The Folded Hypercube ATM Switches

Jahng Sun Park

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Dr. N. J. Davis IV, Chair
Dr. F. G. Gray
Dr. S. F. Midkiff
Dr. R. E. Nance
Dr. B. D. Woerner

September 18, 2001
Blacksburg, Virginia

Key Words: Switching Fabric, ATM Network, Folded Hypercube

The Folded Hypercube ATM Switches

Jahng Sun Park

Abstract

Over the past few years, many high performance asynchronous transfer mode (ATM) switches have been proposed. The majority of these switches have high performance but also high hardware complexity. Therefore, there is a need for switch designs with low complexity and high performance. This research proposes three new ATM switches based on the folded hypercube network (FHC). The performance of the three architectures are studied using a network model and simulation. The major performance parameters measured are the cell loss rate and cell delay time through the switch under uniform, normal, and bursty traffic patterns. To guarantee faster switching of time-sensitive cells, the routing algorithm of the three switches uses a priority scheme that gives higher precedence to the time-sensitive cells. Also, an output buffer controller is designed to manage the buffers in a fair manner. The three proposed switch architectures have lower complexity while providing equivalent or better switching performance compared to other more complex ATM switches described in the literature. This research shows a new approach to designing ATM switches by using the FHC as the switching fabric for the first time instead of using the crossbar, multi-path, or Banyan-based switching fabrics.

Dedications

After a long and painful fight against cancer, my grandfather passed away on Christmas Eve, 1999. He deeply wanted to, but was not able to see me receive the doctorate degree. I was blessed to receive his continual love and support for me. I hope he is looking down on me from heaven feeling proud.

This dissertation is dedicated to my grandfather, Wookyu Park, in his loving memory.
I love you grandpa.

I also like to dedicate this dissertation to my parents, Mr. Johngseh and Mrs. Kyeha Park, to express my deepest gratitude and love for them. Without their love and support, I could not have finished this work.

Acknowledgments

As I finish this long and arduous research, I would like to thank several people who have helped me. First, I would like to express my gratitude and appreciation to my advisor, Dr. Nathaniel J. Davis IV for his patience and guidance. His suggestions and advise allowed me to overcome the difficult times during my research. He also endured the reading and numerous iterations of revisions needed to obtain this final product.

I also would like to thank my committee members, Dr. F. Gail Gray, Dr. Scott F. Midkiff, Dr. Richard E. Nance, and Dr. Brian D. Woerner. Special thanks go to Dr. Gray and Dr. Midkiff. Dr. Gray gave me guidance when I was teaching as an instructor. With his support I was able to gain great experience in teaching. Dr. Midkiff gave me an opportunity to work as a research associate to gain more research experience.

I wish to thank my best friend, Won Jae Lee, in Seoul, Korea. His words of encouragement and support helped me overcome the emotional difficulties I had during the research. My love goes to him, his wife, and his son.

Special thanks go to my dear younger sister, Hyesohng, and my brother-in-law, Namsoo Kim, for their love and support. A special love goes to my niece Cherie who has the world's cutest, prettiest, and loveliest smile. Also, my love goes to my baby sister, Hyejun, whose cute e-mails and e-cards made me smile during the tough times.

Table of Contents

List of Figures

List of Tables

1. Introduction

Asynchronous transfer mode (ATM) has been widely accepted by common carriers as the mode of operation for future communication systems, transporting and switching various types of data. ATM is based on packet switching using small, fixed-size packets of 53 bytes (5 bytes of header and 48 bytes of data) called *cells*, and line speeds equal to 150 Mbps or more.

The ATM switch for Broadband Integrated Services Digital Network (BISDN) differs from the existing ATM Local Area Network (LAN) switches in terms of large network capacity, throughput, and delay performance. Over the years, many high performance ATM switches have been proposed, and comprehensive surveys on ATM switches can be found in [AhD89, AwM95b]. These ATM switch architectures can be broadly classified into three categories: the *shared-memory* type, the *shared-medium* type, and the *space-division* type. The major drawback of the shared-memory and shared-medium architectures is that the shared components of the switch need to operate at a speed equal to N times the speed of the input line for an $N \times N$ switch. Therefore, these two types of architectures are only suitable for small ATM switches.

Many recent studies on ATM switching have been focused on the performances of buffering schemes for various types of space-division switches, mostly the input and/or output buffered switches with non-blocking switching fabrics. Designing a non-blocking switching fabric with good performance is the main goal for most of the researchers. The majority of these studies use some variation of interconnection networks, mainly the crossbar or the Banyan (or the multistage interconnection) networks, as their switching fabrics [AhD89, AwM95b]. However, some studies use disjoint-path based switching fabrics [AhD89, AwM95b].

Interconnection networks were rigorously investigated by computer researchers in the 1970s and 1980s, as the fast development of processing and memory capabilities made them an attractive solution for providing an effective communication medium among processing and memory units of multiprocessor systems. In the 1980s, interconnection networks received a lot of attention by researchers in the communication networking area for two important reasons. The first one was related to the type of communication network envisioned for the 1990s. The second one was related to the status of technology that was likely to be available in the medium term. For the high-speed cell-switching requirement of ATM networks, 150 Mbps and higher,

interconnection networks represent the best solution to implement the input/output (I/O) interconnection device of an ATM switching fabric. This can be attributed to their fast, self-routing packet property as shown by Patel [Pa81]. Furthermore, the parallel processing capability of these networks that exploits their self-routing property makes feasible supporting the load values envisioned for an ATM switch.

However, the basic structures of interconnection networks provide performance that is not suitable for the performance targets of an ATM network, which requires, at least for some services, specific cell loss requirements. To overcome this problem, many researchers have proposed various approaches for building a non-blocking interconnection network for ATM switches. As stated earlier, all of the proposed ATM switches based on interconnection networks use variations of either crossbar-based or Banyan-based networks. Recently, a few researchers have studied ATM switches based on the hypercube, a dynamic routing network [Ma93, PaC95]. This study presents three new ATM switch architectures based on the folded hypercube network (FHC), which is a variation of the hypercube network. The FHC is shown to have better performance than the hypercube with small number of extra links. Also, the routing algorithm and buffer control algorithm developed for these new switches should further enhance the performance. As a result, the new ATM switches should outperform the two types of ATM switches based on the hypercube in terms of delay through the switch and cell loss rate. In addition to presenting new ATM switches, this study also shows a new approach in designing ATM switches. Rather than using the Banyan networks like the majority of switches in the literature, this study is using the FHC as the switching fabric for the first time.

Chapter 2 presents an overview of space-division ATM switches proposed in the literature. Hypercube-type networks have received much attention over the years since they offer a rich interconnection structure with high bandwidth, logarithmic diameter, and high degree of fault tolerance. To further improve the characteristics and performance of the hypercube networks, some variations of hypercube structures have been reported in the literature. Chapter 3 discusses the architectures and the performance of two hypercube-based ATM switches, and also presents different variations of the hypercube networks: the twisted hypercube, the folded hypercube and the bisectional interconnection network (BIN).

Chapter 4 proposes three new ATM switch architectures based on the folded hypercube. The main advantage of the above hypercube variations over the regular hypercube is their

reduced network diameter. However, the FHC and BIN keep the symmetric characteristic of the hypercubes while the twisted hypercubes are asymmetric. The FHC is chosen over the BIN due to its simpler routing algorithm. The three architectures are modeled using the OPNET Modeler simulation tool. Then, they are simulated to study their performances. The two major performance parameters that are measured are the cell loss rate and cell delay time through the switch. These parameters are measured under the uniform and bursty traffic patterns with uniform and random destination distributions. Also, different combinations of external input- and output-link buffer sizes are studied to find the combination that achieve the best cell loss rate.

Chapter 5 presents the simulation results of the three ATM switches. Compared to the ATM switches based on the multistage interconnection networks, the three proposed switch architectures offer lower complexity while having equivalent or better switching performance. The cell delay times through the three new switches are lower compared to switches based on the hypercube or even the Banyan type networks. This is because an $N \times N$ hypercube or Banyan network has a diameter of $n = \log_2 N$ while an FHC of the same size has a diameter of $\lceil n/2 \rceil$. Furthermore, many high performance ATM switches based on the Banyan type network can have much higher cell delay time since they have more than n stages. In fact, in some Banyan network based switches like the sorting-based switches, a cell has to go through a multiple number of Banyan networks. Also, the three new switches have lower cell loss rate than the switches based on the hypercube or the Banyan type networks. These perfornance advantages are achieved with a smaller total number of queues. Finally, Chapter 6 gives a summary and suggests future work.

2. An Overview of Space-Division ATM Switches

This chapter presents an overview of space-division ATM switches proposed in the literature. Each switch is presented briefly in a category that best fits its characteristics. More detailed descriptions of each switch can be obtained from the original papers cited here.

2.1 Buffering Strategies

The main goal of a high-performance packet switch design is to provide many simultaneous input/output paths through the switch fabric and allow the internal paths to be time-multiplexed in a statistical rather than deterministic fashion. Depending on its design, the switch fabric may be blocking; in other words, the fabric may be unable to provide simultaneous, independent paths between arbitrary pairs of inputs and outputs. However, congestion still occurs even if the switch fabric is nonblocking because of the statistical nature of the arriving packets. Two or more packets may arrive on different inputs destined for the same output within the same time slot. The switch may allow only one of these packets to pass through the switch, and buffer others for later transmission. This form of congestion is unavoidable in a packet switch, and to deal with it, an efficient buffering strategy is needed in every switch architecture. There are three main buffering strategies: input, internal, and output buffering.

In a switch that is able to deliver only one cell to each output port in any given time slot, buffers are needed at the input ports (called input buffering), or within the switching fabric at possible points of contention (called internal buffering). However, when a switch fabric can deliver more than one cell to each output port in a given time slot, buffers are also needed at the output ports (called output buffering). A switch may employ one or any combination of the three buffering strategies. Finally, some blocking switch fabrics need only output buffering since they use certain methods to minimize blocking and provide multipaths to each output port.

Generally, internal buffering is not desirable because of one or more of the following reasons [Le90a].

- In multipath architectures, cells may be delivered out-of-sequence.
- It complicates the internal design of the switching elements.
- It complicates the fault-diagnosis testing if a virtual cut-through mechanism is used.

- It limits the maximum length of packets that can use the switch to the maximum internal buffer size.

In the following subsections, overviews of the input and output buffering schemes are presented.

2.1.1 Input Buffering

With input buffering, a separate *first-in-first-out* (FIFO) buffer is placed on each input port of the switch (Figure 2.1). Each arriving cell enters the buffer if space is available and awaits access to the switch fabric. If all *head-of-line* (HOL) cells are addressed to different outputs, the nonblocking switch fabric can send each cell to its respective output. However, if K HOL cells ($1 < K \leq N$) are destined to a particular output, only one is selected and allowed to pass through the switch fabric, while the other $K-1$ must wait until the next time slot. These cells may cause the so-called HOL blocking in the input buffering switches with FIFO queues. When an HOL cell is waiting to be transmitted to its output port, other cells may be blocked behind it despite the fact that their destination ports are possibly idle.

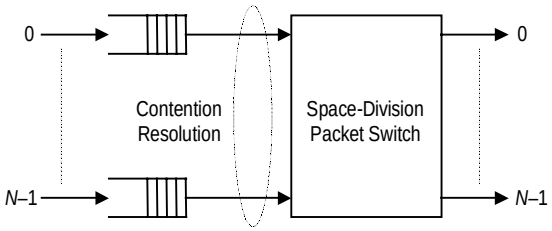


Figure 2.1 A Generic Input Buffering Switch.

Many different selection policies (or arbitration schemes) that try to increase the throughput of the switch by minimizing the HOL blocking have been proposed and studied in the literature. These selection policies may use a non-FIFO method to choose one cell among the contending HOL cells, but the service discipline within each input buffer still employs FIFO method. Regardless of the particular selection policy used, however, the throughput (TP)[1] of an input buffering switch is limited to $(2-\sqrt{2}) \approx 0.586$ due to HOL blocking [HlK88, KaH87]. This is the major disadvantage of the input buffering switch. The steady state queue sizes and the wait time for the cells grow infinitely above this value. This value is also known as the saturation value

[1] Throughput (TP) is the average number of cells that are successfully delivered by the switch per time slot per input line. Maximum throughput (TP_{max}) is the TP under maximum-load conditions.

of p, the cell arrival rate to each input port ($0 < p < 1$). The relation between the average queuing delay Q and the arrival rate p can be shown by the equation below for $N = \infty$, B_{in}[2] $= \infty$, and random selection of the contending HOL cells [HuA87].

$$Q = \frac{(2-p)(1-p)}{(2-\sqrt{2}-p)(2+\sqrt{2}-p)} - 1, \qquad 0 \le p < 2-\sqrt{2}\,. \tag{2.1}$$

The equation is plotted in Figure 2.2.

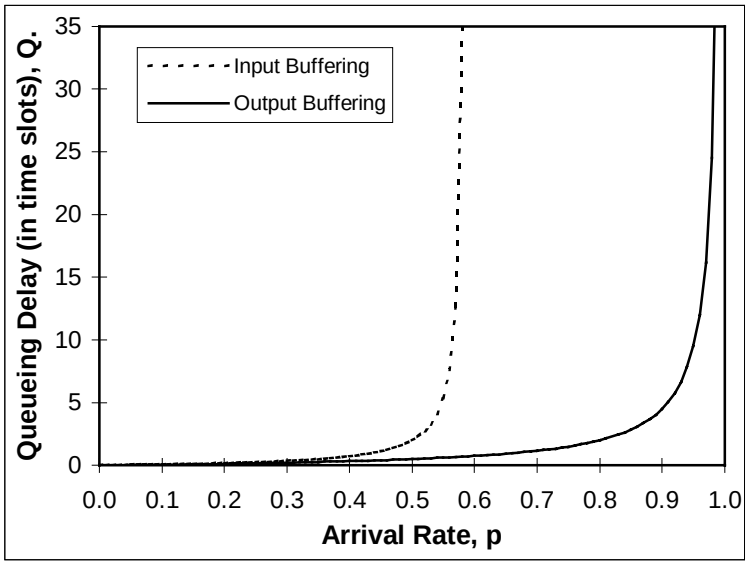


Figure 2.2 Throughput-delay performance of pure input and output buffering for nonblocking switches.

A TP_{max} of $(1 - e^{-1}) \approx 0.632$ can be achieved by not using any input buffering and dropping cells upon contention [Pa81, KaH87]. The following equation gives the throughput with this blocked-loss scheme.

$$TP = 1 - \left(1 - \frac{p}{N}\right)^N, \qquad 0 \le p \le 1. \tag{2.2}$$

However, this slight increase in throughput performance is achieved at the expense of very high cell loss rate, P_{loss}, which is unacceptable in the ATM network environment. Different selection policies still affect other performance measures of the switch when it is operated below the saturation level (when $p < 0.586$). For example, the longest queue selection policy results in less

[2] B_{in} is the capacity of each input buffer, and B_{out} is the capacity of each output buffer.

queuing delay compared to the random selection policy [KaH87] and to the HOL FIFO selection policy [BaM93]. Also, both the global FIFO and the longest queue selection policies have better cell loss performance compared to the oldest queue or the HOL FIFO policies [BaM91, BaM93]. The selection policy with a priority scheme [IyE89] insures the delivery of time-sensitive cells by giving them higher priority. It is certain that different selection policies have different implementation complexities.

