packets.

## 2.4 Source Policing

Effective source policing is critical in order to maintain a satisfactory grade of service (GOS) across an ATM switch. An ATM switch architecture will have been carefully designed and performance engineered to provide the required GOS (sometimes also referred to as quality of service, QOS), and it is essential that neither intentional or unintentional misoperation of subscribers' equipment should jeopardise this by overloading the switch with excess cells.

Before any connection is established across an ATM switch, a call acceptance algorithm must decide whether or not the switch has sufficient resources to support the new call. The switch therefore needs information on the characteristics of the proposed new traffic before it can assess the effect of the new load on the switch block performance. In the case of CBR traffic the switch only needs to know the bit rate of the connection. In the case of VBR traffic more than one parameter is required and contenders include:

1. Mean bit rate
2. Peak bit rate
3. Burst duration
4. Burst frequency

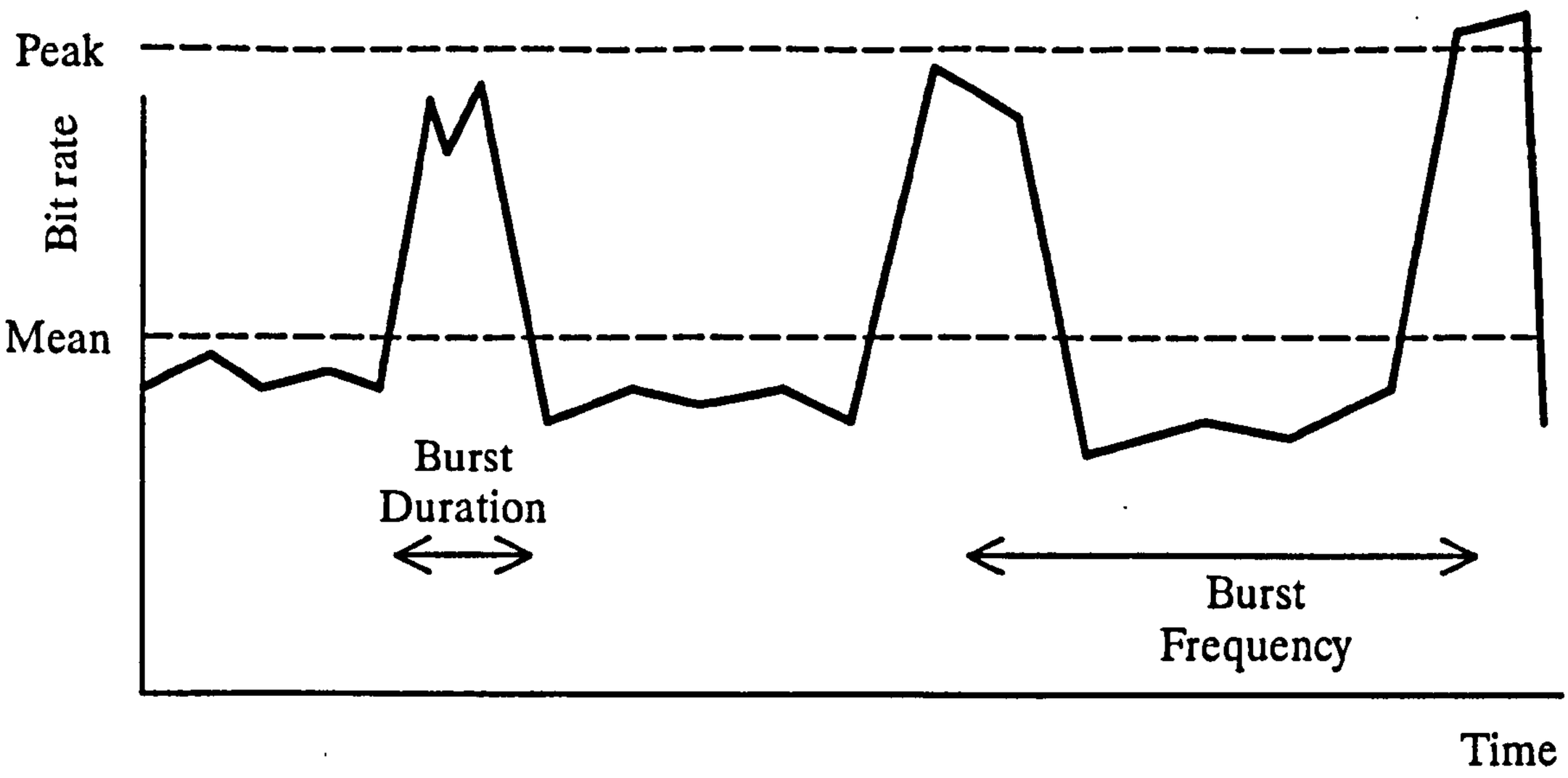



Figure 2.7: Characterising VBR Traffic for Call Acceptance and Policing

Figure 2.7 shows how these 4 possible VBR parameters could characterise the traffic. (Note that the term "bit rate" represents information rate and not line rate.) It can be seen that in most cases the parameters will probably only be approximations.

It can be argued that policing a connection once set up is more important and more onerous than making the call acceptance/rejection decision. Currently, an important paper on source policing is by Niestegge of Siemens Central Research laboratories in Munich [38]. The paper proposes an algorithm for policing both CBR and VBR traffic and suggests parameter values and dimensioning rules for its implementation.

The algorithm is based on a leaky bucket and is used to monitor and enforce the negotiated bandwidth of a connection. The policing device utilising the algorithm is placed as close to the user–to–network interface as possible. Excess bandwidth is dealt with by discarding any user data exceeding the negotiated bandwidth. However, the algorithm is tolerant to a certain amount of delay jitter on the traffic which may manifest itself as temporary increases in the source bit rate.

The well–known leaky bucket method is implemented with a counter having:

1. A threshold value
2. A decrement value
3. A decrement frequency.

On decrementing, the counter is not allowed to go negative. This method allows users to temporarily exceed their negotiated bandwidth by allowing short bursts into the network but the length of a burst is limited according to the selection of the algorithm's parameters.

The $10^{-10}$ quantile of the cell delay jitter is calculated by modelling an M/D/1 queue with a server utilisation of 85%. The value thus produced however is only realistic for a single ATM multiplexor between the subscriber and the network. The quantile value would be significantly higher if more multiplexing or switching stages were involved. The $10^{-10}$ quantile is chosen as this subsequently equates to the probability of cells being discarded without justification.

The proposed algorithm appears to be satisfactory for CBR traffic but the paper also claims VBR traffic can be policed in a similar manner. The method is to monitor both mean and peak bit rates with 2 leaky buckets. This coincides with the complementary call acceptance algorithm proposed in [39], in which VBR traffic is only characterised by the 2 bit rate parameters. Studying figure 2.7 it is possible to see that while the mean and peak bit rate values quantify the *volume* of traffic, they can give little indication as to the *burstiness* of the traffic. Figure 2.8 shows how 2 sources can have very different traffic arrival characteristics whilst having similar mean and peak bit rate values. Source A would be less burdensome for the switch to handle than source B due to the smoother nature of the cell arrivals.

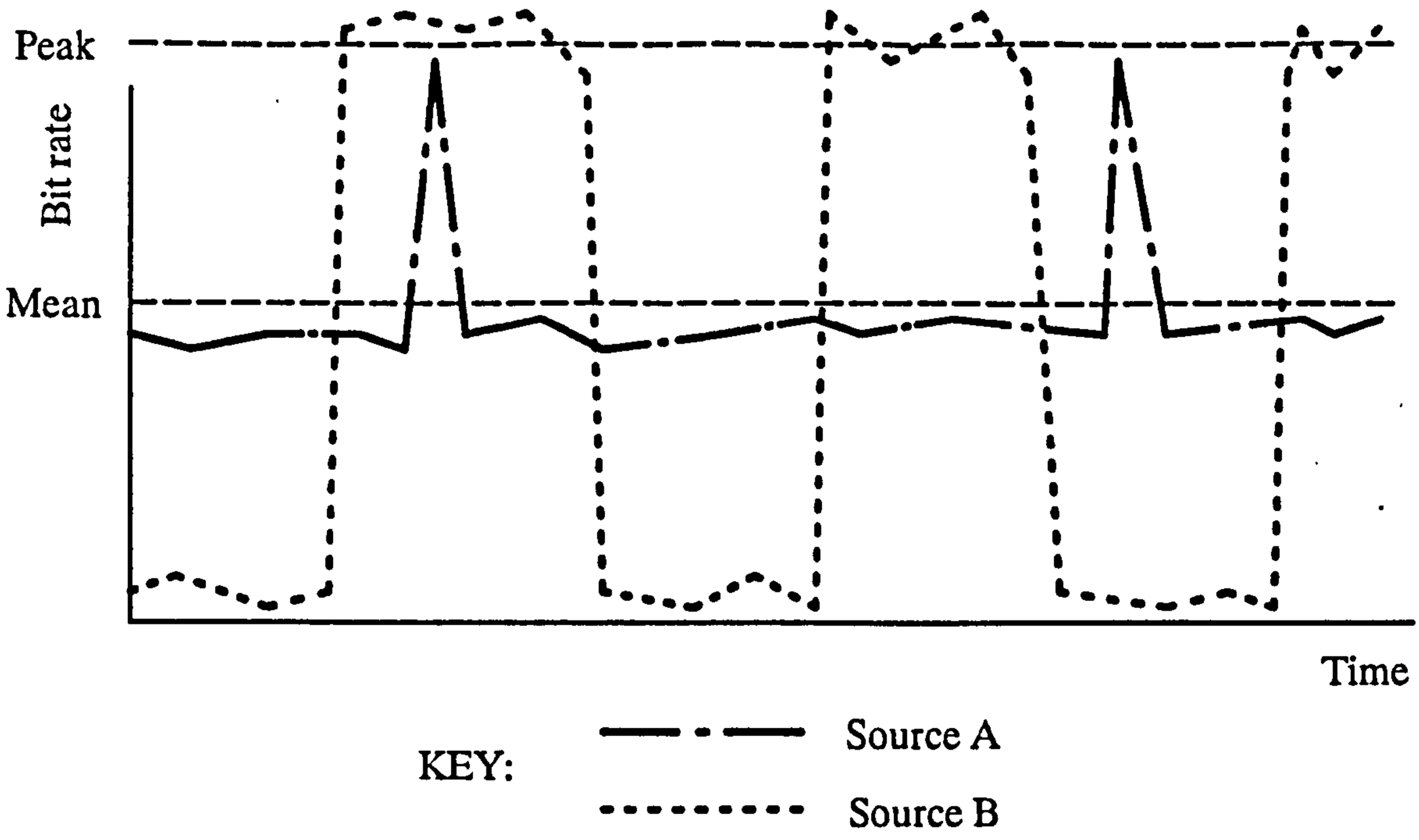



Figure 2.8: Two sources having similar mean and peak bit rates with very different arrival characteristics

## 2.5 Video Source Models

One of the main motives for a move to fast–packet, or ATM based networks is its ability to achieve statistical gain by multiplexing many *variable* bit rate traffic sources together. ATM therefore allows variable rate coding of video signals. This is significant because the application of compression algorithms onto a standard PCM encoded constant bit rate video signals results in a variable bit rate signal. Verbiest discusses the advantages of carrying coded video signals over ATM in [40].

The compression works because of a number of characteristics of a video picture, principally time correlation and space correlation. Time correlation means that there is a high probability that a picture element (pixel) will stay constant from one frame to the next. Space correlation means that there is a high probability that 2 adjacent

pixels will have similar colour and brightness. These 2 characteristics allow the standard PCM signal to be compressed.

The characteristics of the statistical fluctuations of the variable bit rate video signal may affect the performance of an ATM switch and influence its design. Therefore a number of source models have been developed to reproduce the likely behaviour of VBR video codecs.

Source models may be used during simulations to generate ATM cells with an appropriate rate and time distribution.

One of the earliest models with respect to ATM was presented by Maglaris et al. at Globecom '87 [41]. The model presented, apart from being based on the US standard frequency of 60Hz, generates negative cell sending rates. This is handled when it occurs by changing the cell sending rate to zero. However it reveals a fundamental flaw in the model.

Cosmas and Odinma–Okafor have modelled a conditional replenishment codec with a geometrically modulated deterministic process as reported in [42]. In the findings it is reported that the model would require 6144 different cell sending rates. As this number of states is not practical for a model it is reduced to 3. A description of the model and its implementation is given in section 4.1. It should be noted that some assumptions on the sojourn time in a state and the actual cell sending rates to use have to be made to implement the model.

## 2.6 Queueing Theory

It is no coincidence that queueing theory has telephony, and more specifically telephone traffic engineering, as its foundations. One must look upon A K Erlang,

Danish mathematician and teletraffic engineer, as the father of the subject. Then working for the Copenhagen Telephone Company, Erlang first published work on congestion in the telephone network in 1909. (In conventional telephone systems the theory is better referred to as congestion theory as if the system is full new calls are rejected and not queued.) In 1917 the generalised version of *Erlang's loss formula* giving the probability that $j$ out of a total of $s$ trunks are busy was published. Kendall further developed the mathematical aspects in the 1950's with a key paper given in reference [43] and also developed the standard queue classification referred to as Kendall's queue notation. There are many excellent texts on queueing theory, most notably Kleinrock in 2 volumes [44][45] and Cox and Smith [46].

Standard queueing theory makes assumptions about arrival and service processes. Whilst these assumptions conveniently provide mathematically tractable solutions and good approximations to real 'random' arrivals and service times, they are not applicable to ATM switching. The standard queueing model allows the server, whether it is a post office counter clerk or processor handling telephone calls, to serve the next 'customer' as soon as the previous job is finished. In the ATM switch case the service time (at the ATM transport layer) is deterministic but also *slotted* as shown in figure 2.9.

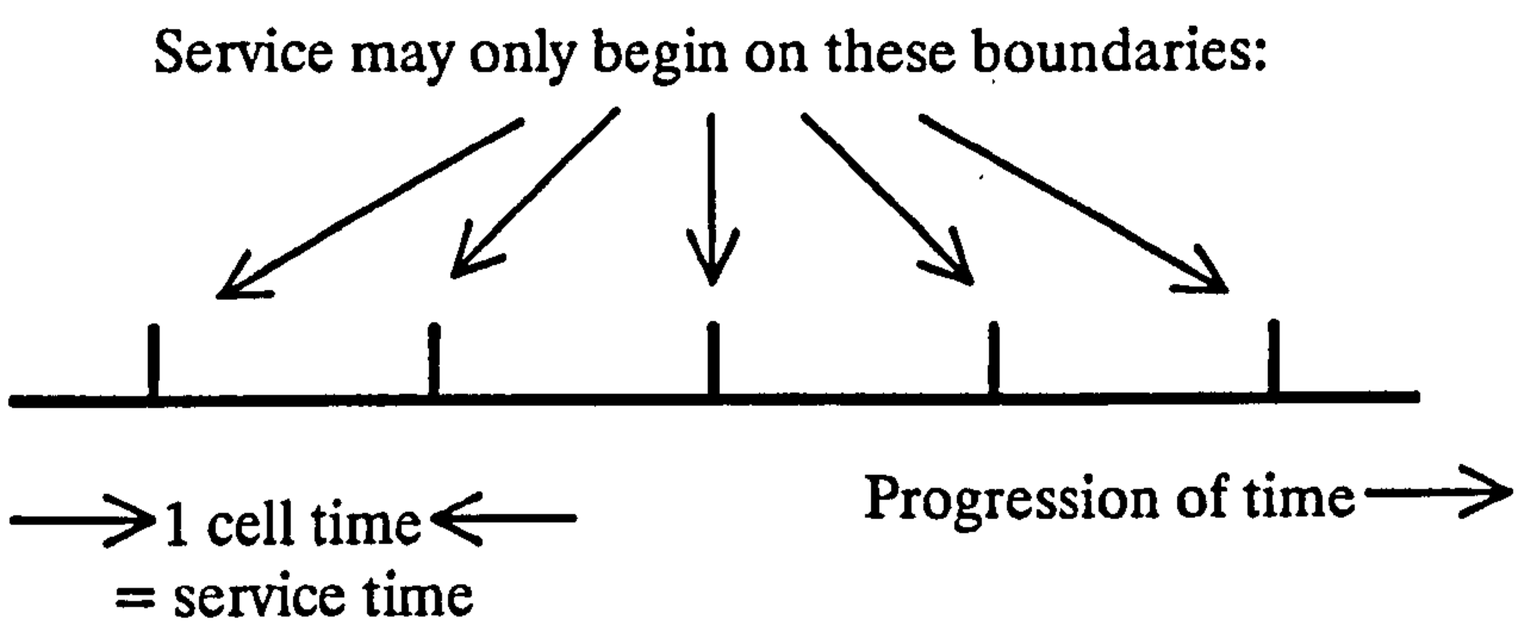



Figure 2.9: Slotted Service times in an ATM Switch block

This effectively renders present queueing theory invalid for this problem and promotes simulation as the clear route forward towards understanding how various networks of these queues behave.

## 2.7 The Development of Simulation Techniques and Languages

The history of modelling goes back many years. One of the earliest examples was C.de Buffon's celebrated estimation of $\pi$ produced by the unlikely means of dropping a needle onto a table [47]. The model of $\pi$, produced in 1777, involves dropping the needle between a pair of parallel lines a large number of times, N, and recording how many times one of the lines is intersected by the needle, K. If the distance between the lines is twice the length of the needle it is possible to derive that the probability of intersection is $1/\pi$ and therefore

$$\pi \approx N/K$$

Volser carried out this experiment in 1850 and obtained a value of 3.1596 for $\pi$. Buffon's needle marked the first recorded use of the Monte Carlo simulation technique. During the Second World War, Von Neumann and Ulam of Los Alamos scientific laboratories used Monte Carlo simulation methods for solving neutron diffusion problems [48] although the term "Monte Carlo" did not appear until the publication of work by Metropolis and Ulam in 1949 [49]. The Monte Carlo technique can be used to model the evolution of a system with time where the nature of the evolution between 2 time instances is stochastic.

Simulation is a broad term with a wide definition covering a diverse range of subjects. Computer simulation, as a more specific branch of the overall subject, became possible in the early 1950's with the advent of high–speed analogue computers. Studying the development of the more popular simulation programming languages

and tools is an appropriate way of illustrating the history of the subject. One of the first commercially available simulation tools was IBM's General Purpose Systems Simulator, GPSS, also referred to as the Gordon Simulator after its founder [50]. The 1960's saw the appearance of many other programming languages developed specifically for simulation purposes. In 1962 the Rand Corporation produced the first version of SIMSCRIPT [51]. In the same year in England CSL, the Control and Simulation Language, was presented by Buxton and Laski [52]. In 1966 the first version of SIMULA, as developed by Nygaard and Dahl of the Norwegian Computer Centre in Oslo, appeared referred to as SIMULA 1 [53]. Even though it is called a simulation language, SIMULA is actually a general purpose language based on Algol. SIMULA 1 was quickly superseded in 1967 by SIMULA 67 and has had its capabilities for simulation greatly enhanced by the DEMOS (Discrete Event Modelling on SIMULA) package [54].

The evolution of digital computers has seen simulation gain in popularity as a multi–disciplinary numerical problem–solving technique and this is demonstrated by the fact that it is taught in many college and university departments today.

## 2.8 Simulation Techniques

In reference [55], De Prycker and De Somer present the behaviour of the BTM switch as described in section 2.2.2 above. In this paper the behaviour of the switch under both static (call–set up phase) and dynamic (information transfer phase) conditions with a wide variety of traffic types is discussed. The results were obtained by simulation. No comparisons between the performance of the BTM switch and other switch types are made. The paper offers the view that the BTM switch is capable of supporting a wide range of traffic types and a high aggregate load. The cell–loss figure

is however assumed to be $10^{-8}$ which is a worse grade of service than the $10^{-12}$ generally used in this thesis.

Various ATM modelling techniques have found favour in research establishments in England. The Queen Mary and Westfield College in London has reported studies on burst–level modelling of ATM switches in the literature. The University of Durham has reported on studies into the simulation speed–up achieved by parallel processing using transputers.

In [56], Earnshaw addresses the problem of synchronising an array of transputers used in this way. A speed–up factor is not given but each transputer is quoted as being capable of generating 200 cells per CPU second. Additionally, Earnshaw states a belief that call set–up and clearing messages take an insignificant time in proportion to the call holding time, and that the propagation delay of a cell may be calculated assuming the speed of transmission is the speed of light, c. With respect to the first point, this is true for conventional telephony calls but is untrue for newer services such as the Fast Select Protocol [57]. This potentially high volume service sends the set–up with a few ATM cells directly behind and therefore the set–up time is not insignificant. Applications could include fax, E–mail and data query/update. With respect to the second point, the speed of transmission in copper varies with the configuration (e.g. twisted pair, co–axial cable) but is approximately 2/3rds of the speed of light. The speed of transmission in optic fibre is fractionally slower than that in copper.

In [58], Hind argues that the need for parallel simulation of circuit switched telecommunications networks is well established. Parallel simulation is one technique offering some benefits when it is required to simulate a large number of nodes but

brings with it additional problems including synchronisation of the nodes to prevent causality errors.

The simulation technique employed by Pitts of Queen Mary and Westfield College, University of London, in [59] to model B–ISDN networks is to describe events as bursts of cells instead of individual cells. The motivation is well–founded; to simulate a high enough number of cells to be able to study extremely low cell–loss probabilities (in the region $10^{-8}$). A detailed comparison of the cell–level and burst–level methods would be long and complex. To summarise, queues in an ATM switch can be regarded as having 2 components, the burst–level component and the cell–level component. The concept is shown in figure 2.10.

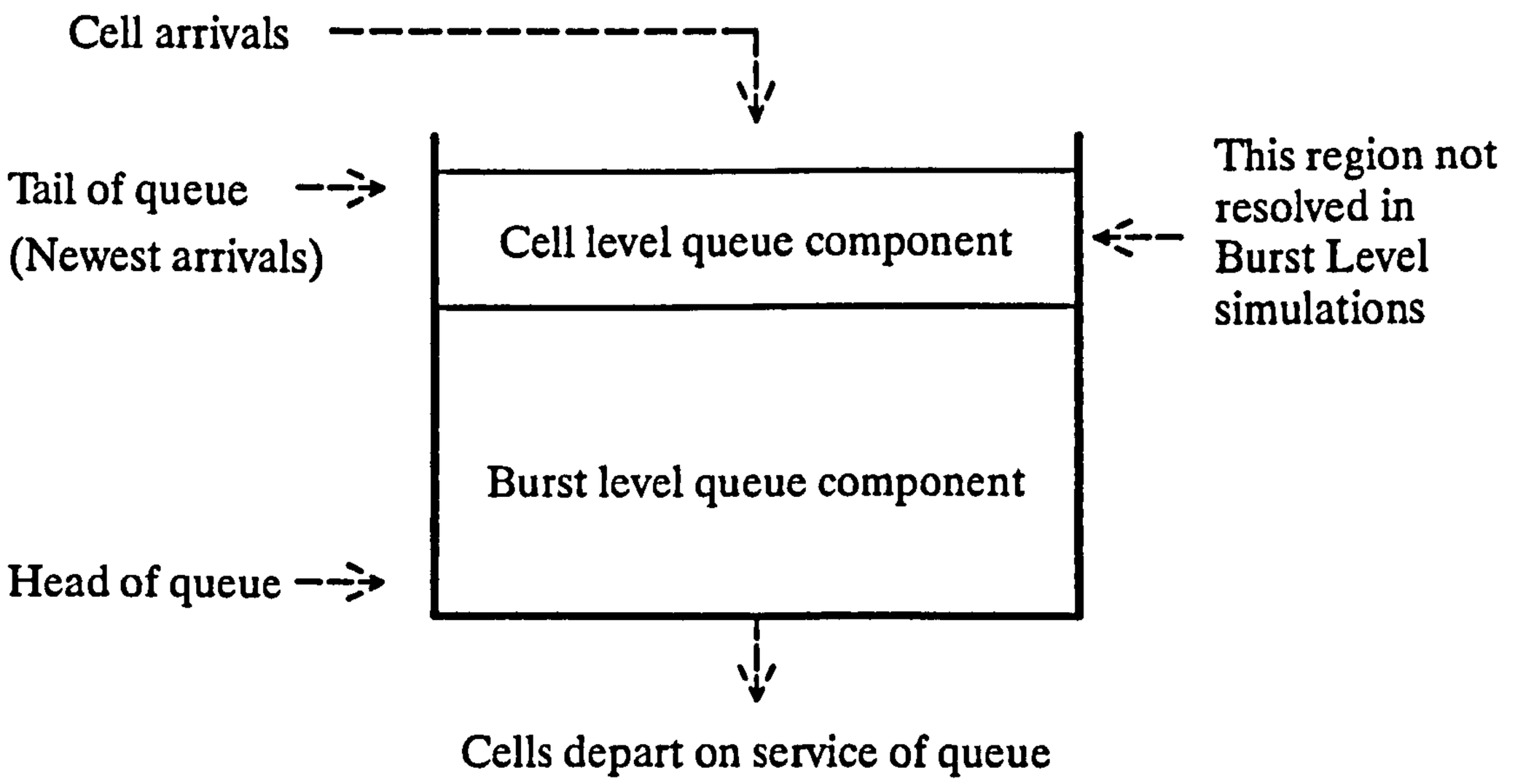



Figure 2.10: Conceptual Queue Components, Burst level + Cell level

The queue is shown having an underlying occupancy resulting from bursts of cells. In addition there is an occupancy due to individual cells. The cell level component is regarded as being much smaller than the burst level component and comprises of cells

that have 'just arrived' at the queue whereas the burst level component is the occupancy caused by groups of cells arriving some time before the present time. In short, the burst level represents the macroscopic and the cell level the microscopic view of the queue occupancy. The burst level simulation technique of Pitts trades precision in the results with a reduction in computational effort required. If large amounts of CPU time are available the cell level based simulation technique will give more precise results. The methodology of the simulator used in this research programme is fully discussed in chapter 3.

## 2.9 Buffer Positioning

On the subject of buffer positioning within an ATM switch, several key papers have laid the foundations. Lutz in [60] compares switches with input buffers, output buffers and central buffers and concludes that central buffers are clearly superior in terms of system, traffic and technological aspects. In [61], Karol, Hluchyj and Morgan model a fixed length packet switch with a cross–bar architecture and compare the performance of queues at inputs with queues at outputs. The principal conclusion is that mean queue lengths are shorter when the queues are at the outputs. A system with queues at the cross–points was unfortunately not simulated.

Lehnert in [62] models input, output and cross–point buffered switching elements. He concludes that input buffers require 70 queue places at 0.6 Erlang offered traffic (the blocking probability at this number of queue places unfortunately is not specified). It is possible to improve this performance by modifying the queue discipline from FIFO to allow the first $n$ buffer places to be accessed by the outgoing lines. When $n = 4$ the throughput increases to 0.85. The output buffered case is quoted as being optimum from a *performance* point of view whereas the cross–point buffered

architecture is said to be optimum from a *throughput* viewpoint. It is noted that the cross–point buffered architecture requires more buffer space in total than the output buffered one.

Pattavina in [63] uses binomial cell–arrival distributions to model input and output queueing. Throughput and delay were modelled using analytic techniques including queueing theory but in section 2.6 we have seen that conventional queueing theory is not an accurate representation of the operation of queues in ATM switching elements due to the slotted nature of the servers. Queue full situations have been modelled by simulation techniques but only with binomial cell arrivals.

## 2.10 The Problems of ATM

The vast majority of papers on ATM in the literature cover the new and exciting aspects. Coverage of the problem areas is very sparse. This section attempts to highlight some of the less glamorous and more problematic aspects of ATM.

### 2.10.1 Packetisation Delay

In [64] Spratt proposes a possible solution to the problem of large packetisation delay incurred on speech connections. The solution is to carry data from more than one speech connection in the same cell, suggesting that 4 or 5 calls per cell may be optimum. This is referred to as the *composite cell* technique and raises some interesting questions on the strategy for allocating new calls into existing composite cell paths across the switch block. In this method the 48 user bytes in the cell are split into 4 or 5 different calls. In the case of 4 calls the packetisation delay is reduced from 6 ms (48 x 125 μs) to 1.5 ms (12 x 125 μs).

This early simulation based work shows that in a predominantly 64 kbit/s speech carrying network the composite cell technique can offer performance improvements by trading an increase in bandwidth used across the switch and complexity of call acceptance/allocation control with a reduction in the accumulated delay at the packetisation stage. One problem with this technique that is not highlighted in the paper is that a synchronous PCM–based transmission medium is required between switching nodes. It seems unlikely that in an ATM network cells will not be transmitted down an ATM pipe. If the composite cell technique were to be extended to allow the possibility of an ATM cell transmission format between nodes, more complex signalling would have to exist between the nodes to transfer information about which part of which cell belongs to which speech circuit. All of this signalling would have to be standardised by CCITT as 2 adjacent switching nodes may be supplied by 2 different equipment manufacturers. This interworking problem would not occur across a switching node as a proprietary protocol could be used locally. The composite cell technique has not been followed up in any other literature.

### 2.10.2 Network Synchronisation

Network synchronisation is required in an ATM network for the accurate reproduction of information carried in constant bit rate (CBR) streams. In an unsynchronised network two possibilities exist for end–to–end synchronisation, adaptive timing and fixed local clocks with 'slips'. Both of these methods are deficient in important characteristics. A definition of timing related terms is given in appendix C.

In the adaptive technique, a local oscillator attempts to reproduce the source clock from the incoming cell stream. A buffer is required to accommodate the variable

delay across the network. While no information is lost (or gained) using this technique a large amount of jitter remains (no jitter attenuation) and indeed further jitter can be generated.

With the fixed clock method the local clock can be a free running oscillator or may be phase locked to the local switching node. In this latter case however, the local node is not synchronised to the distant node and therefore a difference between source and destination clocks will exist. While there will be no jitter on the recovered signal, information will invariably be lost (or gained) due to the frequency difference between the 2 clocks.

Therefore, if the network is unsynchronised, CBR services will suffer either slips or severe jitter. As neither of these are acceptable due to the onerous requirements of many CCITT, BELLCORE etc. specifications, some form of network synchronisation is required. In today's networks, 2.048 Mbit/s streams are frequently used for network synchronisation. There are various advantages to using 2.048 Mbit/s links for synchronisation in ATM networks. Firstly, existing networks use the system making evolution to ATM based networks easier. Secondly, equipment and standards for synchronisation at 2.048 Mbit/s are available and well established. The possibility of carrying circuit emulated 2.048 Mbit/s synchronisation signals is investigated in chapter 6.

Additional methods of synchronisation exist. Remote synchronisation by either satellite or terrestrial radio is a possibility. The main problem with both of these methods is one of security. If a network is dependent on a satellite or radio transmitter, it would not be difficult to bring down the entire synchronisation mechanism either accidentally or maliciously. Additionally, terrestrial radio and satellite systems are subject to interference from man–made sources and also

variable propagation conditions from natural sources such as E−layer propagation and aurora effects.

In "Taking Time Across the Network", Chopping [65] points out the inability of the SDH transmission technique to carry accurate synchronisation signals. As SDH is a potential carrier of ATM cells the disclosure is worrying. The problem is the byte−sized Positive/Zero/Negative justification method employed by the SDH standard. By contrast, the PDH standard uses bit−sized positive justification which provides continuous small adjustments of the clock in the same direction. Chopping proposes a modification to the SDH standard to enable the hierarchy to carry timing information. The modification has not been incorporated even though it dates back to 1990. Chopping also believes that the inability of the ATM technique, by its very asynchronous nature, to carry timing information will result in ATM *not* becoming the general purpose multiplexing method of the next generation of telecommunications equipment.

No satisfactory solution to ATM network synchronisation has yet been found in the literature.

This chapter has reviewed significant papers both past and present from the literature. The next chapter describes in detail the simulation tool developed and used during the course of the research presented in this thesis.

# Chapter 3 The ATM Switch Simulator

This chapter will discuss what simulation is, why simulation is used, its costs and drawbacks, and how simulations are performed. It will go on to discuss the design philosophy and validation of the ATM switch simulator used during this research project.

## 3.1 What is Simulation?

*"Simulation: feigning, mimicry, the making of working replicas or representations of machines or the recreation of a situation, environment etc. for demonstration or analysis of problems."* [66]

Simulation, as the above dictionary quotation illustrates, is the art of representation of a system, either real or imaginary, for the purposes of extracting or deriving information about that system's behaviour and characteristics over a period of time.

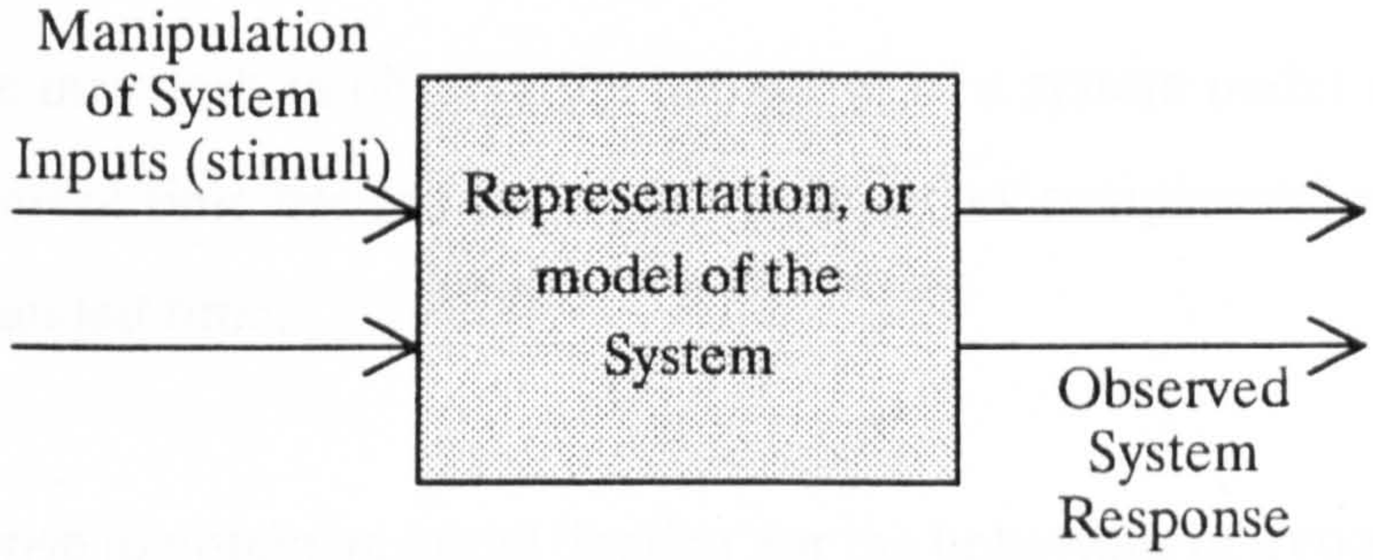



Figure 3.1: Basic Simulation Concept

The representation of the system is referred to as the *model*, and abstracts the system's relevant relationships and attributes allowing the extraction of the properties of interest. The model is necessarily a simplified representation of the system, indeed if the model represented *all* of its characteristics, it would *be* the real system.

## 3.2 Why Simulate?

The motivation to simulate may be driven by one of a number of factors including:

a. It may be too hazardous to study the real system (for example chemical or nuclear reactions).

b. It may be too expensive, time–consuming or disruptive to study the real system (for example aeroplane design, long–term economic effects or traffic flow).

c. One may wish to study a system that is not amenable to an analytic solution (for example complex computer networks).

d. One may wish to investigate the properties of a system that does not yet exist (for example various designs for a new road bridge may be simulated and tested with 3–dimensional scale models).

e. One may wish to observe the behaviour of a system under a different time base to real time, i.e. in either compressed or expanded time.

One may also wish to obtain an initial 'feeling' for the behaviour of a complex system through simulation which could then lead to more precise modelling once the important characteristics and parameters have been identified.

In summary, simulation can provide an environment which permits controlled experimentation on a system to increase the understanding of the system's response to changes in its external stimuli.

## 3.3 The Limitations of Simulation

However appealing the lure of simulation might seem, there are limitations and dangers involved of which one must be aware.

The first danger is cost. Simulation can be an enormously expensive undertaking in terms of man–power and computer processing time, and there is no guarantee that the benefits finally derived from a simulation will more than offset its costs. To illustrate this, the cost of development of a land–combat simulator in the early 1960's was estimated as over 3 million dollars with no results obtained for the first two and a half years [67].

The second danger is one of credibility. The computer simulation model exists only in the form of software programs. As such, the operation is hidden from the user and limitations, inconsistencies and even errors in the model may be overlooked. It is prudent therefore to perform adequate validation of the simulator, to provide de–bug and trace facilities in the code and to regard any results produced as sub–optimum solutions and guide–lines rather than absolute, optimum values. One must recognise that it will be hard to convince others of the validity of such results.

The third danger concerns the use of the model beyond its range of applicability. By way of a simplistic example, the use of a flight–simulator computer game to train airline pilots would clearly be foolish but the bounds of applicability of a chemical reaction simulation say, might not be so clear. This problem could manifest itself after long, personal involvement with a project where the tendency is for the investigator to extract the maximum out of the model.

## 3.4 Approaching a Simulation Problem.

If simulation is determined as the most appropriate method of solving a problem, and it has been described as a "method of last resort" [48], the first step is to produce a *model* of the system. Clearly, many different models of the same system can be produced. A town for example, could be modelled by means of a map, by a miniature 3−dimensional scale model, by a list of its inhabitants or in many other ways.

Once a model has been established one must decide which methodology to apply. Simulation methodology can be categorised into three types [68]:

1) Continuous time,

2) Discrete time,

3) Discrete event.

Continuous time simulations are used where the model is described by ordinary or partial differential equations in which the system state varies continuously with time. Discrete time simulations are used where the model can be expressed by difference equations where the system need only be considered at selected moments in time. Discrete event simulations are used where the model can be described as a sequence of events such that nothing of importance occurs between those events, but where time is represented (at least conceptually) as continuous.

Having defined the model and established the methodology, one can then proceed to:

1. Implement the simulation,

2. Validate the operation of the simulator,

3. Design and perform the simulation experiments,

4. Collect, analyse and summarise the experimental results.

## 3.5 Development and Validation of the ATM Switch Simulator.

This sub–section describes the methodology, the design philosophy and the validation of the ATM switch simulator developed during this research project. The simulator is referred to as ATMoSS and is implemented in the programming language C.

### 3.5.1 Simulator Methodology

ATM switch simulators at the cell level (Note1) lend themselves well to being described by either method 2) or 3) in section 3.4 above [48]. The advantage of type 3) event–driven simulation is that when there is no processing to be done between events, the conceptual system time can shift, or jump forwards, to the next event thereby saving processing time. This is shown in figure 3.2. In discrete time systems, real time is modelled as fixed increments regardless of whether there is any processing to do or not. The discrete time concept is shown in figure 3.3.

(Note1).Two types of ATM switch simulator have emerged, simulators at the cell level in which individual cells arrive and are processed discretely, and simulators at the burst level in which cells arrive in bursts and are processed together.

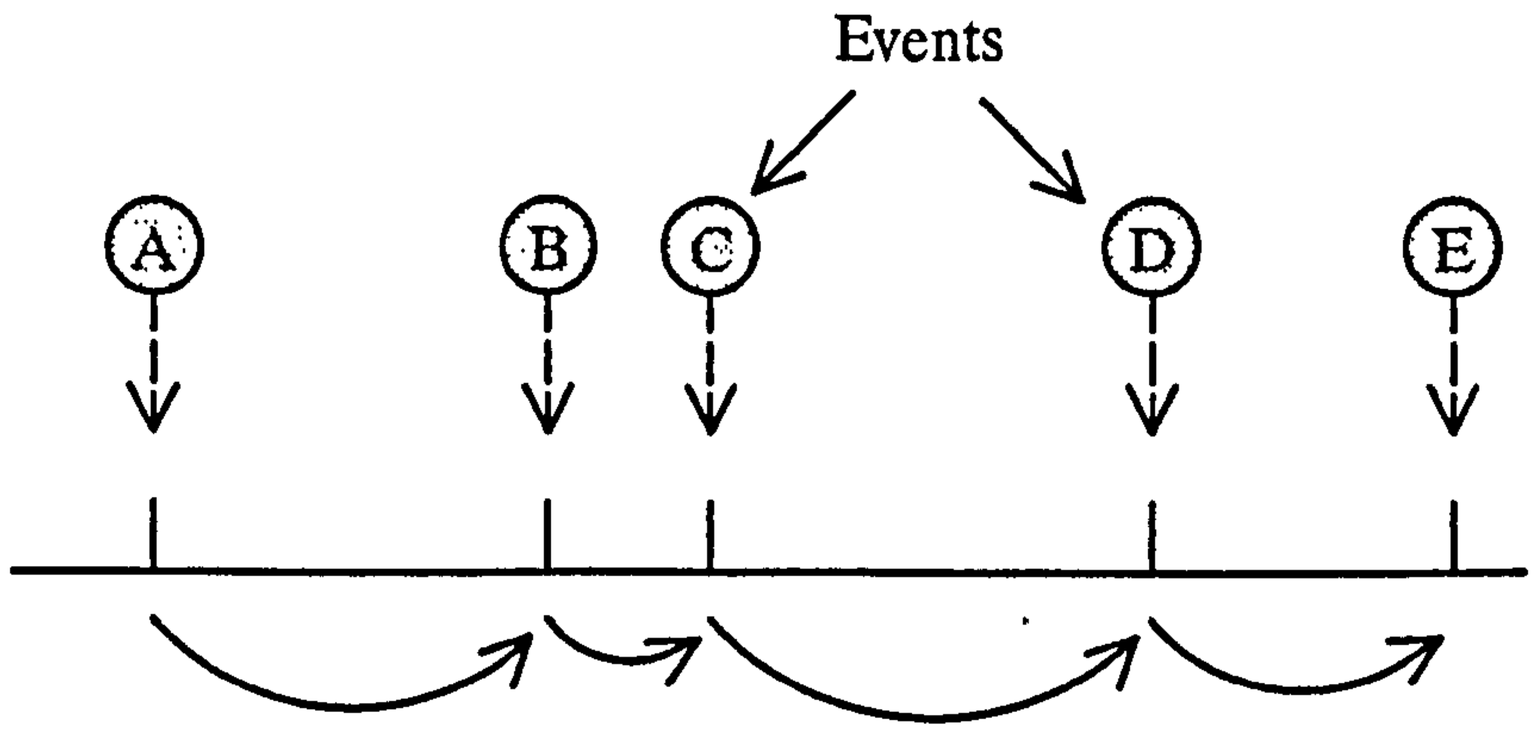



Figure 3.2: Discrete Event Concept

The discrete event approach can give benefits in terms of elapsed central processor unit (cpu) run time at low simulated loads. However, the performance of ATM switch blocks under low loads is of minor importance compared to their performance at high loads and so the advantage is lost in the operating conditions required to be studied.

The discrete event approach also allows the conceptual time to take any value, not just one of the fixed increment values of the discrete time method. But, the reduction of $\delta t$, the unit of processing time in the discrete time method, to small enough values allows the discrete time method to behave approximately as discrete event.

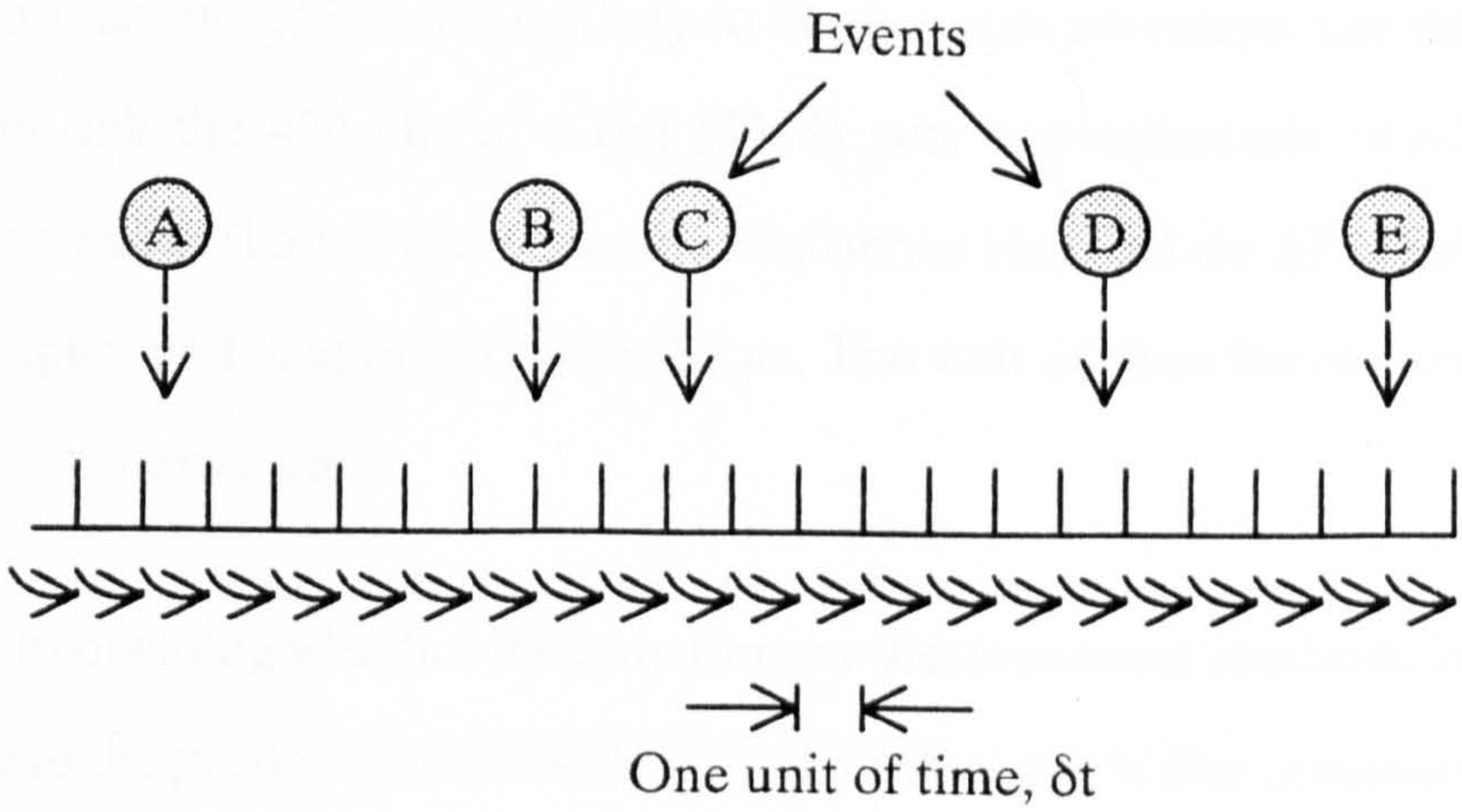



Figure 3.3: Discrete Time Concept

The method chosen for the ATMoSS simulator was the discrete time, (also called unit−time) concept. This method, shown in figure 3.3, more closely resembles how the hardware of a subsequent real system would be implemented. As an actual switch block must operate in a synchronous manner the unit−time approach, which effectively synchronises the actions of the components of the switch, is seen as being more appropriate. The same 5 events are shown on both figure 3.2 and 3.3. Only events B and E align exactly to an event boundary in the discrete time approach, highlighting a limitation of this method for certain systems. Due to the slotted nature of the ATM mechanism, this problem does not occur when simulating ATM switch blocks.

The unit of time for the simulation was chosen as follows. The standard ATM cell as specified by CCITT [17] is 53 bytes long comprising 48 data (user) bytes and 5 header bytes. The physical layer at the UNI (User Network Interface) over which ATM cells are exchanged between users and the network is described in [69]. A bit rate of 155.520 Mbit/s has been selected as one possible speed for the physical medium and

with current technology is initially likely to be the most prevalent. On this speed of transmission link the 424 bits of a cell (53x8) take approximately (424/155)x$10^{-6}$ seconds to transmit. This is known as the "existence time" of an ATM cell on a 155 Mbit/s multiplex and is approximately 2.7 μs. The unit of time for the simulation is taken as the existence time.

In addition to deciding whether discrete time or discrete event methodology is more appropriate to the problem, in the ATM switch block case it is also necessary to decide on whether a cell−level or burst−level simulator is more appropriate. The two techniques are also discussed in section 2.8.

In the cell−level case, cells are modelled individually. In the burst−level case, cells are modelled as arriving in bursts typically with 3 parameters, burst duration, inter−burst gap and burst bandwidth as shown in figure 3.4.

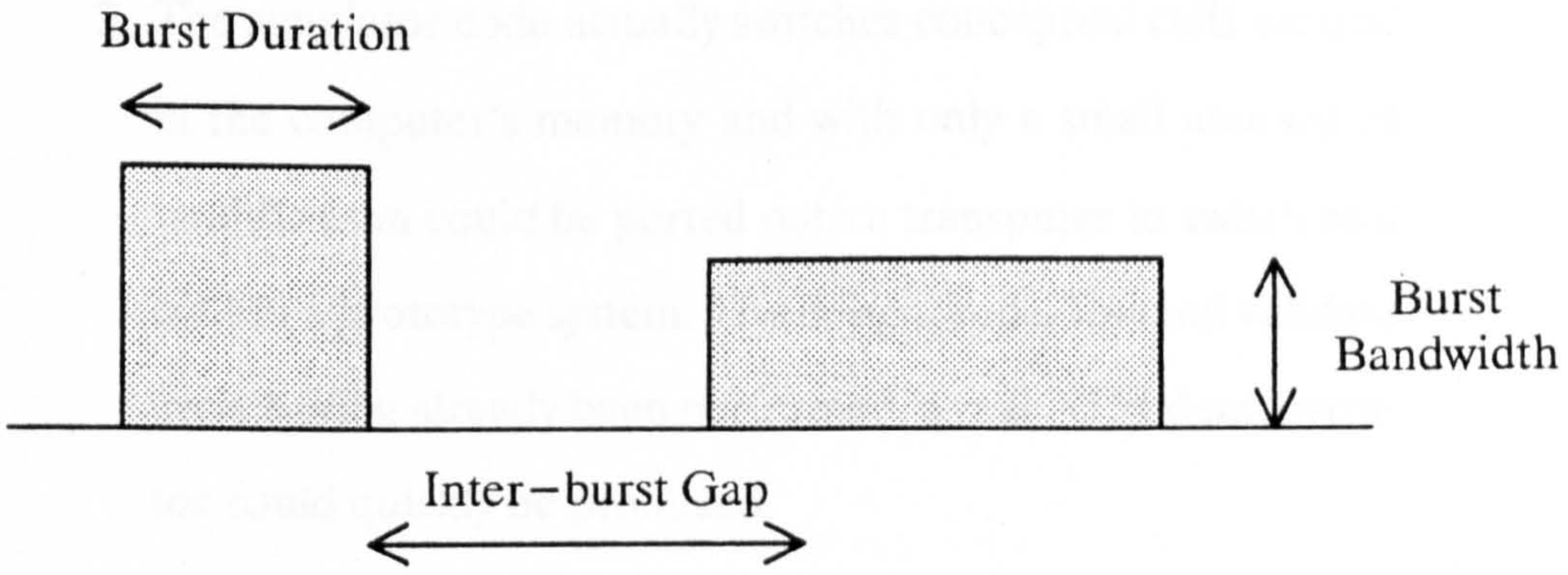



Figure 3.4: Burst−Level Modelling Concept

Both cell−level simulations and burst−level simulations have their advantages and disadvantages. The principle advantages are summarised below:

| Cell Level Advantages | Burst Level Advantages |
| --- | --- |
| High–precision results are obtained. | "Speed–up" of simulation run time. |
| Higher number of draws on the random number generator improves spectral qualities | |

The ATMoSS simulator has been implemented as a cell level simulator. There are 2 main reasons for this:

1. The simulator runs on a SPARC 1+ with a processing power of approximately 16 MIPs. The simulator has sole use of the processor and can also be left to run overnight, over a weekend, etc. Lack of processing power is therefore not a problem in these circumstances.

2. The simulator code actually switches conceptual cells around in the computer's memory and with only a small amount of modification could be ported onto a transputer to switch real cells in a prototype system. The design, code, test and validate cycle having already been performed, a real ATM demonstrator could quickly be produced.

### 3.5.2 Simulator Design Philosophy

ATMoSS was required as a tool to further understand how various architectural changes within an ATM switch fabric affect its performance. It was therefore required to be general purpose, being able to model as many different configurations of switching elements as possible. Unfortunately, an overly–generic product is

extremely difficult and time consuming to produce and test, and some compromise on flexibility needs to be made or there is a danger that the product may never come to fruition at all.

This sub–section describes how ATMoSS was designed to achieve these goals.

### 3.5.2.1 Simulator Physical Representation

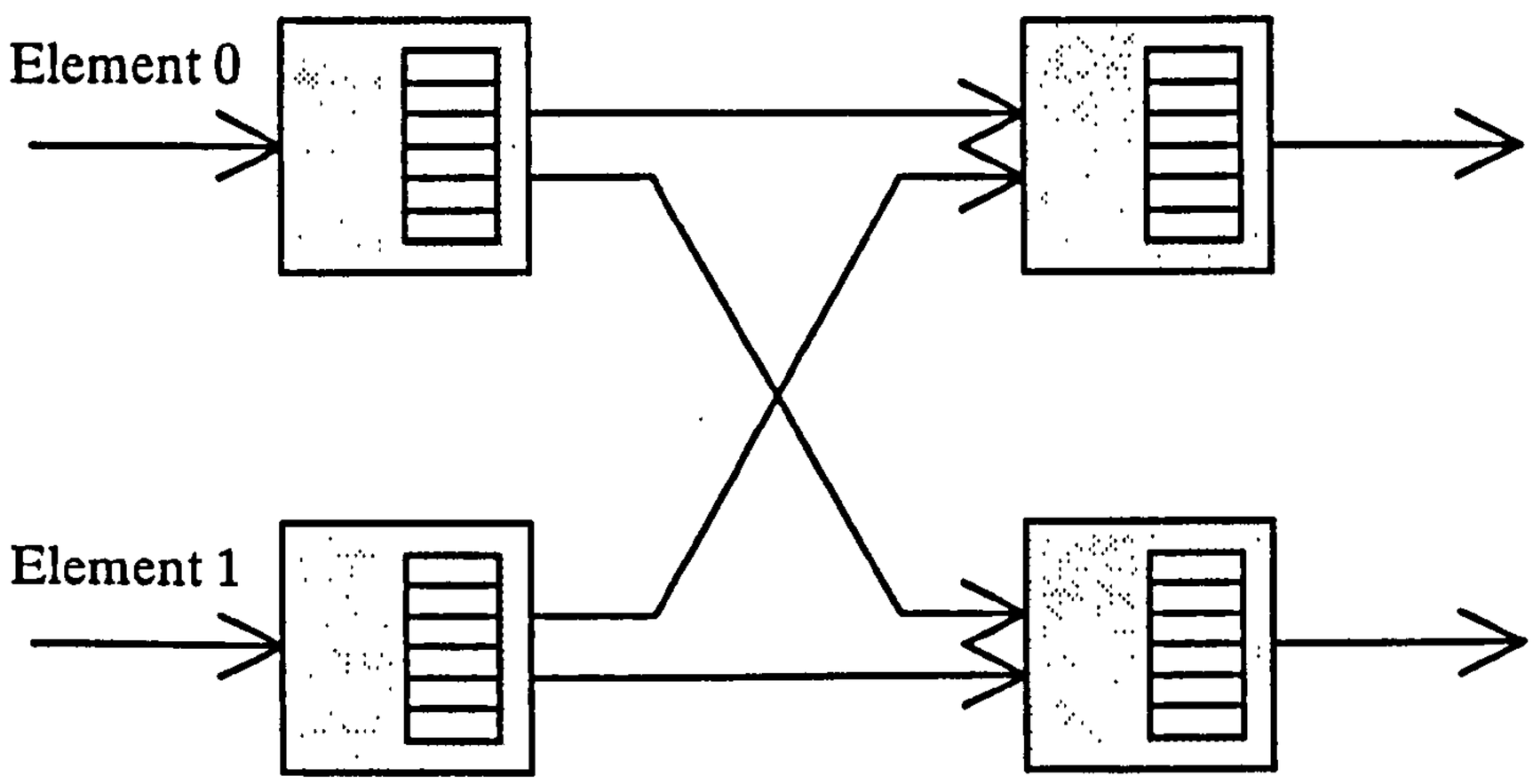



Figure 3.5: Basic 2x2 Simulator Physical Block Diagram

Figure 3.5 shows the simplest ATM switching physical arrangement. The switching elements are arranged in *stages* with the conventional use of that term. The stages are numbered from zero. Stages are composed of *elements* which are also numbered from zero. The simulator allows the number of stages, elements per stage and the interconnect pattern to be varied if required.

### 3.5.2.2Simulator Functional Representation

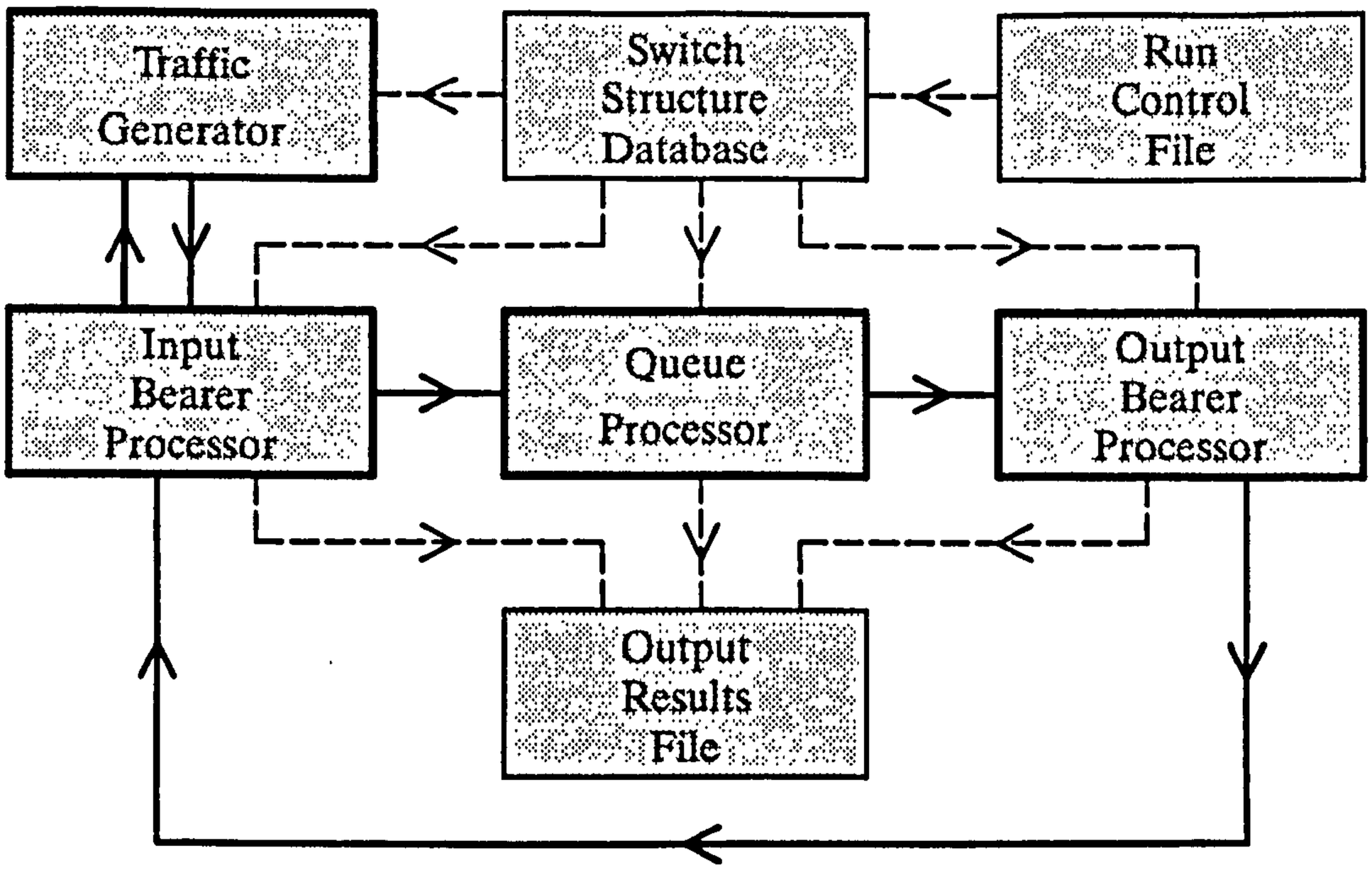



Figure 3.6: Simulator Functional Block Diagram

Figure 3.6 shows the functional block diagram of the simulator. The executeable code is shown in the boxes with the thick outlines, data areas are shown with thin outlines. Cell flow and control of the simulation is represented by solid lines, data is written and read via the dashed lines. There is in addition a common read/write area containing the data that is manipulated during the simulation run. The key individual functional blocks are discussed in more detail below.

### 3.5.2.3The Switch Structure Database

This contains all of the parameters governing the current simulation run, e.g. the number of switch stages, the number of elements per stage, length of run, traffic type and percentage load on bearers, number of queue places in each switching element etc. There are 2 types of parameter, hard−parameters and soft−parameters.

The hard parameters determine the size and structure of the switch fabric model in terms of the number of stages, number of elements per stage and the interconnect pattern. These are written into the source code and may only be changed by a re–compile.

The soft parameters to the simulator are read from the Run Control File at the start of a run. The Run Control File is discussed in detail in section 3.5.2.4. The soft parameters available are:

1) Queue Lengths in each stage.

2) Number of units of time to be simulated.

3) The traffic load on each input multiplex.

4) The arrival distributions of incoming cells.

5) The proportion of cells from each input to each output multiplex.

6) The configuration of the intermediate and final stage queues.

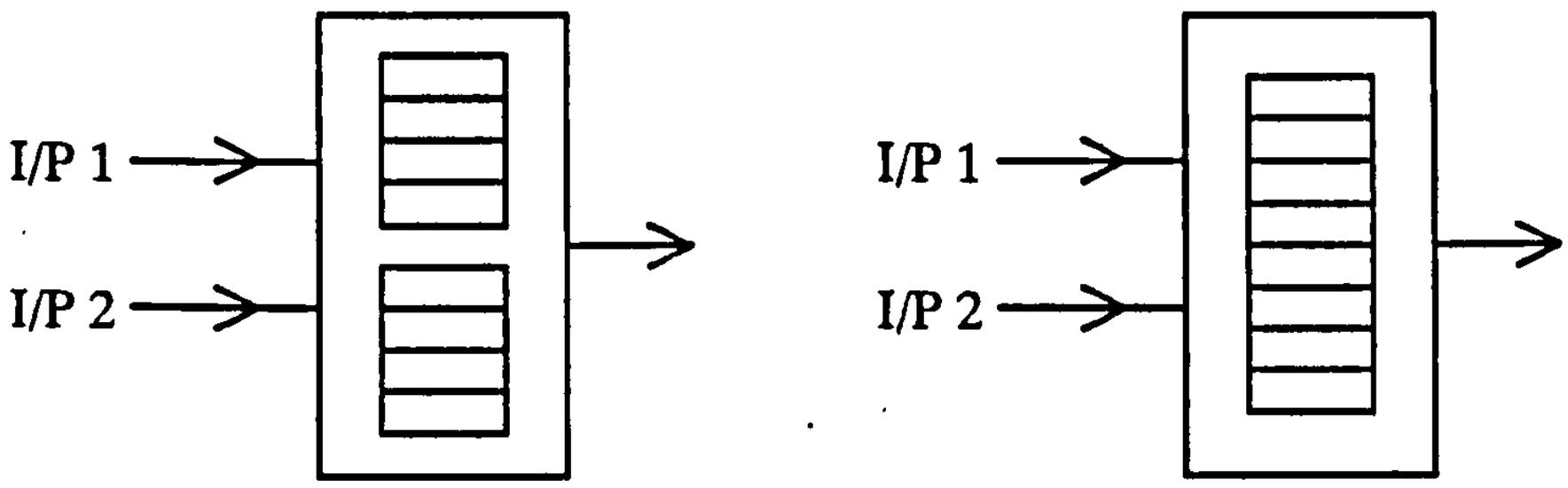
7) The algorithm to apply if the intermediate or final stage queues overflow.

8) The number of constant bit rate (CBR) sources.

9) Jitter analysis mode (enabled/disabled) and start time.

The meaning of most of the parameters is obvious, however it is useful to expand on numbers 6), 7) and 9). Item 8) is discussed more fully in section 3.5.2.5 below.

The queues in the intermediate and final stages of the switch (item 6 above) can be configured in one of two ways as shown in figure 3.7. In mode 1 (LH drawing) there is no queue sharing, in mode 2 (RH drawing) all inputs share the same queue space. It might be anticipated that queue–sharing will be the most efficient in terms of memory utilisation. However, an overload on input 1 produced by users exceeding their permitted bandwidth may seriously affect the grade of service provided to users on input 2 – clearly not a satisfactory situation. The purpose of the two different configurations is to allow the possibility of studying the advantages and disadvantages of the 2 modes.



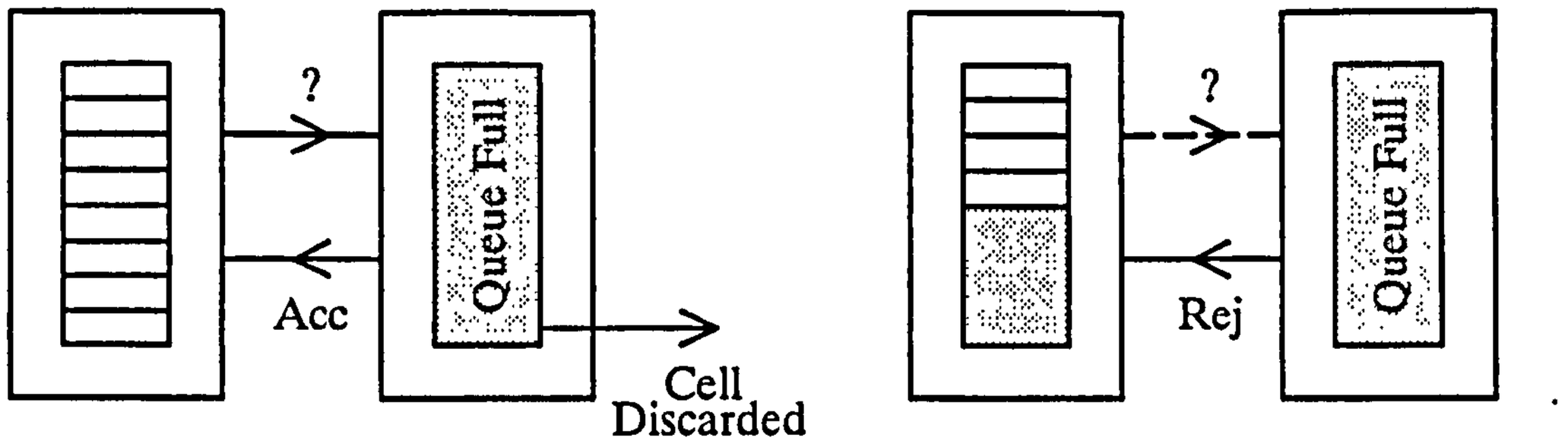

Mode 1: Each Input has a Dedicated Queue

Mode 2: All Inputs Share the Same Queue

Figure 3.7: The Two Configurations of the Intermediate and Final Stage Queues

The intermediate and final stage queues can handle queue–full conditions in one of two ways (item 7 above) as shown in figure 3.8. The essential point is that there are queues in the earlier stages of the switch. When the intermediate or final stage queues are full, potential incoming cells can either be rejected by the stage, in which case they remain in the previous stage (RH drawing mode 2), or the cell can be accepted and the oldest cell in the queue discarded to make room for the new cell (LH drawing mode 1). (An alternative strategy is to discard cells based on the CLP bit but this is

not implemented in the simulator.) Again, the purpose of the two operating modes is to allow the possibility of studying the most efficient distribution of queue places.




Mode 1: Cell Accepted, cell loss occurs

Mode 2: Cell Rejected, remains in previous stage

Figure 3.8: The Two Methods of Handling Queue Full Conditions in Intermediate and Final Stage Queues

Jitter analysis (item 9 above) accumulates the occurrence of specific delays of cells across the switch thus producing a discrete delay or waiting–time distribution. It can be either enabled or disabled as appropriate. This allows suppression of the copious output when jitter analysis is not required. In addition, the start time for the jitter analysis may be specified. This is necessary as the analysis uses the discrete distribution of delayed cells and a significant amount of error is introduced during the simulation warm–up period particularly at high loads. It is theoretically possible to disable jitter analysis by specifying a start time for the analysis as a time after the end of the simulation run. This is not however considered to be a particularly elegant solution.

### 3.5.2.4 The Run Control File

The run control file, which is read at the start of each simulation run, specifies the values of the soft parameters for the particular run. It is composed of Simulator

Command Language (SCL) instructions. These must be written to a precise syntax that the simulator pre−processor can understand. The SCL instruction set is given in appendix A.

The soft parameters listed in section 3.5.2.3 all have default values. The control file may then simply consist of the command "GO", in which case the run proceeds with all soft parameters set to their default values.

### 3.5.2.5 The Traffic Generator

The Traffic Generator is capable of producing both CBR and VBR traffic. Variable bit rate source models are discussed fully in chapter 4. In this sub−section the constant bit−rate traffic generator is discussed.

The principle behind the CBR source model is that constant bit rate traffic is deterministic and regular. The bit rate of the CBR source model implemented in ATMoSS approximates to 64kbit/s. One "golden−rule" of telecommunications is that any new equipment installed in a network must be compatible and interwork with the existing equipment. It is for this reason that ATM, unless installed as a separate overlay network, must be able to support today's ubiquitous 64kbit/s voice telephony traffic.

At 64kbit/s the time taken to fill a standard ATM cell assuming 48 user information bytes is 48 x 125μs, or 6ms. (Given that there are 8000 speech samples per second, each of 8 bits.) Thus there is a 6ms delay in filling a cell, usually called the packetisation or framing delay, and every 6ms a single source generates a new full cell. This implies that one CBR source at 64kbit/s generates approximately 167 standard ATM cells per second.

An ATM pipe at 155Mbit/s has the ability to carry 365566 (to the nearest integer) full cells per second. (Bit rate of the pipe, $155x10^6$, divided by the number of bits per cell, 53x8.) Therefore, one ATM pipe can carry approximately 2189 individual 64kbit/s sources.

Note: The ATM pipe used in the simulator carries ATM cells back–to–back with no overhead on the multiplex. This is shown as a type 1 multiplex in figure 3.9 below and is called a pure–ATM pipe. This type of multiplex has historically been favoured by the French telecommunications equipment manufacturers. The alternative to pure–ATM is to carry cells in a frame structure such as STM–1. STM–1 is the specification for a 155.520Mbit/s synchronous multiplex within the synchronous digital hierarchy (SDH). This is shown in simplified form as a type 2 multiplex in figure 3.9.

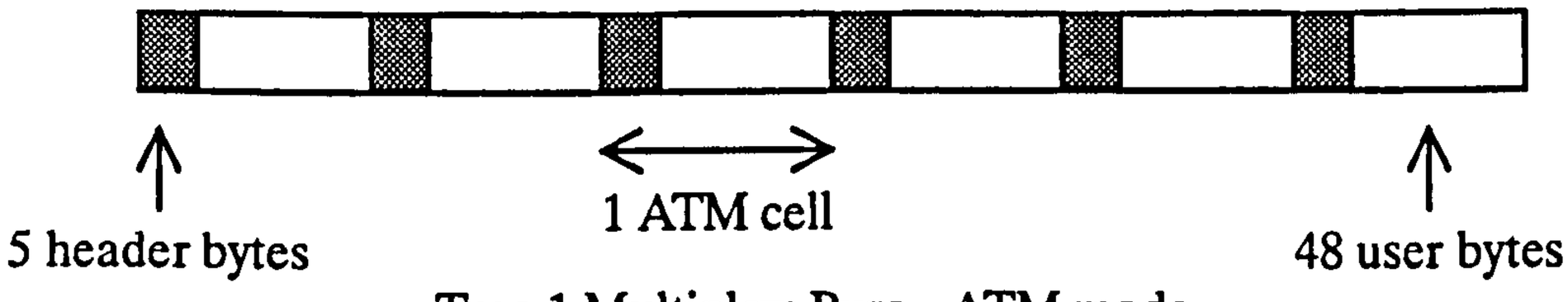



Figure 3.9: Pure–ATM and Frame Structured ATM Pipes

As the ATM pipe can carry 2189 individual 64kbit/s sources, it is divided up for the purposes of the CBR source generator into 2189 'time–slots'. The time–slot

mechanism is implemented with a modulo 2189 counter producing 2189 timeslots with numbers 0–2188. The simulator maintains a map of the 2189 timeslots to indicate which, if any, represent CBR sources. Any slot not used by a CBR source is available for VBR traffic.

Commands in the command file allow the number of CBR sources to be varied in the range 0 to 2189. Value 0 produces solely VBR traffic in the ATM cells, value 2189 produces solely CBR traffic. CBR cells are marked as such on entry to the simulator and have priority over VBR cells by entering a special reserved queue that is always emptied first in each switching stage. The motivation for the implementation of the CBR source model is to study for *n* CBR sources, how the service provided to the VBR sources deteriorates as *n* increases. This will be of real practical value to switch and network designers allowing dimensioning of systems for appropriate grades–of–service.

### 3.5.2.6 The Bearer and Queue Processors

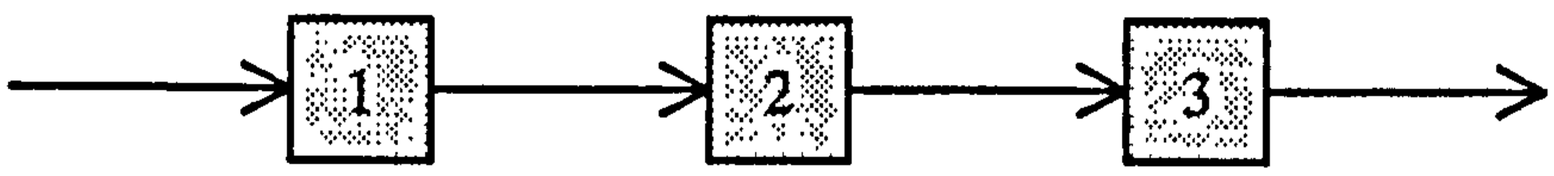
The input bearer processor, queue processor and output bearer processor provide the fundamental cell transfer capability of the simulator. Figure 3.6 shows the direction of **flow of cells** through the switch (input bearer → queueing stages → output bearer). This is opposite to the **order of processing** which is show in figure 3.12, the overall flow diagram.

There are 2 choices for the *order* of processing the bearer and queue processors.

1) Input bearer processor → queue processors *forwards* through the switch in order stage 1, stage 2, ... , stage n → output bearer processor

2) Output bearer processor → queue processors *backwards* through the switch in order stage n, stage n−1, ... , stage 1 → input bearer processor

The 2 choices are shown diagrammatically in figures 3.10 and 3.11 for a 3−stage switch architecture. The order of processing is shown as proceeding down the page.




Physical 3−stage switch architecture

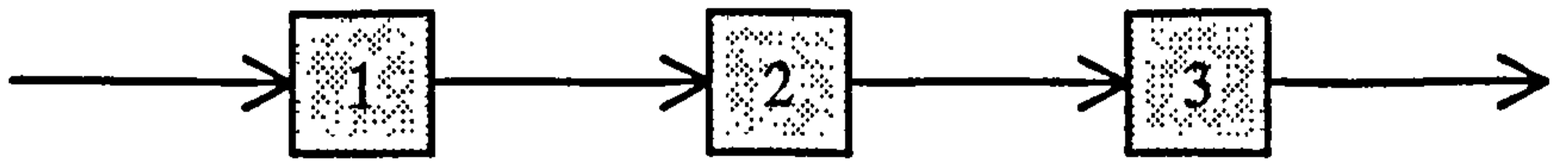
Cell arrives at time t

→ Queue processor 1 transfers cell forwards to stage 2 at time t

→ Queue processor 2 transfers cell forwards to stage 3 at time t

→ Queue processor 3 transfers cell forwards to output bearer processor at time t

Figure 3.10: Processing Order number 1, (forwards through the switch)




Physical 3–stage switch architecture

Cell arrives at time t

Queue processor 3 has no work at time t

Queue processor 2 has no work at time t

Queue processor 1 transfers cell forwards to stage 2 at time t

END OF CYCLE, SYSTEM TIME INCREMENTS

Queue processor 3 has no work at time t+1

Queue processor 2 transfers cell forwards to stage 3 at time t+1

Queue processor 1 has no work at time t+1

END OF CYCLE, SYSTEM TIME INCREMENTS

Queue processor 3 transfers cell forwards to output bearer processor at time t+2

Figure 3.11: Processing Order number 2, (backwards through the switch)

In both cases the synthesised system time increments after completing all required actions in all three processors.

In choice 1, shown in figure 3.10, it would be possible for an ATM cell to arrive, be processed through the switch and dispatched into the network on an outgoing multiplex **in the same unit of time.** This is because the queue processors (number of

queue processors = number of stages of the switch) would pass the cell forwards from one to the next until the output bearer is reached. In choice 2, as shown in figure 3.11, an ATM cell would be transferred through each of the queue processors in 1 unit of time, and would therefore be transported across the switch **in n units of time** where n is the number of stages comprising the switch.

It can be seen that choice 2 more closely represents the operation of a real ATM switch fabric even though the order of processing, being backwards with respect to cell flow, would initially seem to be the wrong way round.

### 3.5.2.7 Simulator Flow Chart

The main flow chart of a basic 2x2 architecture is shown in figure 3.12 below. The "Read Run–Control File" function allows the default parameters to be changed prior to the simulation run. Processing in this function finishes when either a "GO" command or the end of file character "*" are encountered. If the end of file has not been reached, the queues are cleared, all statistics reset and the run commenced. To resolve any unfairness introduced, the order of processing elements within a stage is alternated on each cycle.

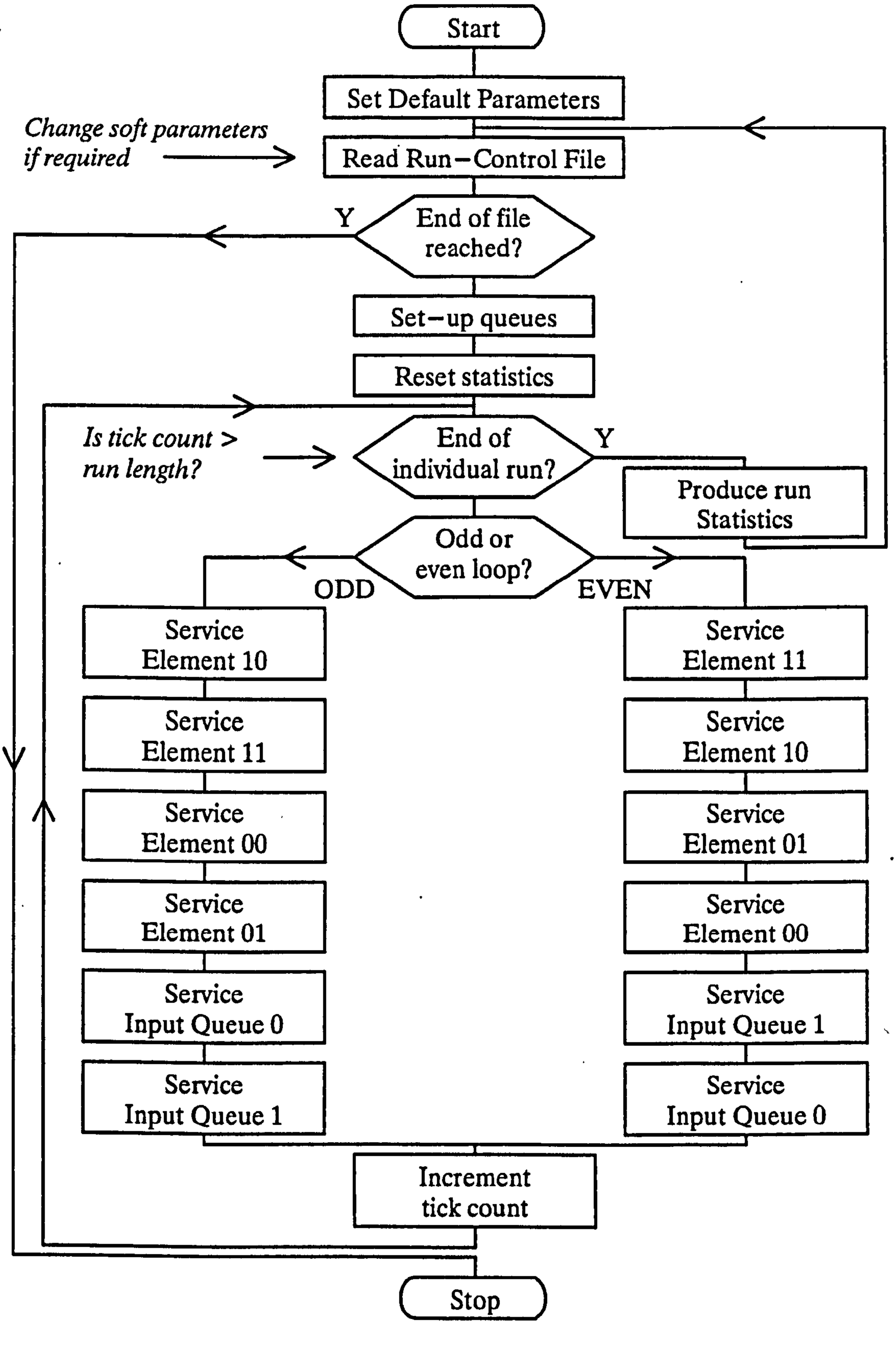



Figure 3.12: Main Flow–Chart of the Simulator

### 3.5.2.8 Simulator Results File

A wide range of statistics are maintained during a simulation run and presented as results when the run terminates. The simulator is also capable of multiple runs with the same soft parameters and can calculate the mean and standard deviation of up to 5 individual results over up to 10 consecutive runs by means of a spot–light function. Examples of results files from ATMoSS are given in appendix B.

Various de–bug and trace facilities are provided to aid validation. These include tracing the executed path through the code (FTRACE and RTRACE), printing queue contents (QTRACE) and highlighting specific events such as cell loss (ETRACE). The facilities may be turned on and off at compile time with *#define* statements.

```
#define QTRACEON    NO    /* Print queue contents each unit of time  */
#define VTRACEON    NO    /* Variables trace                         */
#define RTRACEON    NO    /* Routines trace                          */
#define FTRACEON    NO    /* Footprint trace                         */
#define DTRACEON    NO    /* Diagnostics trace                       */
#define ETRACEON    NO    /* Event reports                           */
```

Figure 3.13: Simulator Trace Options

## 3.5.3 Validation of the Simulator Operation

As discussed in section 3.3, it is essential to gain confidence in the operation of the simulator and the results produced. This was achieved in 3 stages. The first stage involved manual inspection of both the executed path through the code and the contents of variables such as the queues, and utilisation of the in–built diagnostic traces and footprint facilities.

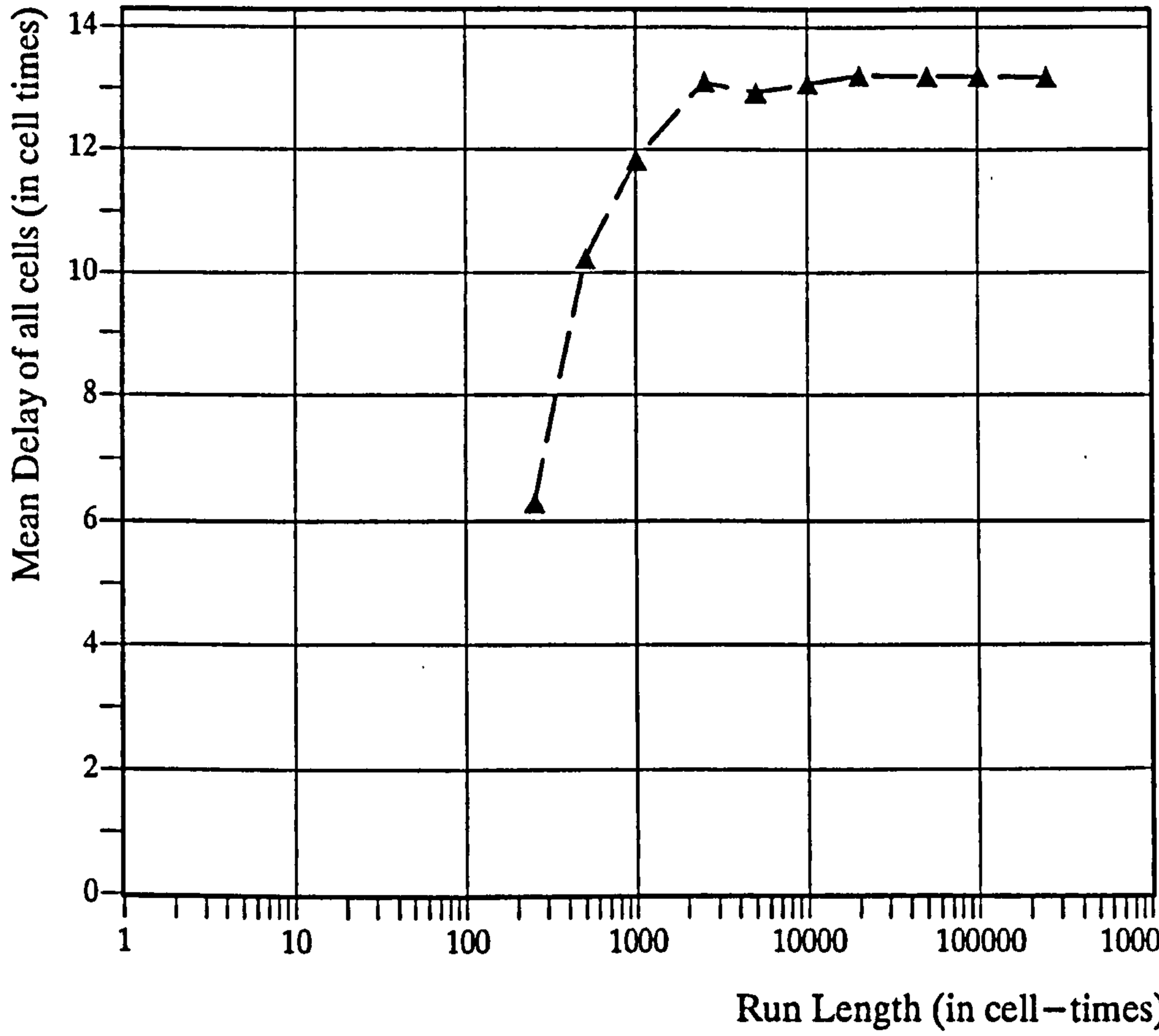
The second stage was to identify the minimum run length required to remove the effect of starting up the simulation with an empty switch in addition to removing as

far as possible the pseudo–random nature of the random number generator used. This allows a non–deterministic start to the simulation.

The third stage involved identifying a scenario capable of both being simulated and analysed mathematically. The analytic solution was then be compared with the simulator's results using a standard statistical significance test.

### 3.5.3.1 Determination of Minimum Run Length

Graph 3.1 shows the variation of cell delay with run length, the results of the experiment designed to establish the minimum run length. Too long a run time would clearly be inefficient in the use of CPU time and elapsed real–time. A compromise must be found to balance the stability and reliability of the results with the efficient use of available resources.




Graph 3.1: Variation of Cell Delay with Run Length

This initial experiment was carried out using full load on the incoming ATM pipes, also called multiplexes. The result *mean cell delay (all cells)* was used in this assessment as it has the longest 'settling−time' of all the results produced. Graph 3.1 shows that cell delay reaches a steady state value at a point somewhere between 20000 and 50000 units of time. It can be seen that there is no advantage to be gained from having run lengths in the region of hundreds of thousands of units. A standard run length of 25000 units was therefore established. This is the default value, but can be changed at run−time with SCL commands in the control file.

### 3.5.3.2 Comparison Between Analytic Results and Simulator Results

For the final validation test, the ATMoSS hard parameters were set to simulate a basic 2x2 ATM switching element, with 2 incoming and 2 outgoing ATM bearers. The soft parameters were set to transfer all traffic from both incoming bearers to one outgoing bearer. The queue at the output bearer had 4 queue places. The load on each incoming bearer was set to the same value and varied from 10% to 95% in steps of 5% requiring *18* experimental runs. The scenario is shown in figure 3.14, the experimental and analytic results are given in table 3.1.

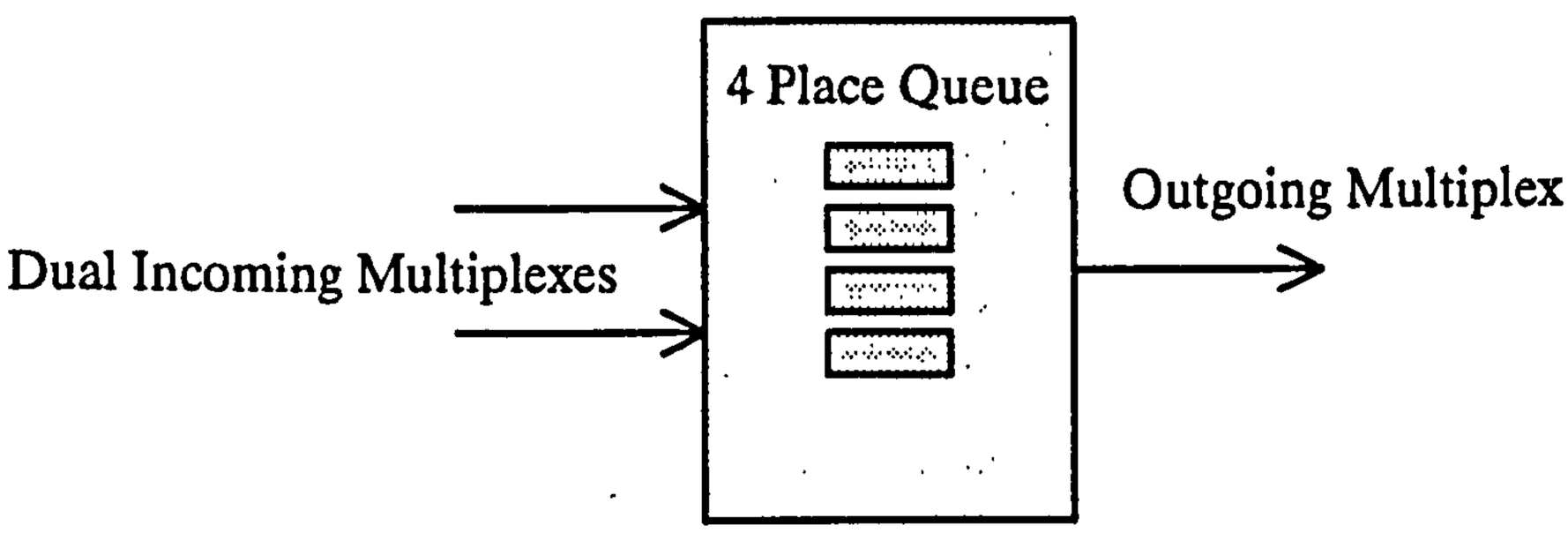



Figure 3.14: Validation Scenario

The parameter *average queue occupancy* of the final stage was recorded. During the course of this stage of the validation, it was found that it is insufficient to specify the average queue occupancy with a single parameter.

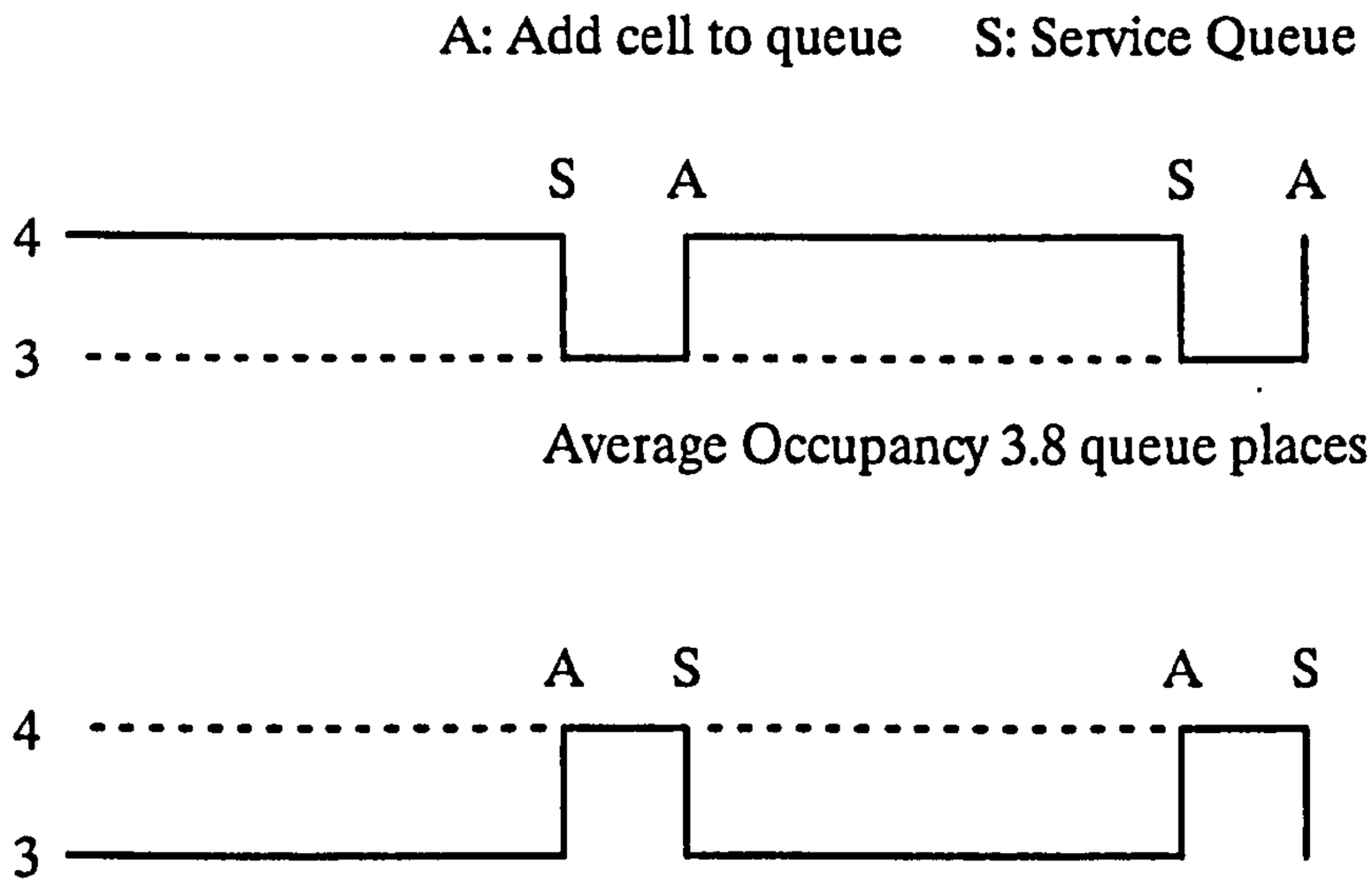



Figure 3.15: Queue Occupancy shown as dependant on where in the cycle servicing occurs with respect to appending new cells.

It can be seen from figure 3.15 that the queue occupancy varies according to where in the cycle appending further cells to the queue (A) is done with respect to where the queue servicing (S) is done.

This required the development of a new theory that "to completely specify queue occupancy in an ATM switching element requires 2 parameters, the *average occupancy before queue servicing* and *average occupancy after queue servicing.*"

In order to produce the model of the scenario for the analytic solution, the cycle time (1 unit) is split into 2 components. Let $\lambda$ represent the time between appending and servicing, and $\mu$ between servicing and appending as shown in figure 3.16.