2.1.2 Output Buffering

With the output buffering, all of the cells are buffered at the outputs of the switch with a separate FIFO provided for each output port (Figure 2.3). Here, it is assumed that the switch fabric is output-nonblocking where all arriving cells in a given time slot are cleared to the output side in one time slot, even if all N cells are destined to the same output port. This can be done by operating the switch fabric at a rate that is N times as fast as the input and output lines. Only one cell, however, can be transmitted via the output line in each time slot and the remaining cells with the same output address must wait in the output buffer, space permitting. Note that it is possible to achieve output buffering without the N times speed-up of the switch fabric. The Knockout Switch [YeH87] is one such switch, and it will be discussed in a later section. The average queuing delay Q is given by the following equation when $N = \infty$ and $B_{out} = \infty$.

$$Q = \frac{p}{2(1-p)}, \qquad 0 \le p < 1. \tag{2.3}$$

This equation is same as the M/D/1 queue equation [HlK88, KaH87] and is plotted in Figure 2.2.

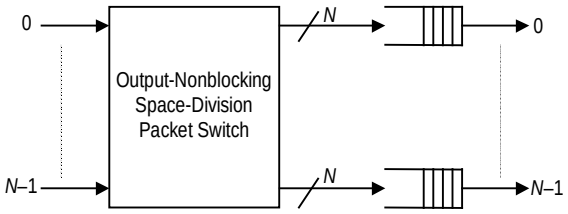


Figure 2.3 A generic nonblocking output buffering switch.

Unlike the input buffering, there is no HOL blocking that limits the maximum attainable throughput with the output buffering. It is only at each output that one finds the unavoidable congestion caused by multiple cells simultaneously arriving on the different inputs addressed to

the same output. Also, the output buffering provides the best waiting time performance achievable by any approach, and lends itself naturally to multicast and broadcast functionalities.

2.1.3 Comparison

As can be seen in Figure 2.2, the throughput-delay performance of the output buffering is significantly better than that of the input buffering under the uniform traffic for all switch dimensions [KaH87, HlK88]. The cost of this better performance is the higher complexity of the output buffering switches than that of the input buffering. The switch fabric itself is more complex since it is sped up by a factor of N in many output buffering switch designs. This further requires the output memories to have a high bandwidth. Each output memory must handle a maximum of N writes and one read operations every time unit compared to each input memory performing only one write and one read operations [To90]. By organizing the memory in a bit-sliced format or by using output port concentrator, as in Knockout Switch [To90], the output memory speed can be reduced.

When infinite size buffers are assumed, the output buffering has a higher throughput than the input buffering under any traffic model. Also, the output buffering needs less number of buffers than the input buffering when the same switch sizes and uniform arrival rates are considered [De91, Li90]. However, [Li90] has shown that under a bursty traffic model with the same size buffers, the output buffering may have higher P_{loss} than the input buffering. The author concluded that this is due to the inherent buffer sharing properties of the input buffering under bursty traffic. Simultaneous arrival of cells destined for the same output are stored over several buffers in the input buffering even though the buffers are not actually shared. For any buffering strategy, bursty traffic generally requires larger buffers to obtain the same P_{loss} compared to uniform traffic. Also, the switch needs to operate at a load much lower than the maximum throughput load to achieve small P_{loss} under bursty traffic.

2.2 Crossbar Based Switches

An $N \times N$ crossbar switch consists of a square array of N^2 individually-operated crosspoints, one for each input-output pair. Figure 2.4 shows a crossbar switch for $N = 4$; the horizontal (vertical) lines represent the inputs (outputs). Each crosspoint has two possible states: cross and

bar. A connection between input port *i* and output port *j* is established by setting the (*i*, *j*)-th crosspoint switch to the bar state.

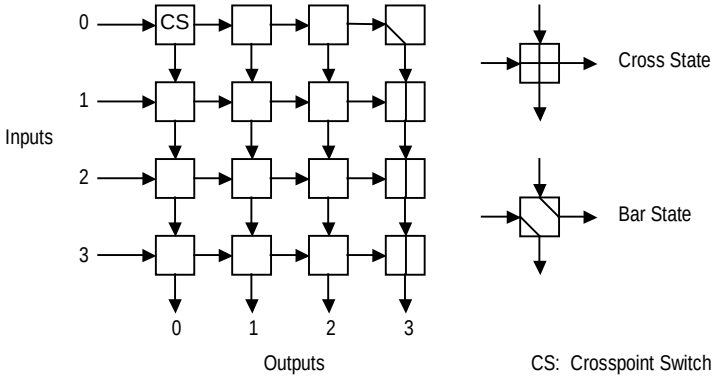


Figure 2.4 A 4×4 crossbar switch.

Over the years, crossbar switches have been favored by switch designers due to their nonblocking, simple architecture, and modular characteristics. They, however, have the following three problems [To90].

- Square growth of complexity (the number of crosspoints is of $O(N^2)$).
- Different input-output pairs may have different transit delays.
- Suffer from output blocking.

The first problem makes the crossbar switch a bad choice of fabric for a large size switch. A fairness problem rises from the second drawback when the switch is self-routing, but it can be solved by using artificial time delays. The last problem requires buffers to reduce cell losses as in any packet switch. The following sub-sections discuss different crossbar-based switches. The switches are classified by their buffering strategies.

2.2.1 Input Buffering

The input-buffered crossbar switch in Figure 2.5 needs a contention resolution scheme to select only one cell out of those requesting the same output port. There are two ways to do this: centralized and distributed. In the first approach, a central controller resolves output contentions in a slot-by-slot basis. This approach makes the switching fabric simple and economical. However, the centralized control becomes another limiting factor to the scalability of the switch. The second approach distributes the control over all output ports by using a separate controller or *arbiter* at each output port. Each arbiter looks at all the cells addressed to the corresponding output

port, selects one of them using some selection policy, and blocks all other cells by a backpressure signal. For each input port, there is one bus to broadcast the cell and its destination address, and one reverse control line for backpressure signal. Since the input-buffered crossbar switch suffers from HOL blocking, a good selection policy is needed to improve its performance.

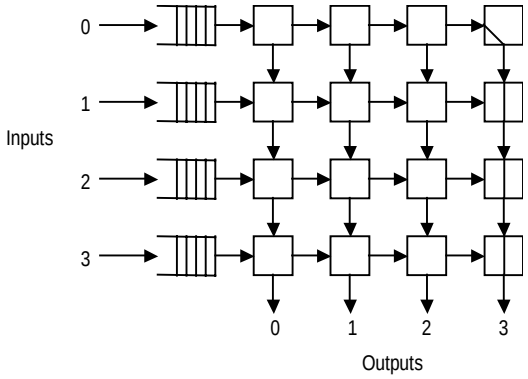


Figure 2.5 An input-buffered crossbar switch.

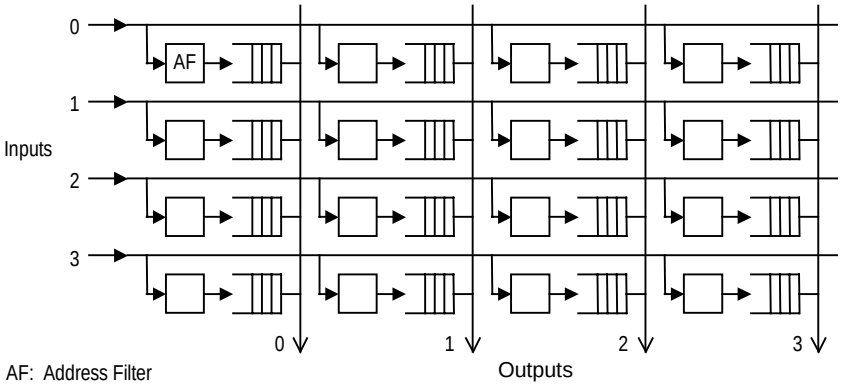


Figure 2.6 A crosspoint-buffered crossbar switch.

2.2.2 Internal or Crosspoint Buffering

Rather than at the input, buffers can also be placed at the possible contention spots within the switch fabric. Figure 2.6 shows an example, the *Bus Matrix switch* [NoT87], where an address filter (AF) and a FIFO queue replace the crosspoint switch. An incoming cell is broadcast to all attached AFs, but the cell passes through only one AF when their addresses match. An arbitration strategy is needed to choose one of *N* HOL cells in the buffers connected to the output buffer. For large enough buffer sizes, this switch design can achieve throughput-delay performance close to that of an output buffered switch [AlY91, DeF93]. However, the total number of buffers it needs

far exceeds that of an output buffered switch for the same level of performance since it needs N buffers per output rather than one buffer per output [To90]. The high complexity, mainly from the large number of buffers, is the main drawback of this architecture, and limits the switch sizes that can be implemented.

2.2.3 Input-Internal Buffering

To reduce the complexity of the crosspoint buffering and to improve the performance of the pure input buffering, [GuB91] has proposed the *Limited Intermediate Buffer* (LIB) switch. The LIB switch has FIFO buffers at the input ports and an intermediate buffer at each crosspoint as shown in Figure 2.7. An HOL cell in the input buffer moves to the intermediate buffer connected to its destination port only when that buffer is empty. A selection policy is needed to choose one of N intermediate cells that would be transmitted through the output port. After studying several selection policies, the authors found that the performance of the switch highly depends on the selection policy used. A 16×16 LIB switch achieved a maximum throughput of 87.5% which is much better than that of the pure input buffering. The authors also found that the maximum throughput of the LIB switch increased as the switch size increased.

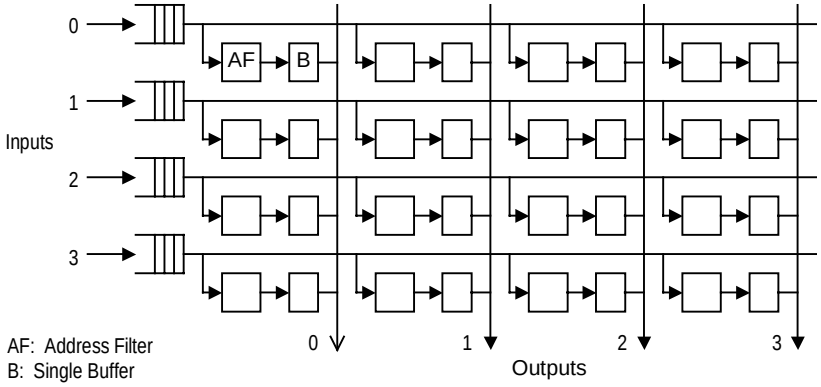


Figure 2.7 The Limited Intermediate Buffer switch.

2.3 Disjoint-Path Based Switches

Some switching fabrics have the capability of providing N^2 disjoint paths for an $N\times N$ switch. No blocking occurs within this type of fabric even among the cells destined to the same output port. Buffering of the cells at the output is required, though, since more than one cell can arrive in a time unit. This type of switch architecture can achieve the best possible

throughput-delay performance because of its pure output-buffered characteristic. The Bus-Matrix switch which was discussed above, also falls into this disjoint-path-based switch category. There are two types of disjoint-path based switches: disjoint in time and disjoint in space. These two types will be discussed in the following subsections.

2.3.1 Paths Disjoint in Time

Any switch can be changed to a disjoint-path based switch whose paths are disjoint in time by speeding up the fabric by a factor of *N*. [ImU88] proposed a multistage self-routing switch with stages that run *N* times faster than the external input/output lines. The switch has $n = \log_2 N$ stages, and each stage has *N* (2×2) switch elements (SEs) as seen in Figure 2.8. The input stage is numbered as *n* and the output stage is numbered as 1. Since this switch provides paths that are disjoint in time, a cell is lost only when the buffer of its destination output port is full. This switch can also perform multicast switching tasks. The disadvantage of this switch, however, is its size limit. Since the stages run *N* times faster than the external lines, building a large switch is impractical.

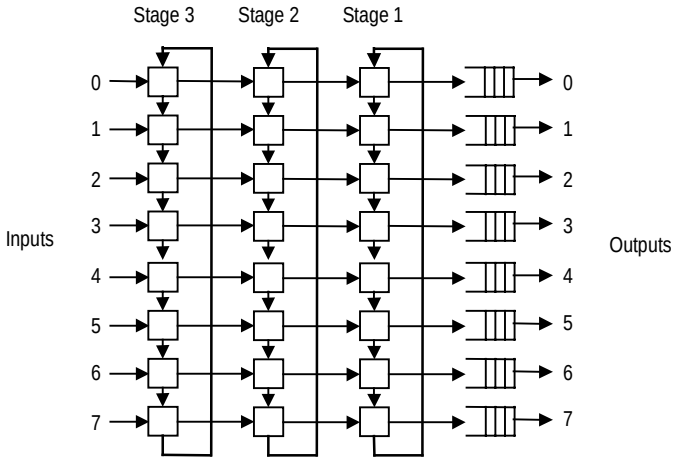


Figure 2.8 An 8×8 input-to-output address-difference-driven switch.

2.3.2 Paths Disjoint in Space

The best known switch with the paths disjoint in space is the *Knockout switch* (Figure 2.9) [YeH87]. The design stemmed from the authors' realization that all practical switches experience a certain amount of cell loss in the switch fabric. The Knockout switch uses a fully interconnected

fabric to broadcast all arriving cells to all output ports. Each output port has a bus interface that can receive packets from each input port. A bus interface of an output port is made up of three major components ~~a~~ filter, a concentrator, and a buffer. The filter has a self-routing capability. The concentrator uses a novel algorithm to select a fixed number of packets, *L*, from the *N* incoming lines. The buffer stores the cells in the order of their arrivals into a FIFO buffer that is shared among all *L* lines.

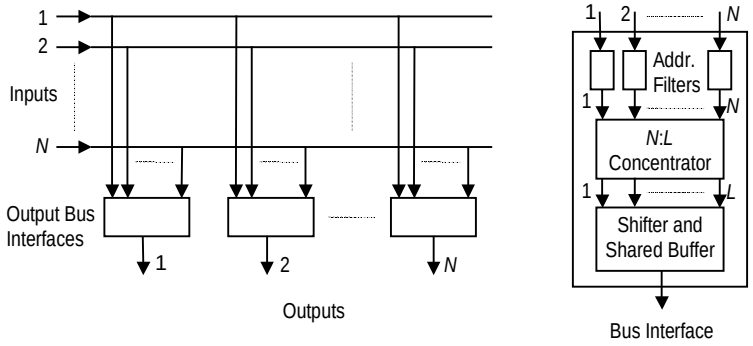


Figure 2.9 The Knockout switch.

The main theory behind the *N* to *L* concentration mechanism is that the probability of cell loss due to output congestion can be kept below the loss expected from other sources such as channel errors. This theory allows the number of separate buffers needed to receive simultaneously arriving cells to be reduced from *N* to *L*. With large *N* and $L = 8$, a concentration cell loss probability smaller than 10^{-6} can be achieved at a load of 0.9. For a nonuniform traffic model, larger values of *L* are required for similar cell loss probability [YoL88]. The cost of this good performance is the high complexity of the Knockout switch for large sizes. The switch needs N^2 busses or physical paths, N^2 address filters, and $N(N \times L)$ concentrators. There are many proposals to reduce the complexity of the Knockout switch.