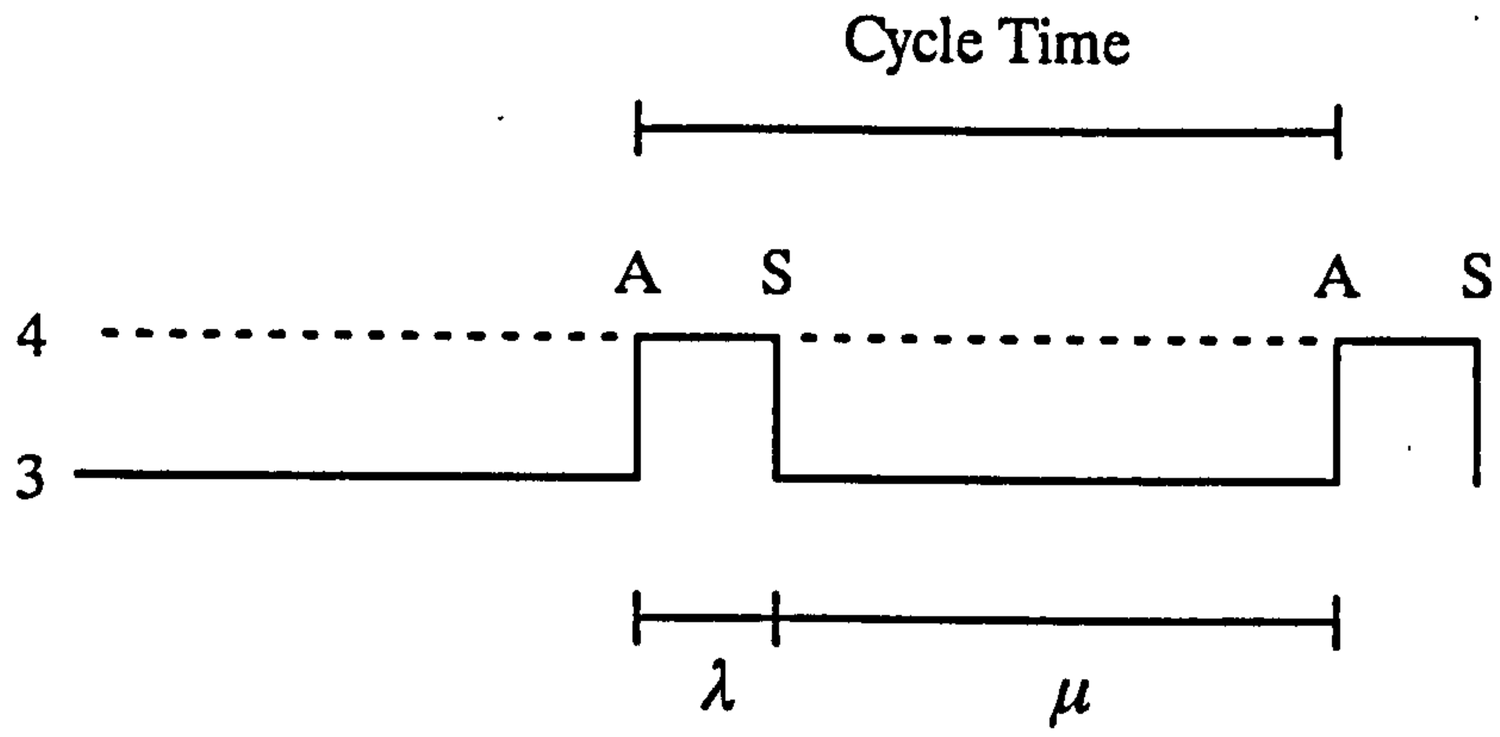



Figure 3.16: Timing of Appending to and Servicing the Queue

where $\lambda+\mu=1$, $0<\lambda<1$ and $0<\mu<1$

$\lambda$ and $\mu$ therefore completely specify the timing of the queue servicing with respect to the queue append.

The analytic values of expected queue occupancy can then be produced as follows. Firstly, 2 Markov chains were produced, one to model the event *queue arrivals* (figure 3.17) and one to model *queue departures* (figure 3.18).

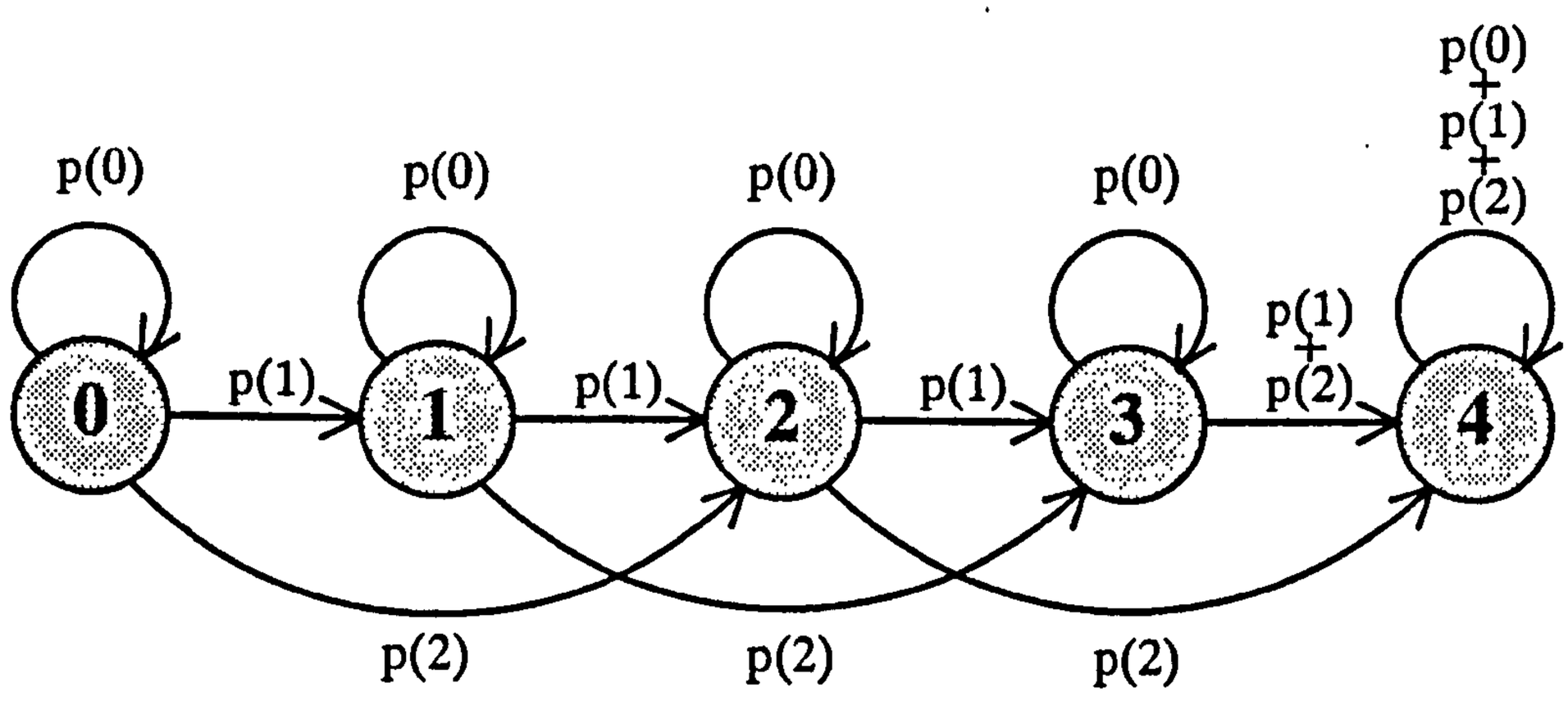



Figure 3.17: Markov Chain Representation of Queue Arrivals

Note: p($x$) represents the probability of $x$ cells arriving in one unit of time.

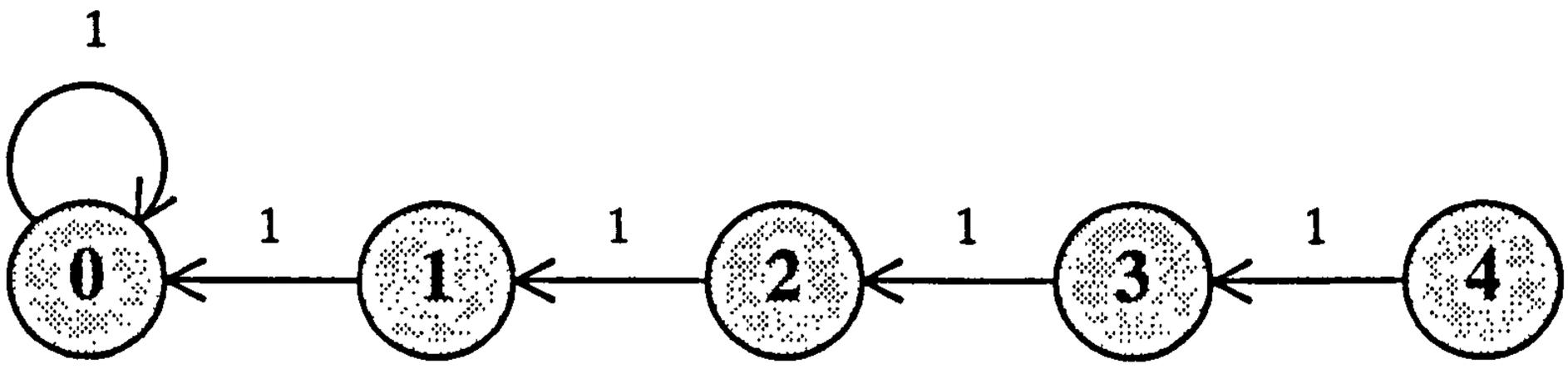



Figure 3.18: Markov Chain Representation of Queue Departures

Secondly, from the Markov chains shown in figures 3.17 and 3.18 above, corresponding state transition matrices can be produced. These are shown in figures 3.19 and 3.20.

| | | At time *t+1* | | | | |
|---|---|---|---|---|---|---|
| | | S0 | S1 | S2 | S3 | S4 |
| At time *t* | S0 | p(0) | p(1) | p(2) | 0 | 0 |
| | S1 | 0 | p(0) | p(1) | p(2) | 0 |
| | S2 | 0 | 0 | p(0) | p(1) | p(2) |
| | S3 | 0 | 0 | 0 | p(0) | p(1) +p(2) |
| | S4 | 0 | 0 | 0 | 0 | p(0) +p(1) +p(2) |

Figure 3.19: $P_A$, State Transition Matrix for Queue Arrivals

| At time *t* \ At time *t+1* | S0 | S1 | S2 | S3 | S4 |
|---|---|---|---|---|---|
| S0 | 1 | 0 | 0 | 0 | 0 |
| S1 | 1 | 0 | 0 | 0 | 0 |
| S2 | 0 | 1 | 0 | 0 | 0 |
| S3 | 0 | 0 | 1 | 0 | 0 |
| S4 | 0 | 0 | 0 | 1 | 0 |

Figure 3.20: $P_D$, State Transition Matrix for Queue Departures

*For example, in the period when cells arrive at the queue (figure 3.19), the probability of remaining in state 3, i.e. with 3 cells in the queue, is p(0) [the probability that no cells arrive].*

Thirdly, the values of p(0), p(1) and p(2) are calculated for the required values of *a* (the arrival rate of cells on the incoming multiplex where 0<=a<=1) from the binomial probability model,

$$p(x) = {}^2C_x \,.\, (1-a)^{2-x} \,.\, a^x$$

where $0 \leq x \leq 2$, and ${}^2C_x$ is the binomial coefficient, $\frac{2!}{x!(2-x)!}$

The values can then be substituted into the matrices $P_A$ and $P_D$. As discussed in the first paragraph of section 3.5.3.2, 18 different values of the arrival rate were used for the validation, $a = 0.5$ to 0.95 in steps of 0.05. This results in 18 $P_A$ $P_D$ pairs.

The 18 $P_A$ $P_D$ pairs are then used to produce the required parameters *average occupancy before queue servicing* and *average occupancy after queue servicing* as follows.

Let $\mathbf{p}_t$ be the row vector containing the queue state probabilities at time t,

$$\mathbf{p}_t = \{ p(S_0)_t, p(S_1)_t, p(S_2)_t, p(S_3)_t, p(S_4)_t \}$$

where $p(Sm)_t$ is the probability of $m$ cells in the queue at time t.

$$\textit{therefore} \qquad \sum_{m=0}^{4} p(Sm)_t = 1$$

If a system can be described by a single state transition matrix, P, from [70]

$$\mathbf{p}_{\text{at time t +1}} = \mathbf{p}_{\text{at time t}} \cdot \mathrm{P}$$

In this case however, there are 2 matrices and the system can be described in 2 ways,

$$\mathbf{p}_{\text{at time t} + \lambda} = \mathbf{p}_{\text{at time t}} \cdot \mathrm{P_A} \quad - \{1\}$$

$$\mathbf{p}_{\text{at time t +1}} = \mathbf{p}_{\text{at time t} + \lambda} \cdot \mathrm{P_D} \quad - \{2\}$$

when the order of processing is *append, service*, and

$$\mathbf{p}_{\text{at time t} + \mu} = \mathbf{p}_{\text{at time t}} \cdot \mathrm{P_D} \quad - \{3\}$$

$$\mathbf{p}_{\text{at time t +1}} = \mathbf{p}_{\text{at time t} + \mu} \cdot \mathrm{P_A} \quad - \{4\}$$

when the order of processing is *service, append*.

If we denote $\mathbf{p}\,\alpha$ as the row vector containing the queue occupancy probabilities *after* queue servicing, (the end−state of equations {1} and {2} is with the queue serviced)

and $\mathbf{p}\,\beta$ as the row vector containing the queue occupancy probabilities *before* queue servicing,

equations {1} and {2} may be combined to produce

$$\mathbf{p}\,\alpha_{t+1} = \mathbf{p}\,\alpha_t \,.\, P_A \,.\, P_D \quad - \{5\}$$

and equations {3} and {4} may be combined to produce

$$\mathbf{p}\,\beta_{t+1} = \mathbf{p}\,\beta_t \,.\, P_D \,.\, P_A \quad - \{6\}$$

As $\lambda$ and $\mu$ no longer appear in equations {5} and {6}, they may now take any values satisfying the following 3 conditions: $\lambda+\mu=1$, $0<\lambda<1$ and $0<\mu<1$. Equations {5} and {6} are now independent of the timing of the queue servicing w.r.t the queue appending, and we may write

$$\mathbf{p}\,\alpha_{t+n} = \mathbf{p}\,\alpha_t \,.\, [\, P_A \,.\, P_D \,]^n \quad - \{7\}$$

$$\mathbf{p}\,\beta_{t+n} = \mathbf{p}\,\beta_t \,.\, [\, P_D \,.\, P_A \,]^n \quad - \{8\}$$

As $n \to \infty$, the system reaches equilibrium and the vectors $\mathbf{p}\,\beta_{t+n}$ and $\mathbf{p}\,\alpha_{t+n}$ contain the steady state probabilities. The values in the vectors $\mathbf{p}\,\alpha_t$ and $\mathbf{p}\,\beta_t$ no longer have significance (any of the 5 possible start states will yield the same result) and may therefore be assigned arbitrarily, {1, 0, 0, 0, 0} for example.

Equations {7} and {8} may then be solved:

$$\mathbf{p}\,\alpha_n = \{1, 0, 0, 0, 0\} \,.\, [\, P_A \,.\, P_D \,]^n \quad - \{9\}$$

$$\mathbf{p}\,\beta_n = \{1, 0, 0, 0, 0\} \,.\, [\, P_D \,.\, P_A \,]^n \quad - \{10\}$$

$[\, P_A \,.\, P_D \,]^n$ and $[\, P_D \,.\, P_A \,]^n$ were computed by a matrix–multiplication program written in C with n=200. (After 200 multiplications the resulting matrix had reached equilibrium.)

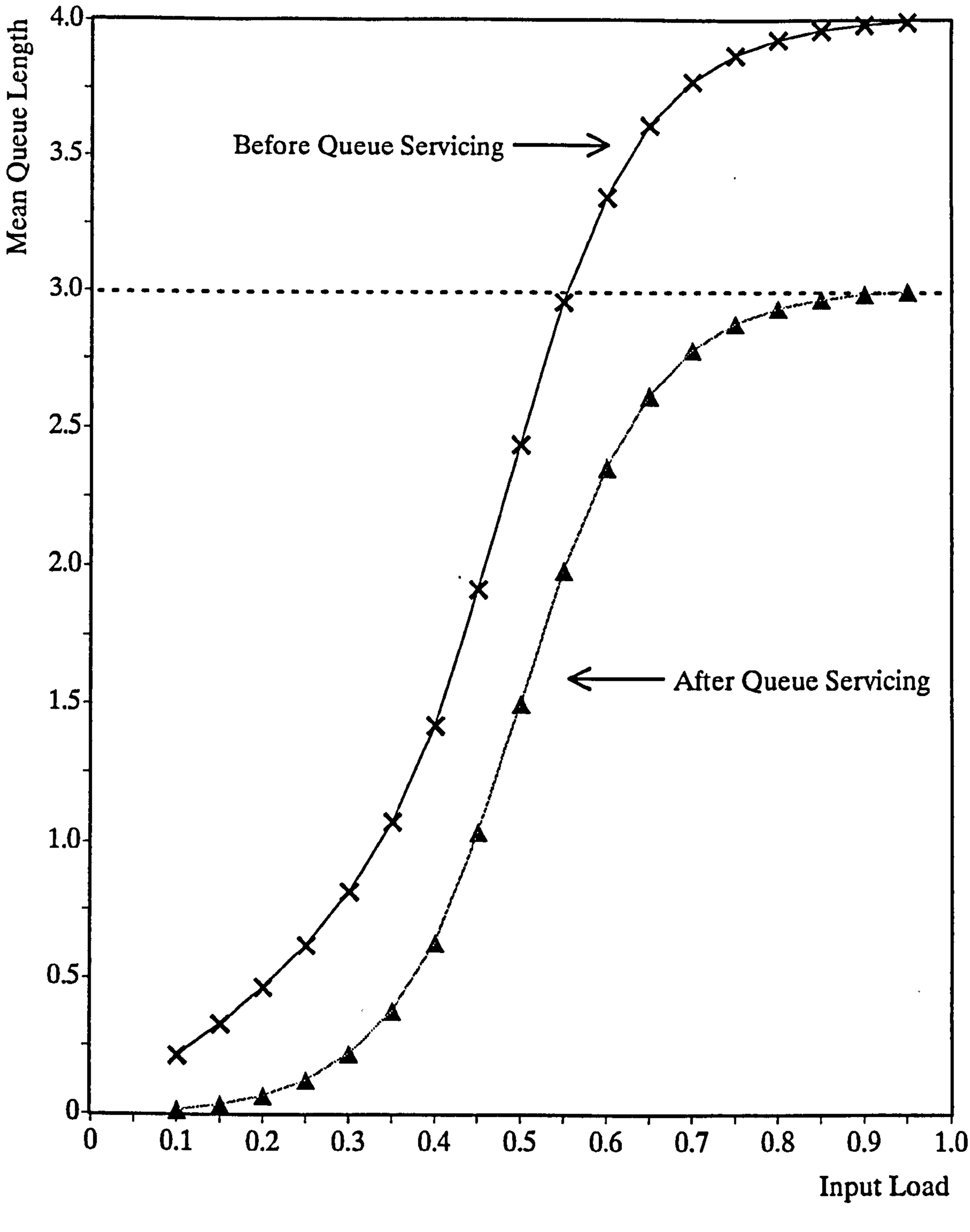
The 18 $P_A$ $P_D$ pairs produce 36 discrete probability distributions that are the steady state probabilities of a particular queue length $m$, represented by the notation

$$p(Sm) \text{ where } 0 \leq m \leq 4.$$

The *expected* queue length (or queue occupancy) for the system is the mean of the probability mass function (pmf) and is given by

$$\textit{Expected queue length} = \sum_{m=0}^{4} m.\ p(Sm)$$

The 36 theoretical expected queue lengths are shown on graph 3.2 and in table 3.1.




Graph 3.2: Theoretical Expected Queue Length vs. Input Load (4 queue places, 2 inputs)

| | Queue Lengths Before Servicing Queue | | | Queue Lengths After Servicing Queue | | |
|---|---|---|---|---|---|---|
| Arrival Rate | Theoret. | Experimental Mean | Std. Dev. | Theoret. | Experimental Mean | Std. Dev. |
| 0.10 | 0.01250 | 0.01180 | 0.000804 | 0.21250 | 0.21057 | 0.003234 |
| 0.15 | 0.03214 | 0.03261 | 0.001313 | 0.33214 | 0.33387 | 0.003657 |
| 0.20 | 0.06661 | 0.06715 | 0.001940 | 0.46660 | 0.46660 | 0.004643 |
| 0.25 | 0.12439 | 0.12387 | 0.004470 | 0.62431 | 0.62442 | 0.006203 |
| 0.30 | 0.22044 | 0.21676 | 0.009002 | 0.81999 | 0.81518 | 0.011861 |
| 0.35 | 0.37986 | 0.38126 | 0.009496 | 1.07773 | 1.07963 | 0.012676 |
| 0.40 | 0.63759 | 0.63958 | 0.015694 | 1.42947 | 1.43059 | 0.018136 |
| 0.45 | 1.01989 | 1.02132 | 0.024346 | 1.89477 | 1.89564 | 0.027515 |
| 0.50 | 1.50000 | 1.50165 | 0.021475 | 2.43750 | 2.43885 | 0.022953 |
| 0.55 | 1.98010 | 1.98224 | 0.017468 | 2.95497 | 2.95742 | 0.018466 |
| 0.60 | 2.36241 | 2.35647 | 0.016209 | 3.35429 | 3.34822 | 0.016580 |
| 0.65 | 2.62014 | 2.62326 | 0.012244 | 3.61800 | 3.62108 | 0.012160 |
| 0.70 | 2.77956 | 2.78072 | 0.006721 | 3.77911 | 3.78022 | 0.006762 |
| 0.75 | 2.87561 | 2.87427 | 0.004901 | 3.87553 | 3.87412 | 0.004891 |
| 0.80 | 2.93339 | 2.93212 | 0.003062 | 3.93338 | 3.93202 | 0.003067 |
| 0.85 | 2.96790 | 2.96718 | 0.001313 | 3.96791 | 3.96710 | 0.001304 |
| 0.90 | 2.98753 | 2.98736 | 0.001281 | 3.98755 | 3.98728 | 0.001281 |
| 0.95 | 2.99726 | 2.99692 | 0.000346 | 3.99727 | 3.99685 | 0.000346 |

Table 3.1: Theoretical and Experimental Expected Queue Lengths

Note: Experimental results are the mean of 10 simulation runs at the same load

By inspection, the results in table 3.1 suggest that the simulator is capable of giving good results with a high degree of correlation to the theoretical results. To test this statistically the null hypothesis $H_0$ was established such that

$H_0$ = *"there is no difference between the theoretical and experimental results "*

and subjected to a two−sided significance test.

The statistical test of significance allows the separation of a difference that could have occurred by chance, from one that could not. It is not possible statistically to prove

equality, only to disprove it with a certain level of confidence. The level of confidence is given a statistical value called the significance value, and from [71] may be calculated:

$$\textit{Significance value, } t = \frac{|x - \mu|}{\sigma}.\sqrt{n}$$

where $x$ = *sample mean*
$\mu$ = *theoretical mean*
$n$ = *sample size*
$\sigma$ = *standard deviation of sample*

The higher the significance value $t$, the more confident the rejection of $H_0$.

From t−tables, the significance value at the 5% level for 9 degrees of freedom (number of trials minus 1) is 2.26 . Substituting the values from table 3.1 into the formula leads us to being unable to reject $H_0$ for 33 out of the 36 results and thus assert for these results that *there is no difference between the simulator results and the theoretical results.*

However, it is necessary to reject $H_0$ for the following points:

0.10 arrival rate queue sampled before servicing

0.95 arrival rate queue sampled before servicing

0.95 arrival rate queue sampled after servicing

As all of the experimental values are very close to the theoretical values, an explanation was sought to explain the deviation at the 2 ends of the input load range. In all 3 cases of rejection the experimental value is lower than the theoretical value. In the 2 cases of rejection at the 0.95 load the relatively high significance value is

caused principally by the very small standard deviation. (Note2)

During initial simulator runs a trend was observed of low actual input load compared to the expected input load. The spectral properties of the random number generator used in the simulator will not be perfect and it is possible that the effects of this are more pronounced at the 2 ends of the input load range. An investigation was carried out to establish whether this was the cause of the 3 rejections.

A series of 10 further simulation runs were performed at input loads 0.1 and 0.95.

Analysing the results at 0.1 input load showed that

the load on Input 0 was approximately 0.0993 instead of the desired 0.1
the load on Input 1 was approximately 0.09995 instead of the desired 0.1

so the load was accurate. The 2 significance values obtained in the repeated trial were actually lower than 2.26 and therefore this time $H_0$ cannot be rejected.

At the 0.95 input load,

the load on Input 0 was approximately 0.95007 instead of the desired 0.95
the load on Input 1 was approximately 0.94970 instead of the desired 0.95

This produces an average load over the 2 multiplexes of 0.949885, which is an error of 0.012%. The 2 significance values were still higher than the 5% significance figure of 2.26 being 4.02 (before queue servicing) and 3.03 (after queue servicing).

<u>Discussion: Results of Significance Test at 0.95 Load</u>

The 2 significance values calculated on the results of the second trials were **3.03** for queue sampling after queue servicing and **4.02** for queue sampling before queue servicing.

(Note2).The standard deviation is small in all cases which leads to the conclusion that the simulator is producing consistent results.

The 3.03 result allows acceptance of $H_0$ at the 1% significance level. The 4.02 result allows acceptance of $H_0$ at the 0.2% significance level. All other results are accepted at the standard 5% significance level.

The significance level in this type of test represents the probability of making a Type I error, *rejecting $H_0$ when it is true*, which at 5% is 0.05. Changing the significance level to reduce the probability of a Type I error increases the probability of a Type II error, *accepting $H_0$ when it is false*. The significance level of a test is selected according to the relative importance of the 2 types of error.

In this case it is more important to prevent the possibility of a Type II error than a Type I error, therefore the 5% significance level is appropriate and on the basis of the test it is necessary to reject the hypothesis for 0.95 load.

However, the test may have failed for reasons other than inaccuracies in the simulator. Several possibilities for the cause of failure exist:

1) The theoretical figure could have been subject to rounding or other errors during calculation.

2) The error in the experimental results could have been caused by a lower than required input load which is more critical at the high end of the load range.

3) The error could have been introduced by simulation inaccuracies which are more apparent at the high end of the load range.

4) The discrepancy, which puts the actual queue occupancy below the expected queue occupancy, could have been as a

result of the simulator warm–up period. During this period the queue occupancy would be lower than when the system had reached steady state conditions.

Possibility 1: Unlikely. If the theoretical figures at the other loads are to be believed there is no reason to question the figure at 0.95 load.

Possibility 2: Most likely. The error on the input load was 0.012% on the low side. The errors on the queue occupancies as table 3.2 shows were 0.0163% and 0.0187%. Between one half and three quarters of the error could therefore have been introduced by the low input load.

Possibility 3: Somewhat possible. There is no difference however between the simulator operation at the high end of the load range from that at the low end of the load range.

Possibility 4: Possible. If the warm–up period had caused problems it would have been to reduce the average queue occupancy figures which is what is observed.

| [illegible] | Sampling Before Servicing | Sampling After Servicing |
|---|---|---|
| Actual | 3.99662 | 2.99670 |
| Theoret. | 3.99727 | 2.99726 |
| Differ-ence | 0.00065 | 0.00056 |
| % Error | 0.0163 % | 0.0187 % |

Table 3.2: Actual and Theoretical Queue Occupancies for 0.95 Load

By inspection it would be unreasonable to dismiss the values entirely, on the basis that they are not representative of the theoretical figures when in fact good correlation can be seen. It is concluded therefore that the simulator is performing correctly up to 0.95

load. The results at 0.95 load may provide guidance but on the basis of the significance test should be regarded with a degree of caution.

This chapter has described in detail the design philosophy of the simulator used during this research project, its operating modes and the method of its validation. Not yet covered are the source models used to generate cell arrivals. The three models investigated, the negative exponential, binomial and the video source models are discussed in the next chapter.

# Chapter 4 Variable Bit–Rate Source Models

The principal characteristic of ATM as a switching and multiplexing technique is its ability to handle both variable and constant bit rate sources. A realistic ATM switch simulator must therefore be able to reproduce cell arrivals as if they were from these source types. Section 2.5 presents a review of two source models from the literature designed to reproduce video traffic in a simulator. Section 3.5.2.5 describes the 64 kbit/s constant bit rate sources implemented in ATMoSS. In this chapter three variable bit rate source models are described and the effect of two of them, the binomial and negative exponential source models, on switch block performance is discussed.

## 4.1 The Video Source Model

The model chosen to represent the arrival of cells from video sources is a 3–state geometrically modulated deterministic process as described in [42]. It is claimed that this model represents the best approximation to video sources currently available.

A schematic diagram of the model's states are shown in figure 4.1.

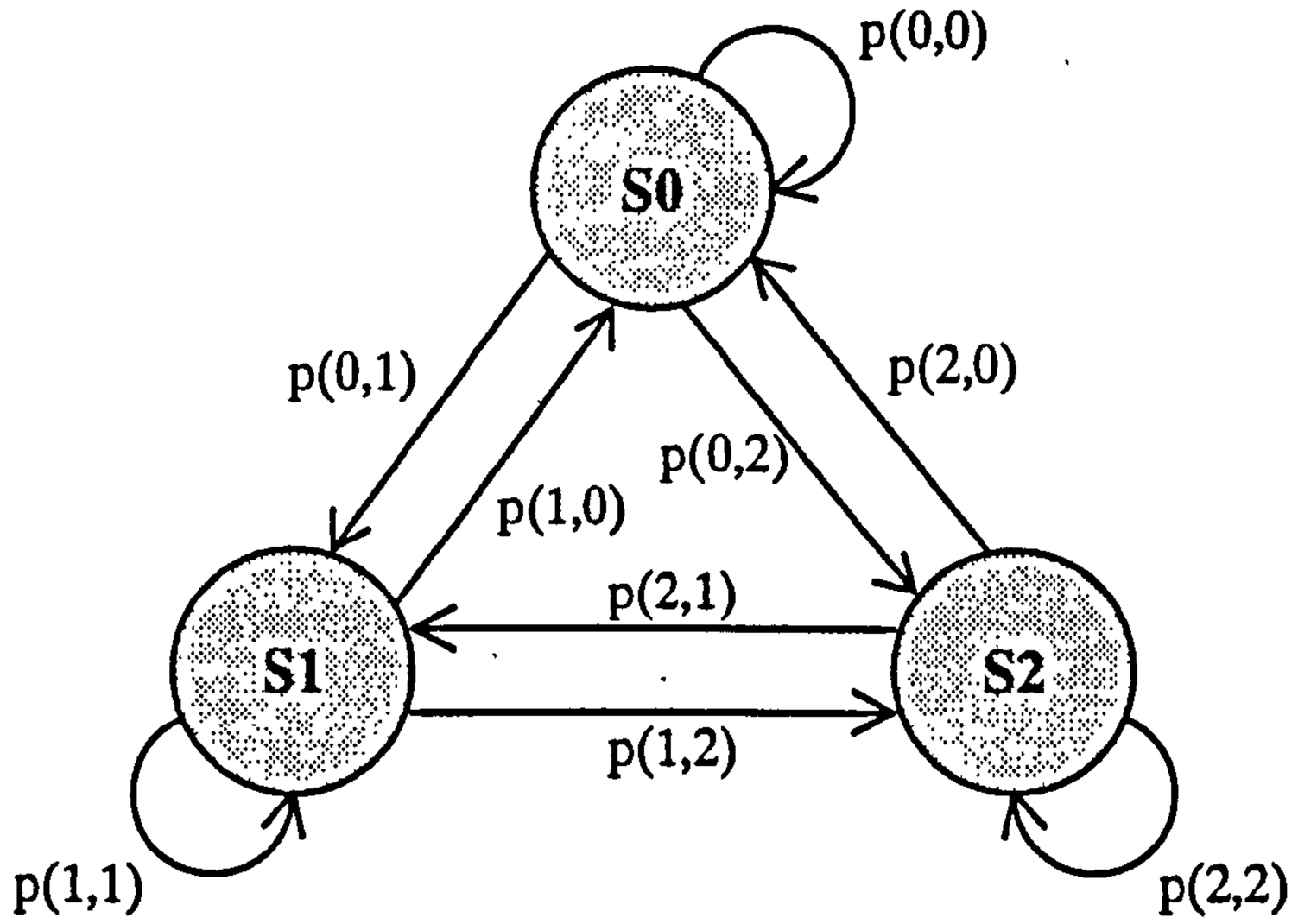



Figure 4.1: 3−State GMDP Video Source Model States

The model is always in 1 of 3 states, S0, S1 or S2. Each state has a cell sending rate represented by N0, N1 and N2. The sojourn time in each state is geometrically distributed and the cell sending rates are deterministic, hence the name of the model.

The transition probabilities between the 3 states are represented by the notation:

$$p(\text{initial state, final state})$$

where $p(0,0) + p(0,1) + p(0,2) = 1$

$p(1,0) + p(1,1) + p(1,2) = 1$

and $p(2,0) + p(2,1) + p(2,2) = 1$

The actual values of N0, N1, N2 and the 9 state transition probabilities were determined by the author of [42] from studies on real video signals and are:

$N0 = 5000$ cells per second

$N1 = 10000$ cells per second

$N2 = 35000$ cells per second

$p(0,0) = 0.8$ $\quad p(0,1) = 0.1$ $\quad p(0,2) = 0.1$

$$p(1,0) = 0.1 \qquad p(1,1) = 0.8 \qquad p(1,2) = 0.1$$

$$p(2,0) = 0.1 \qquad p(2,1) = 0.1 \qquad p(2,2) = 0.8$$

### 4.1.1 GMDP Implementation Problem

The implementation of this source model presents a problem of determining when in the time domain to check for a state transition. A pure ATM pipe at 155 Mbit/s carries approximately 365566 cells per second. Thus a single GMDP source model generates (approximately):

at rate N0, 1 full cell every 73 cells
at rate N1, 1 full cell every 37 cells
at rate N2, 1 full cell every 10 cells

If the check for a state transition is made every 10 cells (the idle time of the high rate state) it is likely that the sojourn time in the low rate state will be so small (in terms of cell intervals) that this state will rarely generate a cell. However, if the state transition is checked for every 73 cells (the idle time of the low rate state) the high rate will dominate the model.

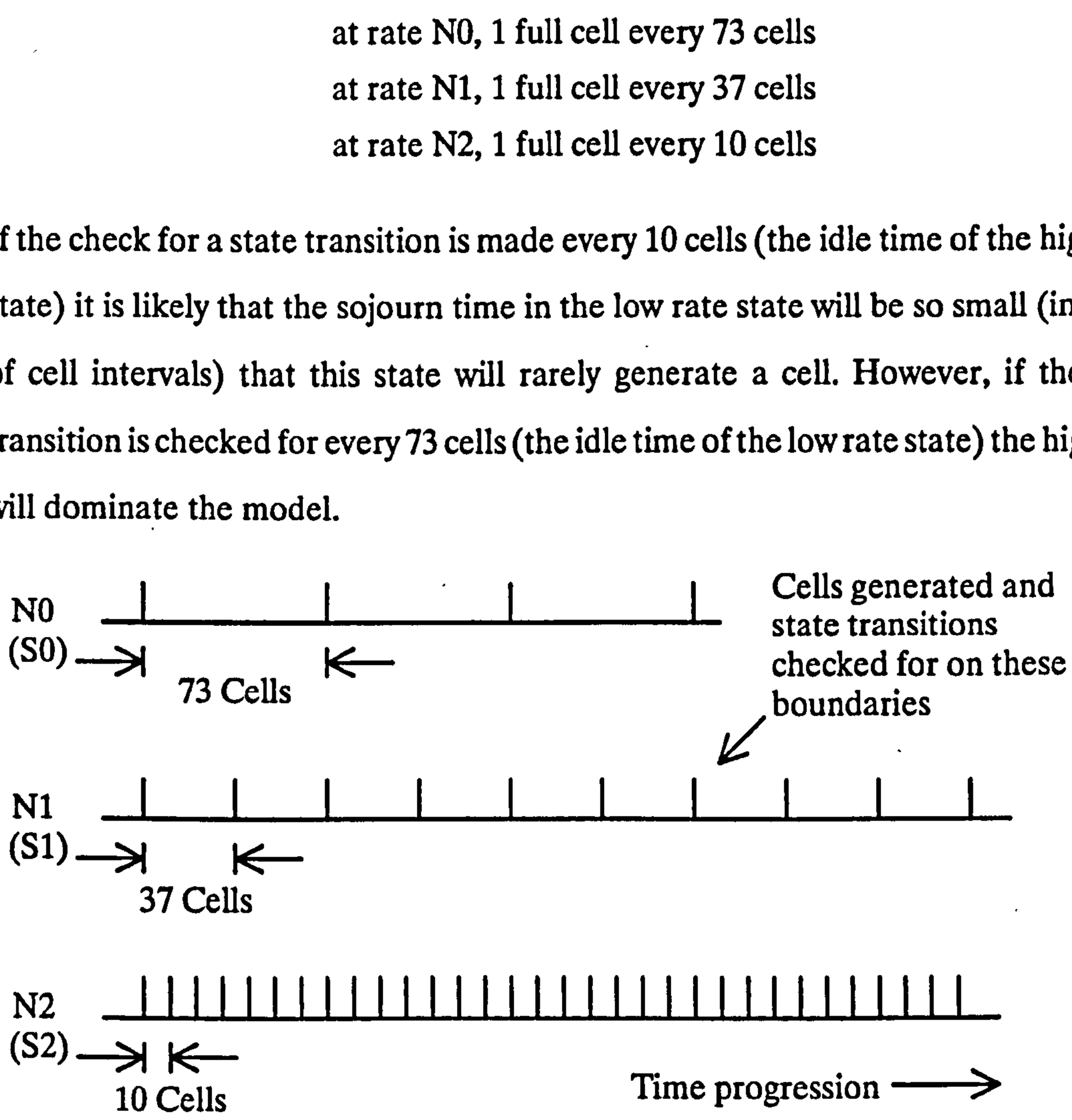



Figure 4.2: GMDP State Timing Relationship

The solution is to check for a state transition immediately after generating a full cell. In this way, the expected sojourn time in the high state, S2, will be much smaller than that in the low rate state but the number of cells generated in each state will be balanced (with the state transition probabilities as stated). The relative timing of the cell generation and check for state transition is shown in figure 4.2.

### 4.1.2 Results from the Video Source Model Implementation

Implementing the model described in section 4.1 above gives a mean cell sending rate per source of approximately 9000 cells per second. The detailed results from a 10 second simulation of a single replication of the model are:

| | |
|---|---|
| No. of cells throughput | = 90159 |
| No. of seconds simulated | = 10 |
| Mean no. of cells per second | = 9016 |
| Cell sending rate LOW | = 5000 cells per second |
| Cell sending rate MEDIUM | = 10000 cells per second |
| Cell sending rate HIGH | = 35000 cells per second |
| Bearer capacity | = 365566 cells per second |
| Percentage load on bearer | = 2.5% |
| No. of cells sent at low rate | = 29868 |
| No. of cells sent at medium rate | = 30086 |
| No. of cells sent at high rate | = 30205 |
| Actual data rate | = 3.462 Mbit/s |

The effective data rate for a single video source can be seen to be approximately 3.5Mbit/s which in turn is approximately 2.5% of the 155Mbit/s bearer capacity. It can also be seen from the results that each of the states generates about one third of the

total number of cells. This gives an assurance that the model parameters are balanced correctly and no individual state is dominating the model.

These individual source model results (effective bandwidth and bearer occupancy) become significant in subsequent investigations where the effect of multiplexing multiple source models onto one ATM bearer is studied (section NO TAG, Multiplexing Multiple Video Source Models).

## 4.2 The Binomial Source Model

This is the simplest of the 3 source models to implement. The name of the source model is derived from the fact that in repeated simulations of N cells, the number of full cells arriving will form a binomial probability distribution with parameters (N,p) where p is the arrival rate of cells and $0 \leq p \leq 1$.

The model is implemented as a simple Bernoulli trial with parameter p, repeated once per cell time.

The motivation for the implementation of this source model is its simplicity, giving a simulation speed−up over the more computationally onerous video and negative exponential source models. How the performance of the ATM switch fabric varies with different source models then becomes an important subsequent investigation.

## 4.3 The Negative Exponential, or Markovian Source Model

Much of the theoretical work on queueing theory has been done using negative exponential, or Markovian, arrival distributions. This distribution is often used as it is a realistic representation of 'real' arrivals, for example the arrival of customers at a Post Office counter or telephone calls at an exchange.

This distribution also yields analytically tractable results. The Erlang blocking formula for example can be proved assuming a negative exponential arrival distribution. It is generally understood that the blocking formula also holds good for other arrival distributions but this is much more difficult to prove mathematically.

One motivation for the implementation of this source model therefore is the potential to compare experimental results with published theoretical results thus increasing confidence in the simulation.

The negative exponential distribution used in the simulator is a modified version of the form:

$$y = k\,e^{-kx}$$

where k is the arrival rate. The modification is required for 2 reasons:

1) The basic formula gives inter–arrival times as any positive value. The simulation can only work with inter–arrival times of positive integers.

2) The simulator has a minimum inter–arrival time of 1 cell duration.

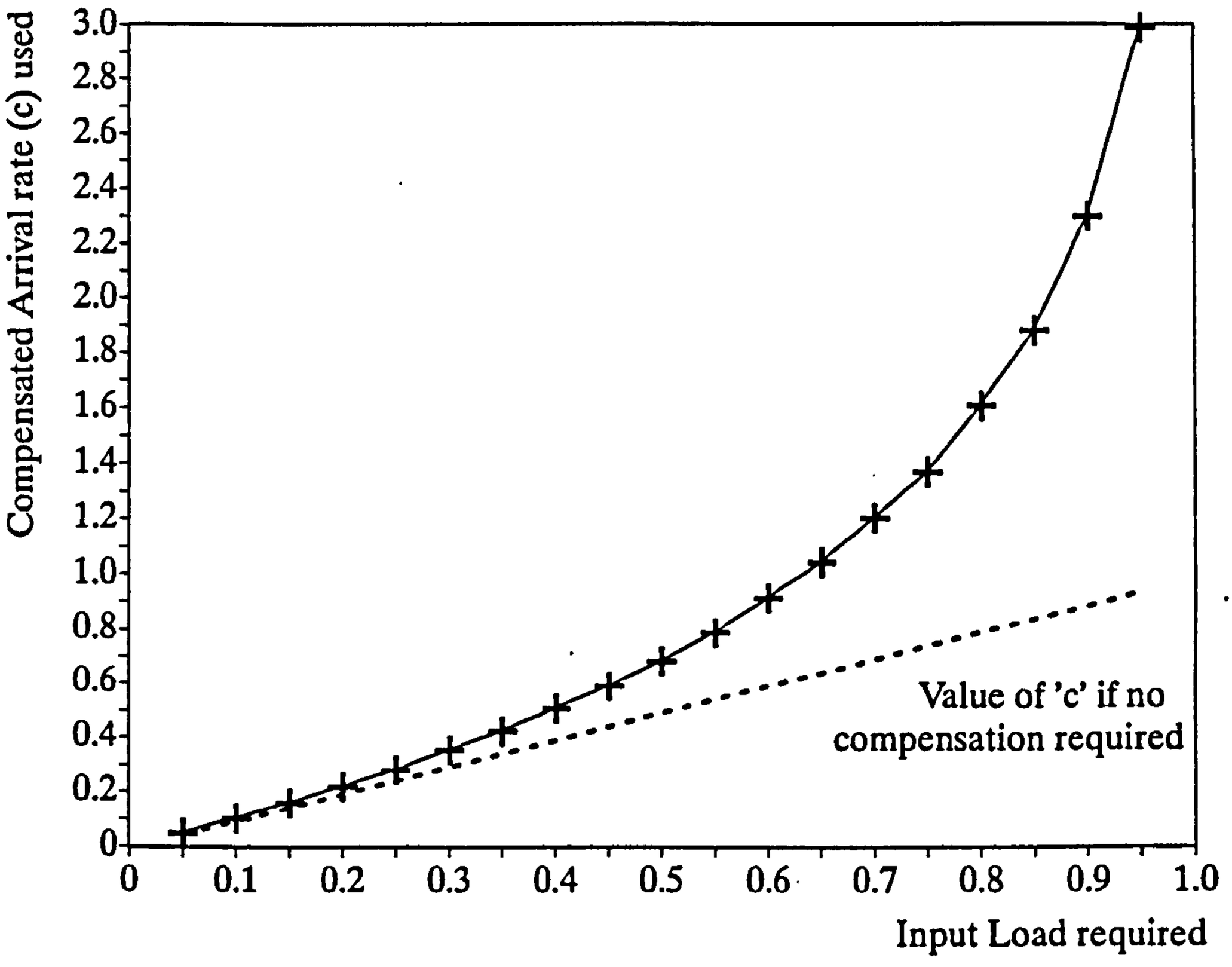
If $t$ is the inter–arrival time from a standard negative exponential distribution and $[t]$ represents the integral part of $t$, the required modification takes the waiting time (in cell times) to the next cell as:

$$\text{waiting time to next cell} = 1 + [t]$$

As $( 1 + [t] ) > t$, the modification has the effect of producing a lower actual arrival rate than that desired and so a compensation factor must be applied. The final formula used to calculate the waiting time to the next cell is:

$$\text{waiting time to next cell} = -\frac{\log_e(r)}{c} + 1$$

where r is an evenly distributed random number in the range $0 < r < 1$ and c, instead of being the arrival rate as expected, is the *compensated* arrival rate. The value of c is clearly a function of input load and was found by repeated trials at specific loads until the desired load was achieved. The pseudo–random number generator used is described in appendix D.



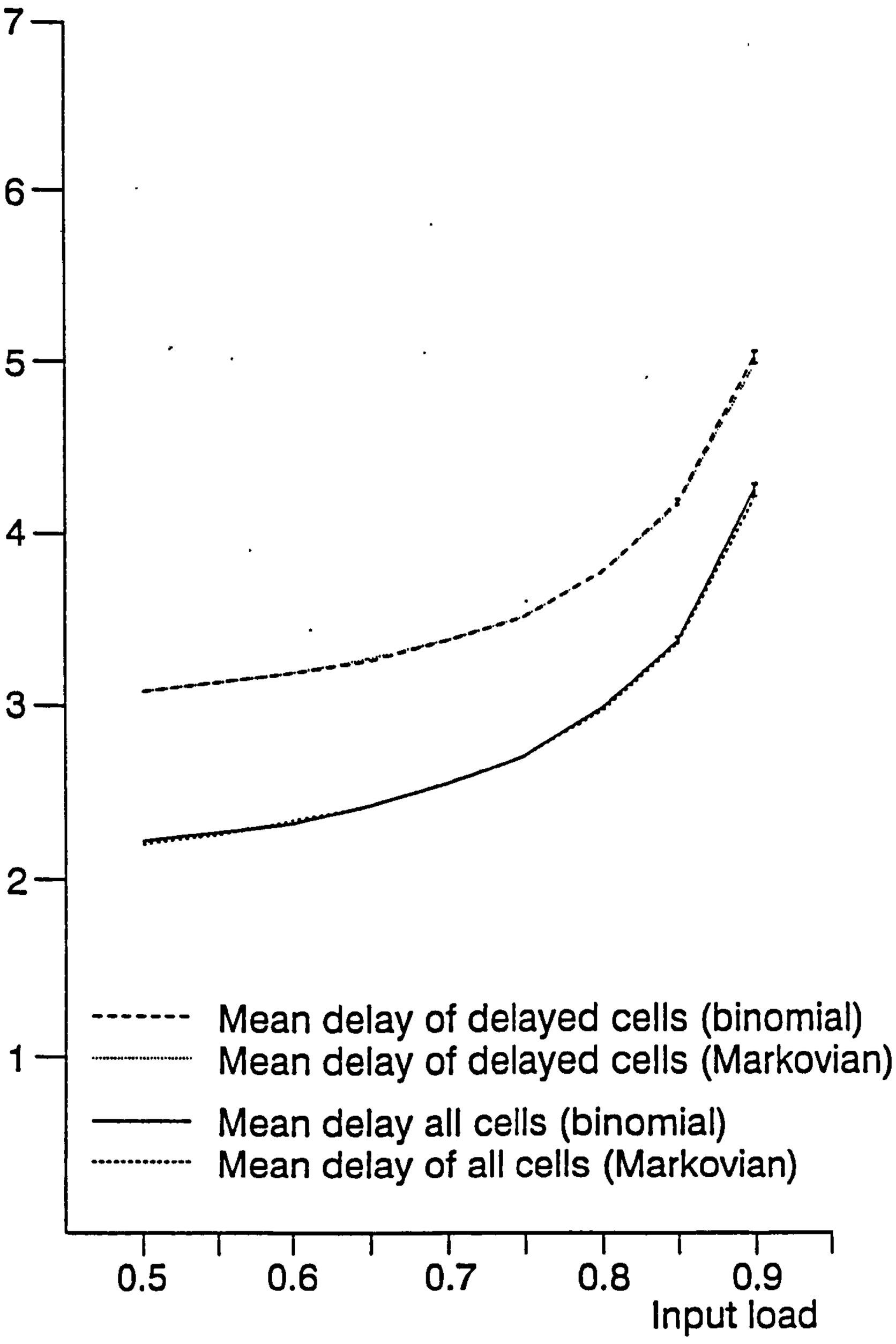

Graph 4.1: Compensated Arrival Rate (c) as a Function of Input Load

It can be seen from graph 4.1 that the magnitude of the compensation required increases as the input load increases. Note that the function is shown joined but was produced as a discrete function for input loads from 0.05 to 0.95 in steps of 0.05.

## 4.4 Comparison of Binomial and Negative Exponential Distributions

In order to assess the effect of different source models on the ATM switch simulator, the two results "mean delay of delayed cells" and "mean delay of all cells" were studied for input loads 0.5 to 0.9 in steps of 0.1 for both source models. The results are shown on graph 4.2.

(Note that due to the considerably higher number of instructions required to implement the GMDP video model it is not used in the subsequent investigations.)

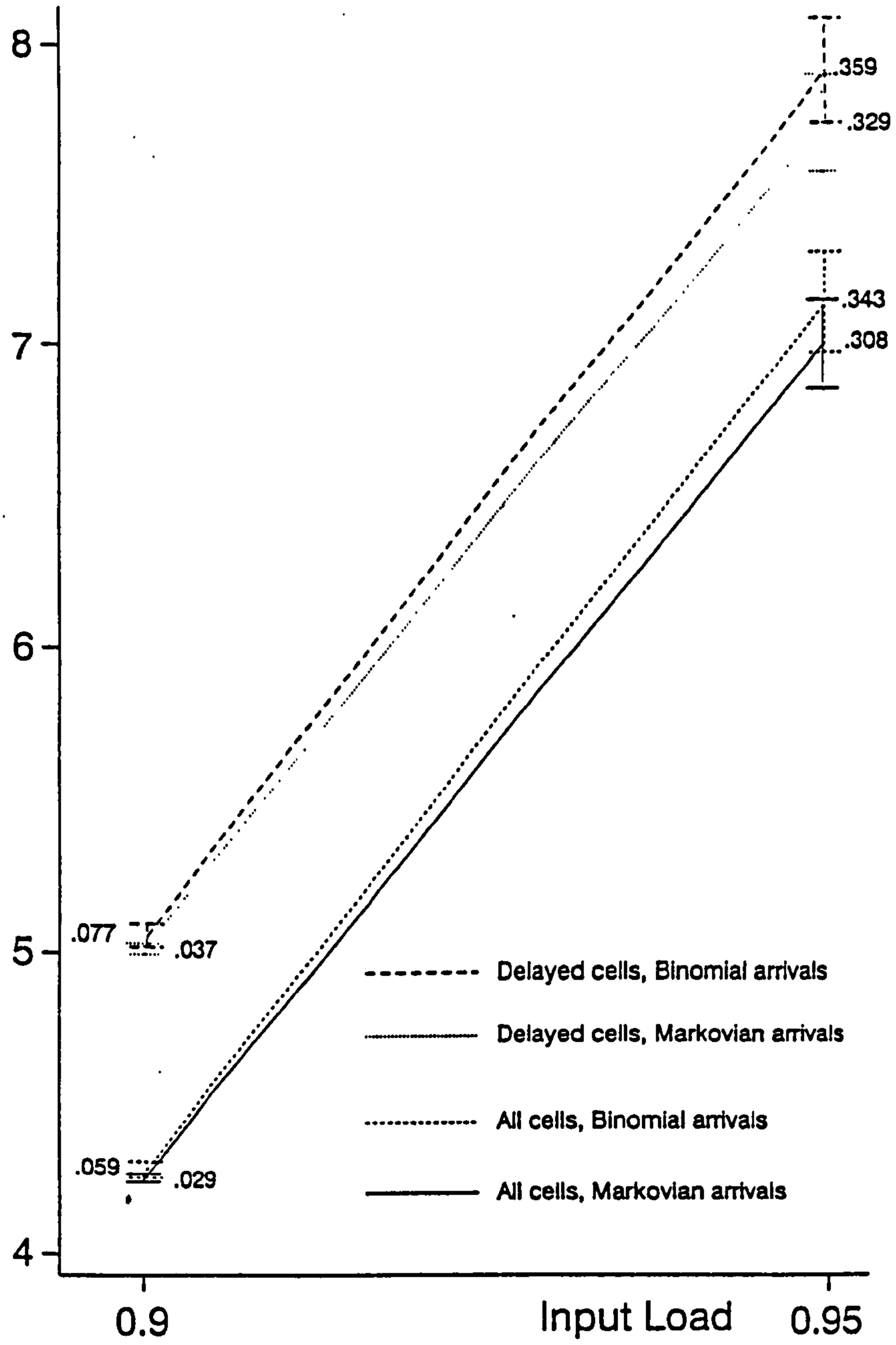



Graph 4.2: Delays for Binomial and Markovian Source Models

Each individual data point shown is the mean of 10 simulation runs with the same parameters. The data points for 0.85 and 0.9 load (binomial source only) are shown with 'error' bars representing the magnitude of the *standard deviation* of the result across the 10 runs.

### 4.4.1 Discussion of Results

The data points for the different source types lie on an almost identical line. The error bars for the binomial source show only a very small fluctuation in the results. [To aid clarity the error bars for the negative exponential source are not shown but are substantially the same, if not slightly smaller, as those for the binomial source.] The 2 lines diverge slightly at the 0.9 level. To assess whether this trend continues to higher loads, a further set of simulation runs at 0.9 and 0.95 loads were performed. These results, together with the error bars for both source types are shown on graph 4.3.




Graph 4.3: Delays for Binomial and Markovian Source Models (Expanded Scales)

With the expanded Input Load scale and slightly expanded Delay scale it is possible to see the magnitude of the difference in delays between the 2 arrival patterns. As the plotted points on the x−axis are the actual load observed (number of full cells divided

by the total number of cells) it is possible to see that both arrival rates accurately produce the desired load. (The tiny variations in input load due to statistical fluctuations can be seen.) This gives confidence in the accuracy of the values of c, the compensated arrival rate, particularly since the higher the input load the greater the compensation required. In 6 of the 8 cases the data points lie within the error bars of the other distribution.

The mean delay values could be the result of two very different delay distributions. To assess whether this is the case, the jitter values (standard deviation of the delay distribution) of the delays for the two arrival distributions were calculated and shown on graph 4.4. Figure 4.3 shows how the parameters shown on graphs 4.2, 4.3 and 4.4 have been calculated.

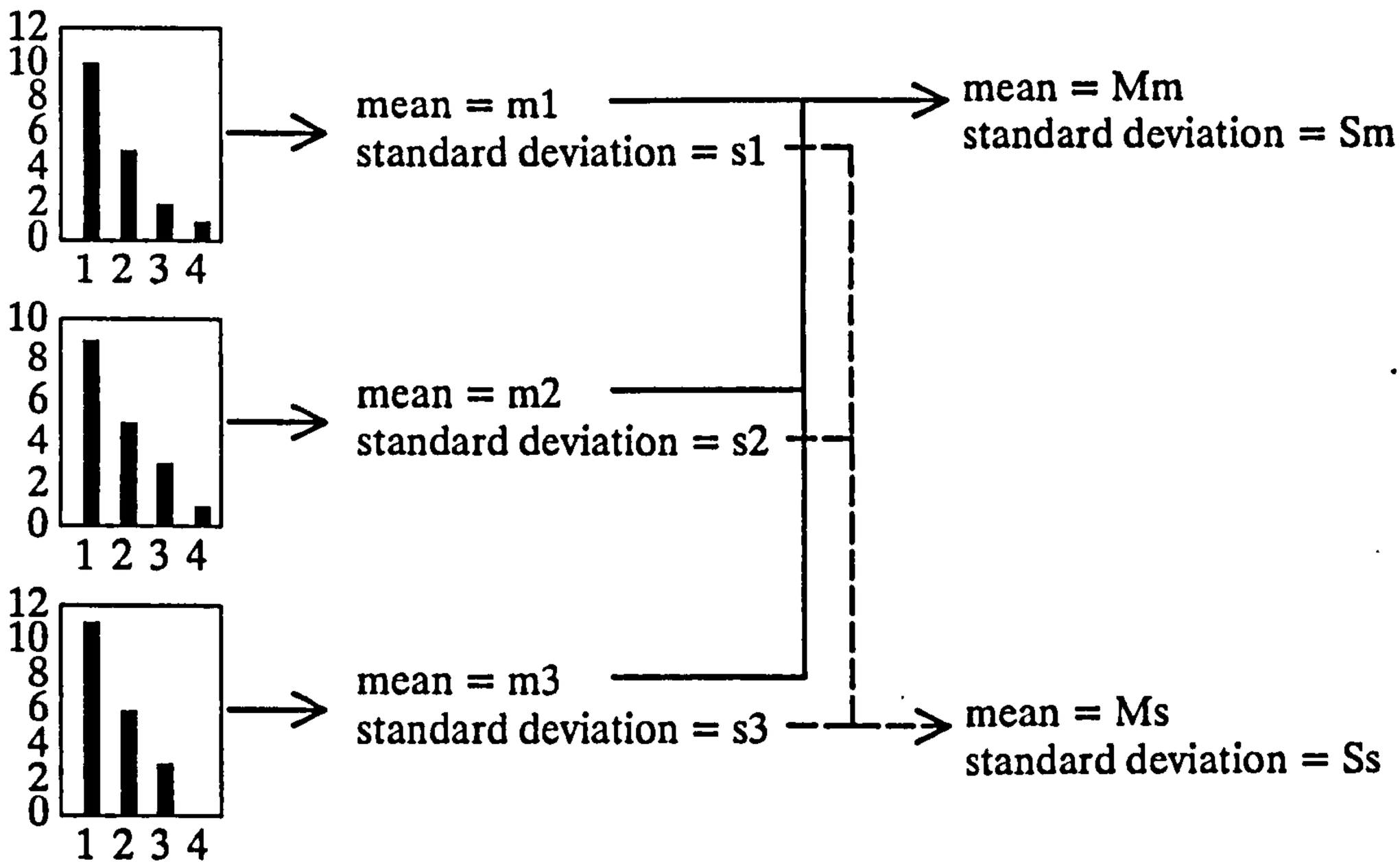
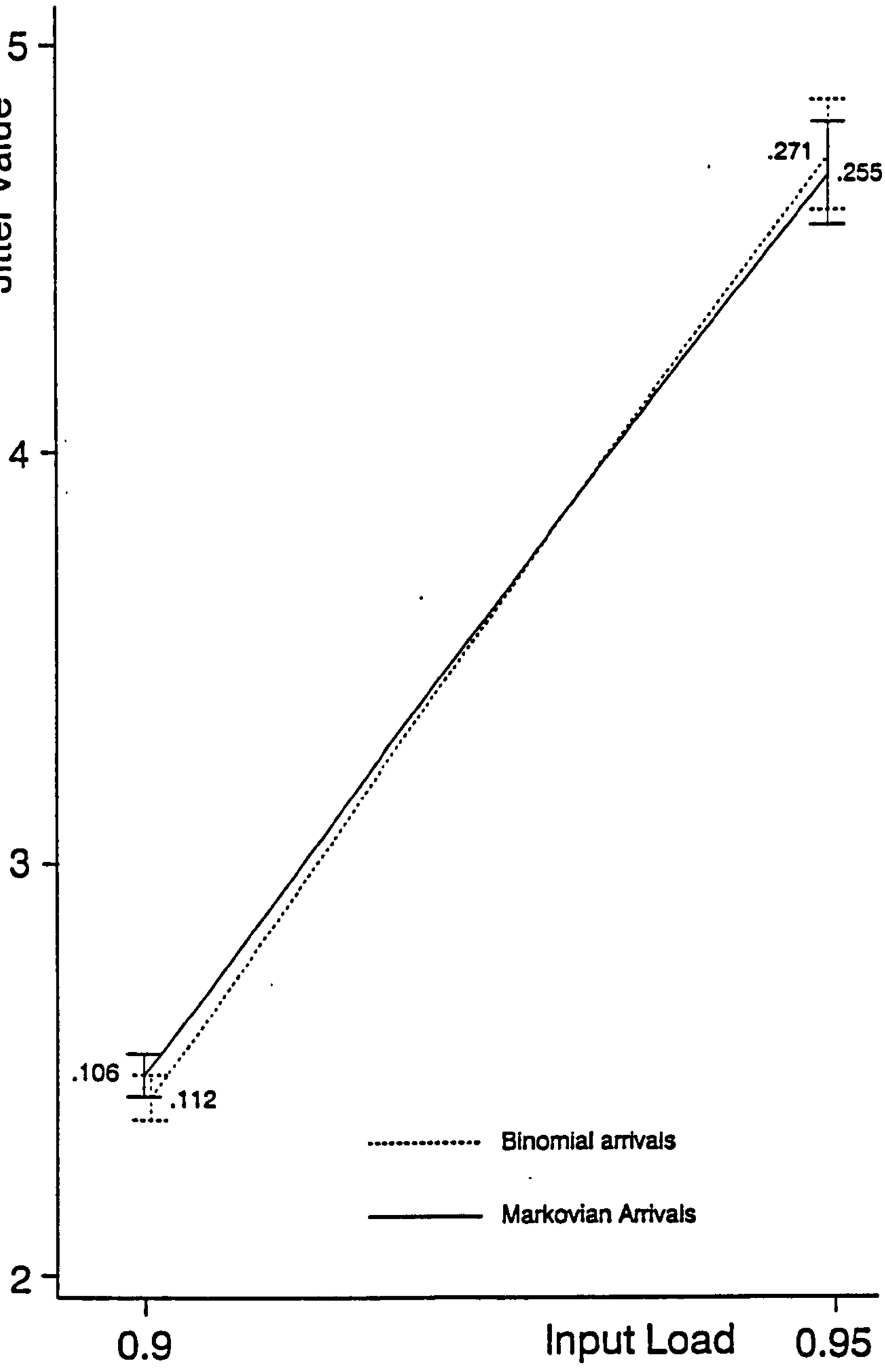



Figure 4.3: Definition of the Averaged Mean and Standard Deviation Values

A single simulation run produces a delay distribution. One way of characterising the distribution is to calculate its mean and standard deviation. The standard deviation

gives a measure of the *spread* of the distribution which for the delay distributions is also a measure of the cell delay variation or jitter and is therefore referred to as the jitter value. The ten simulation runs give 10 mean values and 10 jitter values (standard deviations). If the simulator started each run with the same random number generator seed the 10 values would be identical. Because the seed is changed at the start of each run (actually the number of seconds from January the 1st 1970), the 10 values will be different to some extent and will therefore have a mean and a standard deviation themselves.

The means of the individual runs were averaged to give Mm (see figure 4.3), these are the plotted values on graphs 4.2 and 4.3. The error bars on these graphs represent Sm. The standard deviations of the individual runs were also averaged to give Ms, the jitter values plotted on graph 4.4. The error bars on graph 4.4 represent Ss.




Graph 4.4: Jitter Values for Binomial and Negative Exponential Arrivals

Whereas graphs 4.2 and 4.3 show only the mean delay suffered by the cells, graph 4.4 shows the jitter values which represent the "spread" of the delay values, or the differential delay. This graph gives some evidence that the mean delays are most probably the result of two very similar delay distributions.

The following conclusions are drawn:

1) The performance of the ATM switch block, assessed by observing the parameters mean delay of all cells, mean delay of delayed cells and spread of delay values, is substantially the same for both negative exponential and binomial cell arrivals for loads from 0.5 to 0.85 inclusive.

2) The disparity at 0.9 load is small but consistently gives a slightly worse switch block performance under binomial arrivals than under negative exponential arrivals.

3) It has been suggested by the author in [72], and others [41][73], that for optimum performance the maximum load on a multiplex should be in the region of 0.8. This figure is now generally accepted in the field. From this it is possible to deduce that for input load in the range required to be studied, i.e. 0.85 or less, either arrival pattern will give substantially the same result and therefore the more computationally–efficient binomial arrival distribution may be used with confidence.

This chapter has introduced three source models that may be used within a simulator to generate cells with different arrival distributions. Two of the source models have been compared by studying the simulated performance of the switch block under the two different cell arrival distributions. It was concluded that the performance is substantially the same and therefore the most computationally efficient, the binomial source model, may be used in preference to the negative exponential model. The video source model was discounted due to its complexity. The next chapter proceeds

to use the simulator in addition to analytic techniques to investigate the variation of one of the most important performance parameters of an ATM switch, the cell–loss probability, with buffer sizes.

# Chapter 5 Investigation of Cell Loss due to Queue Overflow

The design of ATM switch fabrics requires trade offs to be made between the number of queue places available in the system and the probability of losing a cell due to queue–full conditions. As the number of queue places in the system increases, the system's tolerance to overload, or peaks (bursts) in the offered traffic also increases. Unfortunately, increasing the number of queue places also produces a number of undesirable effects such as:

i) An increase in the overall possible delay [74]

ii) An increase in the magnitude of the jitter introduced by the node [75]

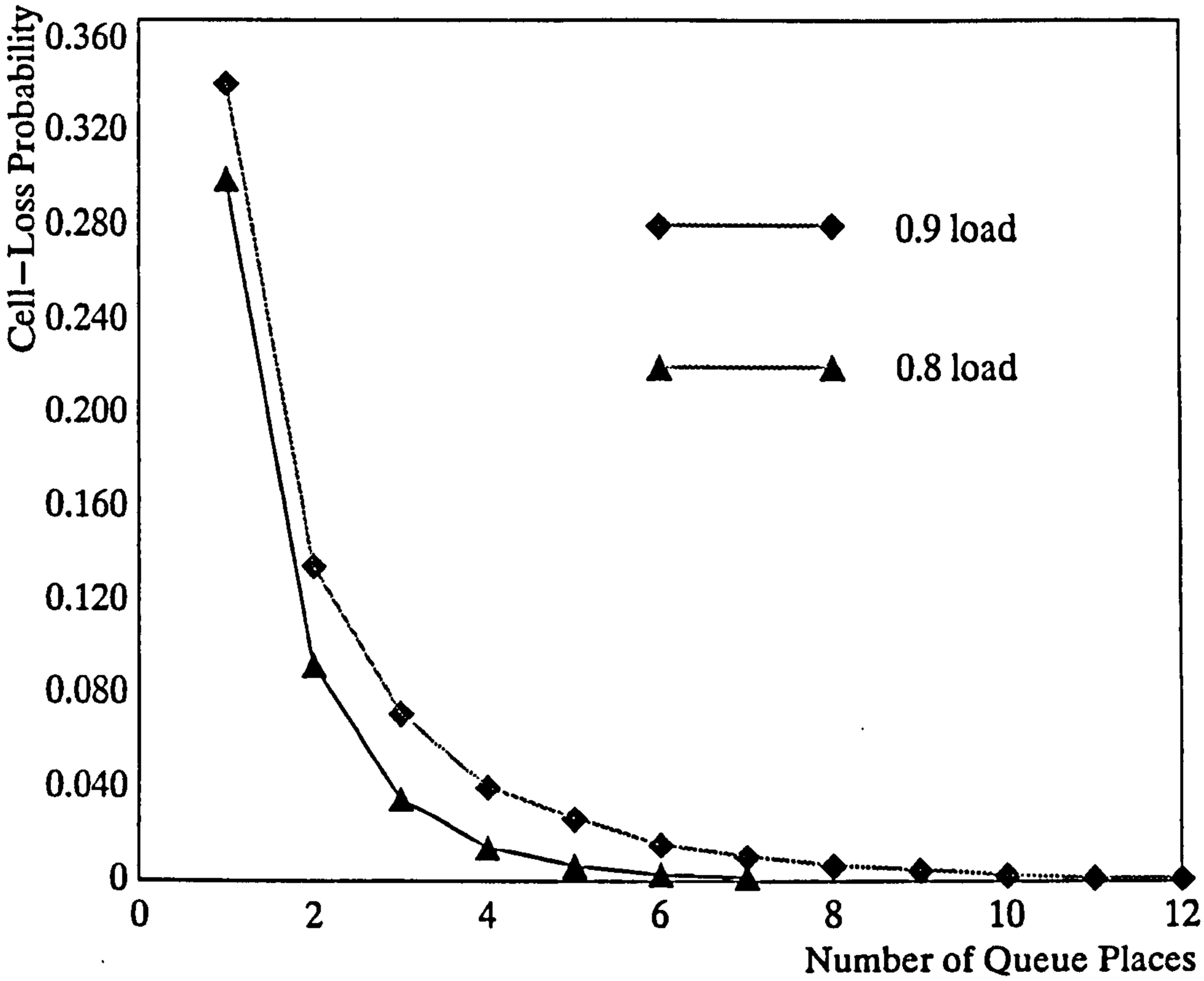
iii) An increase in the cost of the hardware of the switch.

Many studies on the importance of buffer positioning within an ATM switch have been made [61][63] and one conclusion reached is that output queueing provides the most favourable cost/performance trade off (in terms of throughput, delay and delay jitter). However, the investigations presented in this chapter represent a first attempt to find a relationship between cell–loss probability and queue length in ATM switch elements. This has been done to establish a minimum queue size corresponding to the maximum acceptable cell–loss caused by buffer contention. For the purposes of the studies, an acceptable cell loss is assumed to be 1 cell in $10^{12}$. (Other contributions to cell–loss such as header errors will increase cell–loss probability but will not be affected by queue size.)

## 5.1 Cell–Loss Probabilities in the Final Stage by Simulation

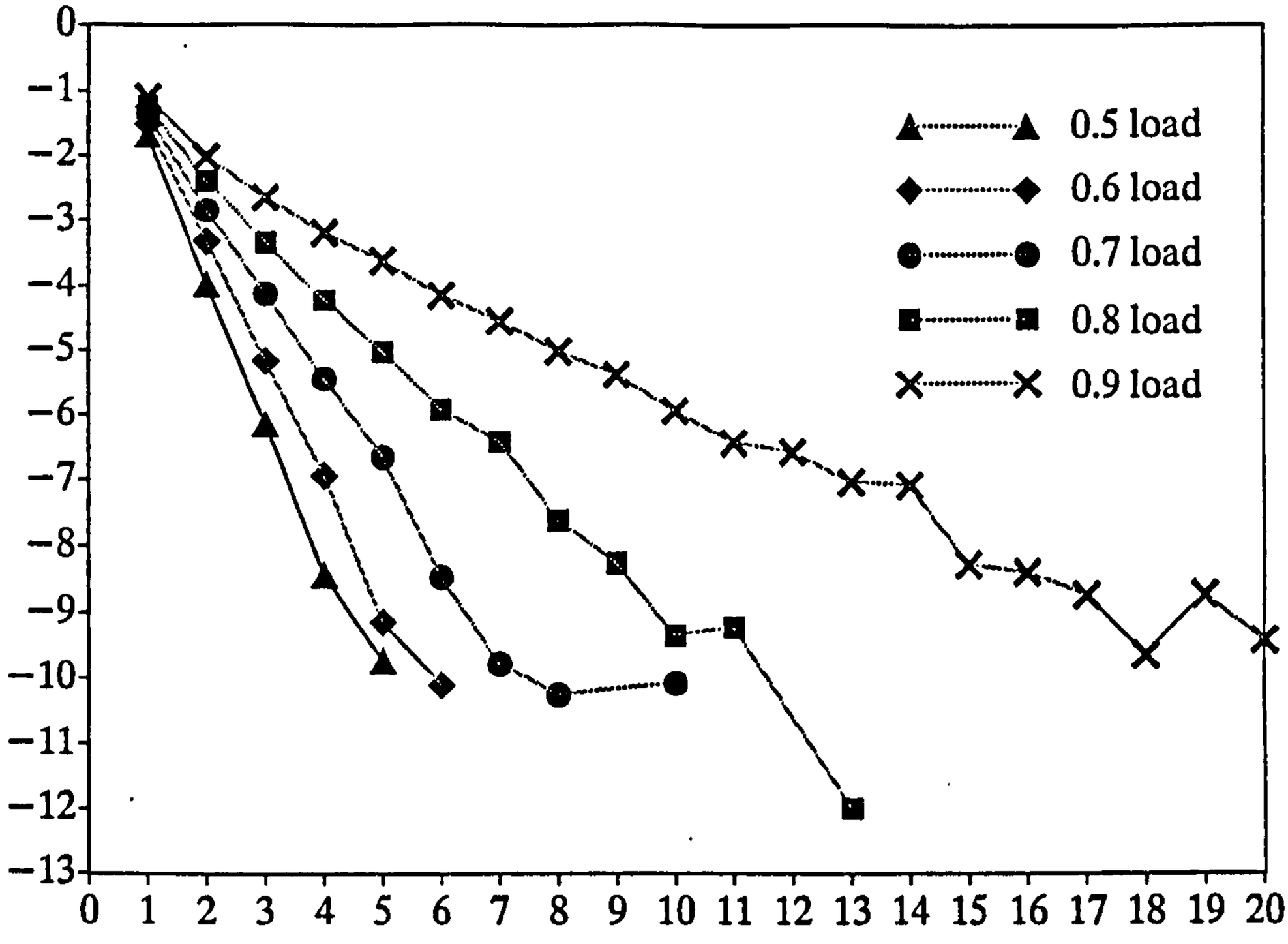
For this investigation, the final stage of the switch was configured with 2 inputs and 1 output (i.e. no queue sharing between output multiplexes). Five traffic loads, 0.5,

0.6, 0.7, 0.8 and 0.9 with traffic having a binomial distribution were used. The number of queue places was varied from 1 to 20 for each input load, and $10^5$ cell times were simulated. The cell–loss probabilities obtained were plotted against the number of queue places. The results for input loads 0.8 and 0.9 are shown on graph 5.1.



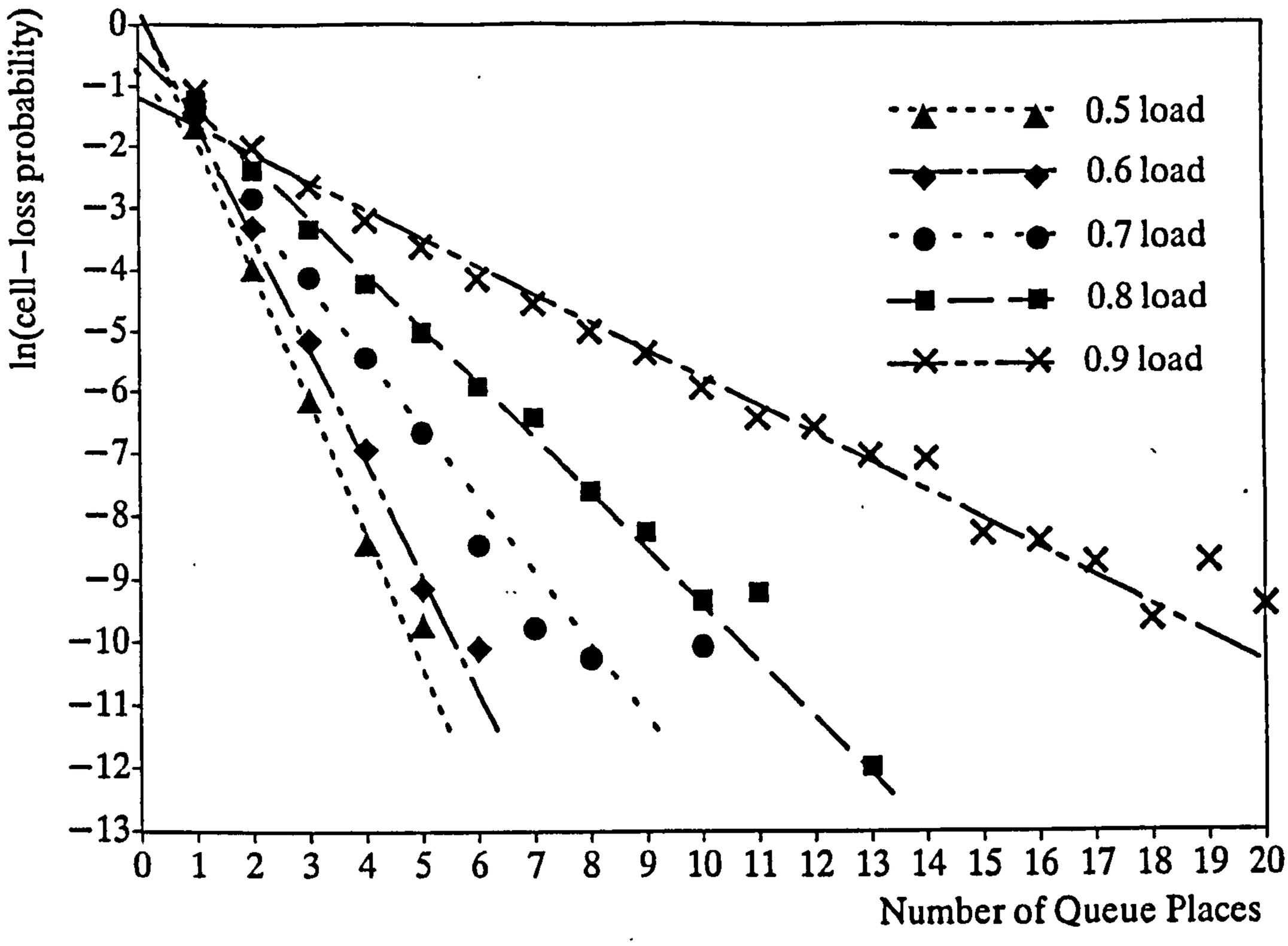

Graph 5.1: Cell–loss Probability as a Function of Number of Queue Places (Simulator Results)

The 2 functions shown on graph 5.1 appear to be exponential and so ln(cell–loss probability) was plotted against the number of queue places as shown in graph 5.2.




Graph 5.2: Ln(Cell−loss probability) vs Number of queue places

The best−fit straight lines for the data points shown on graph 5.2 were found using the least squares method fitting y on x where x is the controlled variable: *number of queue places* and y is the *natural log of the cell−loss probability*. These are shown on graph 5.3.




Graph 5.3: Best−Fit straight lines to the data points

A number of observations about graph 5.3 should be made:

i) The simulation data points for load 0.7 are more disparately spaced than those for the other loads. This is principally due to a high cell−loss occurring with 10 queue places. Further simulation runs at this load and number of queue places have shown that this particular point was misplaced, possibly due to the non−spectral properties of the random number generator used.

ii) For all loads, the data points are more scattered from the straight lines at very low cell−loss probabilities. This is caused by the

small number of cells simulated in comparison with the cell−loss observed at these points on the lines.

Observation ii) above leads to the conclusion that with simulator run lengths of $10^5$ cell times, cell−loss probability values smaller than 1 in $10^3$ will be imprecise. This is a severe limitation of simulation in this instance as target cell loss probabilities for ATM switch elements are in the region of 1 in $10^{10}$ to 1 in $10^{12}$ cells. This would require a minimum of $10^{14}$ cell times to be simulated. With the ATMoSS simulator it is estimated that this would take over 200 years of real time. Alternative methods of investigating very low cell−loss probabilities are therefore required.

To assess the accuracy of the theory that ln(cell−loss probability) is linearly related to the number of queue places, a correlation test and a chi−squared (Pearson goodness−of−fit) test were performed.

### 5.1.1 Statistical Test of Straight Line fit to the Data Points

The degree of correlation of a set of observed values can be represented by a correlation coefficient **r**, which can be found using the formula given in [76]:

$$Correlation\ coefficient\ (r) = \frac{\sum(x_i - x_m)(y_i - y_m)}{\sqrt{\sum(x_i - x_m)^2 \sum(y_i - y_m)^2}}$$

*where $x_i$ and $y_i$ are the observed values,*

*$x_m$ and $y_m$ are the mean values.*

[The value of r must lie in the range −1 to +1. When r = +1 all of the observed points lie on a straight line with a positive slope, when r = −1 the slope is negative.]

The $\chi^2$ value is found using the formula given in [77]:

$$\aleph^2 = \sum \frac{(y_{observed} - y_{expected})^2}{y_{expected}}$$

*summed over all possible values of x*

The test values of r and $\aleph^2$ are given in table 5.1.

| Input Load | Correlation Coefficient | $\aleph^2$ Test Value | $\aleph^2_{0.99,\,n-1}$ Tabulated Value |
|---|---|---|---|
| 0.5 | −0.966 | 0.059 | 0.297 |
| 0.6 | −0.997 | 0.041 | 0.554 |
| 0.7 | −0.962 | 0.888 | 1.65 |
| 0.8 | −0.996 | 0.160 | 3.05 |
| 0.9 | −0.990 | 0.557 | 7.63 |

Table 5.1: Test Values for Best−fit Straight lines vs Experimental Data Points

The $\aleph^2$ goodness−of−fit test shows over 99% confidence in the fit of the experimental points to the best−fit straight lines in each case. The tests on the correlation coefficient have shown that correlation between ln(cell−loss probability) and the number of queue places is significant at the 99.9% level.

### 5.1.2 Cell−Loss Probability as a Function of Load and Queue Size

The statistical tests of section 5.1.1 give a high level of confidence in the existence of a logarithmic relationship between cell−loss probability and the number of queue places for a range of input loads. We may therefore write:

$$\text{Cell−loss Probability} = a\, e^{-bn} \quad - \{11\}$$

where −b is the gradient of the straight lines in graph 3.

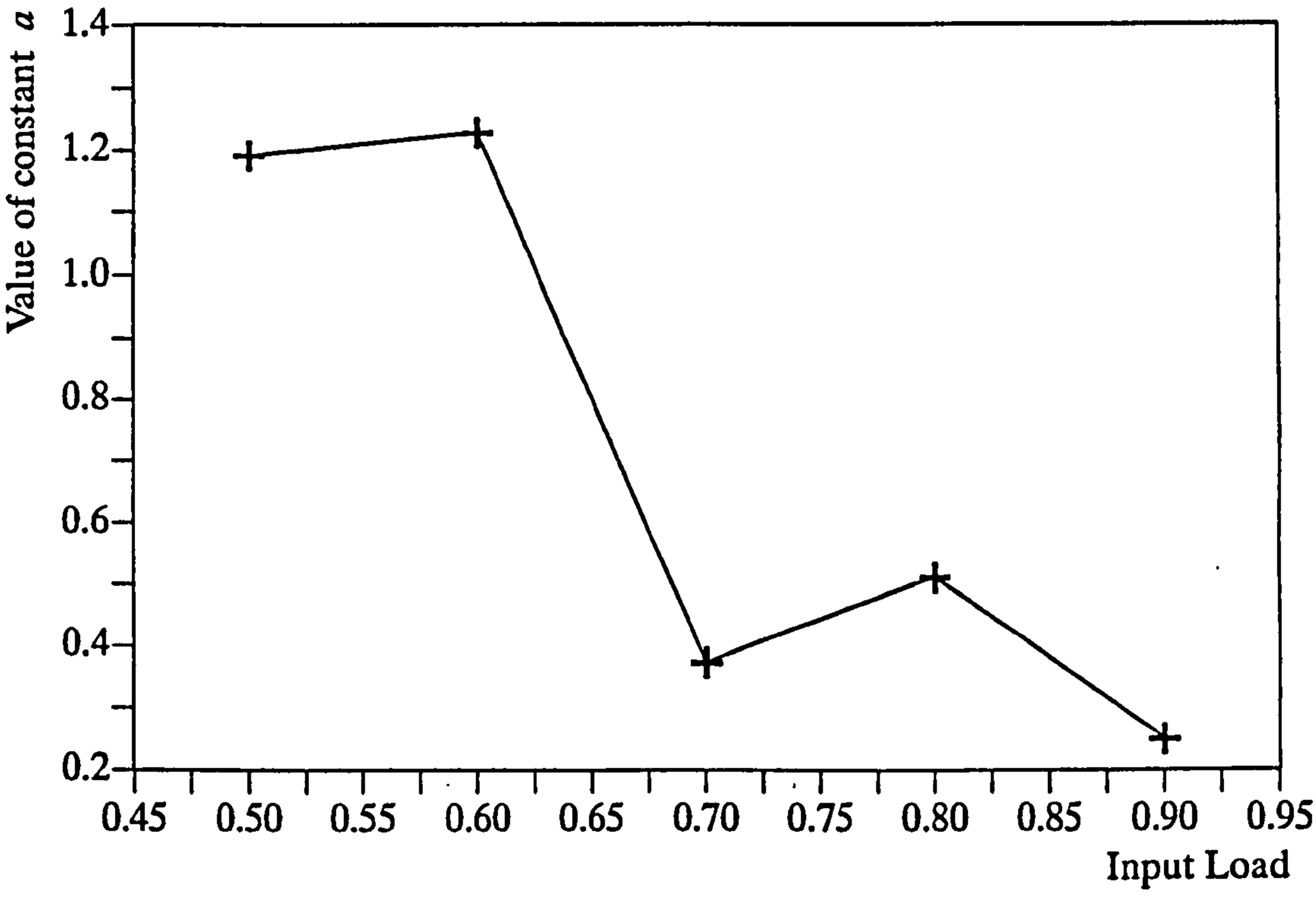
The values of the constants *a* and *b* are dependant in this particular experiment on the offered traffic load and are summarised in table 5.2.

| Input Load | $a$ | $b$ |
|---|---|---|
| 0.5 | 1.192 | 2.054 |
| 0.6 | 1.23 | 1.779 |
| 0.7 | 0.373 | 1.088 |
| 0.8 | 0.511 | 0.846 |
| 0.9 | 0.251 | 0.429 |

Table 5.2: Summary of Values of Constants $a$ and $b$

From equation {11}, if the constants $a$ and $b$ can be defined as functions of Input Load, it would be possible to find the function $f()$ such that

$$f(\textit{input load, number of queue places}) \approx \text{cell-loss probability} \quad - \{12\}$$

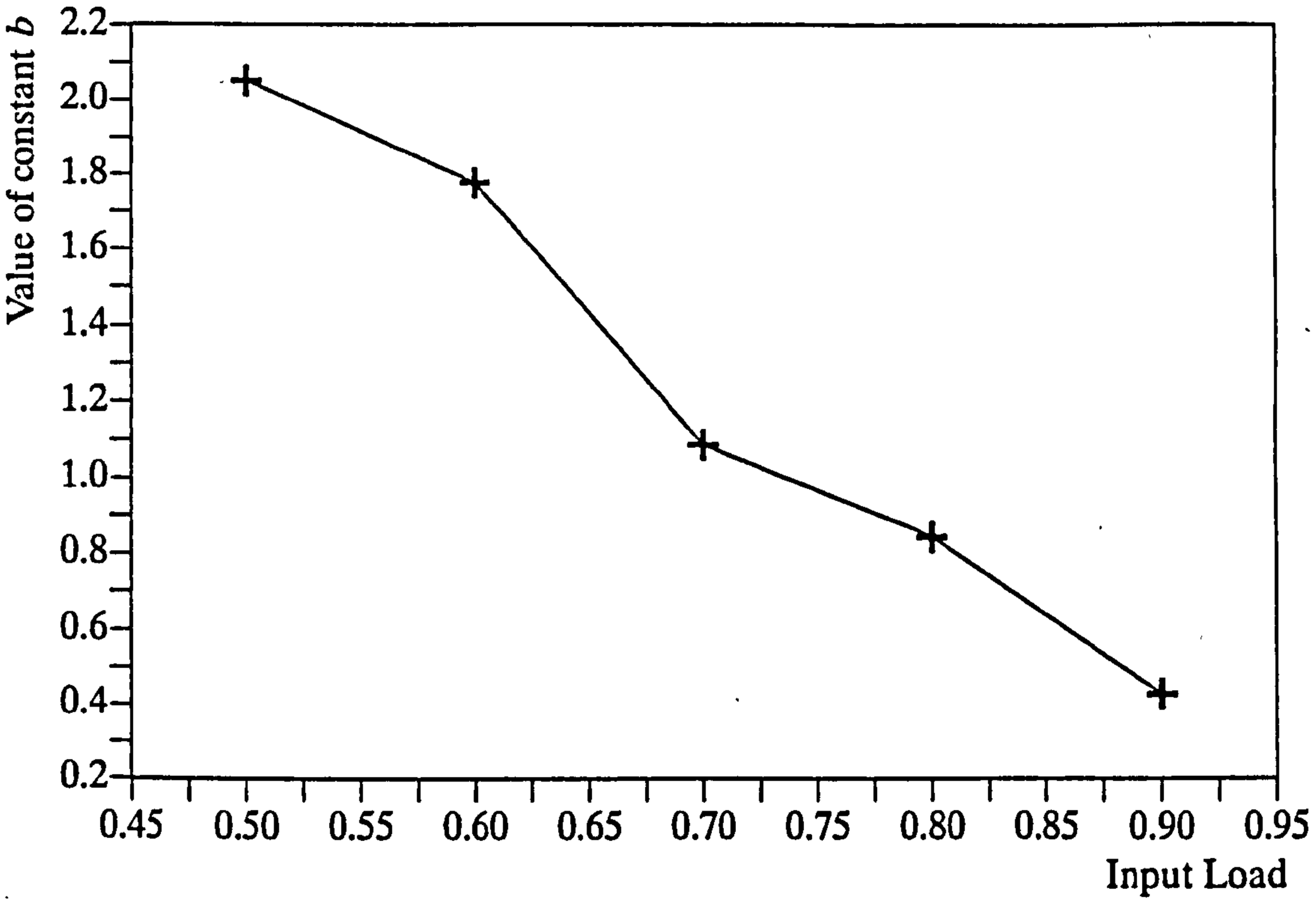



Graph 5.4: Variation of constant $a$ w.r.t. Input Load

Graph 5.4 shows $\textit{function}_1\textit{(input load)} = a$. Fitting a straight line to these points gives

$$a = (-2.601 \text{ x Input Load}) + 2.5321 \quad - \{13\}$$

The correlation coefficient for the points to the straight line is −0.8836 . The function is clearly only very approximately linear, however the accuracy of the value of the constant *a* is not as significant as the accuracy of *b*.




Graph 5.5: Variation of Constant *b* w.r.t. Input Load

Graph 5.5 shows *function*$_2$*(input load)* = *b*. Fitting a straight line to these points gives

$$b = (-4.183 \text{ x Input Load}) + 4.1673 \quad - \{14\}$$

The correlation coefficient for these points to the straight line is −0.9889 . This function is clearly more linear than for constant *a*.

Using *function*$_1$ and *function*$_2$ in equation {11}, we obtain

$$\text{cell-loss prob.} = [(-2.601 \text{ x L}) + 2.5321] \cdot e^{-[(-4.183 \text{ x L}) + 4.1673] \cdot n} \quad - \{15\}$$

where L is the input load and n is the number of queue places. Substituting the following values of L and n into equation 15,

$$L=0.5,\ 1 \leq n \leq 5,$$

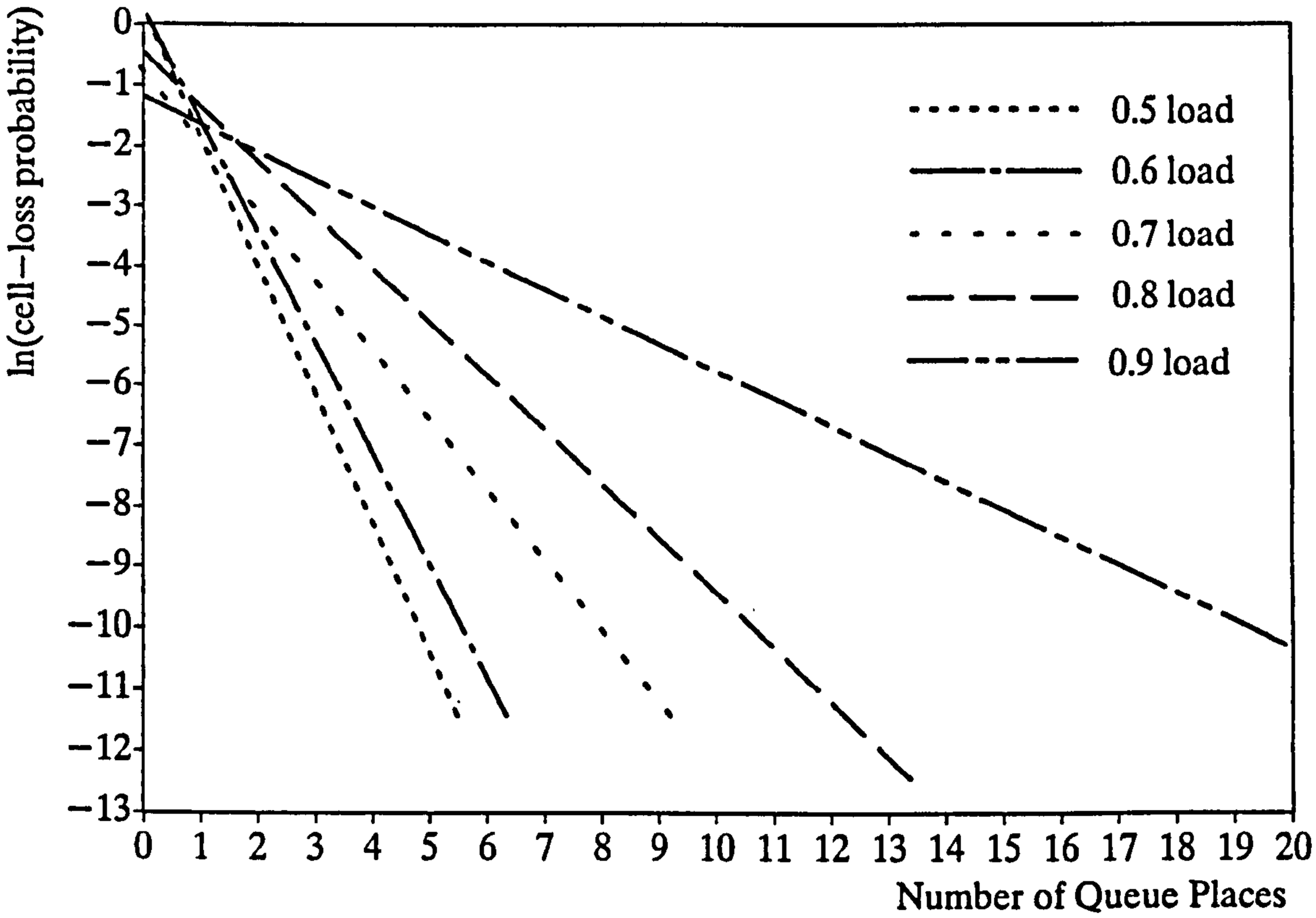
$$L=0.6,\ 1 \leq n \leq 6,$$

$$L=0.7,\ 1 \leq n \leq 9,$$

$$L=0.8,\ 1 \leq n \leq 13,$$

$$L=0.9,\ 1 \leq n \leq 20,$$

graph 5.6 is obtained. Comparison of graph 5.3 with graph 5.6 shows that equation 15 is a good model of cell–loss probability for certain ranges of L and n and thus provides a metric with which it may be possible to predict cell–loss probability analytically for a wider range of these values.




Graph 5.6: Cell–loss Probability given by the Model presented in Equation 15

## 5.2 Cell−Loss Probabilities in the Final Stage by Analysis

Two observations on graph 5.3 above were made indicating the limitations of simulation when investigating very low cell−loss probabilities in the region of 1 in $10^{10}$ to 1 in $10^{12}$ cells. As a result of these limitations an analytic model was developed that in addition to giving results in the cell−loss probability region we are interested in, could also test the model presented in equation 15.

### 5.2.1 Analytic Model Definition

The simulation results in section 5.1 above were produced by modelling 2 inputs into the final stage of the ATM switch while varying the parameters "number of queue places" and "input load". The analytic model introduces a 3rd variable parameter, "number of inputs into the final stage".

**Definitions:**

let L be the total input load into the final stage,

let m be the number of inputs into the queue,

let n be the number of queue places.

The scenario is shown in figure 5.1.