In the Knockout switch, the filtering and concentration functions are done centrally at each output port. However, in the *distributed Knockout switch* proposed by [Ch90] (Figure 2.10), these functions are distributed in small switching elements (SWEs) located at the intersections of the crossbar lines. This switch needs LN vertical lines instead of N^2, and it does not need concentration within each output bus interface. However, these advantages over the Knockout switch come with the expense of LN^2 SWEs.

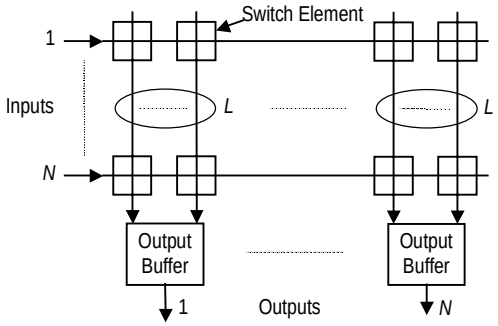


Figure 2.10 The distributed Knockout switch.

[NiS00] proposed two variations of the Knockout switch. The first one incorporates two speedup factors to reduce the number of accesses from a concentrator to a parallel buffer. The second proposed switch is a shared Knockout ATM switch which decrease the switch elements used. Both switches decreased the cell loss probability compared to the Knockout switch.

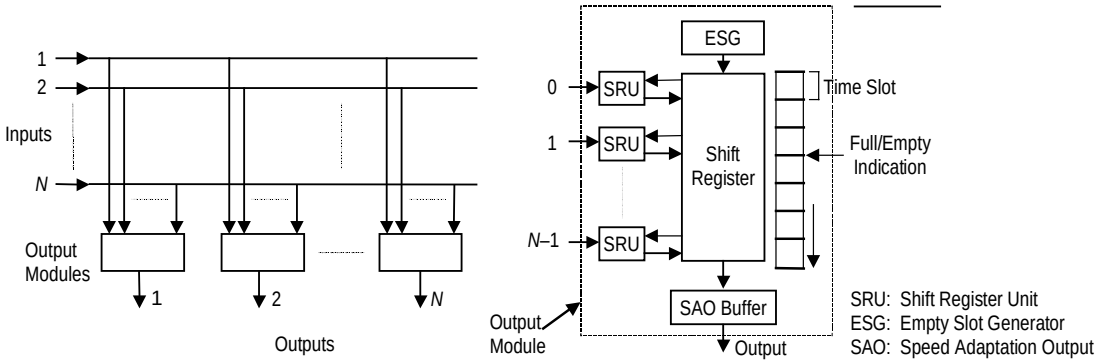


Figure 2.11 An output bus interface of the GAUSS switch.

The *GAUSS switch* proposed by [De90] and the *Cylinder switch* proposed by [MoP90] are two of many disjoint-path switch architectures that use different output bus interface structures and operations from that of the Knockout switch. The idea behind the GAUSS (Grab Any UnuSed Slot) switch design is the observation that, for non-bursty traffic, the probability of two cells originating from the same input port and destined for the same output port in consecutive time slots gets smaller as N increases. Figure 2.11 shows the output bus interface called the GAUSS module. Each input of a GAUSS module has a shift register unit (SRU) connected to the parallel shift register that sends the cells arriving at the SRUs to the speed adaptation output buffer (SAO-buffer). The slots are shifted at a faster rate than the cell arrival rate at the inputs. Each SRU transfers a received cell to any empty slot by means of a 'grab any' mechanism. The speed-up

factor L is the ratio of the shift register speed to the input line speed, and is determined by the desired P_{loss}. The performance analysis of the GAUSS switch is done in [Aav92], and they showed that the GAUSS switch has better average cell loss performance than the Knockout switch for the same N, L, and load value. However, since the upper input lines have better chance of 'grabbing' an empty slot, the GAUSS switch is unfair.

The Knockout switch may discard a cell at a concentrator even if there is available space in the corresponding output buffer. [MoP90] proposed the Cylinder switch (Figure 2.12) that solves this basic problem of the Knockout switch. Unlike the GAUSS switch, the shift registers of the cylinder switch form a ring or a cylinder. The cells rotate in the ring until they are selected and transmitted by the corresponding output element, one in each time slot. The Cylinder switch requires the ring to run at a speed that is much faster than that of the input or output lines, and also requires a minimum of N shift registers in each of the N buffer rings.

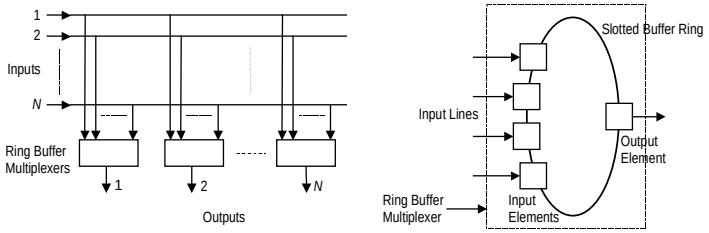


Figure 2.12 An output bus interface of the Cylinder switch.

Chao developed a path-disjoint ATM switch by using multiple switch planes [Ch00]. Each switch plane consists of a matrix interconnection of crosspoint chips (XPCs). Each XPC is composed of a matrix interconnection of crosspoint units (XPUs). The switch has input and output buffers, and uses a simple dual round-robin arbitration scheme to reduce the head-of-line cell block. The switch has good performance, but requires multiple switch planes, and the switch planes must be sped up by at least three times the network speed to achieve the high performance.

2.4 Banyan Based Switches

The earlier works on the *multistage interconnection networks* (MINs) were done in the context of circuit switching [Be64, BrH83, Ch91, Fe81, McM84]. The main goal of these works was to design a nonblocking multistage switch with less complexity than a single-stage crossbar switch. Since then, many MINs were proposed and studied for the purpose of interconnecting

multiple processors and memories in parallel computer systems as the interest in large scale processing systems grew [BhY89, Ch91, DiJ84, Fe81, McM84]. Some of the MINs known in the literature are Banyan [GoL73], Baseline [WuF80], Reverse Baseline [WuF80], Omega [La75], Indirect Binary *n*-Cube [Pe77] and Generalized Cube [Si85] networks. Eventually, these networks have been considered as packet-switching fabrics to achieve high throughput because several packets can be switched simultaneously and self-routed through these networks. The MINs listed above have different interconnection patterns but are topologically equivalent [Si85, WuF80]. Also, they have the same performance in a packet-switching environment under uniform traffic [DiJ84]. Figure 2.13 shows the three most frequently used topologies for N=8.

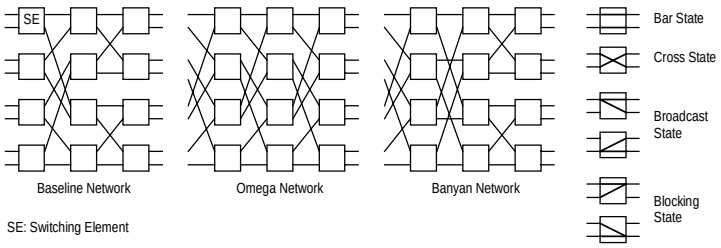


Figure 2.13 Three different topologies of Banyan networks.

A large number of proposed ATM switches are based on the Banyan networks due to their attractive properties [AhD89, AwM95b, Ch91, To90]. The main properties of these networks are [AhD89, AwM95b].

- they consist of $\log_b N$ stages and N/b nodes per stage;
- they have the self-routing property for transferring packets from any input to any output using a unique k digit, base b destination address;
- they have a complexity of $O(\log_2 N)$ compared to $O(N^2)$ for the crossbar switch;
- they can be constructed in a modular way from smaller switches;
- they can be operated in a synchronous or asynchronous mode; and
- they have regular interconnection pattern that are very attractive for VLSI implementation.

While these networks are capable of switching packets simultaneously and in parallel and their path-uniqueness preserves the cell sequences, they are inherently blocking networks. There are two types of blocking: internal link blocking and output port blocking. Figure 2.14 shows an example of the two blocking types. The internal link blocking occurs when two or more cells contend for a particular link inside the network. The output port blocking, however, is a case

where packets are lost due to contention for the same output port. These two effects result in the reduction of the maximum achievable throughput of the switch. One can see this as the trade off cost for reduced number of crosspoints from N^2 (crossbar) to $2N \log 2\, N$ (Banyan).

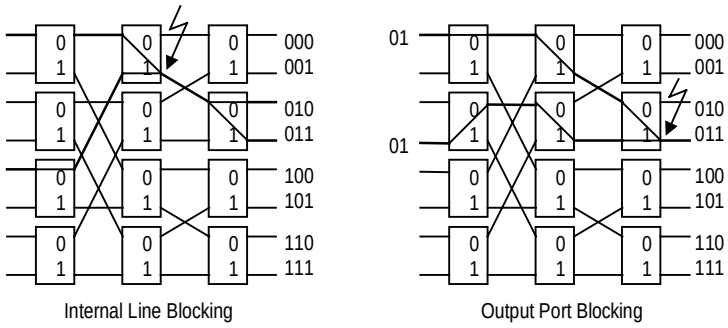


Figure 2.14 Internal link blocking and output port blocking in an 8×8 delta network.

The performances of these switches have been studied in great detail by many researchers [DiJ81a, KrS83, KuJ86, Pa81]. [Pa81] analyzed Banyan networks with a random selection blocked-loss policy in the SEs, and derived the following recurrence relation ($1 \le i \le n$);

$$p_i = 1 - \left(1 - \frac{p_{i-1}}{b}\right)^b \tag{2.4}$$

where $p_0 = p$ (cell arrival rate), p_i is the probability that an output link of stage I carries a cell, and $b \times b$ is the size of each SE. For a given p_0,The throughput is simply $TP = p_n$. The following approximate solution for the throughput of a Banyan network was derived by [KrS83]:

$$TP = \frac{2b}{(b-1)n + 2b/p} \tag{2.5}$$

Also, [KuJ86] derived tight upper and lower bounds on TP. Figure 2.15 shows the plot of Equation (2.4) as a function of $n = \log_2 N$ for $p = 1.0$ and $b = 2$. The throughput of a crossbar switch with blocked-loss policy, from Equation (2.2), is also shown in the figure for a comparison. Both the crossbar and Banyan network suffer from performance degradation as the switch size increases, but the degradation is more severe for the Banyan network. This is because, in addition to suffering from the output blocking like the crossbar, the Banyan network experiences more internal link blocking as the number of stages increases. In particular, $TP_{\max}(N = \infty) = 0.632$ for crossbar and $TP_{\max}(N = \infty) = 0$ for Banyan network. By examining Equations (2.4) and (2.5), one

can see that the throughput improves as *b* increases; in other words, using larger size SEs. This is because larger *b* means less number of stages which in turn means less internal link blockings. (If *b* approaches *N*, the network becomes a crossbar which does not suffer from any internal blocking.)

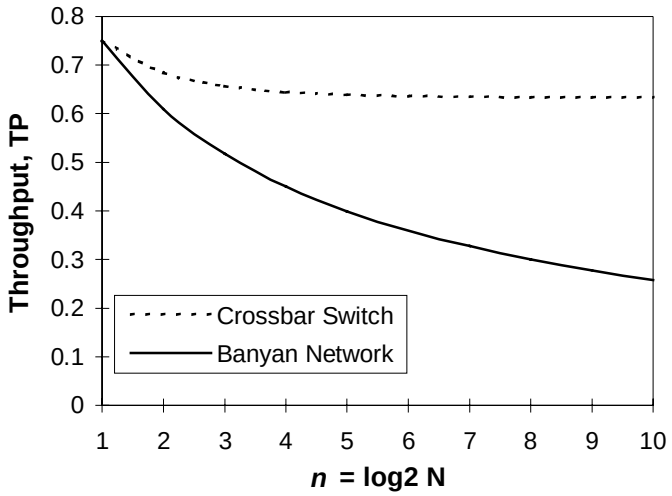


Figure 2.15 Maximum throughput of Banyan networks and crossbar switches.

Many studies were done to improve the performance of the Banyan networks. In particular, [DiJ84] discussed in detail how the performance of a Banyan type network can be improved. There are several ways to improve the throughput of Banyan-type switches [DiJ84, AhD89].

- increase the internal link speeds relative to the external line speeds;
- place buffers in every switching element;
- use a handshaking mechanism between stages or a backpressure mechanism to delay the transfer of blocked cells;
- using multiple networks in parallel to provide multiple paths for any input and output pair or multiple links for each switch connection [KrS83, KuJ86];
- use a distribution network in front of the Banyan network to distribute the load evenly.

The following subsections present different classes of Banyan-based switches that use one or more of these improvement methods.

2.4.1 Buffered Banyans

The most straightforward way to avoid the internal link blocking in a Banyan network is to use SEs with buffers. When there is a link contention in a buffered Banyan network (BBN), the losing cells would be buffered rather then dropped. In general, BBN cells are transmitted synchronously from one stage to the next stage. There are three modes of operation for a BBN: queue loss (QL), local backpressure (LB), and global backpressure (GB). In QL mode, a cell is dropped when it arrives at a full buffer. In LB mode, a cell can only move to the next stage if the destined buffer is not full. However, in GB mode, a cell can still move to the next stage even if the destined buffer is full as long as that buffer will not be full by the time this cell arrives. Among these three modes of operation, QL is the least complex while GB is the most complex to implement. However, performance wise, QL is the worst and GB is the best. One thing to note is that with either LB or GB, a cell can be dropped only at the buffers of the first stage or, if used, at the external input buffers.

The simplest and the most studied type among the buffered Banyan networks is the input-BBN [AtA94, AwM95b, BuT89, DiJ81a] where buffers are placed at the inputs of each SE. Similarly, buffers can be placed at the outputs of each SE (output-BBNs), at the crosspoints of each SE (crosspoint-BBN), within each SE to be shared by all of its inlets and outlets (shared-BBNs), or at both the inlets and outlets of each SE (input-output-BBNs). The performance of a BBN depends on the individual performance of its SEs, and this depends on the buffer size and buffering strategy. Output-, crosspoint-, and shared-BBNs achieve a 100% maximum throughput when infinite buffer size is assumed for any size N [AwM95b, KrS83]. Input-BBNs, however, achieve only a 75% maximum throughput under the same assumptions [AwM95b, DiJ81a]. This is due to the HOL blocking, which was discussed earlier. Under nonuniform traffic, however, all BBNs perform poorly indifferent to the buffering strategy used [AtA94, AwM95b, BuT89]. [AtA94] showed that the unbuffered Banyan networks can perform better than the buffered Banyan under hot-spot traffic.

2.4.2 Multiple Banyans

To over come the disadvantage of the Banyan network having exactly one path for any input and output pair, many Banyan-based switches use more than one Banyan network. There are

three types of multiple Banyan switches. They are parallel loading, sequential loading, and double Banyan, and these are discussed in the following subsections.

2.4.2.1 Parallel Loading

The probability of blocking increases with the load in a Banyan network. As a possible solution to this problem, multiple Banyan networks can be used to distribute the load among them. The *replicated Banyan network* which uses K Banyan networks [KrS83, KuJ86] in parallel is a good example of this type of switch (Figure 2.16). The incoming cells are randomly distributed over the K Banyan networks. At the output ports, either single acceptance (SA) or multiple acceptance (MA) can be used. With SA, only one out of possible K cells that arrive at an output port is accepted, and no output buffering is needed [AwM94a, KrS83, KuJ86]. However, with MA, all cells that arrive at an output port are accepted, and buffers are required at the output ports [AwM94a, LeL92, ToK91]. For both schemes, input buffering has to be used for reasonable values of K to be used in ATM environment. The SA scheme needs 20 Banyan networks while MA needs 4 Banyan networks to achieve $TP_{\max} = 0.60$ for $N = 1024$ [KrS83, KuJ86]. If internal buffers are used, cells belonging to the same virtual connection should be sent to the same Banyan network in order to preserve the cell sequencing.