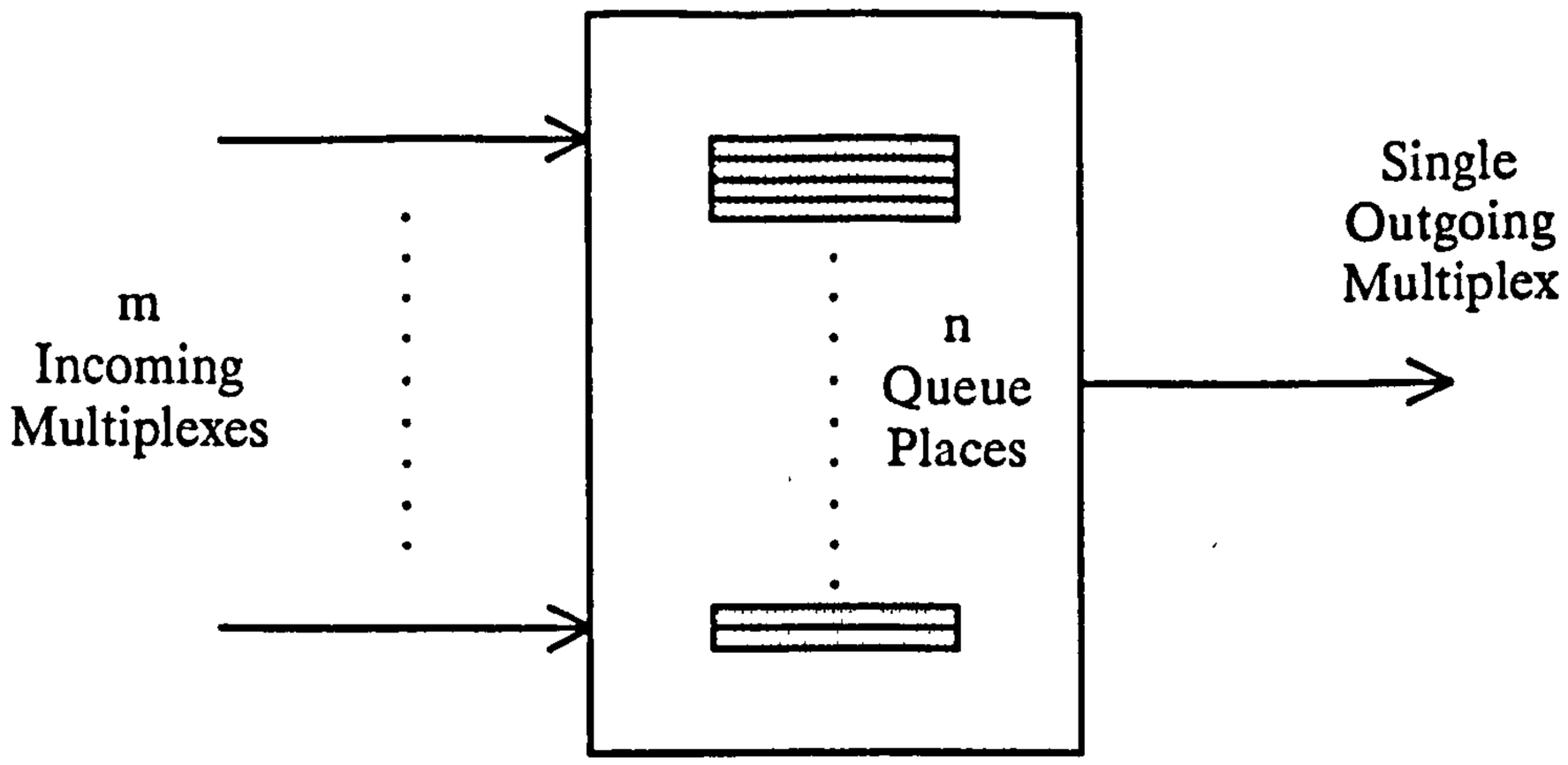



Figure 5.1: Scenario for Analytic Modelling of Cell−loss in the Final Stage Switching Element

Therefore, in 1 unit of time, $x$ cells may arrive at the element where $0 \leq x \leq m$. However, in 1 unit of time, only 1 cell may be output on the outgoing multiplex which implies that the queue occupancy will increase if $x > 1$ by $(x-1)$ cells. In the event that there are less than $(x-1)$ free queue places, cell−loss will occur.

The input load, L, is evenly distributed across all inputs. When no cell loss occurs, output load = input load. Output load is a "utilisation factor" of the outgoing multiplex. 80% or 0.8 load indicates that on average 8 cells out of 10 contain data, 2 out of 10 are empty. One problem with defining the load is specifying the length of time over which the average is taken. For example, 50% load could equally well be full cell / empty cell / full cell / empty cell etc., or 100 full cells followed by 100 empty cells if the interval were 4 cells or 200 cells respectively. The solution to the problem is to maintain a rolling−average of the load over a period (in cell times) equal to the number of queue places.

However 80% output load implies, in general, there are 8 full cells in every 10. If there were 8 inputs into the final stage (m=8) each input would on average only carry 1 full cell in 10. This appears to be seriously under−utilising the interconnecting links into

the final switching stage. To improve utilisation, the conceptual speed of the interconnecting links could be reduced with respect to the outgoing multiplex speed. Thus for m=8 the interconnect speed could be $^1/_8$th of the outgoing multiplex speed. This would result in an interconnect utilisation of 80% but would clearly severely restrict the capability of the switch. Therefore, the interconnect speed in both the analytic model and the ATMoSS simulator is defined as being equal to the outgoing multiplex speed.

To investigate cell–loss analytically, a Markov chain of the queueing system was produced. As cell–loss only occurs when at or approaching a queue–full condition, only this end of the Markov chain is significant. The Markov chain representation of the queue states is shown in figure 5.2. One complete unit of time is shown in figure 5.2 which integrates cell arrivals and departures (i.e. it is assumed that one cell departs from the system per unit of time).

If the probabilities of being in a particular queue state [queue state probabilities $p(S_0)$, $p(S_1)$ etc.] can be determined, and the probabilities of $x$ cells arriving on M inputs where $0 \leq x \leq m$ [$p_0$, $p_1$, ... ,$p_m$] can be determined, they may be combined to produce overall cell–loss probabilities.

Note: To ease the subsequent representation, the parentheses in the previous notation p(x) to represent the probability of x cells arriving in 1 unit of time, are dropped. Thus

*(in the previous notation)* $p(x) = p_x$ *(in the new notation)*

(Again $p_x$ represents the probability of x cells arriving across the group of inputs.)

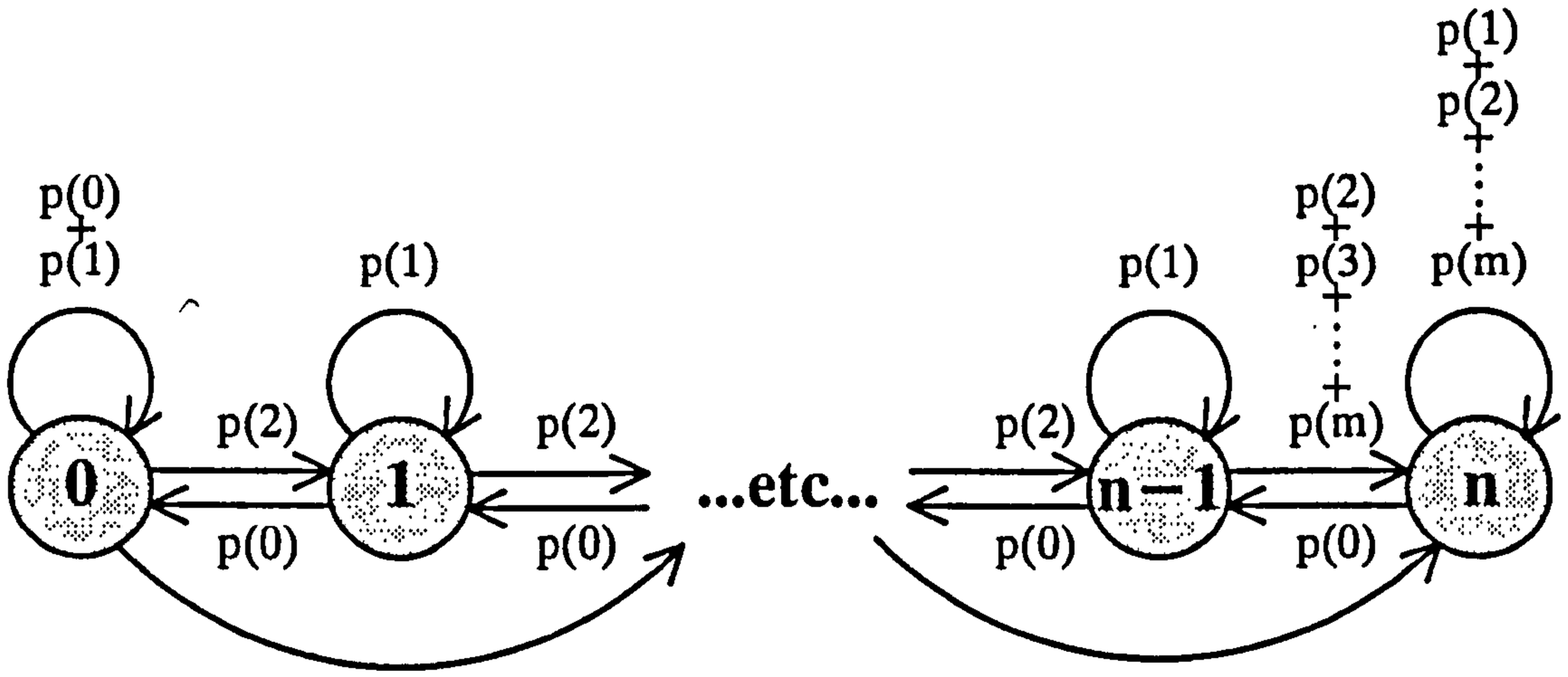



Figure 5.2: Markov Chain Representation of Analytic Model

Note: Not all state transitions are shown in figure 5.2. All states will have state transitions p(0) up to p(m). If no cells arrive in one unit of time, p(0), the queue occupancy will decrease by 1. If one cell arrives, p(1), the queue occupancy will remain the same as one cell will also depart from the system.

### 5.2.2 Derivation of the Analytic Solution from the Analytic Model

The first step in the derivation is to produce a state transition matrix P from the Markov chain in figure 5.2:

$$
\mathrm{P} = \begin{pmatrix}
p0+p1 & p2 & p3 & p4 & p5 & . & pm & 0 & 0 & . & . & . & . \\
p0 & p1 & p2 & p3 & p4 & . & pm-1 & pm & 0 & . & . & . & . \\
0 & p0 & p1 & p2 & . & . & pm-2 & pm-1 & pm & . & . & . & . \\
0 & 0 & p0 & . & . & . & . & pm-2 & pm-1 & . & 0 & 0 & 0 \\
. & 0 & 0 & . & . & . & . & . & pm-2 & . & 0 & 0 & 0 \\
. & . & . & . & . & . & & . & . & . & pm & 0 & 0 \\
. & . & . & . & . & . & & & . & . & pm-1 & pm & 0 \\
. & . & . & . & . & . & & & . & . & . & pm-1 & pm \\
. & . & . & . & . & . & & & p1 & . & . & pm-2 & \sum_{x=m-1}^{m} px \\
. & . & . & . & . & . & & & p0 & . & . & . & . \\
. & . & . & . & . & . & & & 0 & . & p2 & . & . \\
. & . & . & . & . & . & & & 0 & . & p1 & p2 & \sum_{x=3}^{m} px \\
. & . & . & . & . & . & . & . & . & . & p0 & p1 & \sum_{x=2}^{m} px \\
. & . & . & . & . & . & . & . & . & . & 0 & p0 & \sum_{x=1}^{m} px
\end{pmatrix}
$$

Figure 5.3: Analytic Model State Transition Matrix, P

Also from the Markov chain representation, we may form a row vector, $\mathbf{p}_t$, containing the queue state probabilities such that:

$$\mathbf{p}_t = \{ p(S_0)_t , p(S_1)_t , p(S_2)_t , \dots , p(S_{n-1})_t , p(S_n)_t \}$$

where $p(S_y)_t$ represents the probability of y cells in the queue at time t.

As previously stated in the thesis, if P represents the state transition matrix for unit of time $\delta t$:

$$\mathbf{p}_{t+\delta t} = \mathbf{p}_t \cdot \mathrm{P} \quad - \{16\}$$

After n units of time,

$$\mathbf{p}_{t+n.\delta t} = \mathbf{p}_{t+(n-1).\delta t} \cdot \mathrm{P} \quad - \{17\}$$

As n tends to $\infty$, the probabilities reach equilibrium such that

$$p_{t+\delta t} = p_t$$

and equation {16} is written

$$\pi = \pi \,.\, P \quad - \{18\}$$

where $\pi$ represents the row vector containing the steady–state queue–length probabilities,

$$\pi = \{\, p(S_0)\,,\, p(S_1)\,,\, p(S_2)\,,\, \ldots\,,\, p(S_{n-1})\,,\, p(S_n)\,\}$$

$$\text{if } \pi_i = p(S_i) \text{ etc. then}$$

$$\pi = \{\, \pi_0\,,\, \pi_1\,,\, \pi_2\,,\, \ldots\,,\, \pi_{n-1}\,,\, \pi_n\,\}$$

Because $k.\pi = k.\pi \,.\, P$, there are an infinite number of solutions to equation {18}. The uniqueness of the required solution is given by the property that the sum of the contents of a row vector containing state transition probabilities must be equal to 1:

$$\sum_{i=0}^{n} \pi_i = 1 \quad - \{19\}$$

From the state transition matrix in figure 5.3 and equation {18} the following equations may be written:

$$\pi_0 = (p_0+p_1).\pi_0 + p_0.\pi_1 \quad - \{20\}$$

$$\pi_1 = p_2.\pi_0 + p_1.\pi_1 + p_0.\pi_2 \quad - \{21\}$$

$$\pi_2 = p_3.\pi_0 + p_2.\pi_1 + p_1.\pi_2 + p_0.\pi_3 \quad \text{etc up to . . .}$$

$$\pi_{m-1} = p_m.\pi_0 + p_{m-1}.\pi_1 + p_{m-2}.\pi_2 + \ldots + p_2.\pi_{m-2} + p_1.\pi_{m-1} + p0.\pi_m$$

In general for $m-1 \leq j \leq n-1$,

$$\pi_j = p_m.\pi_{j-m+1} + p_{m-1}.\pi_{j-m+2} + p_{m-2}.\pi_{j-m+3} + \ldots + p_1.\pi_j + p_0.\pi_{j+1}$$

The final equation for $\pi_n$ has a different form

$$\pi_n = p_m.\pi_{n-m+1} + \sum_{x=m-1}^{m} p_x.\pi_{n-m+2} + \sum_{x=m-2}^{m} p_x.\pi_{n-m+2} + \ldots + \sum_{x=1}^{m} p_x.\pi_n$$

The above equations are solved iteratively as follows:

If $\pi_1$ up to $\pi_n$ are expressed in terms of $\pi_0$ such that

$$a\,.\,\pi_0 = \pi_1$$
$$b\,.\,\pi_0 = \pi_2$$
$$c\,.\,\pi_0 = \pi_3$$
$$\ldots \text{ etc}$$

then as $\pi_0 + \pi_1 + \pi_2 + \ldots + \pi_{n-1} + \pi_n = 1$

$\pi_0 + a.\pi_0 + b.\pi_0 + \ldots = 1$ and $\pi_0 = 1 / (1+a+b+c+.....)$ – {22}

From equation {20},

$$\pi_0 = (p_0 + p_1)\pi_0 + p_0\pi_1$$

$$\therefore p_0\pi_1 = \pi_0 - (p_0 + p_1)\pi_0$$

$$\therefore \pi_1 = \pi_0 \frac{(1 - p_0 - p_1)}{p_0} \qquad - \{23\}$$

$$\text{and } \therefore a = \frac{(1 - p_0 - p_1)}{p_0}$$

From equation {21},

$$\pi_1 = p_2.\pi_0 + p_1.\pi_1 + p_0.\pi_2$$

$$\therefore p_0\pi_2 = \pi_1 - p_2\pi_0 - p_1\pi_1$$

using equation {23}, $$p_0\pi_2 = \pi_0\frac{(1 - p_0 - p_1)}{p_0} - p_2\pi_0 - p_1\pi_0\frac{(1 - p_0 - p_1)}{p_0}$$

$$\therefore \pi_2 = \pi_0\frac{\left\{\frac{(1-p_0-p_1)}{p_0} - p_2 - p_1\frac{(1-p_0-p_1)}{p_0}\right\}}{p_0}$$

$$\therefore b = \frac{\left\{\frac{(1-p_0-p_1)}{p_0} - p_2 - p_1\frac{(1-p_0-p_1)}{p_0}\right\}}{p_0}$$

If this is done repeatedly, an equation for $\pi_{i+1}$ in terms of $\pi_0$, $\pi_1$, $\pi_2$ etc. to $\pi_i$ is obtained where $\pi_1$ to $\pi_i$ have already been obtained in terms of $\pi_0$, therefore $\pi_{i+1}$ is obtained in terms of $\pi_0$ and the factors a, b, c, etc are obtained in terms of $p_x$.

When all $\pi_i$ for $1 \leq i \leq n$ have been obtained in terms of $\pi_0$, the values of $p_x$ are found.

As there are m inputs carrying binomial traffic, the values $p_x$ for $0 \leq x \leq m$ are calculated using L, the input load:

$$p_x = \binom{m}{x}.(1 - L)^{(m-x)}.L^x$$

$$\text{where}\binom{m}{x} = \frac{m!}{x!.(m - x)!}$$

$$\text{and } 0 \leq L \leq 1$$

Knowing the factors a, b, c etc in terms of $p_x$, using the numeric values of $p_x$ in equation {22} we obtain a numeric value for $\pi_0$. Using equation {23} a numeric value for $\pi_1$ is obtained, and so on up to $\pi_n$.

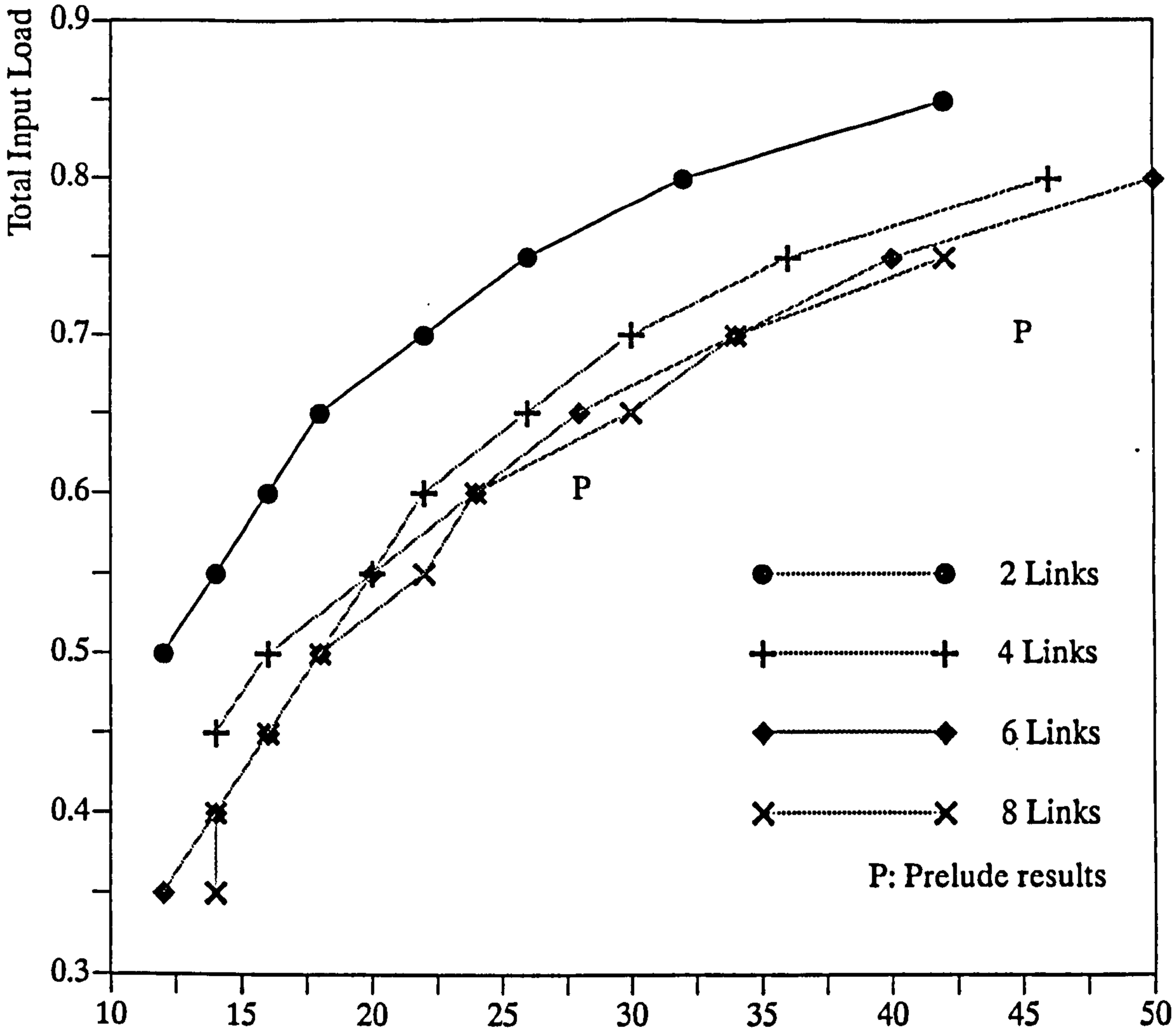
Thus, we now have the steady state probabilities of all possible queue lengths, $\pi_0$ up to $\pi_n$. As cell−loss occurs in the system when $x$ cells arrive with only 1 to $x-1$ free

queue places, for the system with m inputs into the queue and n queue places, we may write:

$$p(\text{cell-loss}) = \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} p_{i+j} \cdot \pi_{n-j+1}$$

### 5.2.3 Numerical Evaluation of the Analytic Model

With the analytic solution derived, it is possible to study very low cell-loss probabilities, in the region $10^{-10}$ to $10^{-14}$ for example. An upper-bound of cell-loss was taken as $10^{-12}$ and the effect of varying m and n was performed. The results are shown on graph 5.7.




Graph 5.7: Variation of Upper−Bound of Cell−loss Probability $< 10^{-12}$ with the number of links into the final stage and the number of queue places.

Graph 5.7 shows the number of queue places required at a specific load for a specific number of input links to keep the cell−loss probability better than 1 cell in $10^{12}$. Note that the points do not appear to lie on a curve as they are presented as integer values and therefore the function appears stepped.

Observations on graph 5.7:

i) Taking a horizontal section of the graph (constant load), as the number of inputs into the final stage increases, the number of queue places required to maintain the same "grade–of–service" [in terms of cell–loss] also increases. From this it is possible to conclude that spreading the same load over an increasing number of input links makes for a more onerous arrival distribution, even though the load on an individual incoming link is reducing.

ii) The number of queue places required is not increasing linearly with the total input load. From this it is possible to conclude that for *reasonable* queue sizes the input load should be restricted to a value less than 90%.

iii) There is an element of "queue–sharing" as the number of input links increases. However, this is balanced by an element of inefficiency as the *throughput* of the final switching stage is measured in terms of the number of cells output and the most efficient configuration (highest 'throughput:number of queue places' ratio) is given with 2 input links.

iv) The most efficient interconnecting link utilisation is given in the 2 input case which would be 40% when the total incoming load is 80%. By contrast, in the 8 input case the interconnecting link utilisation would only be 10%.

An overall conclusion from this analytic experiment is that that most efficient architecture for the final stage switching element is when the number of interconnecting links is kept to a minimum.

### 5.2.4 Comparison of Analytic Results with Other Published Results

The data points shown on graph 5.7 are as a result of analysing and studying a final switching stage with 2, 4, 6 and 8 inputs carrying Bernoulli traffic and therefore interpretation of other published results must take these factors into account.

The results of the Prelude experiment given in [78] show that the outgoing queues need to be dimensioned for a cell−loss better than 1 cell in $10^{10}$ at 32 places for 0.7 load and 64 places for 0.85 load. Interpolating these results for a cell−loss better than 1 cell in $10^{12}$ gives 0.6 load requiring 28 queue places, 0.7 load requires 45 queue places and 0.8 load requires 62 queue places. The first 2 results are shown on graph 5.7 as points marked "P". The buffer requirement for the Prelude switch is larger than those from the analytic solution presented on the graph. This is due to both a larger number of incoming links and a more onerous cell arrival distribution.

In [79], Horn's results of modelling an M/D/1/S queue with cell arrivals having a Poisson distribution shows that 51 queue places are required at 0.85 load to keep cell−loss better than 1 cell in $10^{8}$. The analytic results derived here show that 42 queue places would be required to keep cell−loss probability better than 1 cell in $10^{12}$ and the difference can again be attributed to a larger number of incoming links in Horn's model, a different cell arrival distribution and the difference between ATM and standard queueing theory.

### 5.2.5 Comparison of Analytic Results with the Simulator Results

Equation {15} above presented a model based on simulator results for cell–loss in the region $10^{-1}$ to $10^{-5}$. It has been shown that the equation is a good model for this range and it is now used to predict the number of queue places for a wider range of this parameter's values.

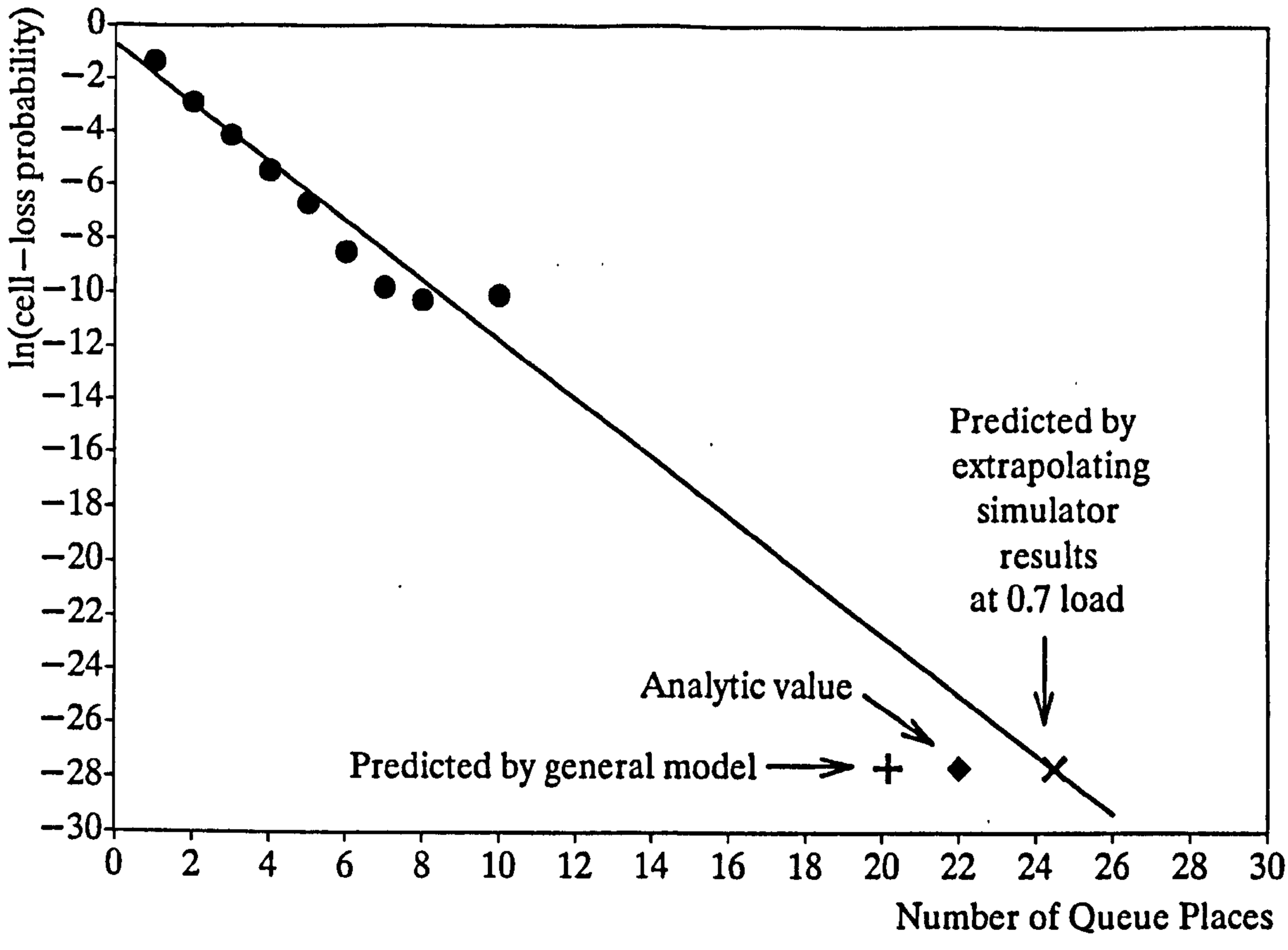
For this comparison, the simulator results were extrapolated in 2 ways:

i) using the best–fit straight line for the individual loads and

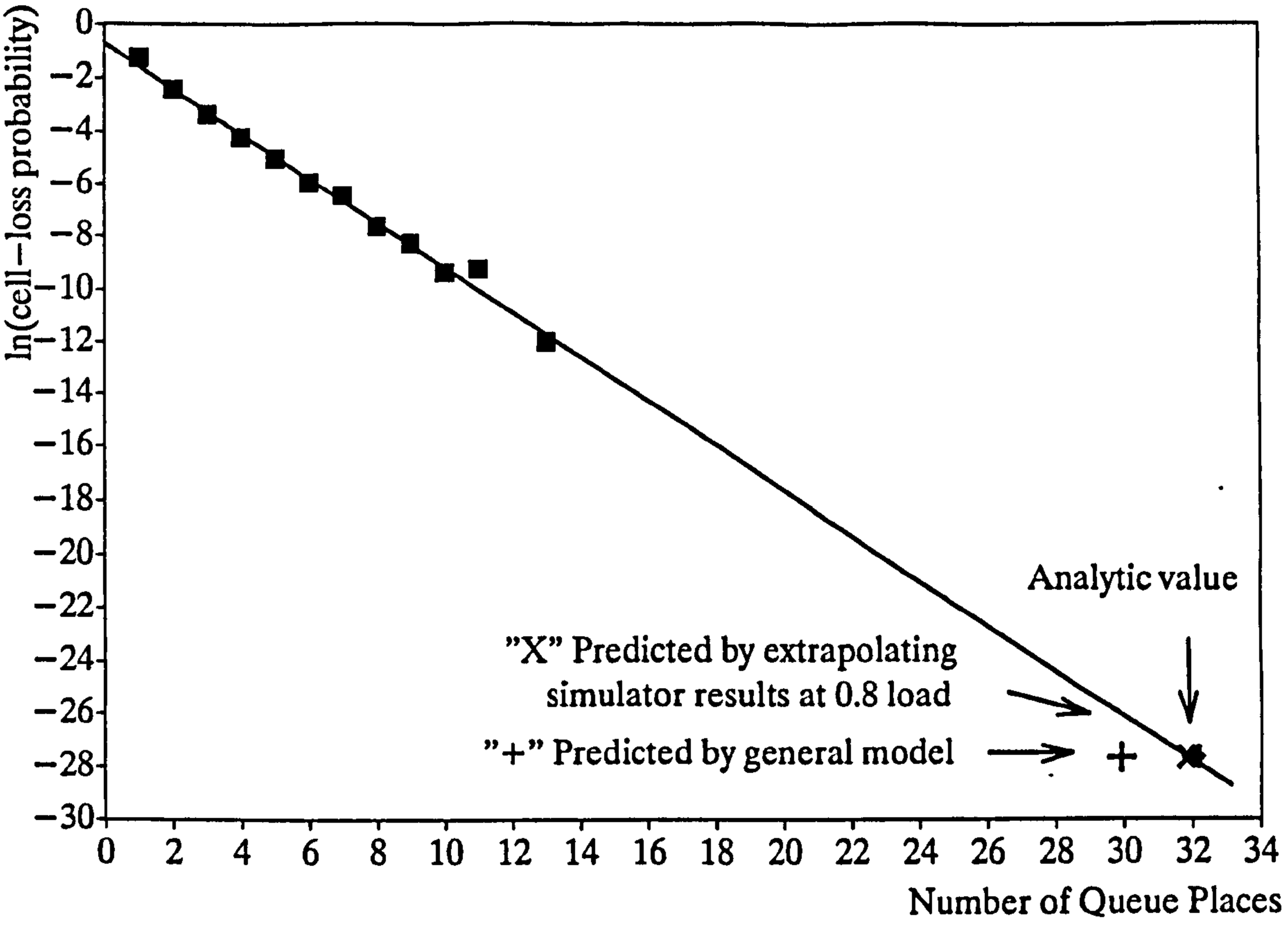
ii) using equation {15}, the general model.

Equation {15} is the generalised formula for all loads and should smooth the results from the individual loads. Both methods i) and ii) above were used to predict the number of queue places required to keep the cell–loss probability better than or equal to 1 cell in $10^{12}$. Two sets of results for input loads 0.7 and 0.8 are shown on graphs 8 and 9.

The points marked "X" were produced by method i) above, those marked "+" by method ii).




Graph 5.8: Simulator Cell−loss Extrapolated to $10^{-12}$, 0.7 Loading




Graph 5.9: Simulator Cell−loss Extrapolated to $10^{-12}$, 0.8 Loading

5.2.5.1 Discussion of Results

In both cases the number of queue places to guarantee a required cell−loss probability predicted by the general model extended from the simulator results is approximately 2 lower than the analytic value. In the case of 0.8 load the value predicted using the simulator results for that load show extremely good correlation with the analytic value. At 0.7 however, the effect of the misplaced point for 10 queue places (graph 5.8) can clearly be seen to have reduced the magnitude of the gradient of the line therefore giving a higher number of queue places than the analytic result. The simulator was re−run at 0.7 load to establish whether the data point for 10 queue places was misplaced in the first instance due to random statistical effects. The later

results indicated that this was indeed the case and therefore the extrapolated result for 0.7 load is subject to a degree of error causing the value to be too high.

Given the degree of correlation between the analytic values and those predicted by the general model, it is concluded that the general model given in equation {15} gives a good representation of cell–loss probability as a function of input load and number of queue places.

This chapter has presented the results of studies into the variation of cell–loss probability with buffer sizes using both analytic and simulation techniques. The next chapter moves on to the new areas of differential delay and timing problems and utilises results presented in this chapter.

# Chapter 6 Studies of Differential Delay (Jitter)

Earlier in chapter 2 some of the problems of ATM as a switching and multiplexing technique were discussed. Probably the most severe problem is differential delay, also called cell delay variation (CDV) or jitter. In this chapter differential delay is studied and the results of attempts to quantify the severity of various differential delays are presented. Comparisons between the re−timing delays of STM and ATM networks are made assuming a standard reference network. An ATM cell re−timing method is presented and the concept modified to give great improvements in performance. The concept of 2 Mbit/s PCM circuit emulation is studied with particular emphasis on the amount of timing distortion introduced by the packetisation/de−packetisation,

## 6.1 Definition and Cause of Jitter

Differential delay, now commonly referred to as 'jitter', is defined as the difference between the expected arrival time of a cell and its actual arrival time. This variable delay is caused when different cells of the same virtual circuit encounter the system queues in different states of occupancy. As the queue occupancy varies over time, the 'waiting−time' in the queues will also vary. Figure 6.1 shows how 2 cells, A and B,

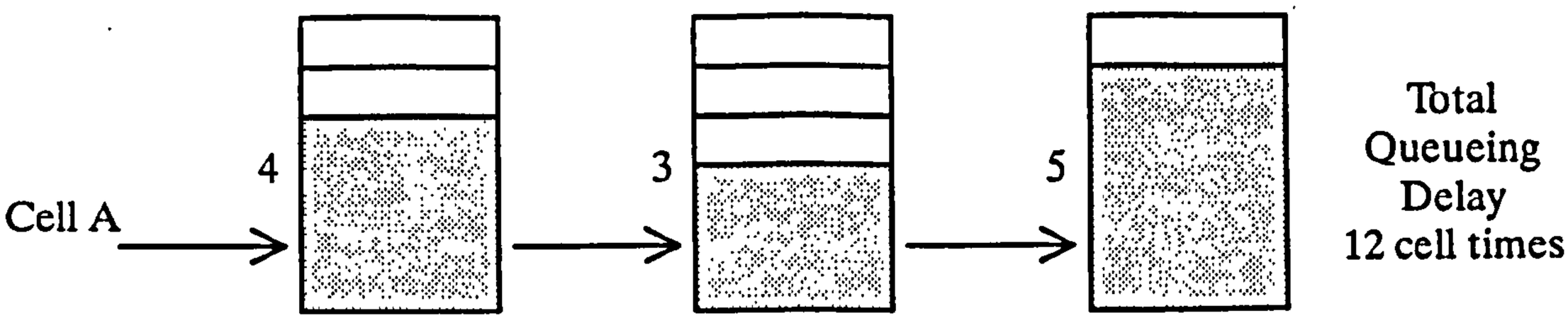



Figure 6.1: Variable Delays suffered by 2 different cells

encounter 3 queues. (As may be typical crossing one switching node for example.) As cell A arrives at the first queue, 4 queue places are already taken and so it has to wait 4 cell times before it is forwarded to the next queue in the system. The total delay for cell A in this case would be 12 cell times. Cell B however, encounters only empty queues and suffers no queueing delay. It can be seen that if A and B were successive cells of the same virtual circuit the original *cell–spacing* between them would be distorted.

Differential delay as studied in this chapter is concerned only with the accumulation of delay caused by multiplexing and variable length system queues. Other sources of delay such as packetisation (framing) delay or propagation delay are not considered here.

The term 'jitter' used in this context has nothing to do with the other type of jitter occurring on transmission systems as a result of noise on the line. A non–ambiguous

term such as 'scatter' for example could have been applied to differential delay but jitter has now become the accepted term.

## 6.2 Why Jitter is a problem

Notwithstanding the different introduction strategies for ATM as discussed in chapter 2, the so–called "Golden Rule" of telecommunications applies equally to ATM as to every other new technique or strategy. The golden rule is that any new equipment introduced into a network must be able to interwork with all of the old equipment and services. This is particularly troublesome for ATM as today's networks primarily carry speech which is extremely sensitive to both absolute and differential delay. Such is the magnitude of the problem that proposals to introduce ATM as an overlay network have been made which effectively negate the argument that the elegance of ATM is its ability to carry all types of present and future traffic.

Jitter is the distortion of cell–spacing. The original cell–spacing contains the vital source–timing information necessary to de–packetise and reassemble the original data stream. If the cell–spacing is distorted the source timing is lost and an alternative method of recovering the source clock is required.

## 6.3 The "Cell–Retiming" Solution

One such method for recovering the source clock of CBR traffic is a cell re–timing buffer at the exit of the ATM network. (Other methods such as Synchronous Residual Time Stamping, SRTS, have also been proposed.) The re–timing mechanism writes the cells arriving from the network into the buffer with a first–in, first–out protocol. The cells are read out periodically at a constant rate by a local clock. The frequency

of the local clock is adjusted with a feedback mechanism according to the average occupancy of the buffer. If the buffer occupancy increases the local clock frequency is increased. Similarly if the buffer occupancy decreases the local clock frequency is decreased. Figure 6.2 shows this principle.

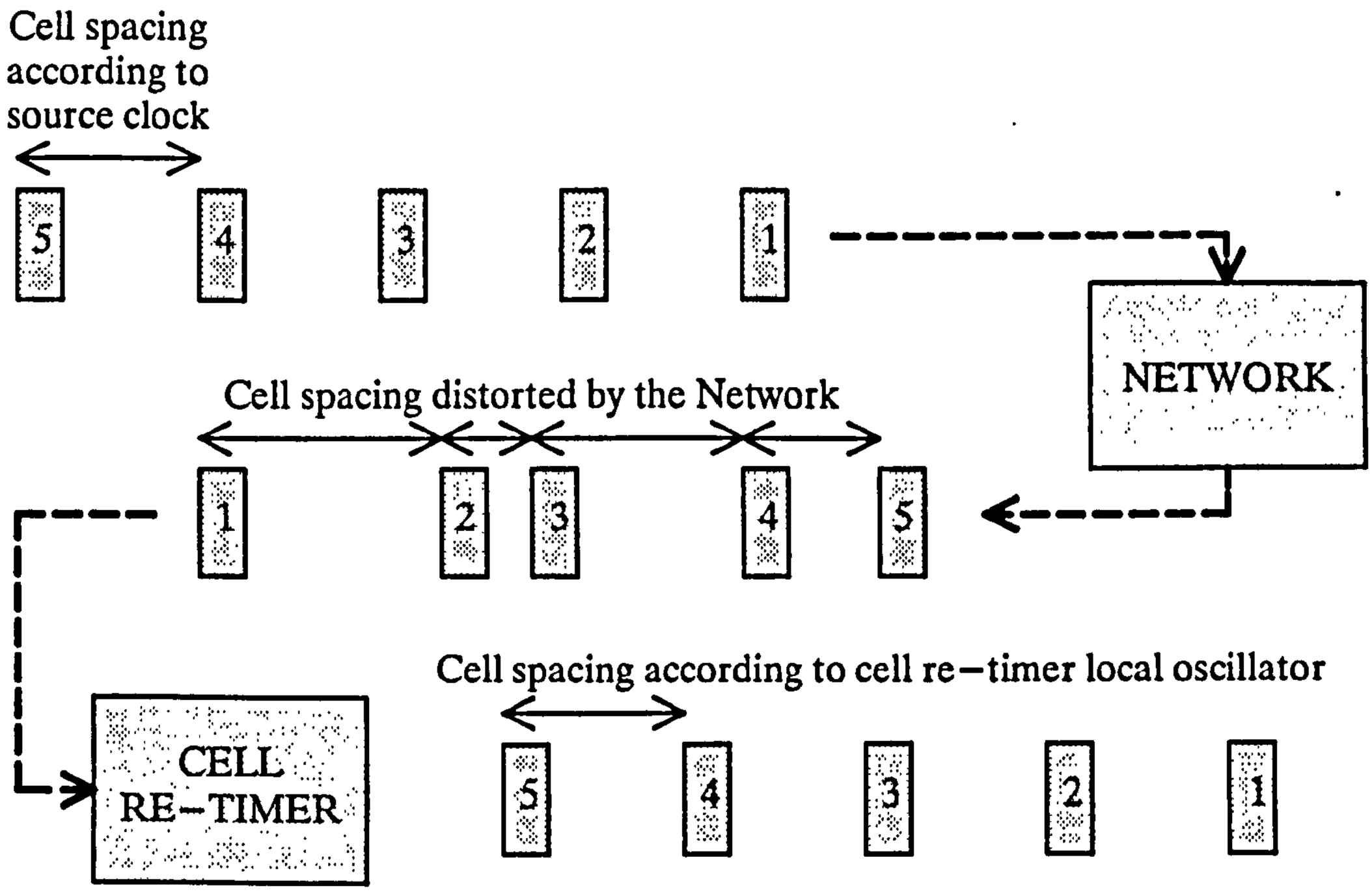



Figure 6.2: Cell Re–timing Principle

There are a number of theoretical problems with this method:

a. The cell re–timer will introduce an additional delay per cell, δr, into the circuit. This variable delay compensates for the variable network queueing delay, δn, such that

$$\delta r + \delta n = \text{constant} \quad - \{24\}$$

This means that the network grade–of–service (in terms of delay) offered on the circuit will be equal to the worst possible

delay the network could inflict on a cell even though the majority of cells would suffer a much smaller delay.

b. The local oscillator of the cell re–timer would be adversely affected by misrouted or lost cells. The oscillator control mechanism would interpret missing cells as if the buffer were being read too quickly, thus slowing the oscillator down temporarily. Extraneous cells would have the opposite effect of speeding the oscillator up. This problem may however be largely removed by the addition of sequence numbers in the cells.

c. Further memory is required in the network on a per active circuit basis. The additional queueing and cell handling would increase the probability of corrupting or losing a cell.

To assess the potential of the method it is necessary to quantify the required depth of the re–timing buffers to give an indication of memory requirements and of the overall delay ATM cells may suffer across the network. This would give a numeric value to the re–timing delay given by equation {24}. As a synchronous network also inflicts a re–timing delay on the traffic it carries it would be possible to compare the values for the 2 different transport and switching methods.

## 6.4 Calculation of Average Synchronous Network Re–Timing Delays

Synchronous networks suffer a certain amount of timing distortion caused by oscillator drift, variable propagation delays caused by different length transmission systems and diurnal temperature variations. These causes manifest themselves as both frequency and phase differences and are removed by alignment circuits at each node in the network. The alignment function is shown in figure 6.3.

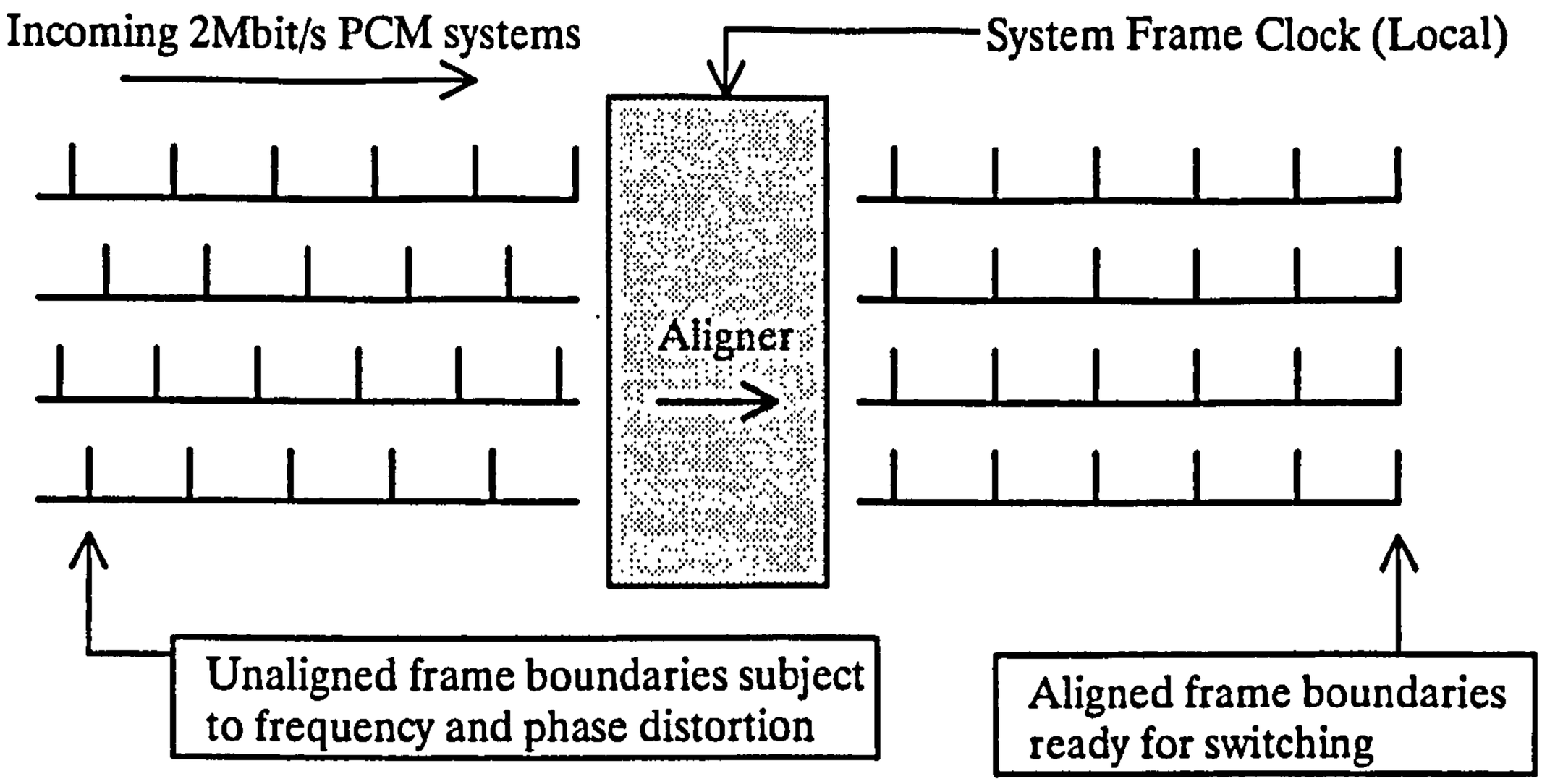



Figure 6.3: Frame Alignment at a synchronous switching node

To compare the performance of synchronous and asynchronous networks in this respect it is necessary to calculate the delay caused by the re-timing mechanisms in both types of network. For this, a *reference network connection* is assumed. A long distance call within the UK national network today typically will pass through 2 local exchanges, 2 trunk exchanges and 5 cross-connect nodes. This is assumed to be the reference network connection for both synchronous and asynchronous networks and is shown in figure 6.4.

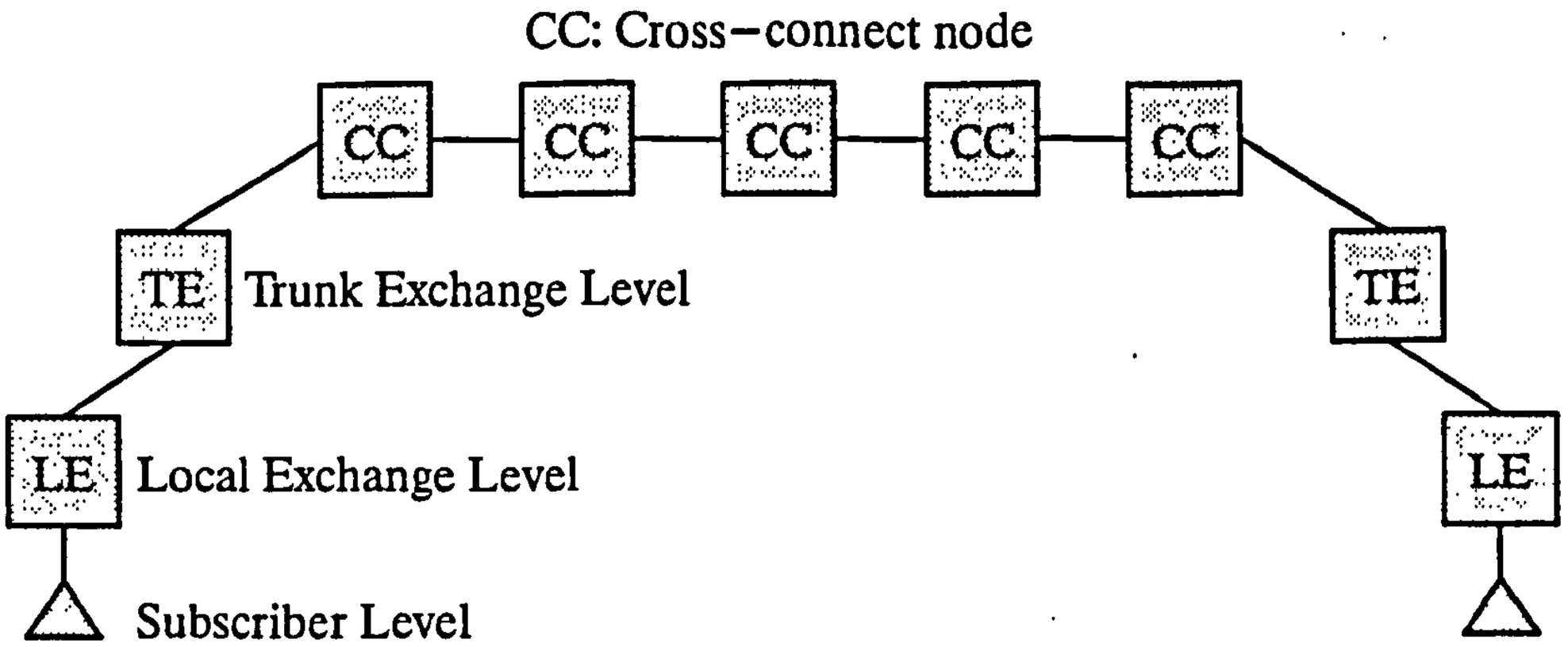



Figure 6.4: Reference Network Connection for calculation of re–timing delays

Only local and trunk exchange nodes have re–timing circuits. These consist of aligners on 2 Mbit/s PCM systems which align incoming frames to the exchange's local system clock as shown in figure 6.3. If the nominal speed of the PCM is fast with respect to the local clock whole frames are periodically omitted. If it is slow whole frames are periodically repeated. These 'slips' can be reported to a management system to assist early detection of faults. Frames suffer an average delay of 1.5 frames per node caused by the alignment circuit.

If $N_A$ = the number of alignment stages,

$A_D$ = the average delay per aligner (in frames),

in a synchronous network the average one–way re–timing delay will be

$$N_A \times A_D \times \text{duration of a frame}$$

$$= 4 \times 1.5 \times 125\ \mu s = 750\ \mu s$$

$N_A$, the 'number of alignment stages', is calculated by assuming the subscriber is directly connected to a concentrator (not remotely connected via a multiplexor which would add 1 more alignment stage). The concentrator is connected to the local

exchange via a 2Mbit/s PCM system terminated with the first aligner. The local exchange is also connected to the trunk exchange via a 2Mbit/s PCM system and terminated with the second aligner.

The 2 trunk exchanges involved are connected together via the cross–connect nodes in the transmission network. The 2Mbit/s PCM leaving the trunk exchange is bit–interleaved with other PCMs to form 34Mbit/s or 144Mbit/s systems of the plesiochronous digital hierarchy. The alignment in these stages is of the order of a few bits (approximately 488 ns per bit) to allow the bit interleaving to work and may therefore be disregarded.

The third and fourth aligners are situated at the receiving trunk exchange (terminating the PCM from the final cross–connect node) and the local exchange (terminating the PCM from the trunk exchange). There are therefore 4 aligners in the reference connection shown in figure 6.4.

## 6.5 Calculation of Asynchronous Network Re–Timing Delays

The re–timing delay in an asynchronous network is constant and given by equation {24} above plus a smaller amount per ATM pipe termination, $A_{A,}$ equivalent to the STM aligner delays. $A_{A,}$ denotes the average ATM pipe alignment delay.

$$\therefore \text{Re–Timing Delay (one–way)} = \delta r + \delta n + ( A_{A,} \times \text{no. of terminations} )$$

$A_{A,}$ may be determined by assuming that cells on an ATM pipe may arrive at any point in the switch timing cycle. Figure 6.5 shows the principle. Cells from 3 different incoming ATM pipes A, B and C are shown arriving periodically.

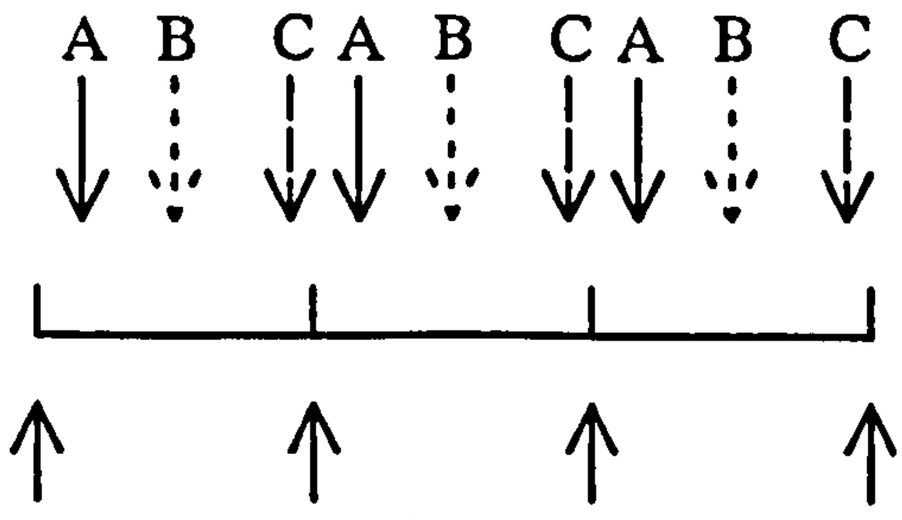



Figure 6.5: Possible Cell Arrival Times on 3 Different ATM Pipes with respect to the Switch Timing Reference

Their point of arrival can be any time δt from the switch timing reference to which all incoming cells must be aligned, where $0 < \delta t \leq 2.7$ μs. If time is assumed to be progressing from left to right, it can be seen that cells from stream A will be held up more than those of stream B which will be held up by approximately half a cell time, or 1.35 μs. The cells from stream C will be held up the least. As the arrivals will be evenly distributed, δt will be evenly distributed in the range $0 < \delta t \leq 2.7$ μs and therefore the mean value, $A_A$, will be 1.35 μs.

To calculate the value (δr+δn) it is assumed that δr is zero when δn has its maximum value, and the maximum value of δn occurs when a cell encounters every system queue in a (queue full − 1) condition, i.e. the cell occupies the last available place in the queue. δn then depends upon

a. The number of nodes in the network

b. The number of switching stages per node causing delay

c. The wait time of a cell in a queue for each queue place (equivalent to the cell existence time on a multiplex which is given by the transmission link speed)

d. The number of queue places per switching stage

To calculate δn, the worst case delay a cell can suffer in the network, the following formula is used:

$$\delta n = a \times b \times c \times d \quad - \{25\}$$

Parameter a. is assumed from figure 6.4 to be 9 nodes. Parameter b. can be derived by assuming there are 3 switching stages per node but with queueing only performed in 2 of them. Parameter c. is 2.7 μs, derived by assuming the transmission link speed is 155 Mbit/s.

As shown earlier in the thesis, parameter d. depends on three further parameters,

1) The equilibrium network load

2) The upper−bound of cell−loss probability (per node)

3) The number of links into the switching stage

In order to calculate a value for parameter d. it is assumed that the network load is 80%, the upper bound of cell−loss probability is $10^{-12}$ and there are 2 incoming links into a switching stage. Referring to the analytic results presented in graph 5.7, these three parameters, (80%, $10^{-12}$ and 2), result in 32 queue places being required per switching stage.

Substituting these assumed values into equation {25} gives:

$$\delta n = 9 \times 2 \times 32 \times 2.7\ \mu s \approx 1555\ \mu s$$

including the ATM alignment delay $A_A$ for 9 nodes, 9 x 1.35 μs, gives a total ATM re−timing delay of

$$\approx 1567\ \mu s$$

or just over double the 750 μs delay incurred in a synchronous system.

## 6.6 A method of Reducing the ATM Re−Timing Delay

There is a possibility of reducing the ATM cell re−timing delay for cells carrying speech. If the stringent requirement of a cell−loss probability of $10^{-12}$ is reduced to $10^{-4}$ for example, then 1 cell in $10^4$ may be discarded *due to excessive delay*. This is justified for the following reasons. For 64 kbit/s speech carried in 48 byte cells (user data excluding header), loss of 1 cell would cause a 6 ms break in the speech. During the author's work on fault location in Plessey's DSS−B in the late 1970's, (the synchronous digital switch used in System X local exchanges), it was found that breaks of 50 to 100 ms (during which diagnostic tests were performed on the switch) were undetectable by users. Discarding 1 cell in $10^4$ would result on average in a 6ms break every 60 seconds or 3 breaks per average 3 minute call.

This does not seem to be excessive, in fact would be a logical thing to do as there seems to be little point in aligning all cells to the very rare occurrences of extreme delay.

To establish the $(1 - 10^4)^{th}$ quantile of end−to−end delay, i.e the value $x$ such that

$$p(\text{end−to−end delay} > x) = 10^{-4}$$

it is necessary to know the frequency distribution of delay across the network. This may be estimated by firstly using the simulator to study the frequency distribution of delay across a single switching stage and then using convolution techniques to obtain an estimate of the distribution of delay across a single switching node, and finally across the reference network.

There are however a number of limitations in this approach. Firstly, overload is unlikely to be evenly distributed in time. That is, if a cell is delayed beyond the $10^{-4}$

quantile it is likely that subsequent cells will also be delayed beyond that point. However we have seen that a break in speech caused by, say, 10 successive lost cells would be unlikely to be detected. This effect is self–regulating to an extent in that if the service becomes unacceptable the subscribers will hang up thus removing part of the overload. Hopefully the system's call acceptance control (CAC) algorithms would reject subsequent new call attempts until the overload was removed. This technique seems harsh but it is used in present STM networks to recover from fault and overload conditions.

Secondly, the convolution method assumes a constant network load across all nodes and is therefore a simplification but in the absence of complete network models at least gives an *indication* of what the delay distribution across an ATM network may be.

### 6.6.1 Production of Single Node Delay Distributions by Simulation

The delay distributions produced from simulator runs were achieved by time stamping the cells at the input to the switch and comparing this with the system time at their departure from the switch. In chapter 3, the warm–up time of the simulator was shown to be 25000 cell times. To ensure that this warm–up time was still valid for the jitter experiment (in which the results are not averaged in the same way), the effects of warm–up times of 0, 5000, 10000, 15000 and 20000 were studied. The overall run length was 25000 cell times.

Graph 6.1 shows the resulting jitter distributions for various warm–up times using 100% load. This load was chosen as it has the longest warm–up time. The effect of an empty switch at start–up can be seen in the delay distribution for zero warm–up time. A small number of cells, 38, 35, 31, 22, 17 and 16 suffered 0, 1, 2, 3, 4 and 5 cell

times delay respectively. This effect is entirely removed with a warm−up time of 5000 cells and therefore a warm−up time of 25000 cells remains valid for this experiment. Additionally, a warm−up period exceeding 25000 cell times would be acceptable.

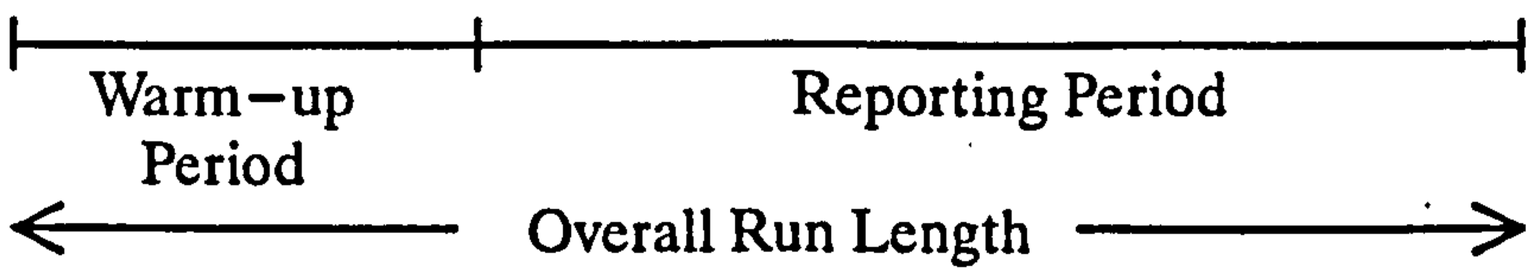
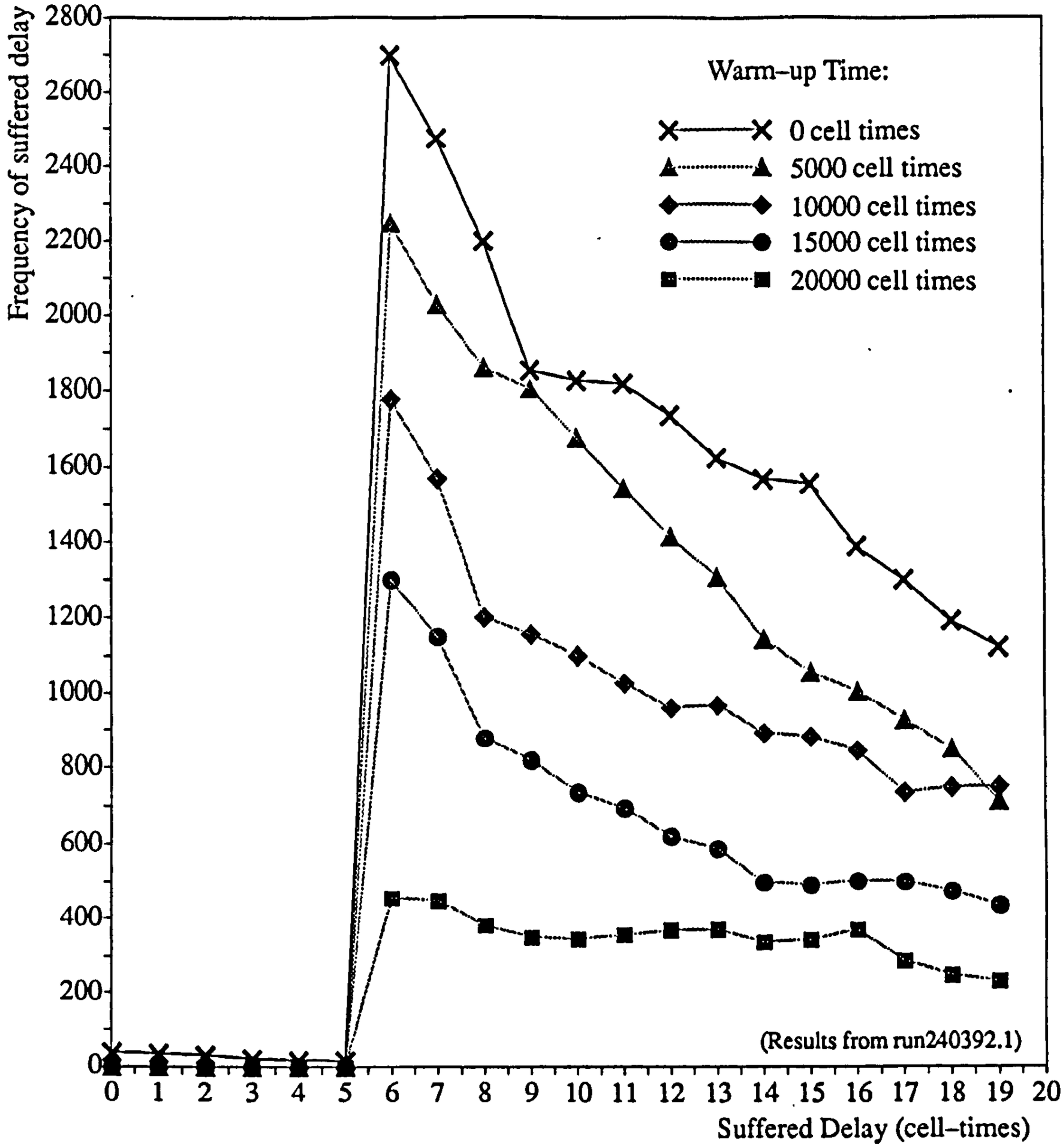


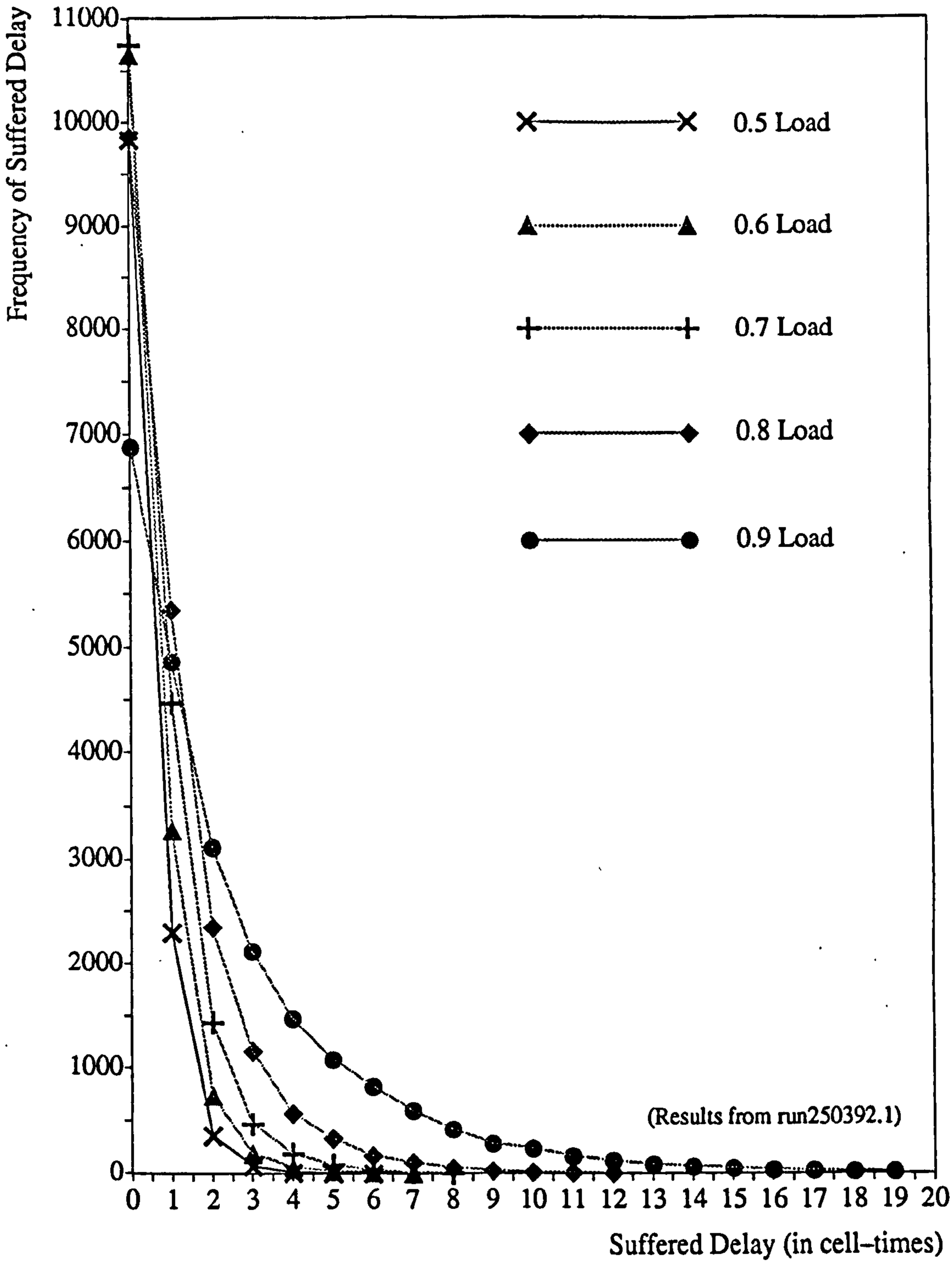

Figure 6.6: Definition of Timing Terms

The different warm−up times produce different curves on graph 6.1 due to the constant overall run length. This produces differing reporting periods and therefore a different total number of cells through the switch in each simulator run. The same experiment was subsequently repeated with overall run lengths changed to give a constant *reporting period* of 25000 cell times. Within the previously mentioned limits on accuracy of the simulator results, the different warm−up times, apart from zero, gave the same delay distribution.




Graph 6.1: The effect of different simulator warm−up times on the jitter analysis

Having established a valid warm−up time for the simulator, delay distributions for various loads were studied.




Graph 6.2: Cell delay distributions for various input loads.

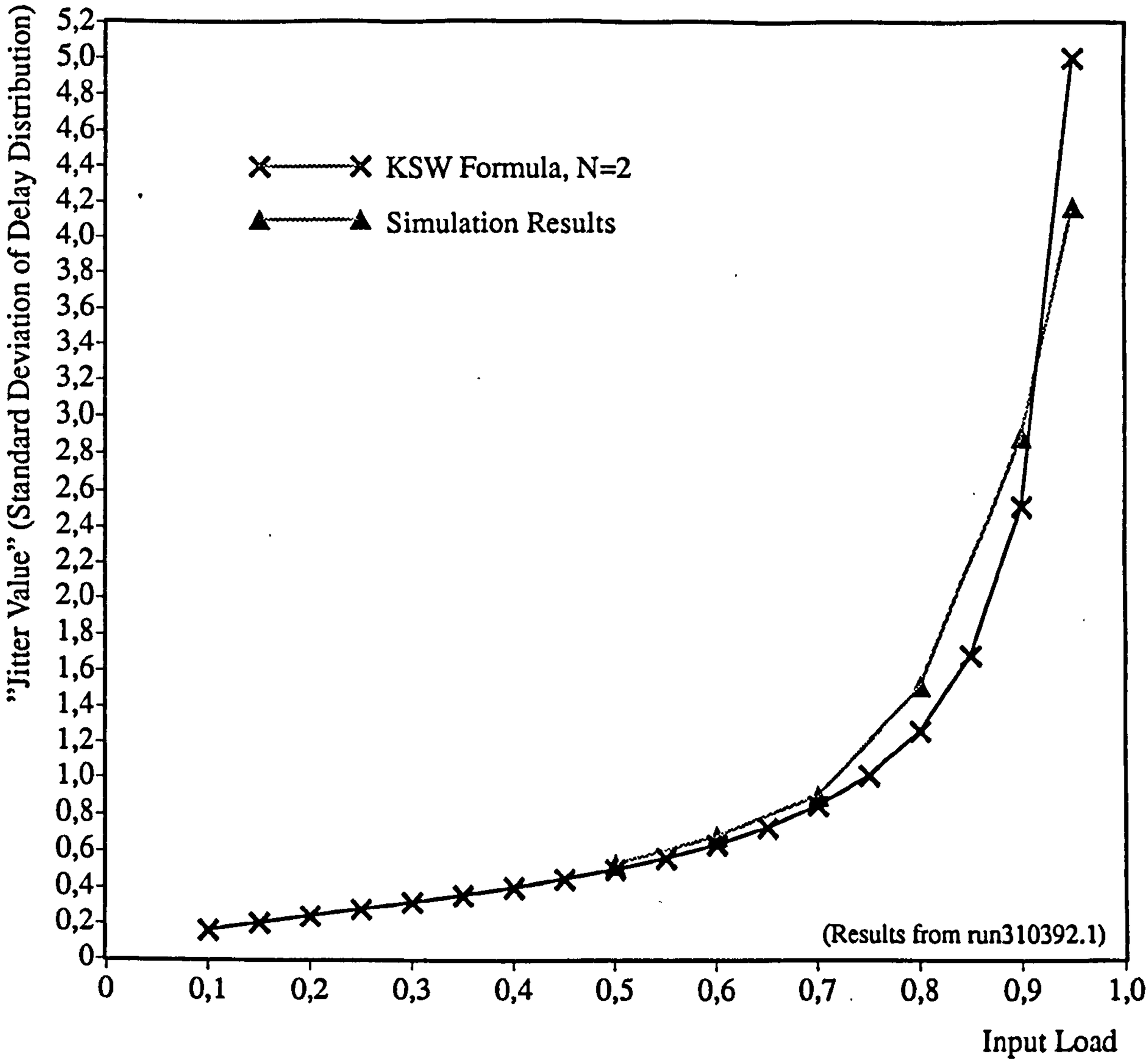
Graph 6.2 shows the delay distributions produced. The warm-up time was 75000 and the reporting period 25000 cell times.

In order to validate the simulator results shown on graph 6.2, the formula for the standard deviation of the waiting time distribution derived by Kruskal, Snir and Weiss in reference [80] is employed. The formula is:

$$\text{Standard Deviation of Waiting Time Distribution} = \sqrt{\frac{(1 - \frac{1}{n})p(6 - 5p(1 + \frac{1}{n}) + 2p^2(1 + \frac{1}{n}))}{12(1 - p)^2}}$$

where p is the load, or utilisation, and n is the number of links into and out of the switching stage. The formula is referred to as the KSW formula.

Using the value $n = 2$ in the KSW formula it is possible to calculate standard deviations of the waiting time distributions for various values of p and compare these with the standard deviations of the distributions obtained by simulations and shown on graph 6.2. The result of the comparison is shown on graph 6.3.




Graph 6.3: Comparison of Standard Deviations of Delay Distributions obtained by Simulation and Analysis

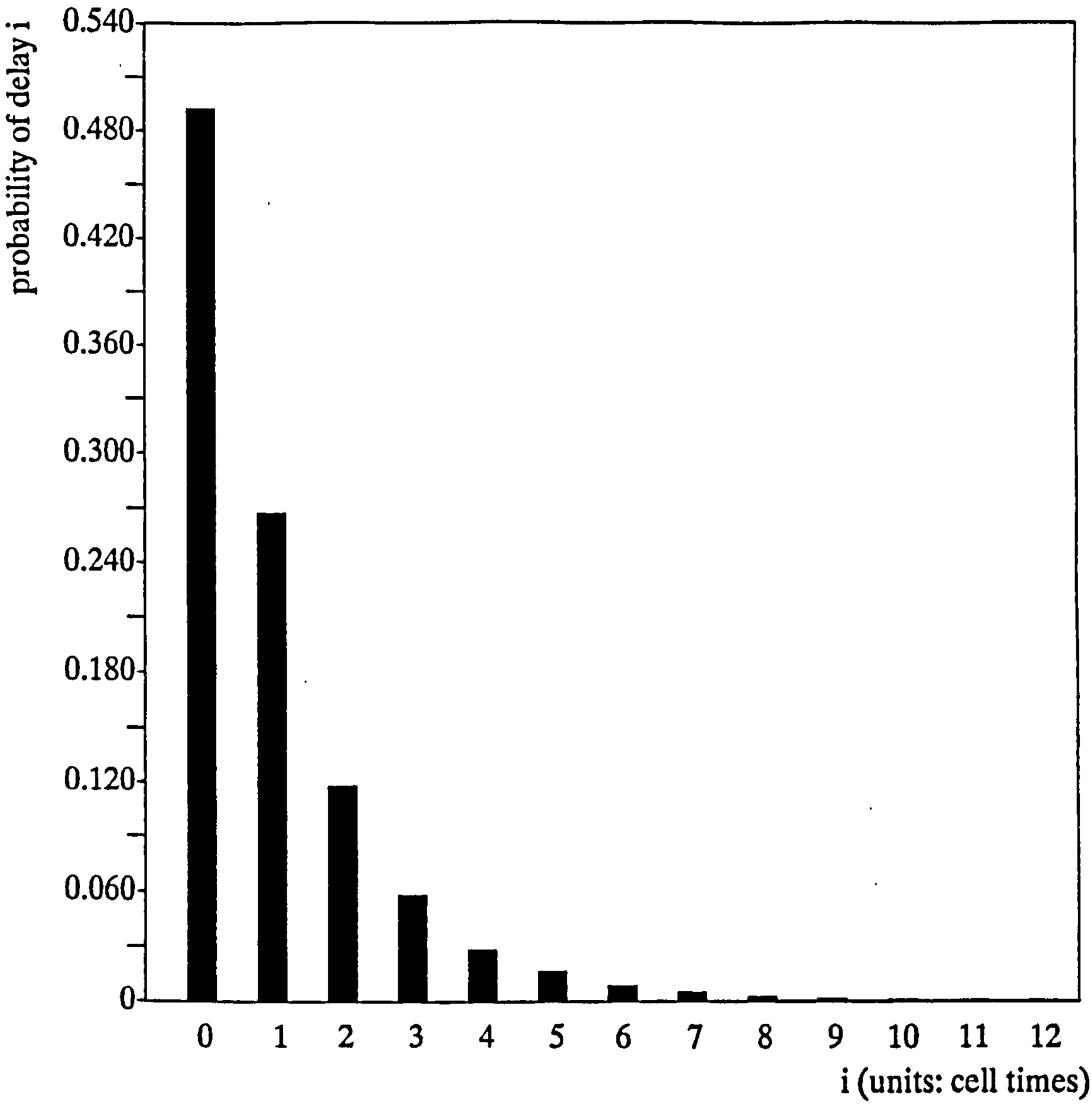
Two significant conclusions can be drawn from graph 6.3:

a. The simulation results are in good agreement with the KSW formula.

b. The KSW formula produces a higher value for the standard deviation at high loads ( > 0.9 here ) due to its *infinite buffer / zero cell–loss* assumption.

The good agreement between the simulation results and the KSW formula gives a high degree of confidence in the accuracy and validity of the distributions shown on graph 6.2.

For the purposes of estimating the ATM re−timing delay, the frequency distribution at 0.8 load shown on graph 6.2 is of interest. The *frequency of suffered delay* (column 2 of table 6.1 below) is normalised to give a discrete probability distribution as shown in graph 6.4.




Graph 6.4: Probability Distribution of Delay for a Single Switching Stage, 0.8 Load (Probability Distribution "A" below)

Graph 6.4 shows the probability distribution for delays across a single switching stage with 2 inputs at 0.8 load. The switching stage had 32 queue places but the maximum queue occupancy during the run in question was 12. The source data used to produce the frequency distribution shown on graph 6.4 is presented in table 6.1.

| Cell Delay (in cell times) | Frequency of Delay |
|---|---|
| 0 | 9856 |
| 1 | 5350 |
| 2 | 2358 |
| 3 | 1162 |
| 4 | 572 |
| 5 | 332 |
| 6 | 169 |
| 7 | 102 |
| 8 | 52 |
| 9 | 25 |
| 10 | 10 |
| 11 | 5 |
| 12 | 2 |

Table 6.1: Source data for Graph 6.4

### 6.6.2 Production of Network Delay Distributions by Convolution

From graph 6.4, the discrete probability distribution of delays across a single switching stage, it is possible to estimate the probability distribution of delays across a single switching node and thence across a number of nodes forming a network.

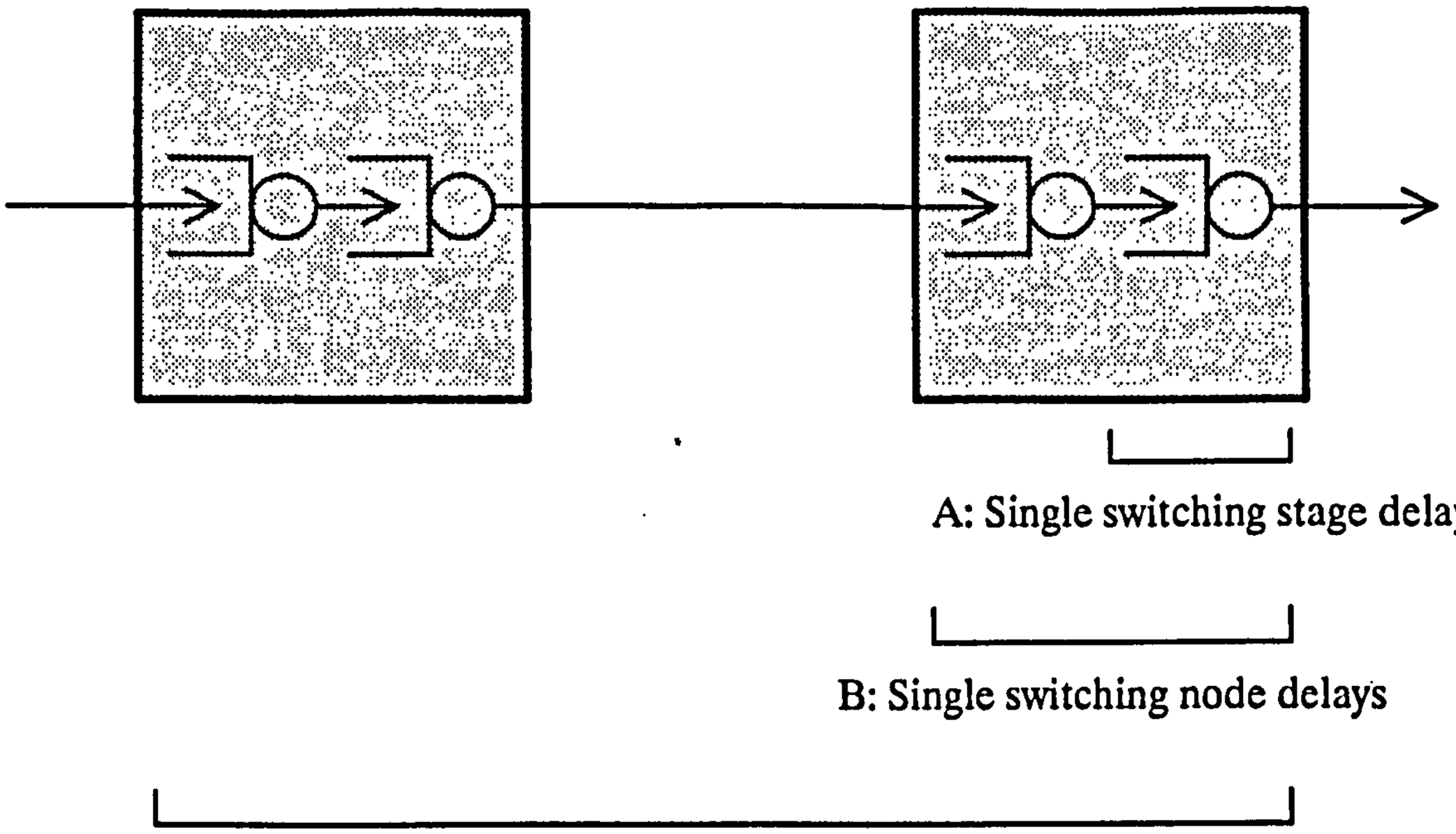


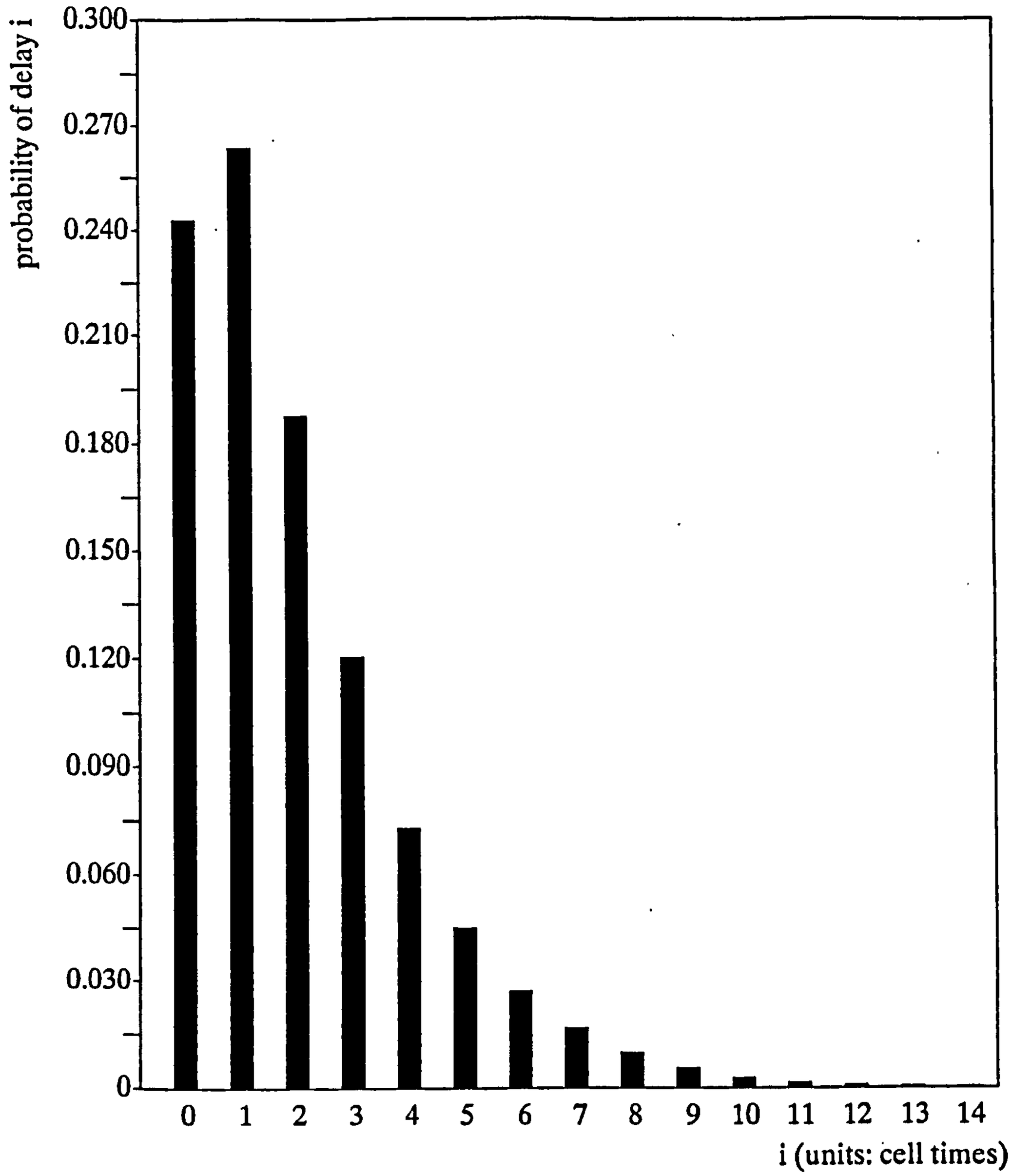

Figure 6.7: The Application of Convolution Techniques to Estimate "C" from "A"

Figure 6.7 shows the principle. The delay probability distribution "A" has been determined by simulation and validated as far as possible against the KSW formula. The delay probability distribution "B" can be *estimated* by auto–convoluting "A". In this case the result of the auto convolution is only an estimate as the distributions are not independent. Having obtained estimates for "B", it can be convoluted to estimate "C". The number of convolutions to produce "C" must equal the number of nodes in the network.

From convolution theory [81],

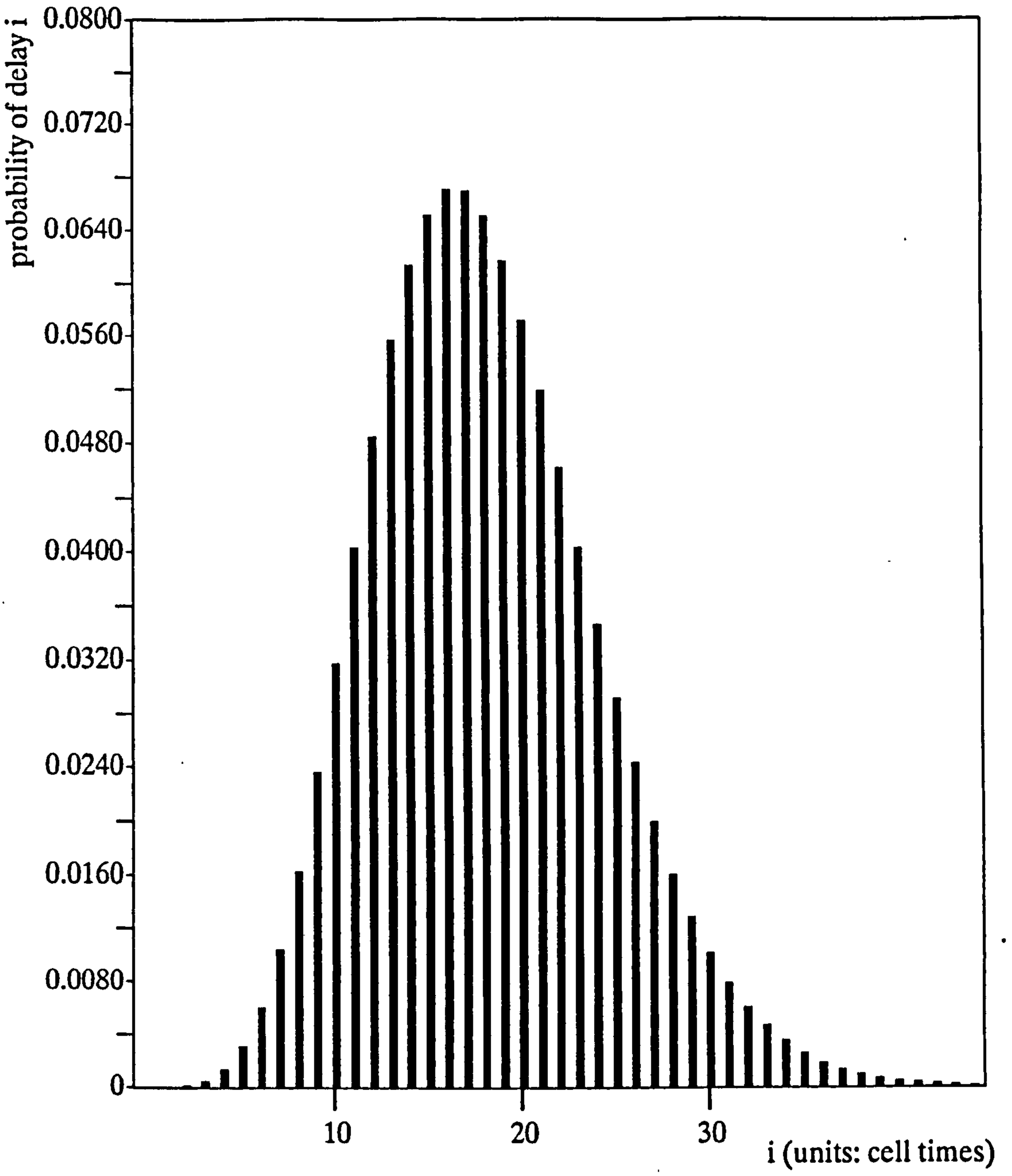
$$p(B = k) = \sum_{j} p(A = k - j).p(A = j)$$

Applying the theory to the probability distribution p(A) shown on graph 6.4, p(B) is obtained.



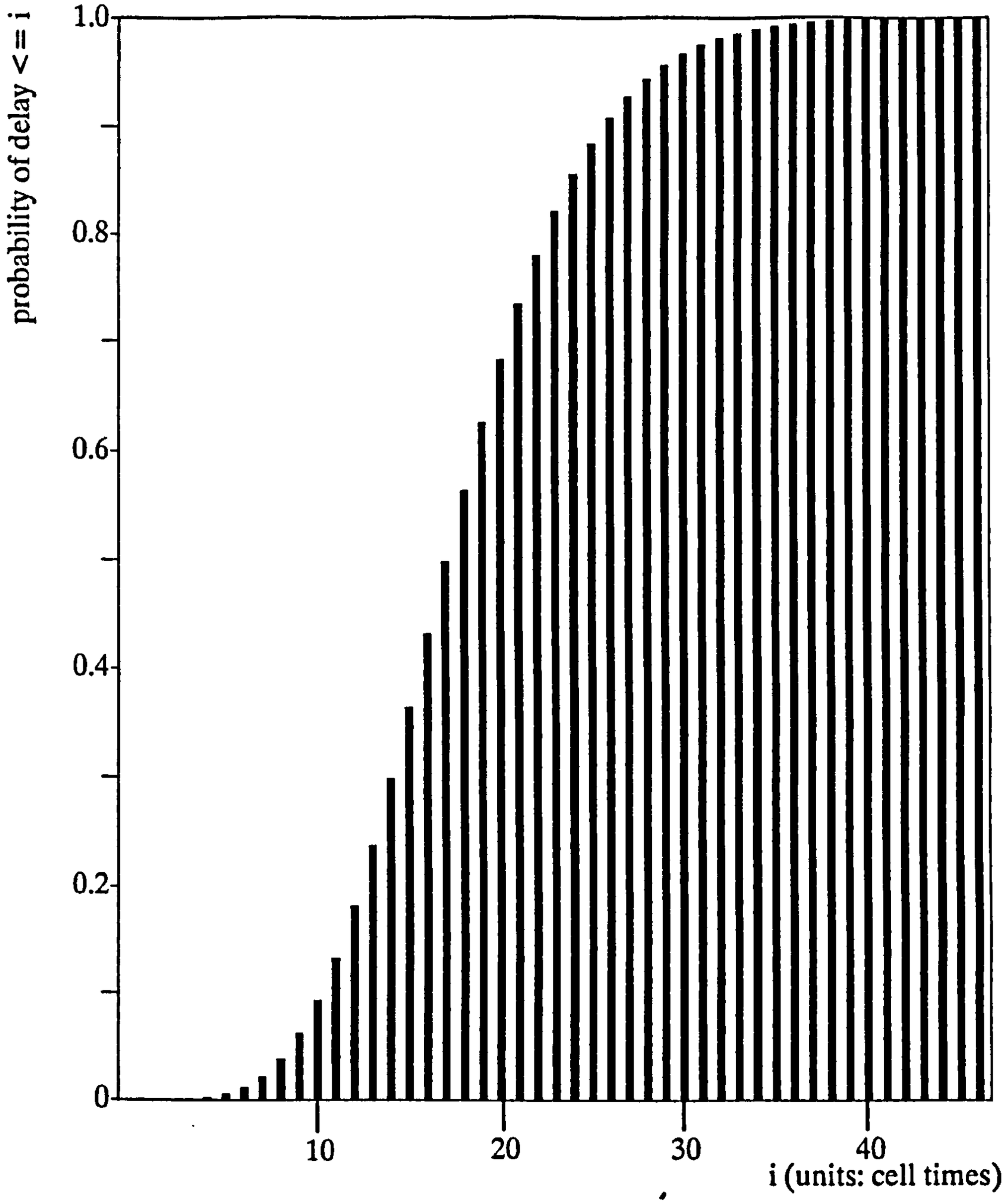

Graph 6.5: Estimated Probability Distribution of Delay for a Single Switching Node, 0.8 Load (Probability Distribution "B" above)

Graph 6.5 shows the values of p(B) that are greater than $10^{-4}$. The reference network shown in figure 6.4 has 9 nodes. Convoluting the probability distribution p(B) nine times produces the estimated probability distribution of delay shown in graph 6.6.




Graph 6.6: Estimated Probability Distribution of Delay across a 9−Node Network, 0.8 Load

To find the value of the $10^{-4}$ quantile of delay it is necessary to form the *cumulative* probability distribution function such that the y−axis becomes the probability of delay less than or equal to i. The cumulative probability distribution function is shown on graph 6.7.




Graph 6.7: Estimated Probability Distribution of Delay across a 9−Node Network, 0.8 Load

From the source data of graph 6.7,

$$p(\text{delay} \leq 46) = 0.999863,$$

$$p(\text{delay} \leq 47) = 0.999903$$

The $10^{-4}$ quantile of delay is therefore between 46 and 47 cell times. This implies that a cell re–timing buffer of depth 47 cells would lose less than 1 cell in $10^{-4}$ in the circumstances detailed above.

The re–timing delay applied in this case can again be calculated using equation {24}. if the network delay, $\delta n$ is assumed to be zero, $\delta r$ would be a maximum and would be the cell–time multiplied by the depth of the re–timing buffer. Substituting the values of 2.7 μs and 47 queue places produces for the re–timing delay, a value of

$$( 126.9 + (A_A \text{ . no. of nodes}) ) \approx 139 \ \mu s .$$

This value, 139 μs compares very favourable with the 1567 μs previously calculated allowing for the worst case delay across the network and shows that significant performance improvements can be made by discarding cells delayed beyond a critical value.