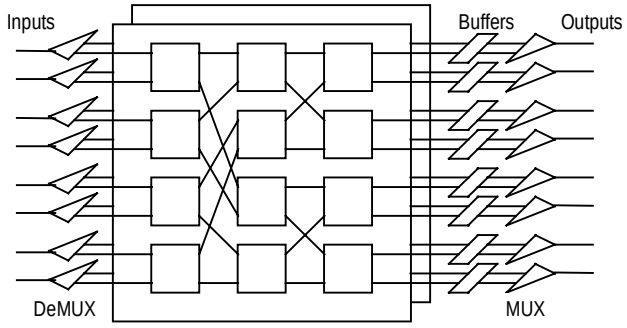


Figure 2.16 An 8×8 replicated Banyan network with $K = 2$.

[OkÇ97] proposed a new ATM switch called the *plane interconnected parallel network* (PIPN). As shown in Figure 2.17, PIPN is composed of two Banyan-based interconnected planes, a distributor at the input side, and an output-port dispatcher. Both Banyan networks have n–1 ($n = \log_2 N$) stages for a size N switch, and each stage has 2^{n-2} SEs. A newly arriving cell is randomly sent to one of the two planes by a DE. When a cell is sent to the front plane, the cell's destination address fields are complemented and its *complement field* is set to one. This complementing of the

destination address fields distributes the arriving traffic efficiently over the entire network. The router is composed of two $2^{n-1} \times 2^{n-1}$ planes, front and back as shown in Figure 2.18(a). The upper (lower) outlets of DEs are connected to the back (front) plane. A cell is routed according to its destination field starting with the second most significant bit. If there is a link contention, the losing cell is dropped. As a cell progresses through the network, its destination address and complement field are not modified.

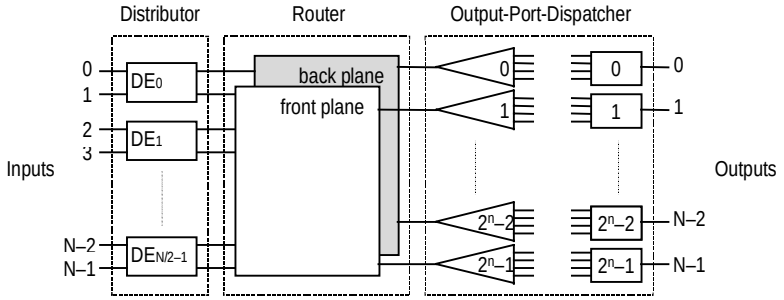


Figure 2.17 General structure of the Plane Interconnected Parallel Network.

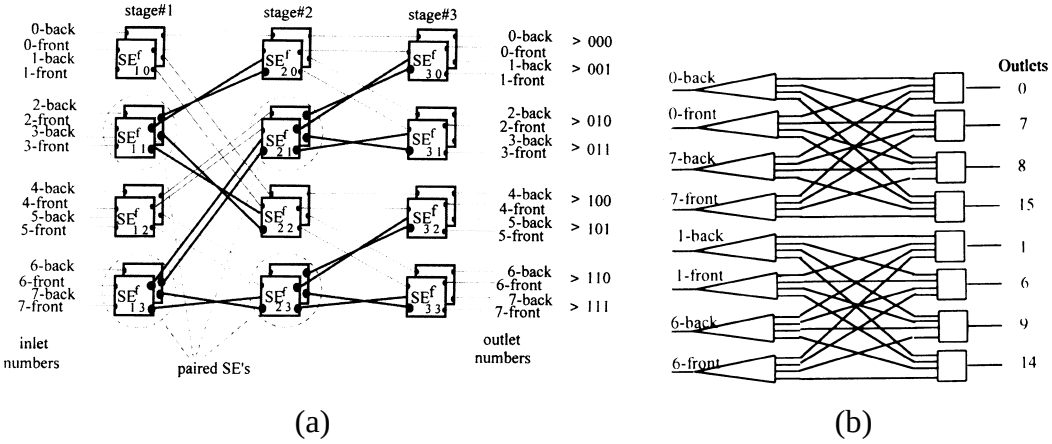


Figure 2.18 (a) Internal structure of a 16×16 router; (b) An example of output-port dispatcher.

Cells exiting the router outlets are routed to the requested output ports by the output-port dispatcher. The output-port dispatcher is made of two sub-units: There is a *decider* (a 1 × 4 demultiplexer) for each outlet of the router (or each inlet of the output-port dispatcher) and the *collector* for each output port. A decider routes a cell by using the cell's complement field and the most significant bit of the cell's destination address. A collector is made of four inlets, a buffer to store cells arriving from four possible deciders, and an outlet that corresponds to an output port of the switch. An interconnection network is still needed between a set of four deciders and a set of

four collectors. For their simulation study, the authors used a Banyan type network as shown in Figure 2.18(b). Also, the collectors need to operate four times faster than the deciders.

The throughput performance of the PIPN is better than that of a Banyan network, but the difference becomes smaller as the network size increases. Under heterogeneous traffic, however, the throughput of the PIPN is significantly better than that of a Banyan network even for the larger network sizes. The cell sequence is preserved since all paths have the same length and neither buffering nor recirculation is done in the network. However, the issue on the probability of cell loss has not been addressed. [Ok01] has modified the PIPN to support multicast packets.

[YoE99] further improved the performance of the PIPN by using the replication technique. They used two and four copies of the PIPN to design ATM switches. Their switches use the replication technique to provide multiple paths between inputs and outputs, and use the PIPN to smooth the heterogeneous traffic models. The availability of more paths between each input-output pairs makes the modified switches more reliable than the original PIPN. For 256×256 size switch, the replicated switch using two PIPNs had a throughput improvement that is close to 30 percent. For the same size switch, when four PIPNs are used, the throughput almost doubled when compared to the original PIPN.

Another parallel loading multiple Banyan switch was designed by [Kr00]. It is called the Stacked-Banyan (SB) ATM switch. An $N \times N$ SB switch is composed of $K = N/2$ parallel Banyans, and uses 2×4 SEs. Two output of an SE are for "horizontal" routing in the same plane and the other two are for "vertical" routing to another plane when blocking occurs. A significant property of the SB is that the parallel Banyans have decreasing number of stages. The first plane has $n = \log_2 N$ stages, the second plane has n-1 stages, the third and fourth planes have n-2 stages, the fifth, sixth, seventh, and eighth have n-3 stages, etc. By having multiple parallel planes, the SB eliminates internal blocking of the cells. All arriving cells enter the switch via the first plane. A cell is routed horizontally within the same plane until it encounters a blocking. When a cell is blocked, it is routed vertically to another plane. For a more detailed routing algorithm refer to [Kr00]. Output buffers are needed since all N input cells can be addressed to one output. The SB switch needed the buffer size to be at least the number of input ports (N) in order to achieve decent cell loss rate. For $N = 16$, 100% loading, the cell loss rate was about 5×10^{-4} when the buffer size is 16. The cell loss rate dropped quite rapidly when the buffer size is increased. However, when the buffer size is decreased to 12, the cell loss rate increased to 3.2×10^{-3} [Kr00].

2.4.2.2 Sequential Loading

Unlike the parallel loading multiple-Banyan switches, cells are offered one by one to each Banyan network in the sequential loading multiple-Banyan switches. A cell is offered to the first Banyan network. If it is blocked, it is offered to the second Banyan network, and so on until the last *K*-th Banyan network. Any traffic blocked by the last Banyan is either dropped or re-offered to the networks in the next cell cycle with new arriving cells. Output buffers are needed since more than one cell can arrive at an output port.

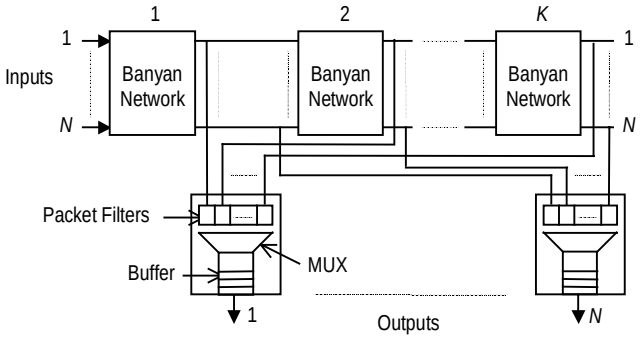


Figure 2.19 The Tandem-Banyan switch.

The *Tandem-Banyan Switch* (TBS) [ToK91], which is shown in Figure 2.19, consists of *K* Banyan networks in series. When a cell loses contention in the *i*-th ($0 \le i \le K-2$) network, it is marked and misrouted so that it will not compete with other cells that are being routed properly within the same network. At the output of the *i*-th network, the unmarked cells are sent to the output ports while the marked cells are sent to network *i*+1 with their marks removed. The marked cells out of the last network are dropped. As mentioned earlier, different topologies of omega network have the same performance under the uniform traffic. However, when they are cascaded like the tandem-Banyan switch, they show significantly different performance [SiZ93, ToK91]. For example, to achieve a $P_{loss} = 10^{-6}$ ($N = 1024$), 10 Modified Data Manipulator or 14 Baseline networks are needed. Even better performance is obtained when Omega networks are used. Another way to reduce the number of networks needed for a given P_{loss} is to re-offer the cells out of the last network back to the switching fabric. One of the disadvantages of the tandem-Banyan switch is that it can deliver cells out of sequence for large *N* unless artificial delays are used. Also, each SE is very complex since it needs to examine the conflict bits in addition to the destination

bits of both cells at its inlets. Finally, the networks in series render the chip-to-chip synchronization difficult.

2.4.2.3 Double Banyan

Figure 2.20 shows a switch made up of two cascaded Banyan networks. The first network is called the randomization or distribution network while the second one is the normal self-routing Banyan network. The first network is used to reduce the effect of incoming traffic imbalances on the performance of the second switching network. The performance of the switch heavily depends on the operation of the first network which depends on the particular switch design. In [Ne88], the first network works as a cell-basis randomizing network where each SE of the first network randomly sets its state ignoring the incoming cells' destinations. Both networks of [Ne88] are internally unbuffered. Another design by [Tu88] uses two buffered Banyan networks. In this design, the first network is called the distribution network in which the SEs route cells alternately to each of their outlets regardless of the cells' destination addresses. Unfortunately, the cells may arrive at the output port out of sequence.

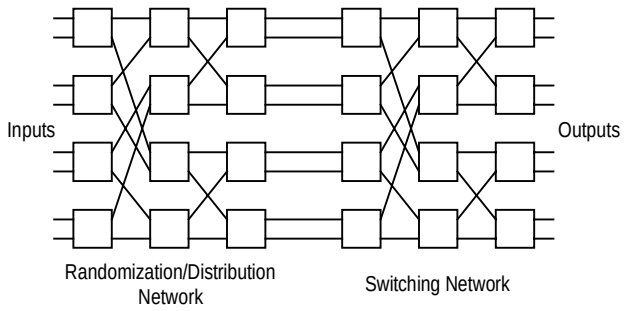


Figure 2.20 A cascade of two Banyan networks.

2.4.3 Dilation

The *dilated Banyan* network has an improved throughput performance over the regular Banyan network by using d internal links instead of one [KrS83, KuJ86]. Figure 2.21 shows an example of the dilated Banyan network with $d = 2$. Each SE is a $(2d \times 2d)$ switch with two output addresses. The dilated Banyan network uses the same routing algorithm of the regular Banyan network. An internal blocking occurs only when $d+1$ or more cells request the same outlet of an SE, and in the event of an internal blocking, d random cells are transmitted while the others are

dropped. Like the replicated Banyan network, one of the two acceptance policies, single acceptance (SA) or multiple acceptance (MA), can be used at the output ports of a dilated Banyan network. As explained earlier, output buffering is necessary with MA since up to d cells are accepted by an output port [AwM94a, GhD91, LeL92, YoU93]. However, with SA, output buffer is not required since no more than one cell is accepted by the output port in a cell cycle [AwM94a, KrS83, KuJ86]. Finally, in the ATM environment, input buffering is required with SA, but not with MA if d is large enough to achieve an acceptable P_{loss}. It has been shown that a dilated Banyan network with SA becomes nonblocking when $d = 2^{\lceil n/2 \rceil}$ where $n = \log_2 N$.

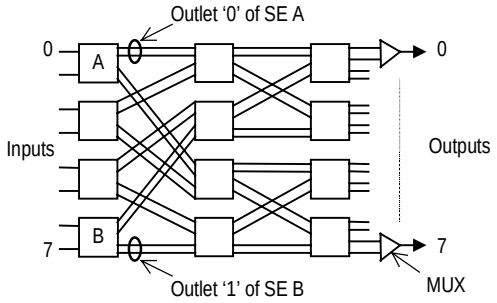


Figure 2.21 An 8×8 dilated Banyan network with $d = 2$.

Using the MA policy can further improve the throughput performance of a dilated Banyan network. The TP_{max} of an output buffered dilated Banyan network with $d = 3$ and $B_{out} = 20$ exceeded 90% in [GhD91]. [SzS89] showed that to have a cell loss rate between 10^{-3} and 10^{-6} under a full loading condition, only d between 4 and 8 is needed. [SzS89] also showed that a TP_{max} approaching 1 can be achieved only with $d = 3$ when single output-buffered SEs are used. However, this architecture has multi-paths with internal buffering, and therefore, may deliver cells out of sequence. Another architecture shown in Figure 2.22 can also achieve a TP_{max} approaching 1 without internal buffering [YoU93]. In this architecture, the contention losing cells are misrouted (deflected), not dropped, and they re-enter the network through the input ports. This architecture, however, may also deliver cells out of sequence.

[SaH99] combined the concentration mechanism of the Knockout switch discussed in Section 2.3.2 and the dilated Banyan to design an ATM switch. Their switch had lower hardware complexity but gave good performance. They also implemented multicast capability in their switch. [AlK99a, AlK99b], however, combined the idea of multiple switching planes with the

idea of dilation to develop their switch. Their switch also has high performance, but needs multiple number of switching planes.

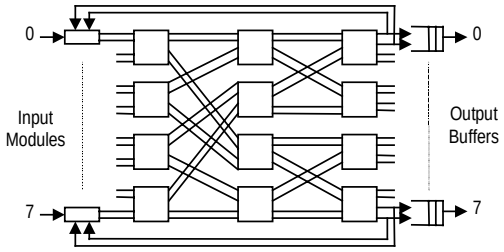


Figure 2.22 An 8×8 dilated Banyan network (d = 2) with deflection routing and recirculation.