## 6.7 2 Mbit/s Circuit Emulation – An Analysis of Phase Distortion

It has often been suggested that ATM can carry plesiochronous or synchronous transmission systems such as the CEPT 2.048 Mbits/s PCM system. In this subsection this claim is investigated with respect to the potential phase distortion that may be incurred. (A definition of synchronisation related terms is given in appendix C.)

In the circuit emulation method the PDH (Plesiochronous Digital Hierarchy) signal is packetised at the entrance to the ATM network, transported across the ATM network in standard cells and reconstituted at the exit to the ATM network. It is clear that the much higher data rate of the PDH signals compared to speech will reduce the packetisation delay considerably. However as with speech the source timing is still lost and the magnitude of the phase distortion applied because of this is of interest.

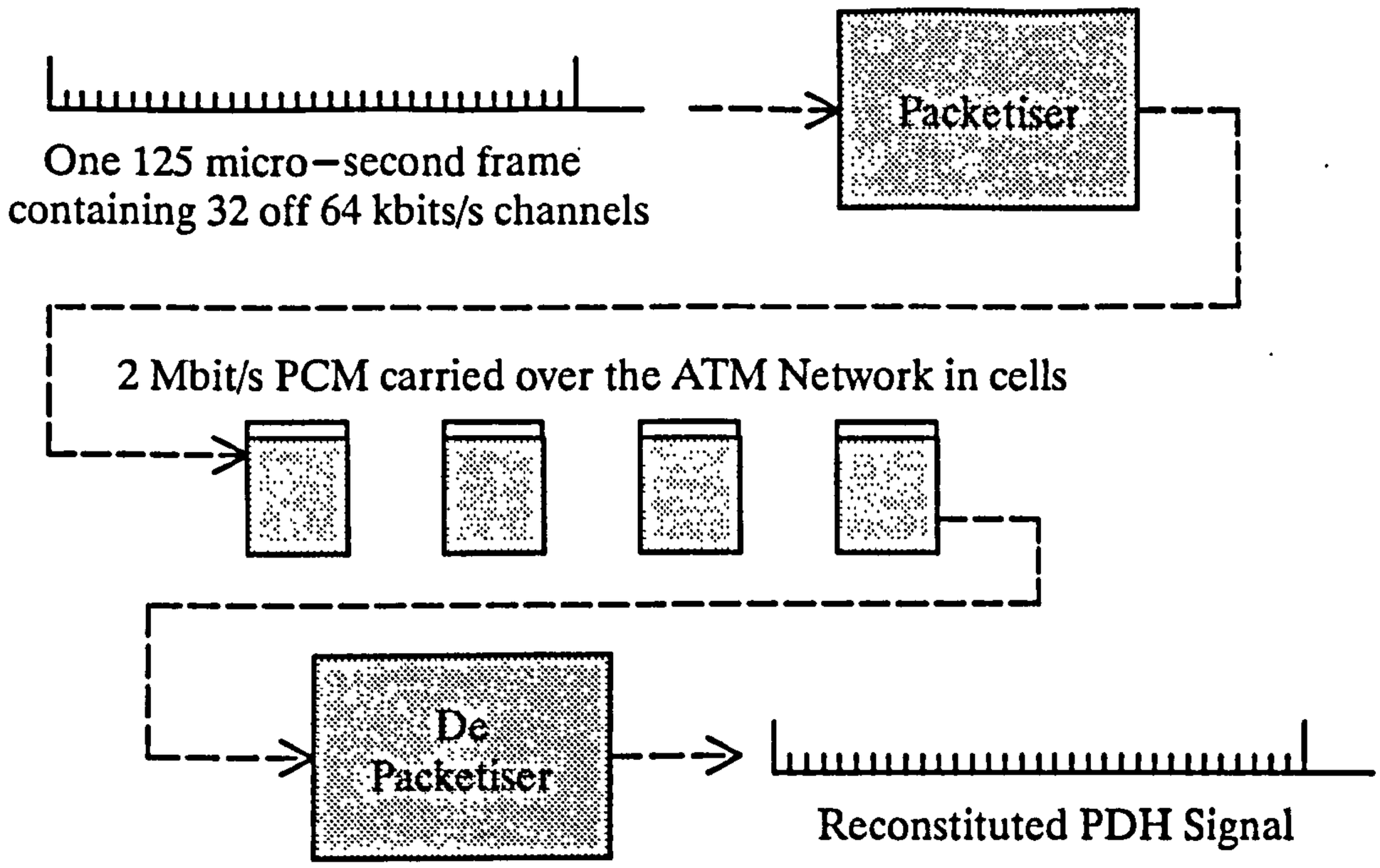



Figure 6.8: 2 Mbit/s Circuit Emulation across an ATM Network

The scenario is shown in figure 6.8. The 2Mbit/s PDH system enters the ATM network at an interworking unit containing a packetiser. The details of the packetiser, such as whether sequence numbers are applied to the cells, are not important to this analysis. The cells are transmitted across the ATM network and the PDH signal reconstituted by the de−packetiser in the interworking unit at the far−end interface between the ATM and STM network. For the purpose of the analysis, the following assumptions are made:

a. The PDH system under study is a precise 2.048 Mbit/s signal with no timing distortion.

b. The transmission system in the ATM network is 155.520 Mbit/s with no timing distortion carrying cells in "pure−ATM" mode.

c. The full ATM cell payload (48 bytes) is used to carry sequential octets from the PDH system.

d. The ATM transmission system is 100% available for the single PDH source, i.e. there is no multiplexing delay.

e. The ATM switching nodes are 100% available for the PDH source, i.e. there is no variable queueing delay.

Let $UI_P$ represent the length of time required to transmit one bit at 2.048 Mbit/s (unit interval) and $UI_A$ the length of time required to transmit one bit at 155.520 Mbit/s.

$$\text{therefore } UI_P = 1 / (2.048 \times 10^6) \text{ s} \approx 488.28 \text{ ns}$$

$$\text{and } UI_A = 1 / (155.520 \times 10^6) \text{ s} = 6.43 \text{ ns}$$

The time taken to transmit one cell in the ATM network is therefore:

$$UI_A \text{ x number of bits per octet x number of octets per cell}$$

$$= 6.43 \times 8 \times 53 = 2.72632 \text{ µs}$$

The time taken to fill one ATM cell with 48 octets taken from the PDH signal is:

$$UI_P \text{ x number of payload bits per ATM cell}$$

$$\approx 488.28 \times 48 \times 8 \approx 187.5 \text{ µs}$$

therefore the packetiser will have to wait for 187.5 / 2.72632, or approximately 68.774025 *cell–times* between requiring a slot on its outgoing ATM side. Unfortunately the packetiser is only able to wait an integer number of cell–times. If cell number *n* is ready for transmission exactly on a cell boundary the subsequent 6 cells will be timed as follows:

| Cell Number | Ready at (in cell intervals) | Transmitted at (in cell intervals) | Cell spacing |
|---|---|---|---|
| n | 0 | 0 | – |
| n+1 | 68.77 | 69 | 69 |
| n+2 | 137.55 | 138 | 69 |
| n+3 | 206.32 | 207 | 69 |
| n+4 | 275.09 | 276 | 69 |
| n+5 | 343.87 | 344 | 68 |
| n+6 | 412.64 | 413 | 69 |

Table 6.2: Cell Spacing for a 2 Mbit/s System carried by the Circuit Emulation Method

Table 6.2 shows that the cell spacing suffers a discontinuity between cells (n+4) and (n+5). The magnitude of the discontinuity is one cell−time, or 2.72632 μs, and the frequency of the discontinuity may be estimated as follows:

Over the long term, cells carrying a circuit emulated signal will have a cell−spacing of 68.774025 cell times. To produce this time average, let there be m occurrences of 69 cell times between circuit emulated cells and n occurrences of 68 cell times.

$$\therefore (n+m) \times 68.774025 = (n \times 68) + (m \times 69) \quad - \{26\}$$

where n and m are both integers with m > n (because 68.77 is closer to 69 than 68). The lowest 2 integers that solve equation {26} are m=9923 and n=2897. There is therefore a discontinuity (change in cell spacing from 69 to 68 cell times) on average every m/n circuit−emulated cells.

Therefore, the discontinuity in cell spacing will occur every 3.425 x 69 cell times, ≈ every 236 cell times, or, as there are approximately 366795 cells per second at 155.520 Mbits/s, at a frequency of approximately 1554 Hz. Therefore a phase−distortion of 2.73 μs at a frequency a 1554 Hz will be applied to the 2.048 Mbit/s signal despite the 5 favourable assumptions above. As $UI_P \approx 488.28$ ns, this equates to jitter of over 5.5 $UI_P$ at 1554 Hz.

Referring to the CCITT specification G.823 [82], the maximum permissible jitter on a 2 Mbit/s system is less than 1.5 $UI_P$ at this frequency. (Note: CCITT specification G.824 gives the corresponding figures for the US PDH.)

## 6.8 Discussion of Jitter Studies

In section 6.4 a value for the average re–timing delay across a synchronous network was determined by assuming a standard reference network model. This figure is 750μs. In section 6.5 a value for the re–timing delay across the same reference network implemented with ATM switches is found. This value is 1567μs. These figures highlight a number of points:

a. The synchronous transmission network (the multiple cross–connect nodes) introduces very little additional re–timing delay due to the nature of the bit interleaved structure of the plesiochronous digital hierarchy. The re–timing delay is relatively unaffected by the number of cross–connect nodes traversed allowing great flexibility for the network operator.

b. The ATM re–timing delay is, somewhat surprisingly given the large number of queueing stages across the network, only twice that of the STM equivalent delay. The main reason for this is the repetition rate of the 2 systems, 125μs for STM and 2.7μs for ATM.

c. Given that speech is not particularly susceptible to data loss, the re–timing delay can be greatly reduced by adopting a strategy of discarding 1 in $10^4$ speech cells due to their delay exceeding a fixed value rather than aligning all cells to the worst possible delay. This gives a re–timing delay across the reference network of 139μs.

d. To allow services sensitive to both time distortion and data loss to be carried across and ATM network, it would be preferable to establish a topology such that the cells cross a minimum number of ATM nodes. This reduces the flexibility for the network operators.

In section 6.7 the jitter applied by an idealised ATM network (using 155 Mbit/s pure–ATM bearers) to a circuit emulated 2.048 Mbit/s system from the CEPT plesiochronous digital hierarchy was found to be 5.5 $UI_P$ at 1554 Hz (exceeds the limits specified by CCITT). This is caused by the basic difference in clock speeds but still shows the magnitude of the effect under "ideal" conditions, i.e.

a. Absolutely stable PDH source 2.048 Mbit/s clock. In reality the source clock will be subject to jitter and wander which will make the problem worse.

b. Absolutely stable ATM bearer clock at 155.520 Mbit/s. Again, this clock will be subject to jitter and wander.

c. A completely free ATM multiplex. In reality the multiplex needs to be used to about 80% occupancy. This implies that the packetiser, when ready with a full cell, will have to wait for a free cell slot to pass before its cell can be unloaded. Whilst it was shown in chapter 2 that the ATM switching and multiplexing technique does not equate exactly to the classical M/D/1 queue, modelling an M/D/1 queue using the Pollaczech–Khinchin mean–value formula given in Kleinrock [44] page 187 to give an *indication* of the affect of non–zero multiplex occupancy gives a probability of only 0.2 that the cell required will be free and a mean expected wait per cell of 2 cell times. Thus the jitter imparted by a real system will be much higher than the 5.5 $UI_P$ calculated.

d. Completely empty ATM switches. No account of the jitter imparted by non−empty ATM switches has been made in the calculation of the 5.5 $UI_P$ at 1554 Hz. Earlier in chapter 6 various examples of waiting−time distributions were given as a function of load. These again, the effect increasing as the number of nodes crossed increases, will have a detrimental effect on the final jitter value.

e. "Pure−ATM" transmission mode. The scenario presented above does not take account of the management overhead in the 155 Mbit/s frame which again reduces the probability that the packetiser can access precisely the cell it requires.

Generalising the above, it can be seen that if by the time a cell reaches the de−packetiser at the exit of the ATM network it has been delayed by only one slot away from its ideal position, 5.5 UI jitter is incurred. The frequency distribution of such 'delays' would be a complex function of source PDH clock accuracy, ATM transmission clock accuracy, ATM multiplex occupancy and ATM switch occupancy.

In any event, idealising all of these four parameters results in an unacceptable accumulation of jitter, and it is therefore concluded that 2 Mbit/s circuit emulation over an ATM network will not meet the stringent jitter requirements of CCITT without additional timing methods of some sort.

By comparison with other literature, Wu and Kerner [83] in contrast advocate the use of circuit emulation techniques over a broadband packet−based system. They predict an increasing demand for unchannelised DS−1 and DS−3 transmission systems (1.5 Mbit/s and 45 Mbit/s systems) and propose a 2−stage fast packet switching architecture to support the circuit emulation. However, the parameters they have optimised are the round−trip delay, keeping this below 10ms, and cell−loss

probability, keeping this better than $10^{-10}$. They have not considered the affect of phase distortion on the reconstituted signal as discussed above.

This chapter concludes the research presented in the thesis. The next, final chapter brings together the results from this and previous chapters, suggests potential new areas of research and finally discusses the significance of the principal conclusions of this work.

# Chapter 7 Conclusions and Final Discussion

## 7.1 Conclusions

The conclusions in this sub–section fall into two categories. The first category contains specific conclusions having supporting evidence in the content of the thesis. The number in brackets following the conclusion refers to the section within which the evidence for the conclusion is contained. The second category contains more general, less rigorous conclusions that are nevertheless still significant having been arrived at during the course of this research.

An ATM switch cannot be modelled exactly using conventional queueing theory as the underlying assumptions are untrue (2.6). ATM switches can however be successfully modelled with simulations at the cell level using the discrete time concept in which the order of processing is the reverse of the flow of cells through the switch (3.5.3.2).

It is possible to design and validate a versatile simulator with many architectural variations and other options but studying all possibilities is extremely time consuming and therefore for a small project limiting the options is a sensible approach (general observation).

In contrast with the previous paragraph, it is much more difficult to incorporate modifications to an existing ATM simulator after the design and validation cycle are complete and therefore one must be certain all options have been included at the design stage (general observation).

For a two–stage ATM switch architecture having a moderate (less than 50) number of queue places per stage a warm–up time of 25000 cell times is appropriate with the

parameter *mean cell delay of all cells* having the longest settling time of all results produced (3.5.3.1).

In ATM switch elements 2 parameters are required to specify queue lengths, the queue length before queue servicing and the queue length after queue servicing. Attempts to find one of these parameters as a function of the other and the input load have shown that the relationship is not straight forward (7.2.7). However it is possible to derive an analytic solution to determine the 2 parameters from the input load using Markov chain techniques (3.5.3.2).

The GMDP video source model requires an assumption to be made regarding the timing of the state transition trials. It was found that the model behaves in a balanced mode, i.e. where all states send approximately the same number of cells, if the state transition trials are performed following the generation of a full cell (4.1.1).

A compensation factor can be found to allow the continuous negative exponential distribution to operate successfully as a discrete distribution in the slotted environment of an ATM switch (4.3)

Two different traffic types, negative exponentially distributed cell arrivals and binomially distributed cell arrivals produce the same values for the results *mean delay of delayed cells* and *mean delay of all cells* in this simulator with input loads in the range 0.5 to 0.85. This allows the use of the more efficient binomially distributed cell arrivals (4.4).

The logarithm of the parameter cell–loss probability is linearly related to the number of queue places (5.1.1). In addition the simulation results have provided a model to predict cell–loss probability given the number of queue places and the input load (5.1.2).

A complex analytic model allowing cell–loss probability to be found as a function of the number of queue places and input load has also been developed (5.2.2). Using this model it was found that for a constant load, increasing the number of inputs into a switching stage increases the number of queue places required to maintain the same cell–loss probability (5.2.3). It is concluded that spreading the load over an increasing number of incoming links is detrimental to performance. Additionally the number of queue places required to maintain the same cell–loss probability does not increase linearly with load and to maintain reasonable queue sizes the input load should be restricted to a value of less than 90% (5.2.3).

The highest *cell throughput : number of queue places* ratio was found to be given with two input links (5.2.3). The most efficient interconnecting link utilisation at 80% output load was found to be 40% also given with two input links (5.2.3). These two facts point to the conclusion that the most efficient architectures will minimise the number of incoming links into a switching stage.

Comparisons between cell–loss probability figures produced by the simulator and those produced by analytic means show good correlation giving credibility to other simulator results that are not possible to obtain analytically (5.2.5).

With a hypothetical network reference connection the retiming delay across a synchronous network was estimated to be 750 μs (6.4). Using the same reference connection the ATM retiming delay at 0.8 load was estimated to be 1567 μs, or just over double that in a synchronous network (6.5). A retiming buffer able to cope with this amount of cell delay variation is required to be 576 cells deep. If 1 cell in $10^4$ can be discarded due to excessive delay the ATM retiming delay comes down to 139 μs requiring a retiming buffer depth of only 47 cells (6.6.2). It is acknowledged that some

assumptions have to be made to simplify the problem and allow the use of techniques such as convolution.

When any queue in the network is increased in size by just one queue place, all CBR retiming buffers need to be increased by one as well. (Assuming no cell discard due to excessive delay.) This implies that the number of queue places in the system as a whole is critical and must be kept to an absolute minimum (general observation).

Even in an ideal ATM network it is not possible to carry 2 Mbit/s CBR services in a circuit emulation mode without additional retiming at the exit of the network. As such source timing is lost (6.7).

## 7.2 Suggestions for Further Research

### 7.2.1 Variable Traffic Profiles

The simulator has only been run with stationary traffic in which the load remains constant throughout a simulation run. The performance of various architectural configurations under variable traffic such as ramp, step and burst traffic profiles as shown in figure 7.1 could be studied.

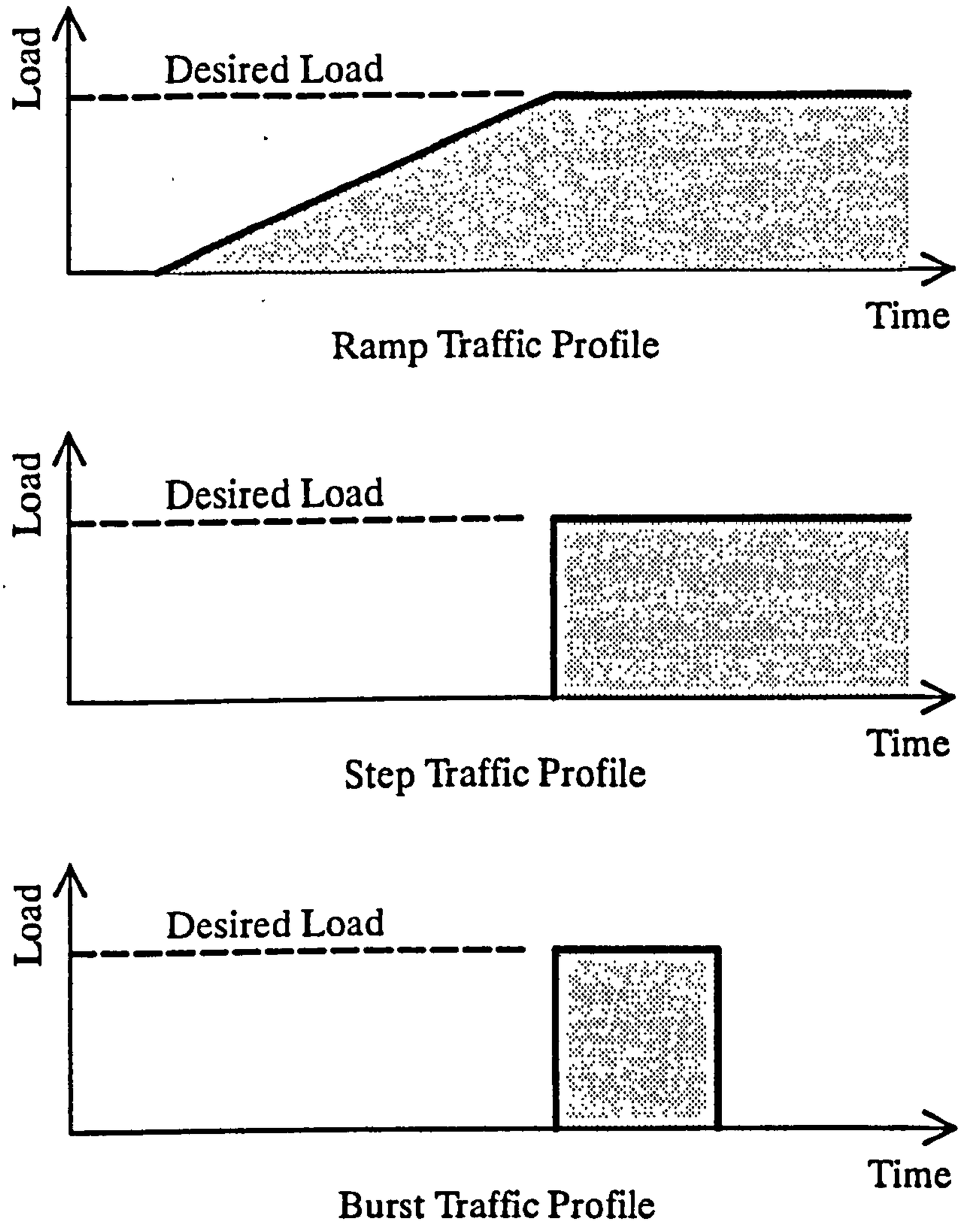



Figure 7.1: Variable Traffic Profiles

The addition of this feature would allow dynamic studies such as the testing of various "feel–good" factors as discussed in section 7.2.3 below.

### 7.2.2 Traffic Segregation into Multiple Streams

Improvements in performance may be anticipated by dividing traffic into a number of different classes, or streams, depending on their service requirements. To the author's knowledge figures of 4 to 7 streams are in discussion to date (November

1992). A possible scheme for 5 traffic segregation streams in informal discussion within CCITT Study Group XVIII breaks down the ATM traffic with priorities for delay as follows:

1. CBR Data – Low rate (up to 64 kbit/s)
2. CBR Data – High rate
3. Signalling messages
4. "Important" VBR Data requiring a superior grade of service to...
5. "Ordinary" VBR Data.

The number of streams is restricted by a number of factors, for example the complexity of the switch block and switch block control and the ability to distinguish the different streams from the limited available data in the cell header. (The current proposal is to allocate unique code–points, to the different streams. The code–points are encoded in the 3–bit Payload Type field thus the number of different possibilities is small.) Further research through simulation could refine the traffic segregation ideas including specifying the number of streams, defining the boundaries of each stream and investigating the very important but hitherto largely neglected area of the charging policy for the different streams.

### 7.2.3 A Method of Monitoring System Load

It was shown in chapter 6 that the magnitude of the jitter imparted to a cell is dependant on the system load. It is possible that this principle could be used as a simple method of monitoring the system load. The control system could periodically send a time–stamped cell into the switch block and observe the magnitude of the jitter suffered. This method would probably only be suitable for ATM pipes with relatively constant traffic levels, possibly those between major switching nodes where some form of source policing had been applied. More generally, a metric extracted from an

ATM switch on a dynamic basis that would indicate the onset of congestion and overload would be extremely useful to the network operators. Many possibilities exist including monitoring queue lengths over a fixed size window.

Having identified such a metric to detect the onset of congestion, network mechanisms would be required to attempt to reduce or at least contain the overload.

### 7.2.4 Queue Sharing

Queue sharing was discussed in section 3.5.2.3. An ideal queue configuration may be to have *partial* queue sharing in which a number of inputs share the same queue space but in addition have a dedicated area to themselves. This achieves the 2 goals of efficient use of storage through sharing and containment of overload on one input such that the performance of the remaining inputs is not impaired. The results of investigations into the optimum queue sizes may be beneficial to switch designers.

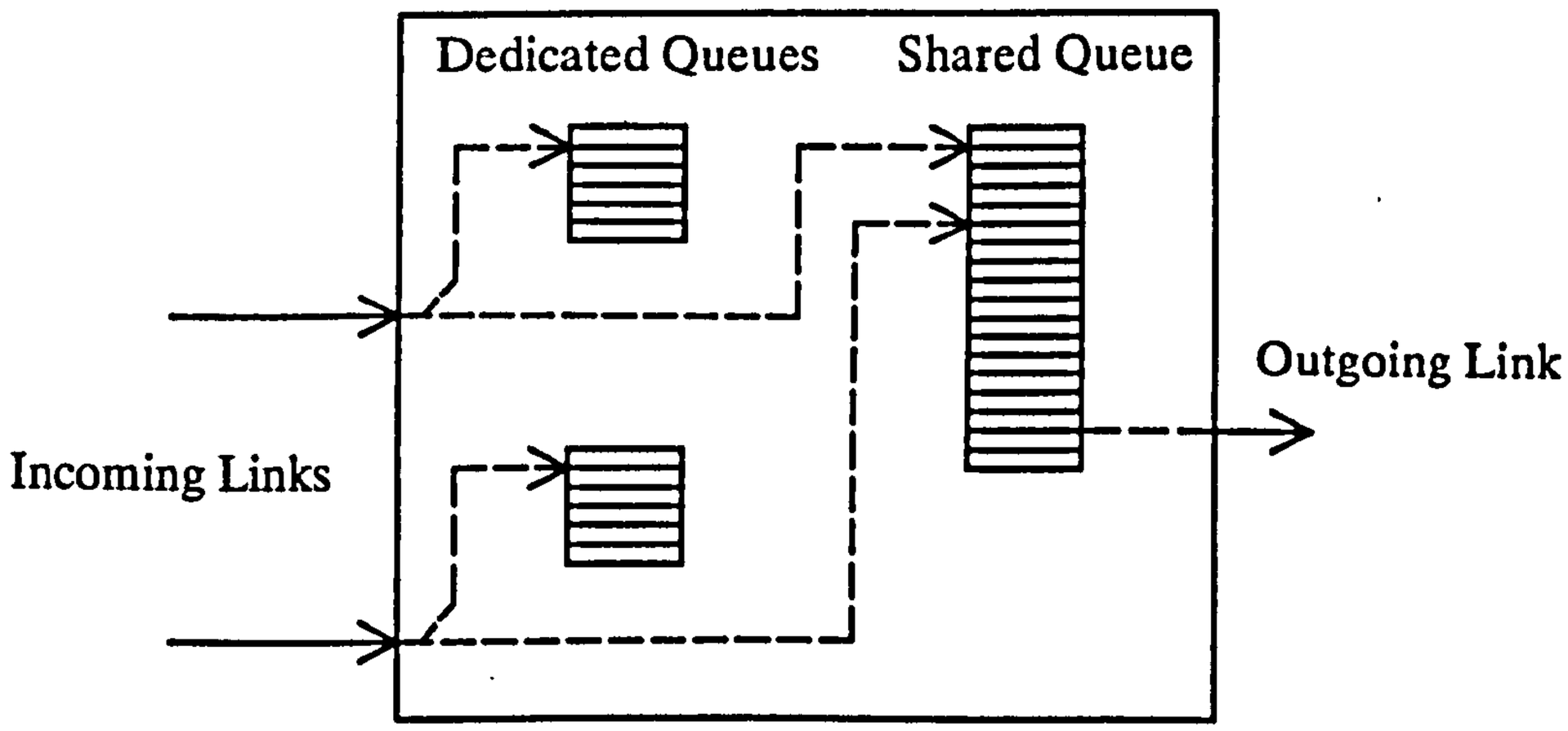



Figure 7.2: Queue Sharing

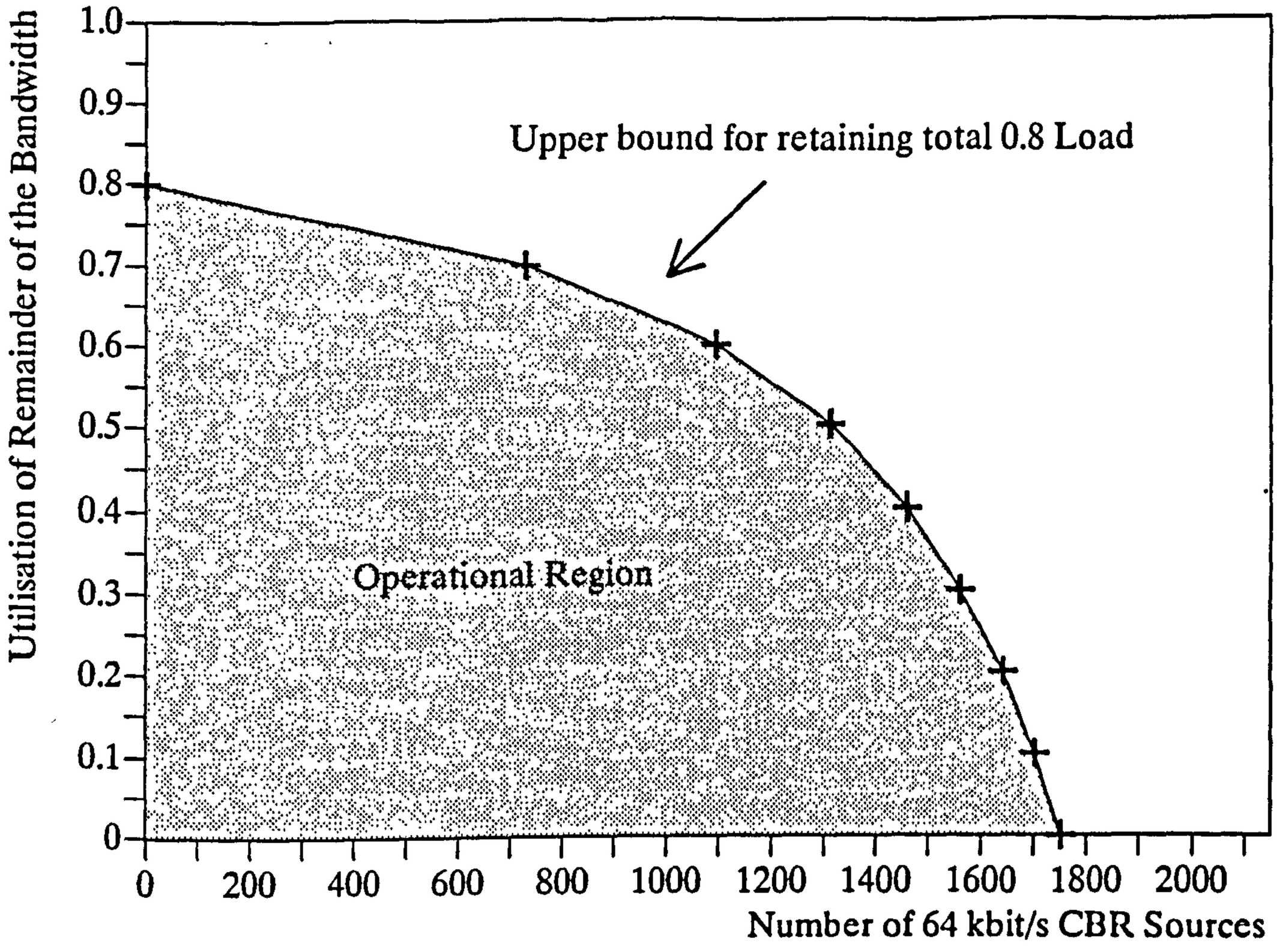
It would be particularly helpful to understand whether this concept gives any performance improvements as anticipated, if so how big each of the queues should be and what queue disciplines to use.

### 7.2.5 The Affects of Constant Bit Rate Sources on Performance

The simulator traffic generators are introduced in section 3.5.2.5. In addition to the variable bit rate source models the simulator also has constant bit rate sources representing 64 kbit/s speech channels. Whether this is the correct method of carrying speech, which by its bursty nature does in fact produce a variable bit rate stream, is still an open question. In [84] Brady presents the results of a study of voice source activity. If speech is considered in terms of active and inactive periods, he shows that the mean active time is 1.2 seconds and the mean inactive time is 1.8 seconds. Thus one–way speech is only active for 40% of the time. Considering a two–way conversation with only one party active at once, the utilisation of the bandwidth reduces to 20%. It is accepted that this is an over–simplification of the problem, however it does demonstrate that significant gains can be made if active speech is packetised and silence is not. (Daigle and Langford also present an interesting study of models for packet voice systems in [85].)

Given that it is necessary to interwork with the present voice network, this subsection presents the results of some preliminary studies into methods of carrying 64 kbit/s speech across an ATM switch.

The simulator was designed to generate CBR traffic representing speech channels. This was done by conceptually *framing* the 155 Mbit/s ATM pipe into 5988 μs blocks. This results in 2189 cells per frame (to the nearest integer). Incoming CBR cells are marked as "delay priority" and are entered into a limited–length priority buffer. At each switching stage a cell in the priority buffer (if any) is moved forwards in preference to all other cells. It therefore suffers no cell delay variation providing other CBR circuits are not being set up or cleared down. The remaining ordinary VBR cells clearly suffer worse service as a result.




Graph 7.1: Upper Bound for total 0.8 load with varying number of CBR Sources

Simulation runs have shown delay priority to be a viable technique for a single stream. If the VBR traffic load is kept on the upper bound of graph 7.1 the quality of service provided is no worse than if the total load were 0.8 made up solely of VBR cells. If a large proportion of the 0.8 load is made up of VBR traffic, a VBR cell is delayed mainly by encountering busy VBR queues. On the other hand if most of the 0.8 load is CBR traffic, a VBR cell encounters quieter VBR queues hence rising to the head of the queue quicker, but there it has to wait while any CBR cells are unloaded from the priority queue. Further research could quantify the performance effect in the rest of the operational region shown on graph 7.1. Further research into the possibilities of more than two streams is discussed in section 7.2.2.

### 7.2.6 Multiplexing Multiple Video Source Models

The GMDP video source model is described in section 4.1 of this thesis. This sub–section presents initial results of studies into the effects of multiplexing multiple video sources onto one ATM pipe. This is included to highlight a significant problem of the ATM technique, that is that a single source suffers differential delay in the early multiplexing stages before any ATM switching has taken place. This means that in certain circumstances the differential delay figures, and consequently the depth of the re–timing buffers, could be worse than those presented in chapter 6 which were based only on differential delay imparted as a result of the switching stages.

The video source model program, written in C and capable of being introduced into the simulator program to precede the switching stages and generate cells with the approximate characteristics of video sources, formed the basis of a larger, stand–alone program simulating N video sources. The block diagram of the arrangement is shown in figure 7.3.

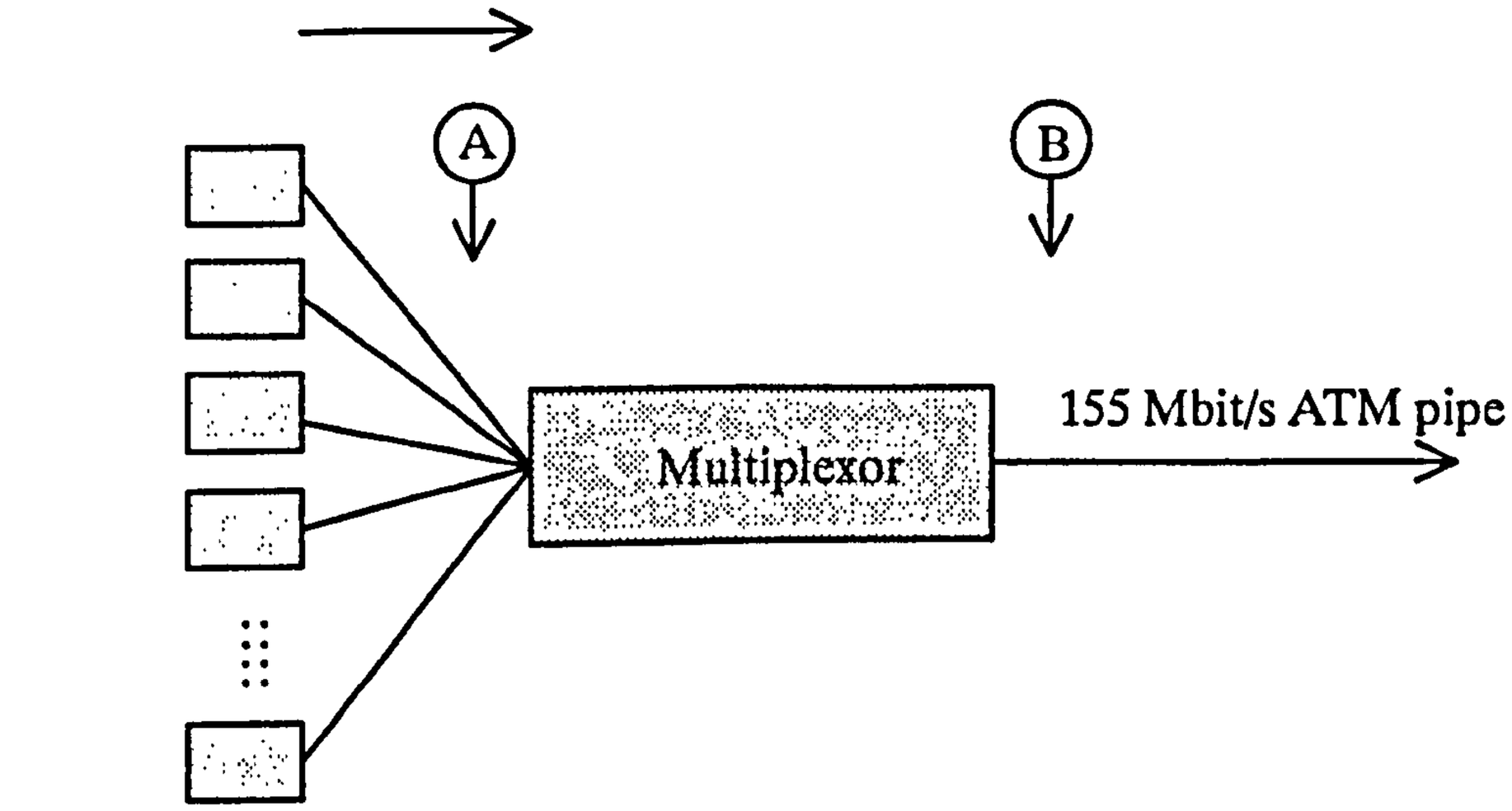
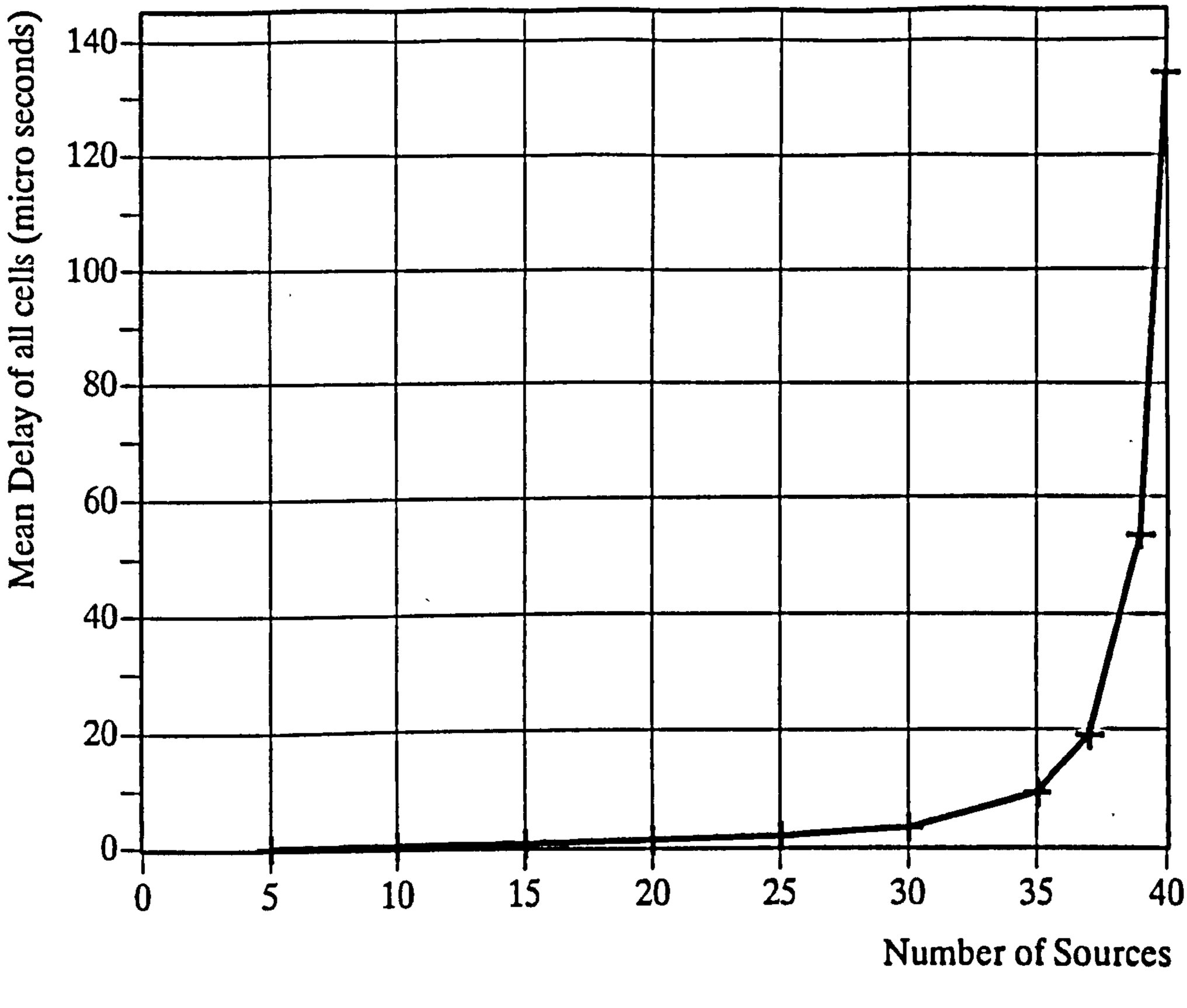



Figure 7.3: Arrangement for Multiple Video Source Multiplexing Studies

Each video source generates cells independently, full cells compete at the multiplexor for slots on the outgoing ATM pipe. This scenario might represent a cable TV provider with the ATM network providing the point to multi−point switching and distribution functions. From the results of the studies of an individual video source presented in chapter 4 it was found that the GMDP source generates approximately 9016 cells per second or approximately 2.5% of the bandwidth of a 155 Mbit/s ATM pipe. In the model therefore, the number of sources was variable between 1 and 40.

The model was set to simulate 2 seconds of real time (731132 cell times) with from 5 to 35 sources in steps of 5 plus 37, 39 and 40 sources. Full cells from the sources were time stamped as they were generated (point A in figure 7.3). This time stamp was compared at point B with the system time and the delay noted. The variation of the parameter *mean delay of all cells* (represented here in micro−seconds) with the number of sources is shown on graph 7.2.




Graph 7.2: Variation of Mean Delay of all Cells with the number of Video Sources

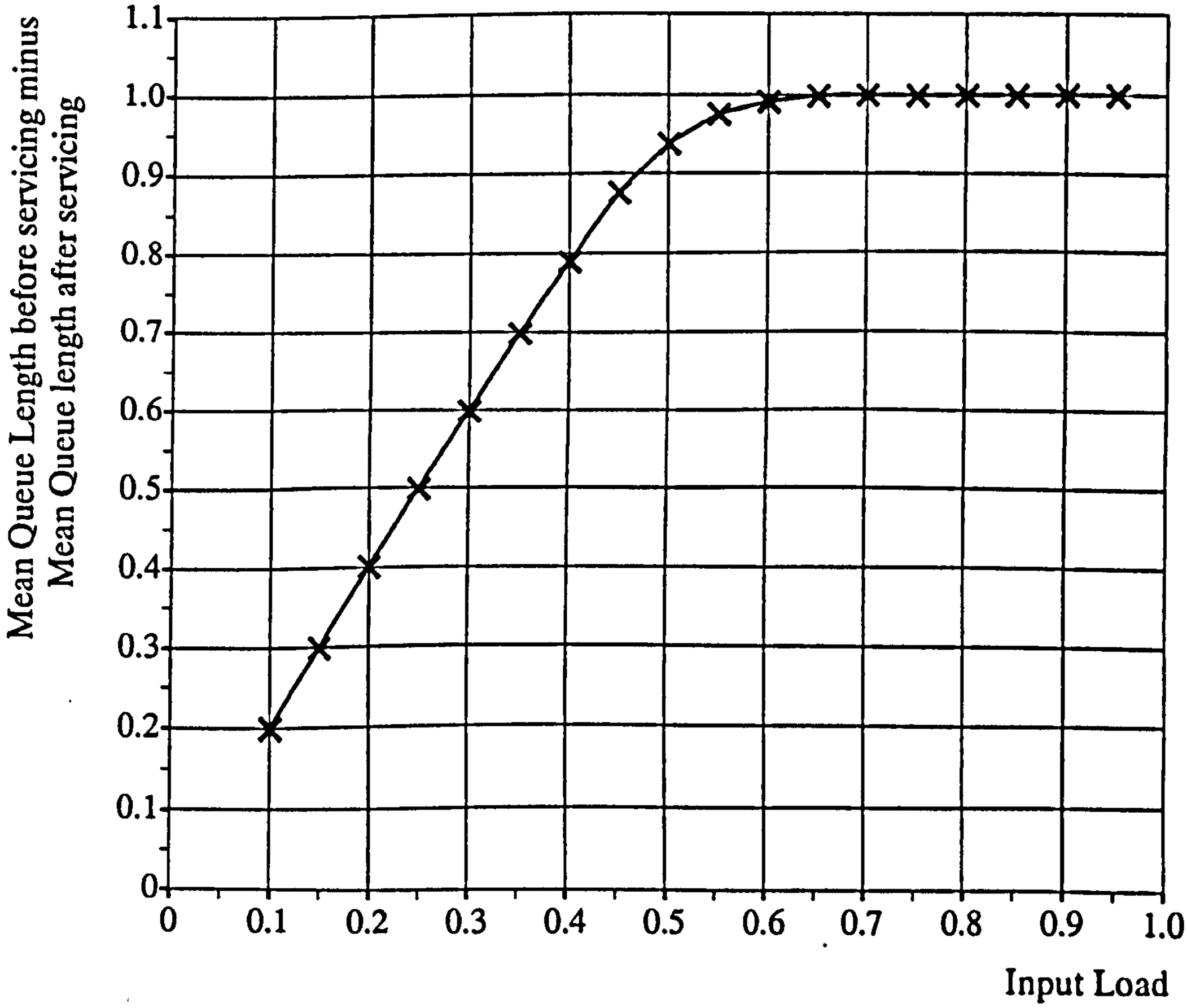
Graph 7.2 shows that the perceived grade of service, in terms of delay, is relatively constant up to 30 sources (equivalent to 75% utilisation of the bandwidth). However this deteriorates rapidly above 35 sources and adding a single source from 39 to 40 sources more than doubles the delay *for all sources – not just the additional one.*

The algorithm used in the multiplexor was simply to rotate around each source in strict numerical order to see if the source had a cell to send (top to bottom in figure 7.3) each time starting at source number zero. It was found, rather surprisingly, that with a large number of sources the best service was not given to the lowest numbered source but to sources numbered in the approximate range 4 to 6. Further research in this area could attempt to quantify the amount of degradation of service suffered at both

switching and multiplexing stages as a function of input load to produce a single metric applicable to both types of equipment.

### 7.2.7 Relationship Between Queue Lengths Specification

Section 3.5.3.2 presented an analytic solution to the problem of specification of queue lengths in ATM switching elements. It was found that two parameters are required to fully specify the queue occupancy, the occupancy before queue servicing and the occupancy after queue servicing. The two parameters are plotted on graph 3.2. The difference between the two results is plotted on graph 7.3 below.




Graph 7.3: Queue Lengths, Before Servicing – After Servicing

The relationship between the two parameters is not obvious from the graph and further research could develop and generalise this analytic theory to include a derivation of this relationship.

## 7.3 Closing Remarks

The telecommunications industry finds itself at a cross–roads. On one hand the industry needs to provide the network operators with exciting new features, improved performance and revenue–earning services to give them the justification to invest large amounts of money in new equipment. On the other hand, as systems become more and more complex so the costs of developing them grow and grow. More skilled engineers are required with better, more expensive, development tools. High–volume sales are required to recover research and development costs. This has lead to the numerous mergers, take–overs and collaborative projects seen over the last decade as major manufacturers join forces to try and defeat the opposition.

The development of ATM will undoubtedly be an enormously expensive undertaking, probably requiring the full commitment of the company's resources. If ATM is taken up in general, a company following this path would be well placed in the market. If however network operators are nervous about ATM and stay with the 'conventional' techniques, the ATM company could quickly be out of business.