2.4.4 Sorting

Some of the Banyan class of networks like a Banyan or an Omega network have a very useful property. Given that there are no multiple cells destined for the same output port, these networks are nonblocking if the entering cells are sorted (either in ascending or descending order) according to their output addresses, and concentrated (compact with no gaps between active inputs) [Le88, Na88]. The Batcher network [Ba68], based on the bitonic sorting principle, is the well-known architecture that can sort in a distributed and parallel method. An (N×N) Batcher network has $\log_2 N(\log_2 N+1)/2$ stages, and each stage has N/2 binary comparators or sorting elements. The Batcher network can sort any set of cells based on any information, usually the output address, contained in the headers, and group them in order at the bottom or the top of its output ports. Therefore, the combination of a Batcher network and a Banyan network solves the problem of internal blocking. As long as there are no multiple cells destined for the same output port, this Batcher-Banyan (BB) network is as effective as the crossbar switch. The particular approach used to resolve output conflicts in a BB switch strongly depends on the buffering strategy the switch uses. Figure 2.23 shows an example of a BB network.

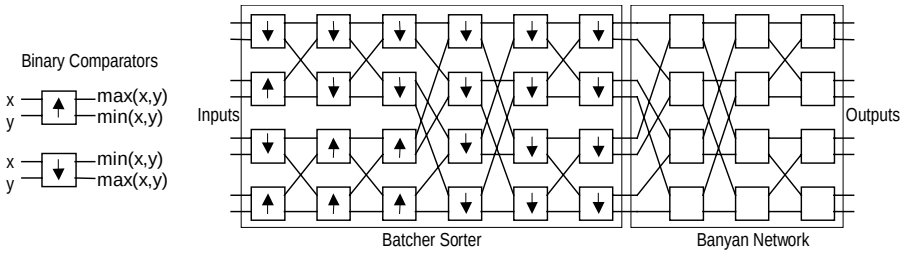


Figure 2.23 The Batcher-Banyan network.

2.4.4.1 Internal Buffering

The first proposed sorting Banyan based switch is the *Starlite* switch [HuK84] (Figure 2.24). The concentrator reduces the size of the other networks in the switch, assuming that there is a high possibility of having many idle input ports. The Starlite switch consists of the concentrator, the sorting (Batcher) network, the trap network, and the expander. The concentrator is built from running sum adders followed by a reverse Omega network. The trap network is made up of a single stage of comparators followed by an Omega network, and the expander is just an Omega network. The Starlite switch, however, may deliver cells out of sequence due to the recycling approach, and requires internal buffers for the recycled cells. Finally, a significant amount of the input ports must be used for recirculators in order to minimize cell loss within the recirculation buffers. This results in under-utilization of the switch capacity, but the recirculation method provides buffer sharing and smoothes bursty traffic.

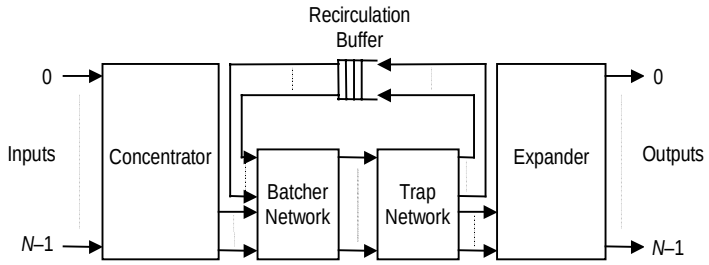


Figure 2.24 The Starlite switch.

[BiP96] has recently proposed an ATM switch based on the Batcher-Banyan structure that performs the parallel self-routing of all the cells entering the network. The switch consists of a Batcher (sorting) network, a trap network, a Banyan network, and a recirculation network as shown in Figure 2.25.

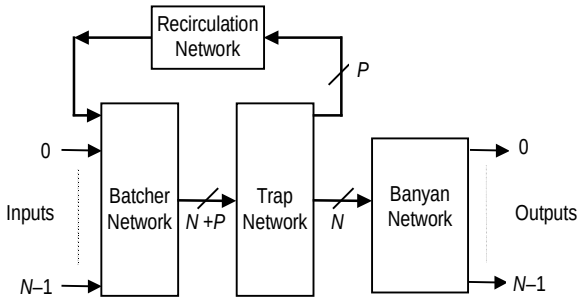


Figure 2.25 The general structure of the interconnection network switch.

The trap network chooses the oldest one among the cells that are destined to the same output port and routes it to the Banyan network. The losing cells are routed to the recirculation network. Choosing the oldest cell maintains the cell sequence. When there are more than P losing cells, the trap network uses a selection algorithm to choose the cells to be dropped. [BiP96] considered three selection algorithms.

- *windowing algorithms*: cells are discarded from the buffer according to the particular position they assume after the Batcher sorting;
- *dynamic priority algorithms*: the discarded cells are chosen based on the cell's age, either oldest (DPO) or newest (DPN); and
- *scanning algorithms*: the discarded cells are selected with the aim of equally sharing the recirculation network capacity among all the cell classes.

The scanning algorithms are highly complex to implement and are only defined to provide the performance upper bounds for the other two algorithms. The windowing algorithms are easier to implement than the dynamic priority algorithms. The DPO algorithm is not perfectly fair but gives the best TP_{max} and P_{loss} performance, which is close to the upper bound set by the scanning algorithms. For a switch size of $N = 32$ and $P/N = 2.5$, DPO algorithm has $P_{loss} = 5\times10^{-5}$ and $TP_{max} = 0.80$ at a load of 80%. The DPO gives the minimum delay through the switch while providing multiple priority service.

2.4.4.2 Input Buffering

Using FIFO input buffers to resolve conflicts before switching cells would be the most direct method to implement a BB network. This approach, however, has two shortcomings: (1) it needs a good contention resolution scheme suitable for a BB network, and (2) TP_{max} is considerably less than unity because of the HOL blocking as mentioned earlier [HlK88, HuA87, KaH87]. [HuA87] first proposed a contention resolution scheme for an input-buffered BB network. It is called *3-phase algorithm*, and the switch using this scheme is simply called the *BB switch*. As can be seen in Figure 2.26, same-number input and output port controllers are located in the same module PC_i ($i = 0, 1,\cdots, N-1$) that interfaces both input port i and output port i of the switch. Details on this algorithm can be obtained from [HuA87].

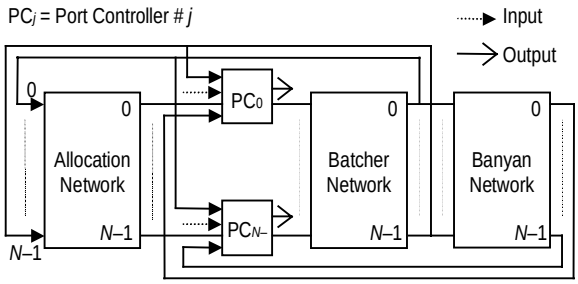


Figure 2.26 The 3-phase Batcher-Banyan switch.

There are some disadvantages in the 3-phase algorithm. First, there is a fairness problem in the 3-phase algorithm. When contending for the same output port, the cells in either the lower or higher address input ports have a higher priority allocating the switching bandwidth [Pa89]. Also, the 3-phase algorithm requires a large processing overhead, about 33% for an ATM switch of size N = 1024 [HuA87]. A 2-phase contention resolution algorithm proposed by [AwM94b] overcomes the above disadvantages. It has less processing overhead than the 3-phase algorithm, can support prioritized traffic in a straightforward manner, and provides fairness. Both methods mentioned above, however, can only achieve the maximum throughput of 58.6% because of the HOL blocking. Also, the BB switch has shortcomings in its ability to share buffers, needs large buffer to insure a small cell loss probability, and has poor performance under imbalanced traffic [LaL90].

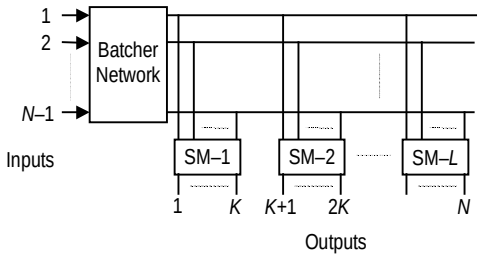


Figure 2.27 The Shared Concentration Output Queueing switch.

2.4.4.3 Output Buffering

The *Shared Concentration Output Queueing* (SCOQ) *switch* was proposed by [ChM93] after observing that the high complexity of the Knockout switch is mainly because the switch needs a bus driver to drive N loads, and because it has a separate concentrator in each of the N output bus interfaces. As shown in Figure 2.27, the SCOQ switch is made up of an (N×N) Batcher

network and L switching modules (SMs). The size of each SM is ($N \times K$) where $K = N/L$. Each SM concentrates and routes the cells destined to one of its K outputs. This results in one concentrator for K output ports instead of one output port as in the Knockout switch. Also, each output of the Batcher network only has to drive L loads. Each SM has output buffers, a group of address filters, and L ($K \times K$) Banyan networks. At most L cells destined to the same output port are routed successfully. The other cells are dropped. The SCOQ switch has been shown to compare well against the Knockout switch in terms of complexity.

2.4.4.4 Internal-Output Buffering

Instead of just using internal buffering, the *Sunshine* switch [GiH91] uses both internal and output buffering, and achieves high performance. As shown in Figure 2.28, the switch uses a combination of a Batcher network and parallel Banyan networks. The Batcher network sorts the incoming cells according to their output addresses and priority levels. Conflicts due to multiple cells destined for the same output port is resolved by the trap network. It selects K highest priority cells destined for each output port in a cell cycle. Unselected cells are recirculated. The concentrator separates the selected cells and the unselected cells, and the selector sends the selected cells to the Banyan networks for routing and the unselected cells to the recirculation buffer. There are K Banyan networks following the selector so up to K cells can be switched to each output port during a cell cycle. The Sunshine switch is shown to have a very low P_{loss}, and to be very robust under bursty traffic if there are large enough number of Banyan networks [GiH91]. Note that if $K = 1$, the Sunshine switch becomes the Starlite switch.

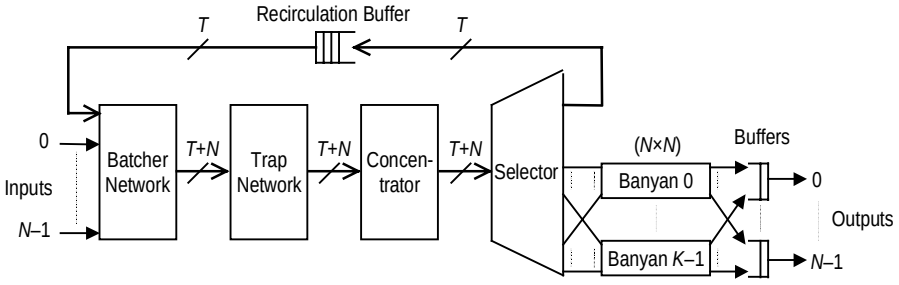


Figure 2.28 The Sunshine switch.

[BeA99] proposed a variation of the Sunshine switch. Their switch uses two input switching fabrics (ISFs), primary and secondary. The primary ISF is of size $N \times N$, and the

secondary ISF is of size $(N/z)\times(N/z)$ where z is a non-zero power of two. The secondary ISF allows a portion of the input lines of the primary ISF to recycle cells. For the same cell loss performance, the switch by [BeA99] requires less hardware than the original Sunshine switch.

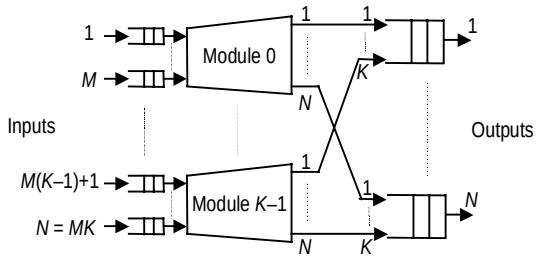


Figure 2.29 Lee's modular switch architecture.

2.4.4.5 Input-Output Buffering

[Le90b] has proposed a modular architecture for large sized switches (Figure 2.29). The inputs are divided into K equal groups, and each group is connected to a switch module of size $(M\times N)$ where $M = N/K$. Each switch module is made up of a cascade of an $(M\times M)$ Batcher sorter, a stack of M binary trees, and K parallel $(M\times M)$ Banyan networks. The Batcher sorter orders the cells in each subset of inputs, and the binary tree partitions these cells into small sub-subsets. In each module, the K Banyan networks then routes the sorted cells of the sub-subsets to the output buffers in parallel. This architecture can be seen as a combination of the BB network (when $K = 1$) and the Knockout switch (when $K = N$). Since the switch modules are interconnected only at the outputs, the switch has characteristics of relaxed synchronization requirements, easy operation and maintenance, and high reliability.

Unlike the above switch that has no means to control output buffer overflow, the switch architecture proposed by [Pa90a, Pa90b] uses mixed input-output buffering with a backpressure method to prevent output buffer overflow. As seen in Figure 2.30, the switch consists of a Batcher network, a Banyan network, and a channel allocation network. Three different networks form the channel allocation network.

1) a $(2N\times 2N)$ merge network with $n+1$ stages of (2×2) SEs;
2) a $(2N\times 2N)$ path allocation network with $L = \lceil \log_2 M \rceil$ stages of adders; and
3) a $(2N\times 2N)$ concentration network with $n+1$ stages of (2×2) SEs.

The size of the Banyan network is ($N2^L \times N2^L$) where only N inputs are used. A probe-ack contention resolution algorithm sets up conflict-free paths for the cells through the interconnection network while insuring that no output buffer overflow occurs resulting in cell losses. This makes it possible to optimize the cell storage capacity at the input and output buffers, and easier to define congestion control procedures.

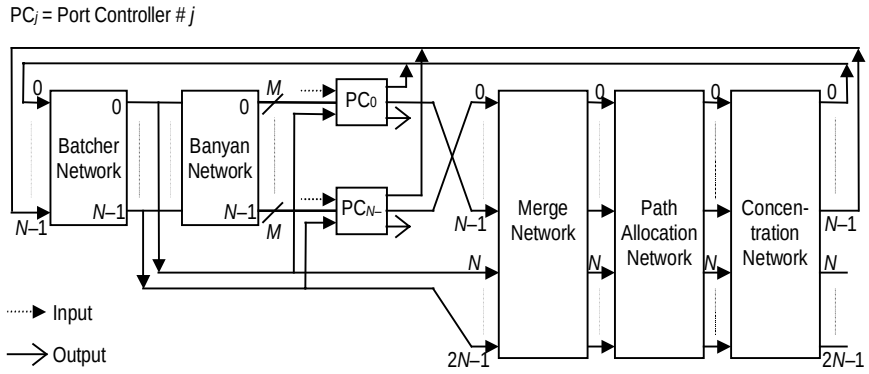


Figure 2.30 Pattavina's input-output buffering switch.

2.4.5 Deflection-Routing

Rather than dropping or buffering a contention loosing cell in a multistage interconnection network, the cell can be misrouted or deflected. In this approach, the misrouted cell has a chance to be routed correctly to its destination in a later stage. The previously mentioned tandem-Banyan switch and dilated Banyan network with re-circulation employ this approach.