Since the early RACE 1 projects of the late 1980's, much research has been done on ATM. Much has been published by the pro–ATM lobby and little by the sceptics. A warning shot was fired by Mr Oliphant of the BBC in October 1990 who in the third paper of the IEE's Integrated Broadband Services and Networks conference [86] stated that ATM would give 'extra difficulties' for operators providing constant bit–rate oriented services, which may well lead them to set up private networks rather than use the B–ISDN.

Overall, ATM suffers real problems when trying to carry time sensitive services and one must question whether, without additional timing techniques, it can ever satisfactorily interwork with, never mind replace, the existing synchronous voice network.

# Appendix A: Simulator Control Language (SCL) Commands

N *<integer>*

This command sets the run length parameter e.g. "N 20000". The maximum value allowable is 999999. The default value is 25000 (minimum run length, see section 3.5.3.1).

Q*<stage number>* *<integer>*

This command changes the number of queue places in the specified stage e.g. "Q0 25" allocates 25 queue places in all first stage switching elements. The default values for a 2–stage switch are Q0 7 and Q1 7.

L*<input bearer number>* *<integer>*

This command changes the load on the specified input bearer e.g. "L0 75" sets the traffic load on incoming bearer 0 to 0.75, or 75%. The load is expressed in terms of percentage. The default values for a 2–stage switch are L0 70 and L1 70.

P*<input bearer number>* *<integer>*

This command changes the proportion of the traffic on the specified input bearer to output bearer 0 e.g. "P1 50". The default values for a 2–stage switch are P0 50 and P1 50.

O *<YES/NO>*

This command sets an operating mode of the final stage queues which can either be overwritten when full (e.g. O YES, represents "Overwrite Yes") or the overload can be transferred back to the queues in the previous stage (e.g. O NO). The default is O NO.

S *<YES/NO>*

This command sets an operating mode of the final stage queues to either operate as a single queue (e.g. S YES) or as one queue for each incoming bearer (e.g. S NO). The default is S NO.

A *<M/B>*

This command sets the arrival distribution of the incoming traffic e.g. A M sets the arrival distribution to Markovian. Allowed distributions are M, Markovian and B, Binomial. The default distribution is binomial (A B).

J *<YES/NO/START>* *<integer>*

This command sets the jitter analysis parameters. It is possible to disable jitter analysis when not required to suppress the large amount of output it can produce. The J START command specifies the run counter value at which the collection of statistics should commence. This is to remove the effect of starting up the simulation with empty queues. The default values are equivalent to the J NO and J START 20000 commands.

C *<integer>*

This command sets the number of 64 kbit/s synchronous sources per incoming multiplex. The default number of sources is zero.

GO

This command starts the simulation run.

*

This command terminates the run by signalling to the pre–processor that the end of the command file has been reached. A non–fatal warning message is output if the end of the input file is reached without reading the terminating character.

# Appendix B: Example ATMoSS Results Output

## 1) Results including "Spotlight Function"

*(Note: Only the full results of Run #10, the last of the 10 runs, is shown)*

| | | | |
|---|---|---|---|
| Length of run | = 100000 | Av. packet delay (all) | = 7.117 |
| Arrival distribution | BINOMIAL | Av. packet delay (delayed) | = 7.874 |
| No. delayed through 10 | = 82810 | % delayed through 10 | = 87% |
| No. delayed through 11 | = 82701 | % delayed through 11 | = 87% |
| Stage 1 queues overwritten | = NO | Single queue in stage 1 | = YES |
| No of places stage 0 queue | = 20 | No of places stage 1 queue | = 20 |
| Average Qlength 00 (B) | = 0.000 | Average Qlength 10 (0) (B) | = 5.534 |
| | | Average Qlength 10 (1) (B) | = 0.000 |
| Average Qlength 01 (B) | = 0.000 | Average Qlength 11 (0) (B) | = 5.205 |
| | | Average Qlength 11 (1) (B) | = 0.000 |
| Average Qlength 00 (A) | = 0.000 | Average Qlength 10 (0) (A) | = 4.583 |
| | | Average Qlength 10 (1) (A) | = 0.000 |
| Average Qlength 01 (A) | = 0.000 | Average Qlength 11 (0) (A) | = 4.256 |
| | | Average Qlength 11 (1) (A) | = 0.000 |
| Packet count into 00 | = 95017 | Packet count out of 10 | = 95050 |
| Packet count into 01 | = 94986 | Packet count out of 11 | = 94940 |
| Total packets in | = 190003 | Total packets out | = 189990 |
| % Actual loading of 00 | = 95.0% | Proportion I/P 0 to O/P 0 | = 50% |
| % Actual loading of 01 | = 95.0% | Proportion I/P 1 to O/P 0 | = 50% |
| No of lost packets in 00 | = 1 | No of lost packets in 10 | = 0 |
| No of lost packets in 01 | = 2 | No of lost packets in 11 | = 0 |
| Total no. of lost packets | = 3 | Unaccounted for packets | = 13 |
| % of packets lost in 00 | = 0.001% | % of packets lost in 10 | = 0.000% |
| % of packets lost in 01 | = 0.002% | % of packets lost in 11 | = 0.000% |
| Packet loss in 00 | = 0.000011 | Packet loss in 10 | =0.000000 |
| Packet loss in 01 | = 0.000021 | Packet loss in 11 | =0.000000 |
| Average cell−loss | = 0.0 | | |

Number of 64 kbit/s CBR sources = 0

**********

Spotlight function:

Result 1: Average packet delay – all

Result 2: Average packet delay – delayed

Result 3: Actual loading 00 (percent)

Result 4: Actual loading 01 (percent)

| | Result 1 | Result 2 | Result 3 | Result 4 |
|---|---|---|---|---|
| Run #: 1 | 7.010631 | 7.769845 | 94.941002 | 94.997002 |
| Run #: 2 | 6.595552 | 7.333641 | 94.871002 | 94.992996 |
| Run #: 3 | 6.993146 | 7.769717 | 95.052002 | 94.942001 |
| Run #: 4 | 6.833552 | 7.582024 | 94.995003 | 94.973000 |
| Run #: 5 | 6.956907 | 7.722158 | 95.038002 | 95.010002 |
| Run #: 6 | 7.492694 | 8.262111 | 95.137001 | 95.010002 |
| Run #: 7 | 7.366226 | 8.134670 | 94.982002 | 94.970001 |
| Run #: 8 | 7.773806 | 8.582283 | 95.031998 | 94.987000 |
| Run #: 9 | 7.279572 | 8.039699 | 94.989998 | 95.084999 |
| Run #:10 | 7.117001 | 7.873803 | 95.016998 | 94.986000 |
| Mean : | 7.141910 | 7.906995 | 95.005493 | 94.995308 |
| S.D. : | 0.343093 | 0.358830 | 0.070315 | 0.037446 |

**********

End of input file reached, run finished

## 2) Results including Jitter Analysis

| | | | |
|---|---|---|---|
| Length of run | = 100000 | Av. packet delay (all) | = 6.783 |
| Arrival distribution | BINOMIAL | Av. packet delay (delayed) | = 7.683 |
| No. delayed through 10 | = 79723 | % delayed through 10 | = 84% |
| No. delayed through 11 | = 79788 | % delayed through 11 | = 84% |
| Stage 1 queues overwritten | = NO | Single queue in stage 1 | = NO |
| No of places stage 0 queue | = 7 | No of places stage 1 queue | = 7 |
| Average Qlength 00 (B) | = 0.000 | Average Qlength 10 (0) (B) | = 2.286 |
| | | Average Qlength 10 (1) (B) | = 2.307 |
| Average Qlength 01 (B) | = 0.000 | Average Qlength 11 (0) (B) | = 2.257 |
| | | Average Qlength 11 (1) (B) | = 2.283 |
| Average Qlength 00 (A) | = 0.000 | Average Qlength 10 (0) (A) | = 1.811 |
| | | Average Qlength 10 (1) (A) | = 1.834 |
| Average Qlength 01 (A) | = 0.000 | Average Qlength 11 (0) (A) | = 1.784 |
| | | Average Qlength 11 (1) (A) | = 1.809 |
| Packet count into 00 | = 95025 | Packet count out of 10 | = 94777 |
| Packet count into 01 | = 95042 | Packet count out of 11 | = 94727 |
| Total packets in | = 190067 | Total packets out | = 189504 |
| % Actual loading of 00 | = 95.0% | Proportion I/P 0 to O/P 0 | = 50% |
| % Actual loading of 01 | = 95.0% | Proportion I/P 1 to O/P 0 | = 50% |
| No of lost packets in 00 | = 259 | No of lost packets in 10 | = 0 |
| No of lost packets in 01 | = 284 | No of lost packets in 11 | = 0 |
| Total no. of lost packets | = 543 | Unaccounted for packets | = 563 |
| % of packets lost in 00 | = 0.273% | % of packets lost in 10 | = 0.000% |
| % of packets lost in 01 | = 0.299% | % of packets lost in 11 | = 0.000% |
| Packet loss in 00 | = 0.002726 | Packet loss in 10 | =0.000000 |
| Packet loss in 01 | = 0.002988 | Packet loss in 11 | =0.000000 |
| Average cell–loss | = 0.0 | | |

----------JITTER REPORT----------

Jitter analysis commenced at clock time: 75000

Mean of the jitter distribution = 4.33008

JITTER VALUE (Method 1): 0.96331

JITTER VALUE (Method 2): 4.17119

| Jitter–map Element 0, Delay | Frequency of Delay | Prop. of total |
|---|---|---|
| 0 | 8307 | 0.17540 |
| 1 | 6946 | 0.14666 |
| 2 | 5524 | 0.11664 |
| 3 | 4424 | 0.09341 |
| 4 | 3762 | 0.07943 |
| 5 | 3285 | 0.06936 |
| 6 | 3017 | 0.06370 |
| 7 | 2485 | 0.05247 |
| 8 | 2044 | 0.04316 |
| 9 | 1612 | 0.03404 |
| 10 | 1318 | 0.02783 |
| 11 | 1127 | 0.02380 |
| 12 | 904 | 0.01909 |
| 13 | 683 | 0.01442 |
| 14 | 533 | 0.01125 |
| 15 | 396 | 0.00836 |
| 16 | 312 | 0.00659 |
| 17 | 259 | 0.00547 |
| 18 | 208 | 0.00439 |
| 19 | 169 | 0.00357 |
| 20 | 42 | 0.00089 |
| 21 | 3 | 0.00006 |
| 22 | 1 | 0.00002 |

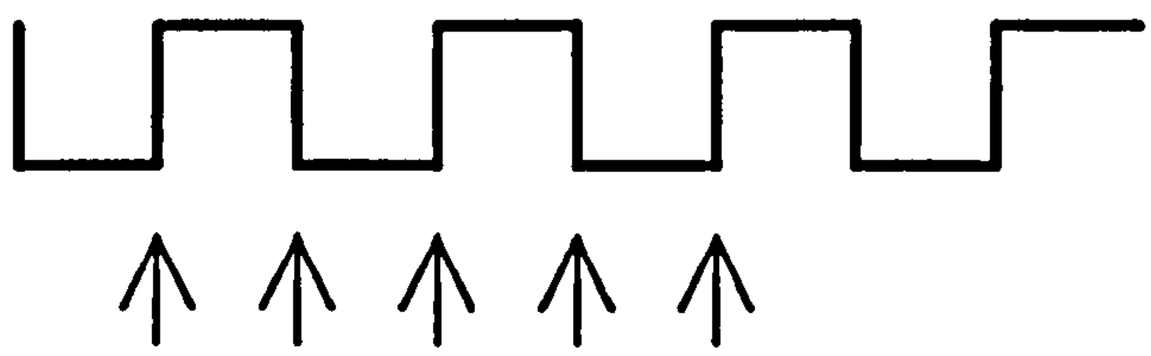
Number of 64 kbit/s CBR sources = 0

**********

End of input file reached, run finished

# Appendix C: Definition of Synchronisation Related Terms

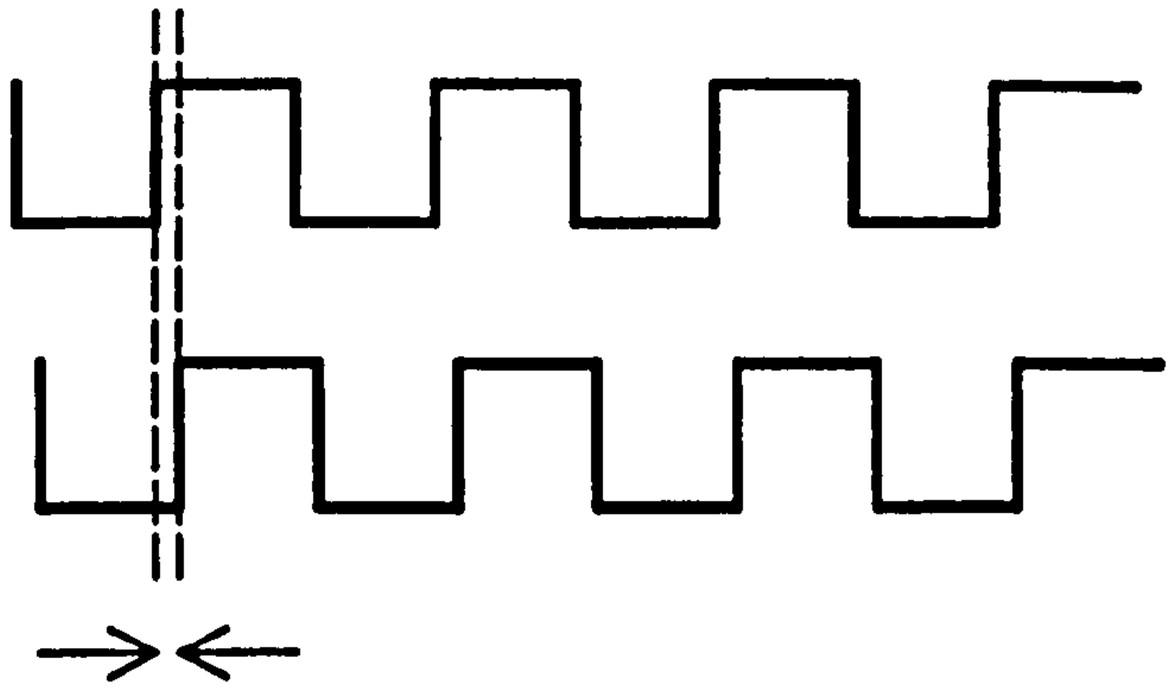
Isochronous: A single signal is isochronous if the time intervals between consecutive significant instants either have the same duration or are an integral multiple of the shortest duration. In practise variations in the time interval have to be constrained within specified limits.



Significant Instants

Figure C.1: Isochronous Signal

Synchronous: Two signals are said to be synchronous if their corresponding significant instants occur at precisely the same average rate. The timing relationship between corresponding significant instants usually varies between specified limits.



Phase Error

Figure C.2: Synchronous Signals

This variation is referred to as phase error or phase distortion and can be caused in a number of ways.

Firstly the 2 signals could have taken a different length of path to the comparison point. As a signal is carried at approximately 2/3rds the speed of light in copper it takes $5 \times 10^{-9}$ seconds to travel one metre. As the bit time (duration of one bit) at 155.520 Mbit/s is $6.43 \times 10^{-9}$ seconds, a difference in path length of only one metre between two systems carrying 155 Mbit/s signals results in a phase error of nearly one bit.

Secondly, noise imparts jitter and wander onto the signals (jitter here is the conventional transmission variety and not ATM cell delay variation).

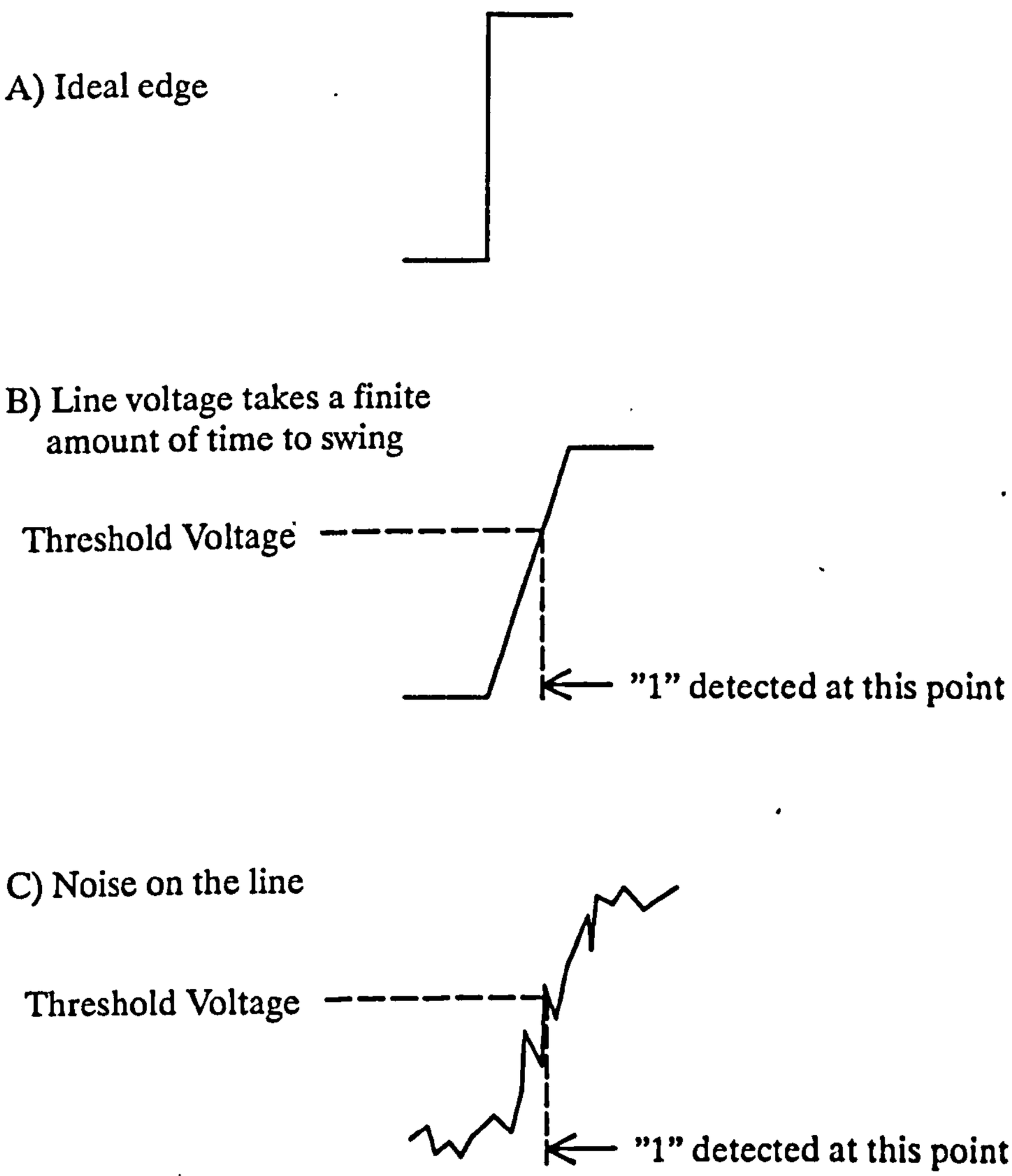



Figure C.3: The Effect of Noise

Figure C.3 shows this effect. As the line voltage takes a finite amount of time to swing there exists a threshold voltage which once exceeded causes the receiver to register the state–change to "1". Noise on the line can cause the receiver to register the state change either prematurely or late thus causing phase error.

Plesiochronous: Two signals are said to be plesiochronous if their corresponding significant instants occur at nominally the same rate. Any variation in rate is contained within specified limits. The timing relationship between corresponding significant instants is not constrained. Signals stemming from 2 different oscillators running at nominally the same rate are usually plesiochronous.

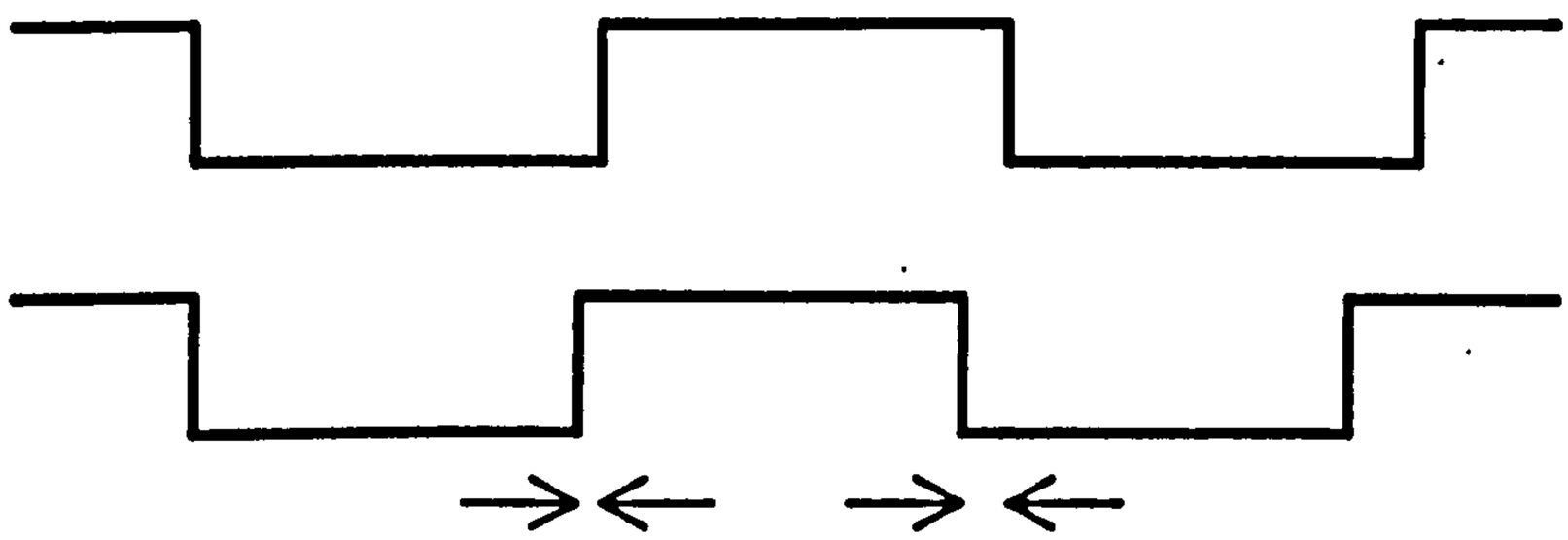



Figure C.4: Plesiochronous Signals

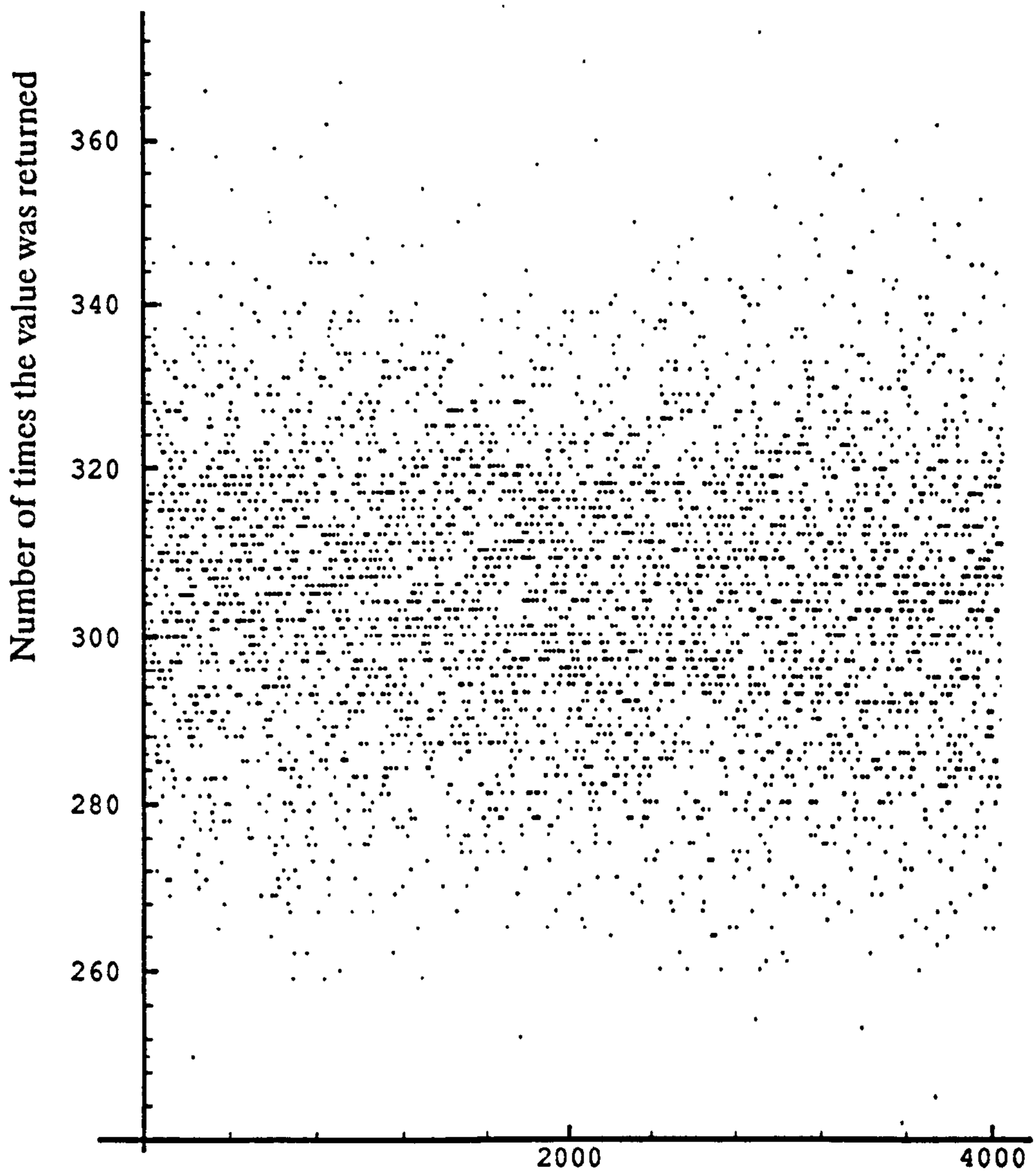
Non–Synchronous: Two signals are said to be non–synchronous if their corresponding significant instants do not necessarily occur at the same average rate. (Also called asynchronous.)

# Appendix D: Pseudo−Random Number Generator Details

The pseudo−random number generator used in the simulator is a multiplicative congruential generator with a period of $2^{32}$ that returns pseudo−random integers in the range 0 to $(2^{15} - 1)$. When used in the negative exponential source model the returned integer is divided by a constant to produce a number in the range 0 to 1.

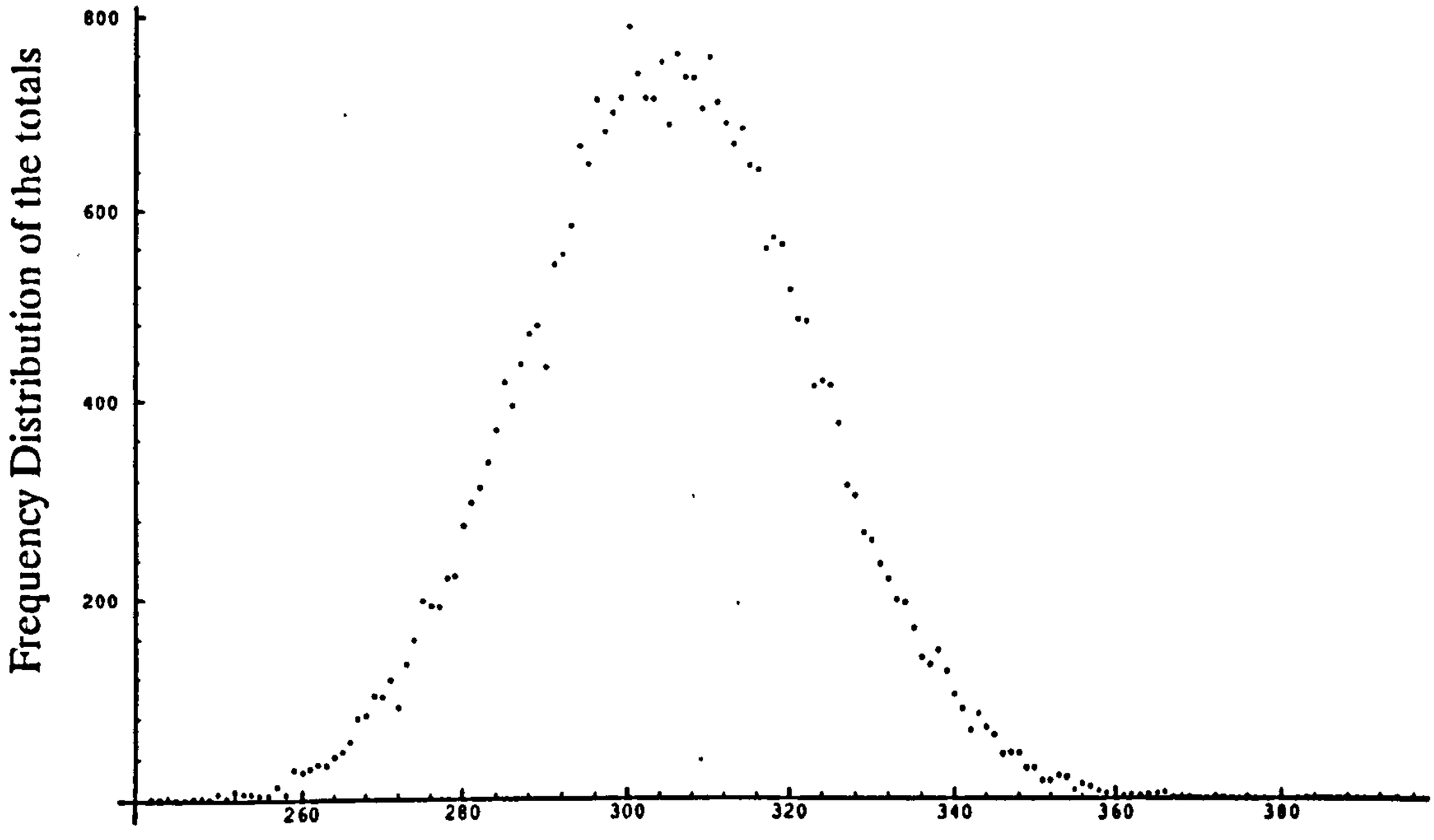
The spectral properties of the generator were tested by calling it $10^7$ times and recording how many times each integer was returned. With 32768 values called $10^7$ times it would be expected that most numbers would be returned approximately 300 times. If the generator continually fails to draw certain integers or favours others this would show up in the final totals for each number.

The results of the test show that the integer drawn the least was drawn 242 times and the integer drawn the most was drawn 394 times. A graph with all 32768 possible values shown on the x−axis and the number of times the value was returned shown on the y−axis was produced. A section of the graph for x values in the range 0 to 4000 is shown on graph D.1. Visual inspection of this graph and the others making up the rest of the range show no obvious holes or clustering.




Graph D.1: Random Number Generator Results – I

The frequency distribution of the number of times each integer was drawn is shown on graph D.2. The graph shows an even distribution around the mean value of 305.




Graph D.2: Random Number Generator Results – II

The results of the test have revealed no obvious flaws in the pseudo–random number generator and it is concluded that it is acceptable for use in the simulator.

## Abbreviations

| | |
|---|---|
| ATD | Asynchronous Time–Division Multiplexing (synonymous with ATM) |
| ATM | Asynchronous Transfer Mode |
| ATMoSS | ATM Switch Simulator |
| B–ISDN | Broadband ISDN |
| CAC | Call Acceptance Control |
| CBR | Constant Bit Rate |
| CC | Cross–connect |
| CCITT | International Consultative Committee for Telecommunications and Telegraphy |
| cct | Circuit |
| CDV | Cell Delay Variation |
| CLP | Cell Loss Priority |
| CNET | Centre National d'Etudes des Telecommunications |
| CPU | Central Processor Unit |
| DEMOS | Discrete Event Modelling on SIMULA |
| DSS–B | Digital Switching Subsystem – Local Exchange |
| FIFO | First–in, first–out (queue discipline) |
| GFC | Generic Flow Control |
| GOS | Grade of Service |
| GPSS | General Purpose Systems Simulator |
| ISDN | Integrated Services Digital Network |
| HEC | Header Error Control |
| IBCN | Integrated Broadband Communications Network |
| KSW | Kruskal, Snir and Weiss formula |

| | |
|---|---|
| LE | Local Exchange |
| LH | Left−hand |
| MAN | Metropolitan Area Network |
| N−ISDN | Narrowband ISDN |
| NNI | Network Node Interface |
| PCM | Pulse Code Modulation |
| pdf | Probability Distribution Function |
| PDH | Plesiochronous Digital Hierarchy |
| PLT | Payload Type |
| pmf | Probability Mass Function |
| POH | Path Overhead (in the SDH frame) |
| PSN | Packet Switched Network |
| PSS | Packet Switched Service |
| QOS | Quality of Service |
| RACE | Research into Advanced Communications in Europe |
| res | Reserved |
| RH | Right−hand |
| Rx | Receiver |
| SCL | Simulator Command Language |
| SDH | Synchronous Digital Hierarchy |
| SIU | Slide−In−Unit (Circuit Board) |
| SRTS | Synchronous Residual Time Stamp |
| STM | Synchronous Transfer Mode |
| SUB | Subscriber |
| TDM | Time Division Multiplexing |

| | |
|---|---|
| TE | Trunk Exchange |
| Tx | Transmitter |
| TXE | Telephone Exchange Electronic |
| UI | Unit Interval (Duration of one bit on a transmission system) |
| UNI | User Network Interface |
| VCI | Virtual Circuit Identifier |
| VLSI | Very Large Scale Integration |
| VPI | Virtual Path Identifier |
| w.r.t | with respect to |

## References


1 Atherton W A: "Ten great ways to kill an idea". *IEE Review*, January 1989, pp.15–17.

2 Scudder F J, Reynolds J N:"Crossbar Dial Telephone Switching System", *Bell System Technical Journal*, January 1939.

3 Robertson J H: 1948, The Story of the Telephone, The Scientific Book Club, London.

4 Cattermole K W: 1969, Principles of Pulse Code Modulation, page 19, Iliffe, London.

5 Miner W M: "Multiplex Telephony", United States Patent 745,734 (1903).

6 Reeves A H: French Patent 852,183 (1938), British Patent 535,860 (1939), American Patent 2,272,070 (1942).

7 CCITT: Blue Book, Recommendations G.703, G.704, G.732 and G.823, ITU, Geneva, 1989.

8 Britain 1990, HMSO, London.

9 Fox B: Watch out for the Video Phones. *New Scientist*, 6th March 1993, No. 1863, p.27.

10 Kulzer J J, Montgomery WA: Statistical Switching Architectures for Future Services. In: Proceedings of the 11th International Switching Symposium (ISS '84) vol. 3. Held May 1984, Florence, Italy, pp. 32C2/1–32C2/7.

11 Turner J S: New Directions in Communications. In: Proceedings of the 1986 International Zurich Seminar on Digital Communications. Held Mar. 1986, Zurich, Switzerland, pp.25–32.

12 Gallagher ID: Multi–service Networks. *British Telecom Technology Journal*, Jan. 1986,4(1), pp.43–49.

13 Perucca G, Belforte P, Garetti E, Perardi F: Research on Advanced Switching Techniques for the Evolution to ISDN and Broadband ISDN. In: Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '86) vol. 1. Held Dec. 1986, Houston, Texas, USA, pp.0329–0333.

14 Decina M: Evolution Towards Wideband Communication Networks. In: Proceedings of International Switching Symposium (ISS '87) vol. 2. Held Mar. 1987, Phoenix, Arizona, USA, pp.0417–0422.

15 Littlewood M, Gallagher I D, Adams J L: Network Evolution Using Asynchronous Time–Division Techniques. *British Telecommunications Engineering*, Jul. 1987, 6, pp.95–104.

16 CCITT: Blue Book, Recommendation I.121, ITU, Geneva, 1989.

17 CCITT: Blue Book, Recommendation I.361, ITU, Geneva, 1989.

18 Cochrane P, Heatley D J T: Future Directions in Long–Haul Optical–Fibre Systems. *British Telecommunications Engineering*, Jan. 1991, 9, pp.268–280.

19 Wörner K, Colquhoun A: High–Speed Si–Bipolar and GaAs Technologies. *IEEE Journal on Selected Areas in Communications*, Jan. 1986, SAC–4(1), pp.24–31.

20 De Prycker M: 1991, Asynchronous Transfer Mode, Solution for Broadband ISDN, Ellis Horwood, London, p.146.

21 Coudreuse J–P, Servel M: Prelude – An Asynchronous Time–Division Switched Network. In: Proceedings of the IEEE International Conference on Communications '87: Communications – Sound to Light (ICC '87), 2, held June 1987, Seattle, USA, pp.0769–0773.

22 Gonet P, Coudreuse J–P, Servel M: Implementing Asynchronous Transfer Mode Concepts: Main Results of the Prelude Experiment. In: Proceedings of the IEEE/IEICE Global Telecommunications Conference (GLOBECOM '87), 3, held Nov. 1987, Tokyo, Japan, pp.1871–1875.

23 Thomas A, Coudreuse J–P, Servel M: Asynchronous Time–Division Techniques: An Experimental Packet Network Integrating Videocommunication. In: Proceedings of the 11th International Switching Symposium ('ISS 84), 3, held May 1984, Florence, Italy, pp.32C2/1–32C2/7.

24 Plehiers P, Fastrez M, Bauwens J, De Prycker M: Evolution Towards a Belgian Broadband Experiment. In: Proceedings of the International Switching Symposium (ISS '87), 2, held Mar. 1987, Phoenix, Arizona, USA, pp.0380–0387.

25 Bauwens J, De Prycker M: Broadband Experiment using Asynchronous Time Division Techniques. *Electrical Communication*, (GB), 1987, 61(1), pp.123–130.

26 Yeh Y S, Hluchyj M G, Acampora A S: The Knockout Switch: A Simple, Modular Architecture for High–Performance Packet Switching. *IEEE Journal on Selected Areas in Communications*, SAC–5(8), Oct. 1987, pp.1274–1283.

27 Fischer W, Fundneider O, Göldner E–H: A Scalable ATM Switching System Architecture. *IEEE Journal on Selected Areas in Communications*, 9(8), Oct. 1991, pp.1299–1307.

28 Clos C: A Study of Non–Blocking Switching Networks. *The Bell System Technical Journal,* Mar. 1953, pp.406–424.

29 Benes V E: Optimal Rearrangeable Multistage Connecting Networks. *The Bell System Technical Journal,* Jul. 1964, pp.1641–1656.

30 Duthie P J, Wale M J: 16 x 16 Single Chip Optical Switch Array in Lithium Niobate. *Electronic Letters,* 4th July 1991, 27(14), pp.1265–1266.

31 Benes V E: 1965, Mathematical Theory of Connecting Networks and Telephone Traffic, Academic Press, New York and London, p.31.

32 Goke L R, Lipovski G J: Banyan Networks for Partitioning Multiprocessor Systems. In: Proceedings of the 1st Annual International Symposium on Computer Architectures held Dec. 1973, pp.21–28.

33 Batcher K E: Sorting Networks and Their Application. In: Proceedings of Spring Joint Computing Conference, AFIPS, held 1968, pp.307–314.

34 Huang A, Knauer S: Starlite: A Wideband Digital Switch. In: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '84), held Atlanta, Georgia, USA, Dec. 1984, pp.121–125.

35 Tobagi F A, Kwok T, Chiussi F M: Architecture, Performance and Implementation of the Tandem Banyan Fast Packet Switch. *IEEE Journal on Selected Areas in Communications*, 9(8), Oct. 1991, pp.1173–1193.

36 Ahmadi H, Denzel W E: A Survey of Modern High–Performance Switching Techniques. *IEEE Journal on Selected Areas in Communications*, 7(7), Sept. 1989, pp.1091–1103.

37 Lizcano P J, Chas P L, Jimenez J: An Approach to a High Throughput Matricial Packet Switch. In: Proceedings of the IEEE International Conference on Communications '88 (ICC '88), 2, held June 1988, Philadelphia, USA, pp.0604–0611.

38 Niestegge G: The 'Leaky Bucket' Policing Method in the ATM (Asynchronous Transfer Mode) Network. *International Journal of Digital and Analogue Communication Systems*, **3**, (1990), pp.187–197.

39 Wallmeier: A Connection Acceptance Algorithm for ATM Networks based on Mean and Peak Bit Rates. *International Journal of Digital and Analogue Communication Systems*, **3**, (1990), pp.143–153.

40 Verbiest W: Video Coding in an ATD Environment. In: Proceedings of the Third International Conference on New Systems and Services in Telecommunications, held Liege, Belgium, Nov. 1986, pp.249–253.

41 Maglaris B, Anastassiou D, Sen P, Karlsson G, Robbins J: Performance Analysis of Statistical Multiplexing for Packet Video Sources. In: Proceedings of the IEEE/IEICE Global Telecommunications Conference (GLOBECOM '87), held Tokyo, Japan, Nov. 1987, **3**, pp.1890–1899.

42 Cosmas J P, Odinma–Okafor A: Variable Rate Video Signals Characterisation Using a GMDP Model. In: Digest of the Seventh U.K. Teletraffic Symposium held April 1990, Durham, England, pp.5/1–5/8.

43 Kendall D G: Some Problems in the Theory of Queues. *J. R. Statist. Soc. B*, **13**, 1951, pp151–185.

44 Kleinrock L: 1975, Queueing Systems Volume 1: Theory, J. Wiley and Sons Inc., New York/London/Sydney/Toronto.

45 Kleinrock L: 1975, Queueing Systems Volume 2: Computer Applications, J. Wiley and Sons Inc., New York/London/Sydney/Toronto.

46 Cox D R, Smith W L: 1961, Queues, reprinted 1971, Chapman and Hall Ltd., London.

47 Mitrani I: 1982, Simulation Techniques for Discrete Event Systems, Cambridge University Press, England, p.3.

48 Neelamkavil F: 1987, Computer Simulation and Modelling, J. Wiley and Sons Inc., New York/London/Sydney/Toronto.

49 Metropolis N, Ulam S: The Monte Carlo method. *The Journal of the American Statistical Association*, 44(247), 1949, pp.335–341.

50 Gordon G:A General Purpose Systems Simulator. In: Proceedings of the Fall Joint Computer Conference, held Dec. 1961, Washington D.C, pp.87–104.

51 Markowitz H M, Hausner B, Karr H: 1963, SIMSCRIPT – A Simulation Programming Language, Prentice–Hall, New Jersey U.S.A.

52 Buxton J N, Laski J G: Control and Simulation Language. *Computer Journal*, 5(3), 1962.

53 Dahl O–J, Nygaard K: SIMULA – An Algol based simulation language. *CACM*, 9(9), 1966, pp.671–678.

54 Birtwistle G M: 1979, Discrete Event Modelling on SIMULA, Macmillan Computer Science Series, London.

55 De Prycker M, De Somer M: Performance of a Service Independent Switching Network with Distributed Control. *IEEE Journal on Selected Areas in Communications*, **SAC–5**(8), Oct. 1987, pp.1293–1301.

56 Earnshaw R W, Mars P: Simulation of ATM Networks on Transputer Arrays. In: Digest of the Seventh U.K. Teletraffic Symposium held April 1990, Durham, England, pp.1/1–1/5.

57 Roberts L, MacDonald T, Bernstein G: Fast Select Virtual Circuit Routing for B–ISDN Networks. In: Proceedings of the International Switching Symposium (ISS '92), **1**, held Oct. 1992, Yokohama, Japan, pp.219–223.

58 Hind A: Parallelization of a Circuit–Switched Telecommunication Network Simulator. In: Digest of the Ninth U.K. Teletraffic Symposium held April 1992, Guildford, England, pp.7/1–7/7.

59 Pitts J M, Sun Z: Burst–Level Teletraffic Modelling and Simulation of Broadband Multi–Service Networks. In: Digest of the Seventh U.K. Teletraffic Symposium held April 1990, Durham, England, pp.7/1–7/5.

60 Lutz K A: Considerations on ATM Switching Techniques. *International Journal of Digital and Analogue Cabled Systems*, **1**, (1988), pp.237–243.

61 Karol M J, Hluchyj M G, Morgan S P: Input vs. Output Queueing on a Space–Division Packet Switch. In: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '86), held Houston, Texas, USA, Dec. 1986, 2, pp.0659–0665.

62 Lehnert R: Performance Analysis of Switching Elements for ATM Switches. In: Proceedings of Stochastische Modelle und Methoden in der Informationstechnik, held 12th–14th April 1989, Nürnberg, Germany, pp.137–143.

63 Pattavina A: Performance Evaluation of ATM Switches with Input and Output Queueing. *International Journal of Digital and Analogue Communication Systems*, **3**, (1990), pp.277–286.

64 Spratt W J: Synchronous ATM – An Analysis of a Call–Packing Algorithm. In: Digest of the Eighth U.K. Teletraffic Symposium, held April 1991, Nottingham, England, pp.2/1–2/7.

65 Chopping G: Taking Time Across the Network. In: Proceedings of the IEE Integrated Broadband Services and Networks Conference, held 15th–18th Oct. 1990, London, England, pp.242–247.

66 Chambers 20th Century Dictionary, W & R Chambers Ltd., Edinburgh, 1983, p.1207.

67 Maisel H, Gnugnoli G: 1972, Simulation of Discrete Stochastic Systems, Science Research Associates Inc., United States, p.6.

68 Garcia R F, Garcia M R: 1990, Network Modeling, Simulation and Analysis, Marcel Dekker Inc., New York, p.2.

69 CCITT: Blue Book, Recommendation I.432, ITU, Geneva, 1989.

70 Cox D R, Miller H D:1980, The Theory of Stochastic Processes. Chapman and Hall, London/New York, p79.

71 Brookes C J, Betteley I G, Loxston S M: 1979, Fundamentals of Mathematics and Statistics, J. Wiley and Sons Inc., New York, London, Sydney, Toronto.

72 Burton N J: Computer Modelling: The Performance of ATM Switch Fabric Architectures. In: Digest of the Eighth U.K. Teletraffic Symposium held April 1991, Nottingham, England, pp.12/1–12/12.

73 Palmer R: The Application of Multistage Interconnection Networks to Fast Packet Switching. *Journal of Electrical and Electronics Engineering, Australia*, June 1988, 8(2), pp.119–129.

74 Kröner H, Eberspächer M, Theimer T H, Kühn P J, Briem U: Approximate Analysis of the End–to–End Delay in ATM Networks. Workshop on ATM Network Planning and Evolution, Queen Mary and Westfield College, University of London, England, 16th–17th April 1991.

75 Vlaming M S M G: Voice Echo and ATM Networks. Workshop on ATM Network Planning and Evolution, Queen Mary and Westfield College, University of London, England, 16th–17th April 1991.

76 Chatfield C: 1978, Statistics for Technology, Chapman and Hall, London.

77 Mood A M, Graybill F A, Boes D C: 1982, Introduction to the Theory of Statistics, McGraw–Hill, London.

78 Gonet P, Coudreuse J–P, Servel M: Implementing Asynchronous Transfer Mode Concepts: Main Results of the Prelude Experiment. In: Proceedings of the IEEE/IEICE Global Telecommunications Conference (GLOBECOM '87), held Nov. 1987, Tokyo, Japan, **3**, pp.1871–1875.

79 Horn G: Tandem Queues and Buffer Size in ATM Switching Nodes. *Stochastische Modelle und Methoden in der Informationstechnik*, Nürnberg, Germany, 12th–14th Apr. 1989, pp.131–136.

80 Kruskal C P, Snir M, Weiss A: The Distribution of Waiting Time in Clocked Multistage Interconnection Networks. In: Proceedings of the 1986 International Conference on Parallel Processing, held Aug. 1986, pp.12–19.

81 Grimmett G R, Stirzaker D R: 1982, Probability and Random Processes, Oxford University Press, England, p.45.

82 CCITT: Blue Book, Recommendations G.823, ITU, Geneva, 1989, p.40.

83 Wu LT, Kerner M: Emulating Circuits in a Broadband Packet Network. In: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '88), held Nov.–Dec. 1988, Hollywood, Florida, USA, **2**, pp.0679–0688.

84 Brady P T: A Model for Generating On–Off Speech Patterns in Two–Way Conversation. *The Bell System Technical Journal,* Sep. 1969, pp.2445–2472.

85 Daigle J N, Langford J D: Models for Analysis of Packet Voice Communication Systems. *IEEE Journal on Selected Areas in Communications*, **SAC–4**(6), Sep. 1986, pp.847–855.

86 Oliphant A: Can Broadcasters use the Broadband ISDN? In: Proceedings of the IEE Integrated Broadband Services and Networks Conference, held 15th–18th Oct. 1990, London, England, pp.8–10.