2.4.5.1 Folded Shuffle Switch

The Folded Shuffle switch is an input-output buffered switch with a cylindrical interconnection network [CaP91]. The switch has connections between the outputs of the last stage to the inputs of the first stage (Figure 2.31). An ($N \times N$) switch consists of $n = \log_2 N$ stages and N unbuffered (3×3) SEs per stage. Successive stages are interconnected through the perfect shuffle pattern [St71]. All the SEs in a particular row are linked to a corresponding input and output of the switch. Each input buffer must be able to send up to n cells in a cell cycle. Similarly, an output buffer must be able to receive up to n cells in a cell cycle. At the start of a new cell cycle, each SE calculates *cell distances* (distance to destination in number of stages) of all the cells at its internal inlets. If a cell has zero cell distance, the SE sends it to the output buffer via its external

outlet, but if not, the SE sends it to the next stage via its internal outlet according to the shortest path algorithm. If there is a conflict for an outlet at an SE, a *randomly* chosen cell is deflected via an internal outlet. Randomly deflecting cells causes cells to arrive at the output buffers out of sequence, but the switch does not have any measures to correct this problem. An SE may receive a new cell with non-zero cell distance via its external inlet if it has an idle internal outlet. In other words, any cell loss is prevented between the input buffers and the network. To achieve a very small P_{loss} at moderate loads, a small number of input buffers and a large number of output buffers are needed.

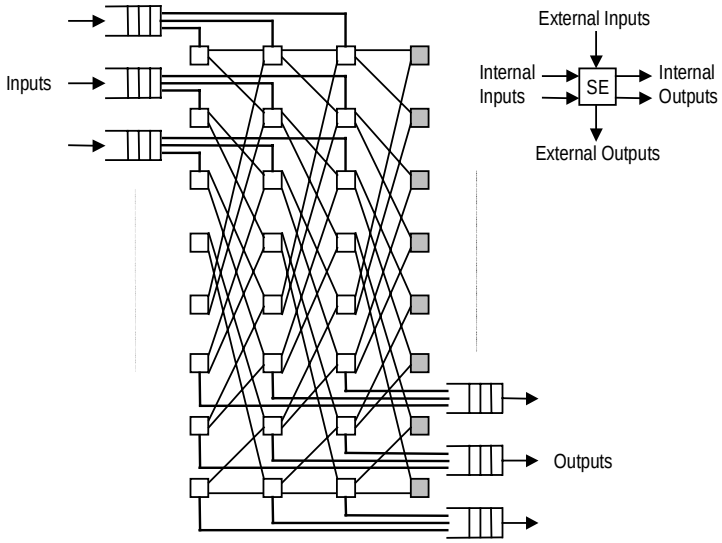


Figure 2.31 An 8×8 Folded Shuffle switch.

2.4.5.2 Shuffleout Switch

Shuffleout switch [DeG91a] also uses the shortest path algorithm with deflection and has successive stages interconnected through the perfect shuffle pattern. As seen in Figure 2.32, it has K ($\geq n = \log_2 N$) stages, each with $N/2$ unbuffered (2×4) SEs. Two outlets of each SE, called local outlets, are connected to the corresponding output ports. Each cell is routed via its shortest path measured in remaining number of stages to destination. If there is a contention, the SE selects the cell that is *closest* to its destination, and deflects the other cell. If a cell does not reach its destination even after K stages, it is dropped. Since up to K cells can reach an output port, some kind of concentration scheme is needed to reduce the system complexity. The switch is robust in nonuniform traffic, but needs a few more stages than the number of stages need for uniform traffic.

Also, the switch is unfair [BaD92]. The cells destined to the central output ports are better served by the network than the cells destined for the peripheral output ports. Unless there are constant delay of $(K - s)\tau$ present on the links between any SE at stage s ($1 \le s \le K$) and the corresponding output port, the sequence of the cells may not be preserved.

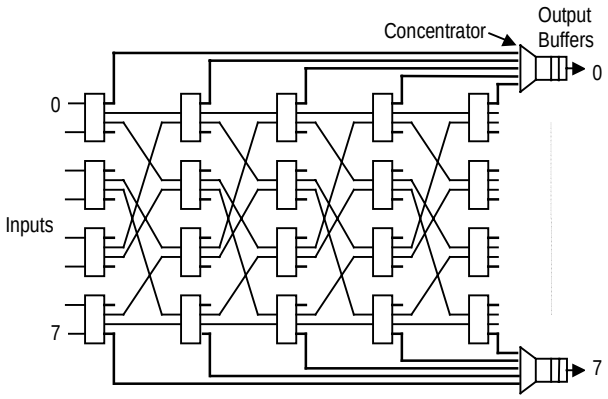


Figure 2.32 An 8×8 Shuffleout switch with K = 5.

Instead of dropping cells, the *closed-loop* Shuffle-out switch recirculates them [DeG91b]. The authors considered two different designs: buffered and expanded. In the buffered closed-loop Shuffle-out switch, an input line and a link from an outlet of an SE in the last stage shares an input of the first stage. The recirculated cells have the priority, and there are input buffers to handle contentions at the input port. In the second design, the recirculated cells are sent to separate, dedicated switch inputs requiring exansion of the network size. For a given P_{loss}, the expanded version needs less number of stages compared to the buffered version at the cost of network expansion. Compared to the *open-loop* switch, both designs, need a smaller number of stages to achieve the same P_{loss}, but in both of the closed-loop designs, cells need to be resequenced at the output port.

The architecture of the *Multi Single-Stage-Shuffling Switch* (MS4) [AwM95a] is similar to the Shuffleout switch. The difference is that the MS4 uses 2×2 SEs and that only the outputs of stages n to K are also connected to the same-number output ports. When compared to the TBS by [ToK91], the MS4 needed much less number of stages to obtain same cell loss rate. As an example, for N = 1024 and $P_{loss} = 10^{-6}$ at full load of uniform traffic, the TBS required 100 stages while the MS4 required only 56 stages [AwM95a]. For $P_{loss} = 10^{-5}$ at 50 percent load of non-uniform traffic, the TBS needed 100 stages while the MS4 needed 43 stages [AwM95a].

When compared to the Shuffleout switch, the MS4 needed a few more stages to achieve the same cell loss rate. However, [AwM95a] claimed that the overall complexity of the MS4 is lower than the Shuffleout switch –e.g., using small size SEs. They also claimed that while the MS4 always preserved the cell sequence, this is not generally the case with the Shuffleout switch.

Another variation of the Shuffleout switch is the I-Cubeout switch by [TzP99]. Their switch gives slightly better performance over the Shuffleout switch with the same number of stages, but the overall hardware complexity is lower than the Shuffleout switch. A variation of the Shuffleout switch with significant performance improvement is an output buffering ATM switch by [Ça99]. It is called *modified shuffle-exchange network* (MSN), is similar to both the Shuffleout and MS4 switches. The main difference is that the MSN has connections to the output ports at every other shuffle-exchange stage. It has K ($\geq n = \log_2 N$) stages, each with $N/2$ unbuffered (2×2) SEs. The main goal of the switch architecture is to reduce the number of the internal conflicts. Each link of every other stage has a filter to route successful packets to their destinations. Both analytical and simulation results showed the MSN to have much lower packet loss rate than that of the MS4 and TBS [Ça99].

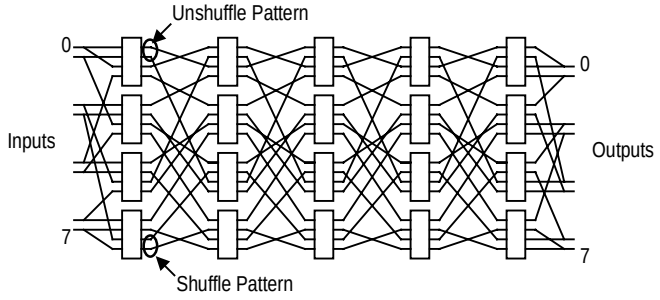


Figure 2.33 An 8×8 Dual Shuffle-Exchange Network with $K = 5$.

2.4.5.3 Dual Shuffle-Exchange Network

[LiL92] has designed a Dual Shuffle-Exchange Network (DSN) from two networks, a shuffle-exchange network (SN) and an unshuffle-exchange network (USN), interleaved together. As shown in Figure 2.33, a DSN is constructed from K ($\geq n = \log_2 N$) stages with $N/2$ (4×4) SEs in each stage. Each SE is connected to the next stage according a shuffling (unshuffling) pattern through its lower (upper) half outlets. Routing bits used in successive stages progress from the most significant bit (MSB) to the least significant bit (LSB) in the SN, and from the LSB to the

LSB in the USN. The cells are routed normally in the SN while the USN routes any deflected (contention losing) cell back to its normal path in the SN in a single step. The cells that reached their destinations are collected and multiplexed by bypass links (not shown in the above figure). Like the Shuffleout switch, the cells that failed to reach their destinations after K stages are dropped, and the cells can reach their destinations out of sequence. The DSN is shown to attain the $N \log_2 N$ lower bound on switch complexity for an arbitrary small P_{loss}.

2.4.5.4 Bridged Shuffle-Exchange Network

[ZaM93a] proposed the Bridged Shuffle-Exchange Network that uses less complex SEs and simpler routing algorithm compared to the DSN (Figure 2.34). A losing cell is sent to the same-position SE in the next stage via the middle link, or "bridge," so that the cell can be routed normally. However, deflection is still needed since all three cells at an SE can request the same outlet. Here, one of the three cells is deflected. Cells are dropped if they did not reach their destinations by the last stage. The Bridged Shuffle-Exchange Network also achieves a small P_{loss} with complexity of $O(N \log_2 N)$ like the DSN. The authors also proposed a closed-loop version of the Bridged Shuffle-Exchange Network where the cells recirculated rather than being dropped [ZaM93b]. It is shown that the closed-loop version needs less number of stages for the same P_{loss} compared to other deflection-routing switches. The authors recommended a re-sequencing method in order to retain the cell sequences.

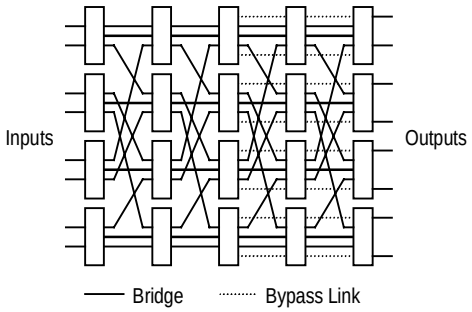


Figure 2.34 An 8×8 Bridged Shuffle-Exchange Network with K = 5.

2.4.5.5 Cyclic Banyan Network

[PaY99] proposed a deflection routing ATM switch called, *Cyclic Banyan Network*. The $N \times N$ Cyclic Banyan switch is based on the $N \times N$ delta network by adding links connecting all SEs

within a stage. These additional links are called the chain links, and provides multiple paths between an input and an output. When a cell experiences link contention, it is deflected via the other normal link or via one of the two chain links. [PaY99] developed fully adaptive routing algorithm to route the cells to their destinations. The Cyclic Banyan had the highest throughput when compared to the Banyan and three different replicated Banyans (two, four, and eight replications). The Cyclic Banyan needed more hardware components than the Banyan and Replicated-2 Banyan, but less hardware components than the Replicated-4 and Replicated-8 Banyans.

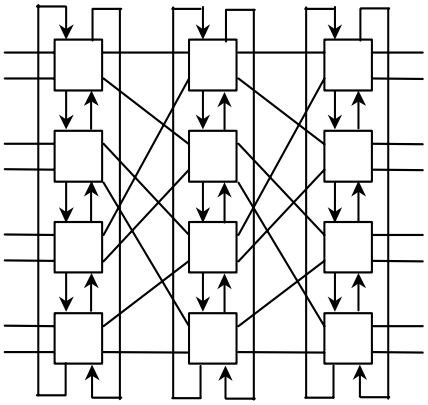


Figure 2.35 An 8×8 Cyclic Banyan Network.

2.5 Summary

This chapter presented an overview of space-division ATM switches proposed in the literature. First, different buffering strategies were discussed. The output buffering has a higher throughput than the input buffering under any traffic model. This disadvantage of the input buffering is due to the Head of Line (HOL) blocking as discussed earlier. Then, different switch architectures using the switch fabrics based on the interconnection networks —mainly, the crossbar, disjoint-path, and Banyan networks —were presented. Different architectures have different performance levels and complexity, and the switches with better throughput or cell loss performance tend to be more complex to implement.

As can be seen above, the majority of the space-division ATM switches in the literature are based on the Banyan networks (or multistage interconnection networks). Most of these switches have a high degree of complexity since they use a multiple numbers of Banyans or high numbers

of stages. Therefore, there is a need for switch designs with low hardware complexity and high performance. Recently, in order to design less complex switches, there has been a move toward switch designs based on another type of interconnection network, specifically, the hypercube, a single-stage interconnection network. This can be attributed to the low complexity, large bandwidth, and inherent self-routing capability of the hypercube.

There are two ATM switches based on the hypercube network in the literature. These switches are shown to have comparable performances but have lower hardware complexity when compared to many Banyan-based switches. A variation of the hypercube, call the folded hypercube network (FHC) is shown to have better performance than the hypercube without much increase in the hardware cost [ElL91, LaE89]. Since ATM switches based on the FHC show promise for outstanding performance with low hardware cost, and this research proposes three new ATM switches based on the FHC. The two hypercube based ATM switches started a fresh approach in designing ATM switches. This study not only continues this novel approach, but introduces new ATM switches with higher performance and lower hardware cost.

3. Hypercube Based ATM Switches and Variations of Hypercube

Hypercube-type networks have received much attention over the years since they offer a rich interconnection structure with high bandwidth, logarithmic diameter, and high degree of fault tolerance. In the literature, two ATM switches have been proposed utilizing these properties of the hypercube. Also, in order to further improve the characteristics and performance of the hypercube networks, some variations of the hypercube structure have been reported in the literature. Section 3.1 discusses the architectures and the performance of two hypercube based ATM switches. Then, Section 3.2 presents different variations of the hypercube networks: the twisted hypercube, the folded hypercube and the bisectional interconnection network. The folded hypercube is discussed in more detail, including its performance since it will be the basis of new ATM switch architectures proposed in this study. Before the FHC is modified into new ATM switches, a simulation model is developed to study its performance. Section 3.3 shows how a simulation model is developed for the FHC and gives the analysis of the simulation results. A summary is given in Section 3.4.

3.1 Hypercube Based ATM Switches

Two ATM switches proposed by [Ma93] and [PaC95] are based on the hypercube network. They provided a different direction in the design of interconnection network based space-division ATM switches. The following sub-sections briefly overview the designs and the performance of the switches by [Ma93] and [PaC95].

3.1.1 HiPower ATM Switch

An ATM switch called HiPower (hypercube-interconnected photonic ATM cell switch with electronic routing network), proposed by [Ma93], offers a good combination of photonics and electronics by adopting a two-layered network in its internal structure. HiPower uses a hypercube network along with detour routing, and has been designed to derive the maximum performance from the two-layered structure. The switch uses a two-layered, *d*-dimensional hypercube structure with 2^d nodes (Figure 3.1). Each node pairs an *input line* and an *output line* on which ATM cells are transmitted from/to adjacent ATM switches. Each node also adjoins *d* nodes whose *node*

addresses differ in one and only one bit from that of the node via bi-directional *inter-node links*. The optical layer devotes its high bandwidths to transport the high-speed cells, while the electrical layer devotes its high functionality to routing control. The routing control information part of an ATM cell is copied to form a small *electrical cell* that travels in the electrical layer. The larger original cell, called an *optical cell*, always travel in the optical layer via the same route as the corresponding electrical cell so that the electrical layer in a node controls optical cells according to the control information given by electrical cells.

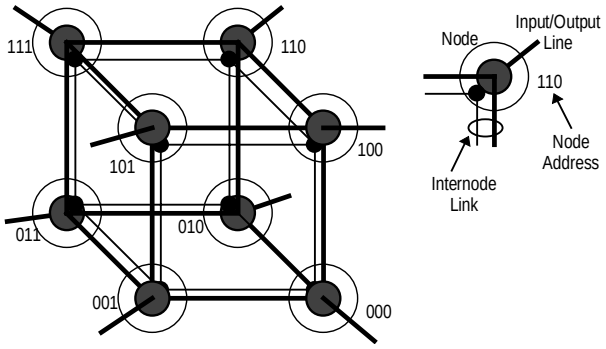
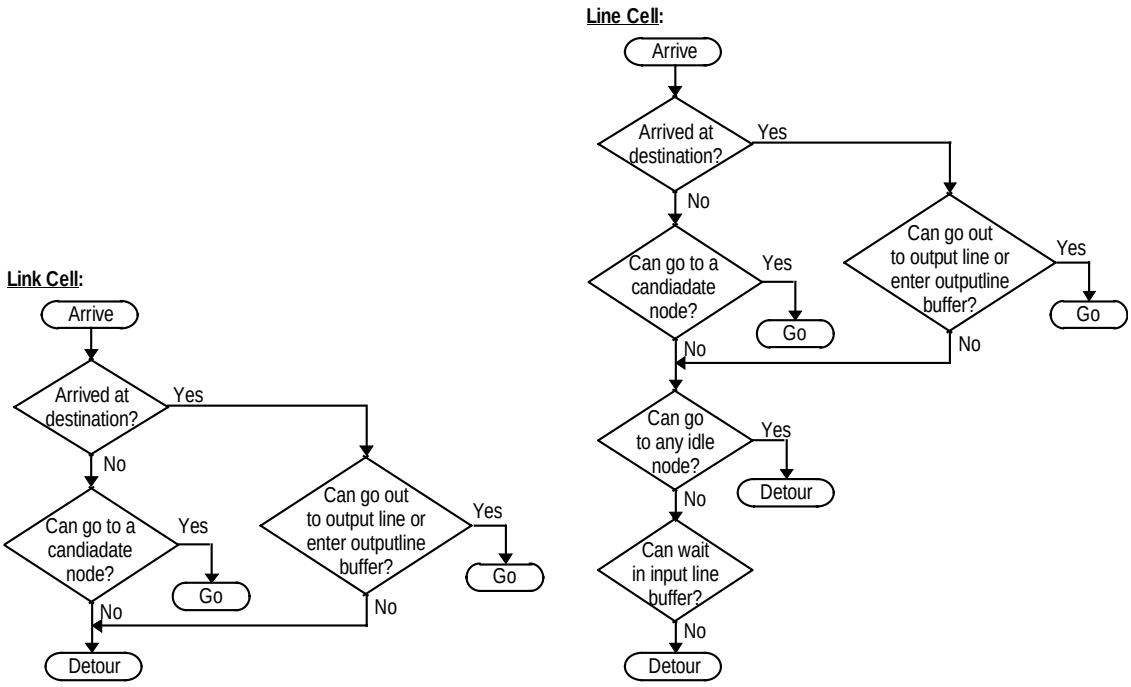


Figure 3.1 Three-dimensional hypercube network [Ma93].

When a node receives a cell from an input line or from one of the input internode links, the node compares its own address with the cell's destination address and finds among adjacent nodes some candidates to transfer. Link blocking occurs when multiple cells in a node have the same candidate and some cells can not go to their candidate nodes. HiPower uses *detour* routing (also known as *deflection* routing in the literature) to remove buffer memory for link blocking from the interconnection network and to reduce the amount of output buffers on the output lines. Since the hypercube network has an inherent routing capability regardless of its path history, it is easy to introduce detouring. In the detour routing scheme, blocked cells are forced to take detour paths. Thus, no buffer memory is needed for link blocking and the amount of output buffers can be reduced. To obtain a sufficiently small cell loss rate, HiPower has a certain size of buffer memory at each input line for the line cells which encounter link blocking and find no idle link for detouring. The link cells have priority over the line cells. Since more than one cell may arrive at its destined output node in a time unit, HiPower also has buffer memory at each output line. When a newly arriving line cell finds the input line buffers are full, it is aborted. On the other hand, when a line cell or a link cell arriving its destination finds the output buffers are full, it is routed back to

the network. Figure 3.2 shows the cell routing flow diagram in HiPower. The size of the input line buffers are designed so that the cell abort rate becomes extremely small, while the overflow rate of the output line buffer can be larger. Also, there should be enough number of buffers at the output line so that the number of the overflowed cells is negligible compared with the total number of cells in the switch.



(a) Link cell routing (b) Line cell routing

Figure 3.2 Cell routing flow diagram in HiPower [Ma93].

With detouring, there is no need for link buffers for cells that experience link blockings. However, detouring increases the delay through the switch and the inter-node link usage that results in more frequent link blockings. In order to reduce the possibility of detouring, a good routing algorithm is needed so that as many cells as possible can go to their candidate routes. [Ma93] proposed four different routing algorithms to determine a preferable set of routes which gives less link blockings. These four algorithms are based on two independent sorting-based strategies that determine a preferable set of routes, which gives less link blockings, and these two are as follows:

- Distance to Destination (DD) method: The cell with the shortest distance to its destination is routed first because the shortest DD cell has the fewest candidate routes and is most

likely to encounter a link blocking. The DD is the number of unmatched bits when a node's own address and a cell's destination address is compared.

- Number of Request (NR) method: The output link with the fewest requests is chosen first and a cell for the chosen output link is determined among the cells offering requests to be sent through it. This is because the least-requested output link is likely to lose the requests and to be left unused.

The four proposed algorithms for HiPower are as follows.

1. The pure DD method: Independent of the NR for output links and randomly chooses one of the available candidate links for a link cell.
2. The pure NR method: Independent of the DD of each cell and randomly chooses one of the requesting cells for an output link.
3. The DD-NR method: Routing is based on the DD method and the NR of the output link is considered when the cell has multiple candidate output links.
4. The NR-DD method: Routing is mainly based on the NR method and the DD is considered only when the chosen output link has multiple cells requesting that link.

The performances of HiPower under the uniform traffic load for these four algorithms and the random sorting method has been studied by computer simulations [Ma93]. The performance was measured in terms of delay and the cell loss rate in the switch. The results indicated that all of the four algorithms provide remarkable improvements over the random method from the viewpoint of delay and cell loss. Also, the DD-NR method gave the least delay while the NR method gave more delay than the other methods. A 7-dimensional HiPower had no input buffer overflow for all five routing algorithms at a loading of 0.8 with two input line buffers and eight output line buffers. No input buffer overflow means that no cells were lost. At a loading of 0.9, the DD method had the best cell loss rate of 1.5×10^{-5} while the random method had the worst cell loss rate of 2.4×10^{-4} for a 7-dimensional HiPower with two input line buffers and eight output line buffers.

Matsunaga also proposed priority schemes to reduce the variance of delay which caused cell-arrival disordering at end terminals [Ma93]. When several cells have a same value of DD in sorting according to DD, the number of cells having many detourings will be reduced if the cell that has experienced more detourings up to then is given priority over the other cells that had fewer detourings. Two priority sorting schemes were evaluated.

1. Multiple-Level scheme: A cell that has experienced detourings *n*-times is given a priority level *n*.
2. Two-Level scheme: A cell is given a priority if it has experienced detouring.

The simulation results showed that the priority sorting caused no difference in the average number of hops, but reduced its variance, especially in the infinite buffer models. Also, the multiple-level scheme reduced the delay variance more than the two-level scheme in the finite buffer models, but the multiple-level scheme is more complex to implement. However, the difference between the two schemes diminishes as the number of output line buffers increases and, in the infinite buffer model, the multiple-level scheme and the two-level scheme equally reduced the delay variance. With two-level scheme, the cell loss rate is 2.4×10^{-4} with eight output line buffers and zero with sixteen output line buffers (7-dimensional HiPower with two input line buffers at a load of 0.9). This suggests that HiPower only needs a few input line buffers for a small cell loss rate if it has a moderately large number of output line buffers. All the simulations were done under the uniform cell arrival probability and uniformly distributed destinations. The author, however, did not provide an algorithm for re-ordering cells that may arrive out of sequence at the destination node, and also, did not consider the throughput performance of the switch.

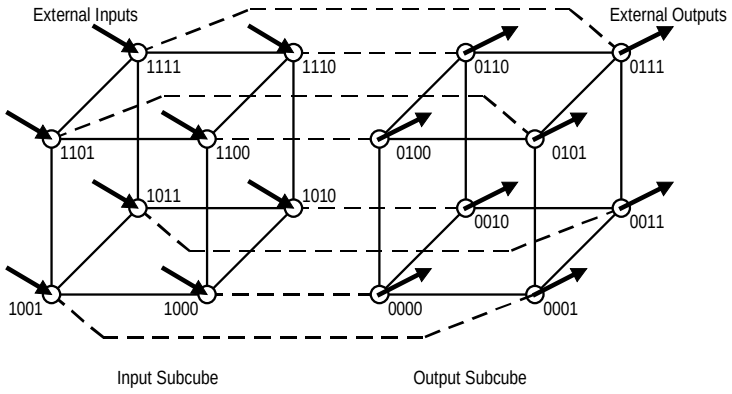


Figure 3.3 Organization of an 8×8 switch [PaC95].

3.1.2 A Hypercube ATM Switch with Shared Input and Dedicated Output Buffers

Another ATM switch proposed by Pao and Chau uses an (*n*+1)-dimensional hypercube to implement an $N\times N$ switch, where $N = 2^n$ [PaC95]. The hypercube is divided into two halves, the output subcube consisting of the nodes numbered from 0 to $N-1$, and the input subcube consisting of the nodes numbered from N to $2N-1$ (Figure 3.3). Each input (output) node has an external

input (output) link. The cells arriving at the input ports are routed towards their destinations in a store-and-forward (SAF) manner. To implement the SAF routing function, each node contains n+1 SAF buffers. Also, each input (output) node has an input (output) buffer of size B_i (B_o). The switch uses deflection routing to minimize the number of buffers needed.

An age tag and a sequence number are attached to each cell upon arrival from the external link. The age of a cell, initially set to zero, is incremented by 1 in every cell time until it is stored in the output buffer of the destination node. Each input node maintains N sequence numbers, one for each output port. When a cell arrives at an input port, it will be either scheduled to be sent to a neighboring node or stored in the input buffer if empty space is available, otherwise it is dropped. In each cell time, the input node selects the n+1 *oldest* cells among the cells in the input buffer and the set of cells it received from its incoming links. The oldest cell is given the highest priority in choosing the communication link to reduce its distance from the destination. The selection of cells is based on the age of the cells rather than the destination ports, and as a result, the design does not suffer from the HOL blocking problem as in the conventional input or input-output buffering architecture. The selected cells will be sent to the neighboring nodes, while up to B_i of the remaining cells will be stored in the input buffer and wait for the next cell time. It is possible that a younger cell is being routed farther away from its destination. However, as the cell gets older, it will be given higher priority and be able to progress towards the destination node. The cells entering the hypercube at node u may be temporarily stored in the input buffer of another input node v. Hence, the SAF buffers and the input buffers behave as distributed shared-buffers.

The output node performs two functions, (1) routing of the arriving cells, and (2) selection of a cell from the output buffer to be sent to the external link. The routing function of an output node for the cells that have not reached their destinations is similar to that of an input node. The cells arriving at an output node are sorted according to their ages, and the oldest cell has the highest priority in choosing a link. When a cell arrives at its destination, it is routed in the following order.

1. Dropped if its sequence number is smaller than the expected value,
2. Stored in the output buffer if it is not full,
3. Derouted to other nodes if the age of the cell is not greater than the threshold value, *AgeThreshold*; if not, it is dropped.

AgeThreshold is used to prevent any unfair sharing among the virtual queues. If buffers are not shared fairly, shared buffers could actually perform worse than buffers that are not shared at a high

applied loading [Li94]. Under heavy traffic load, cells originated from a few input ports may use up all the buffers and affect the traffic flow from other input ports. To eliminate this problem, the congesting cells should be removed if they stayed in the buffer more than *AgeThreshold*. This would allow the cells from other input ports to be queued in the output buffer and not be derouted constantly. The optimum value of *AgeThreshold* depends on the overall traffic condition of the switch. When the input buffers start to drop new cells, the system throughput is near its optimum. When more cells are being dropped at the input, the value of *AgeThreshold* should be decreased. In contrast, when no cells are dropped at the input for a set period, the value of *AgeThreshold* can be increased gradually. The authors recommended a minimum value of *AgeThreshold* to be about four plus the network diameter which is *n* for an *n*-dimensional hypercube.

Since the cells may arrive at the destination out of sequence, the output node is required to restore the original sequence when sending cells to the external link. However, since the routing function always gives higher priority to older cells, the degree of out of sequence arrivals would not be severe. The simulation study showed that the number of cells transmitted out of sequence was less than or equal to four out of 10 million cells injected into the network. The cells admitted to the output buffer are re-sequenced before they are sent out to the external link in the following manner. The output buffer of each output node is actually made up of *N* virtual queues, one for each input line, and each virtual queue maintains an expected sequence number of the next cell to be transmitted. When a cell arrives at its destination node, it enters the corresponding virtual queue, and the cells in each virtual queue are ordered according to their sequence numbers. A virtual queue is active whenever a cell occupies it. A timeout scheme is used for each virtual queue to prevent cells waiting in the queues forever. If the sequence number of the HOL cell of a virtual queue is equal to the expected sequence number, or if it has been queued for more than the timeout period or *TimeoutH*, the cell is enabled for transmission with *high* priority. If the sequence number of the HOL cell is greater than the expected value, and if that cell has been in the queue a period less than or equal to *TimeoutH* but greater than *TimeoutL*, then the cell is enabled for transmission with *low* priority. The cell that has been queued for a longer period of time has the preference among the enabled cells with the same priority. If there is a cell in the *high* priority class, then the cell with the longest waiting time in this class is selected. If there is no cell with *high* priority and the output buffer is full, then the cell with longest waiting time with *low* priority is selected. The new expected sequence number for the corresponding virtual queue is now one

plus the sequence number of the selected cell. The values of *TimeoutH* and *TimeoutL* are closely related to *AgeThreshold*. In their simulation study, the authors set *TimeoutH* = *AgeThreshold* and *TimeoutL* = 0.5 × *TimeoutH*.

The simulations were done for both uniform and bursty traffic models for a 32×32 switch with different combinations of B_i and B_o with $B_i + B_o = 48$. When $B_i = 2$ and $B_o = 46$, the best cell loss probability was achieved ~~a~~about 0.006 under uniform traffic at a load of 0.95 and about 4.0×10^{-4} under bursty traffic at a load of 0.5. The authors also compared the performance of their switch with an *idealized* input-output buffered non-blocking switch architecture by [BaM94], and showed that their design achieved better saturated throughput and lower cell loss probability. The authors concluded that their switch's improved performance was due to the following factors:

1. The design does not suffer from the HOL blocking problem.
2. The SAF buffers and the input buffers behave as distributed shared-buffer that effectively smooth out uneven traffic.
3. The self-regulating property of the routing algorithm prevents the unevenly distributed traffic at a few ports from congesting the network and blocking the other traffic paths.

In general, ATM cells can be divided into two general classes of traffic: time-sensitive and time-insensitive classes. Because of this, the routing algorithm of a switch should give the time sensitive cells higher priority than the time insensitive cells. The switches by [Ma93] and [PaC95], however, have not addressed the issue of time sensitivity of ATM cells. The new ATM switches that will be proposed later incorporate this issue of time sensitivity of ATM cells.

3.2 Variations of the Hypercube

Many interconnection networks such as the tree and multidimensional mesh networks can be embedded in the hypercube [DeJ86, SaS88]. Popular networks such as PM21, Illiac, and shuffle-exchange can be emulated by a hypercube with no hardware overhead [Si85]. Another appealing feature of the hypercube is its homogeneity and symmetry. In contrast with the tree or shuffle-exchange networks, no node or edge plays a special role in the hypercube. Some variations of the hypercube structures have been reported in the literature. A general class of hypercube structures is presented in [BhA84]. In that, the interconnection is based on a mixed radix number system and the technique results in a variety of the hypercube structures for a given number of processors, namely *N*. The cube-connected cycles [PrV81] is another variation of the hypercube in

which the number of connections per processor is reduced to 3. These structures, however, possess either a large diameter, or a small diameter at the expense of a large amount of extra hardware. On the other hand, in many I/O intensive parallel processing applications, an efficient and cost-effective interprocessor communication is essential. The demand for further reduction of diameter and traffic congestion, with little or no hardware overhead, motivated new variations of the hypercube.

3.2.1 The Twisted Hypercubes

Several researchers proposed variations of the hypercube that do not require any extra hardware to minimize the diameter of the hypercube which is n for an n-dimensional hypercube. These new architectures are generally known as the *twisted hypercubes* (or *twisted cubes*) and, depending on the proposed architecture, their diameters, which are less than that of the hypercube, vary. These twisted cubes are achieved by twisting a number of carefully chosen edges (Figure 3.4). The twisted cube proposed by [EsN88, EsN91] has a diameter $n-1$, and [ChC93, Ch95] proposed their version of the twisted cube with a diameter of $2n/3$. The two twisted cubes of [CuL91, CuL92, CuL93, CuL95] have a diameter of either $(n+1)/2$ or $(n+2)/2$ for $n \geq 4$ depending on which of the two is considered. Finally, the architectures proposed by [AbP89, EfB88, Ef89, Ef91, HiK87, SiG95] have a diameter of $(n-1)/2$.

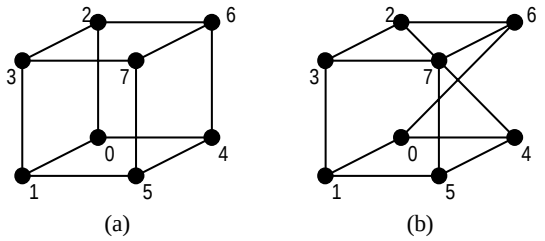


Figure 3.4 (a) A binary 3-cube and (b) a twisted 3-cube.

Even though theses architectures achieve smaller diameters compared to the hypercube without any extra hardware, the symmetry of the hypercube is lost and the edge selection process is not straightforward except for the following two architectures. [EsN88] twists only a pair of edges so their architecture can be achieved easily. [SiG95] developed an edge selection algorithm that creates a twisted cube without loosing the symmetry, but the algorithm itself is very complex, and the routing algorithm is also complicated and has a complexity of $O(n^2)$. With the exception of

the architectures of [Ef89, Ef91] whose routing algorithm also has a complexity of $O(n^2)$, the routing algorithms of other architectures have a complexity of $O(n)$. There exists the possibility that the twisted cube may perform worse than the hypercube because of its asymmetry and because its smaller distances may lead to more congestion. [AbP89] showed by simulation that the twisted cube of [HiK87] did perform worse than the hypercube. However, as [CuL91, CuL92, CuL93] showed, some of the twisted cubes did not suffer from additional congestion even with asymmetry in their architectures. In order to reduce the diameter of the hypercube while keeping the symmetry of the network and keeping the extra hardware at a minimal, [GhB89] proposed the *bisectional interconnection network* and [ElL91. LaE89] proposed the *folded hypercube* (FHC).

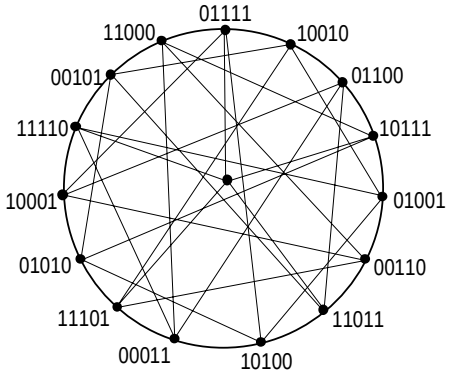


Figure 3.5 Bisectional network with $d = 5$, $N = 16$ (with Hamiltonian cycle) [GhB89].

3.2.2 The Folded Hypercube and Bisectional Interconnection Networks

The bisectional interconnection network (BIN) proposed by [GhB89] is based on a class of codes, called *binary even-weight codes*. The topology is defined in terms of a parameter d, which will represent the degree of the network and will be considered to be odd. The number of nodes in the network is equal to 2^{d-1} which is just half of that in a binary d-dimensional cube. The topology is constructed as follows.

- Consider a binary code of odd length d, having codewords with all possible even weights. Represent each codeword as a node of a graph.
- Connect any two nodes together if the *Hamming distance* between their respective codewords is $d-1$.

The above procedure generates a network that is regular with a degree of d (odd), and that has a diameter of $k = (d-1)/2$. This network is shown in Figure 3.5.

According to the adjacency rule, the codewords of two neighboring nodes have a Hamming distance of *d*=1 meaning that their codewords have exactly one bit in common, either 0 or 1. The network has an algorithm for distributed self-routing of messages, without any routing tables or databases, etc., like the hypercube. However, the BIN's routing algorithm is not as simple as the hypercube's routing algorithm.

Latifi and El-Amawy proposed a new hypercube-type structure, the *folded hypercube* (FHC), which is basically a standard hypercube with some extra links established between its nodes [ElL91, LaE89]. The hardware overhead is almost 1/*n*, where *n* is the dimension of the hypercube, which is negligible for large *n*. A folded hypercube of dimension *n*, FHC(*n*), can be constructed from a standard binary *n*-cube by connecting each node to a node that is farthest from it (Figure 3.6). Every node in an FHC is connected to *n* nodes at Hamming distance 1 and one node at Hamming distance *n*. The latter node is reached via a *complementary link*. A "regular" hypercube link is labeled *i* (it connects two nodes whose addresses differ in only the *i*th bit) and the complementary link is labeled *c*. The FHC is regular with degree *n*+1, and has a diameter of $\lceil n/2 \rceil$ where $\lceil x \rceil$ denotes the smallest integer greater than or equal to *x* [ElL91, LaE89].

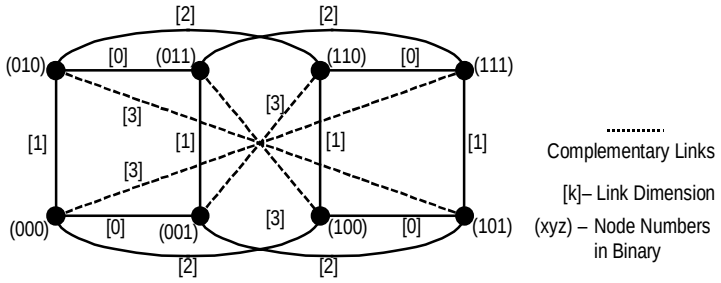


Figure 3.6 The structure of an FHC(3) [ElL91].

The shortest distance between two nodes in a hypercube is the same as the Hamming distance between them, or ||*a*||. However, in an FHC, this is not true since two nodes can have a Hamming distance of *n* when the diameter is only $\lceil n/2 \rceil$. In an FHC(*n*), the shortest distance between any two nodes is min(||*a*||, *n* –||a|| + 1). From this, the following useful property of FHC is stated by [ElL91, LaE89]:

$$\text{dist}(u,v) = \|a\| \qquad \text{for } \|a\| \le \lceil n/2 \rceil \tag{3.1}$$

$$\text{dist}(u,v) = n - \|a\| + 1 \qquad \text{for } \|a\| > \lceil n/2 \rceil \tag{3.2}$$

for every pair of (u,v). This means that in the routing algorithm, the shorter path between two nodes can be selected. This is an important feature that avoids unnecessary communication delays and message congestion.

Like the BIN, the FHC has the self-routing capability, and the authors gave the following algorithm that performs the routing between any pair of nodes, namely u and v. Let $a(u)$ and $\|a(u)\|$ denote the binary address of node u and the number of 1s (Hamming weight) in $a(u)$, respectively. Also, let $\oplus$ denote the *bitwise Exclusive-or* (*XOR*) operation on binary numbers.

- ***One-to-One Algorithm*:** [ElL91, LaE89]

  input: $a(u)$, $a(v)$;

  Begin

  $a(w) = a(u) \oplus a(v)$;

  If $\|a(w)\| \leq \lceil n/2 \rceil$ **Then**

  Route the message sent from u via a path composed of links with labels corresponding to bit positions which are 1's in $a(w)$.

  Else

  Send the message to u' via the c-link; route the message via a path composed of links with labels corresponding to bit positions which are 0's in $a(w)$.

  Endif.

  End.

The reasoning behind this algorithm is as follows. Without loss of generality, let $\|a(w)\| = r$ and bits $w_{r-1}\cdots w_1 w_0$ are all 1's and bits $w_{n-1}\cdots w_{r+1} w_r$ are all 0's. If $r \leq \lceil n/2 \rceil$, a path is formed between u and v which is composed of any one or $r!$ permutations of $w_{r-1}\cdots w_1 w_0$. In the case of $r > \lceil n/2 \rceil$, a c-link must be used somewhere along the path. It follows that the shortest path in this case is any permutation of $(c, w_{n-1}\cdots w_{r+1} w_r)$. In forming a path between u and v, it does not matter where to use a c-link, for the length of the path so obtained is always the same.

The above algorithm can be re-written to be performed at each node so that messages are routed dynamically rather than along a set path. Here is the modified algorithm with $a(u)$ denoting the address of the node performing the routing and $a(v)$ denoting the destination address of a message to be routed.

- ***Dynamic One-to-One Algorithm at Each Node*:**

  input: $a(u)$, $a(v)$;

  Begin

  $a(w) = a(u) \oplus a(v)$;

  If $\|a(w)\| \leq \lceil n/2 \rceil$ **Then**

  Send the message via any one of the *i*-links that corresponds to a bit position that is 1 in $a(w)$.

  Else

  Send the message via the *c*-link if it is free; if not send the message via any one of the *i*-links that corresponds to a bit position which is 0 in $a(w)$.

  Endif.

  End.

Sending the message via the *c*-link or via any *i*-link makes the algorithm dynamic. This is possible since it does not matter when the *c*-link is used and since the length of the path so obtained is still the same as the original algorithm.

In addition to the smaller diameter of the FHC compared to the hypercube, the authors showed that the average distance[3] in the FHC is at least 15% less than that of the same size hypercube. The cost factor[4] for the FHC is also appreciably less than that of the *n*-cube and tends to half for large *n*. Finally, the FHC saturates for higher values of network utilization compared to the hypercube due to the complementary links through which a message residing in one node can reach the farthest node in one hop without having to traverse *n* links (which is the case in the *n*-cube). For a network size of n=10, the hypercube has about 60% higher queuing time delay than the FHC at network utilization of γ=0.20. At γ=0.25, hypercube has already saturated while the time delay of the FHC has increase only about 40% compared to its time delay at γ=0.20. Finally, [La90] has shown that the FHC is more fault-tolerant than the hypercube. The mean time to failure is at least 22% better than that of the conventional hypercube.

---

[3] The average distance is the summation of distances of all nodes from a given source node over the total number of nodes minus one.

[4] The cost factor can be defined as the product of the diameter and degree of the node (i.e., the number of links per node) for a symmetric network.

3.3 Creating and Validating the FHC Simulation Model

This section shows how a simulation model is developed for the FHC using OPNET Modeler, and gives the analysis of the simulation results. For comparison, the hypercube is also modeled and simulated. Finally, the simulation results of the two network models are compared to the published analytical results as verification. This simulation study is also summarized in [PaD99].

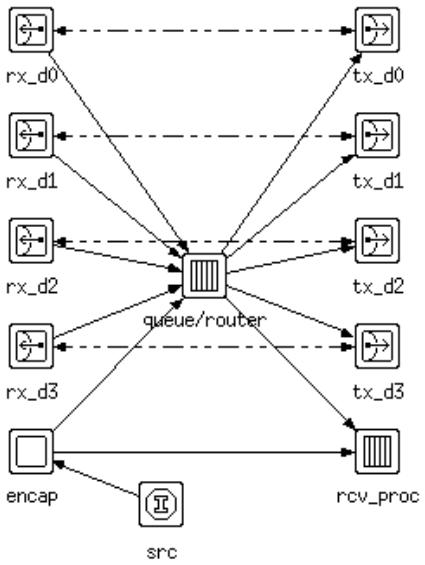


Figure 3.7 A node of FHC(3) modeled in OPNET.

3.3.1 Modeling the FHC with OPNET

The FHC is modeled using OPNET Modeler. OPNET is a network modeling and simulation tool that provides an environment for analysis of communication networks [OPN99a]. OPNET Modeler provides a three-layer modeling hierarchy, the network domain, the node domain, and the process domain. The network domain, the highest layer, allows definition of network topologies. The node domain is the second layer, and allows definition of node architectures or data flow within a node. The process domain allows the specification of logic or control flow among components in the form of a finite state machine.

Figure 3.7 shows how a node of FHC(3) is modeled in OPNET. rx_d0 to rx_d2 (tx_d0 to tx_d2) modules are receivers (transmitters) that are connected to *i*-links. rx_d3 (tx_d3) module is a receiver (transmitter) that is connected to a *c*-link. src module is a cell generator that generates

cells using the given interarrival time, *p*, and the exponential probabilty function. It can also be modified to generate bursty traffic.

encap module generates destination addresses using either the uniform or normal distribution and tags the generated address to each cell received from src module. If a cell is tagged with an address that is the same as the particular node's address, it is passed on to rcv_proc module. If not, it is sent to queue/router module to be queued and routed toward its destination. These two modules, src and encap, model an external input line connected to each node while rcv_proc module models the external output line. rcv_proc module destroys the cells that have reached their destination to free up the memory. This destroying function is equivalent to sending the cells via the external output line. Finally, queue/router module represents the store-and-forward (SAF) buffers and the router that performs the Dynamic One-to-One Routing Algorithm.

The functions or processes of encap, queue/router, and rcv_proc modules are written in *Proto-C* of OPNET Modeler. Proto-C is based on a combination of state transition diagrams (STDs), a library of high level commands known as *kernel procedures*, and the general facilities of the C programming language. Figure 3.8 shows an example of STDs, the STD of encap module. When a simulation starts, init state is invoked, and it initializes the distribution function which is used to generate destination addresses. The process waits in the idle state until a cell arrives from src module. When a cell arrives, xmt state is invoked. In this state, a destination address is generated and tagged to the received cell. The cell is, then, either sent to queue/router or rcv_proc module depending on its destination address. The process returns to init state and waits for another cell.

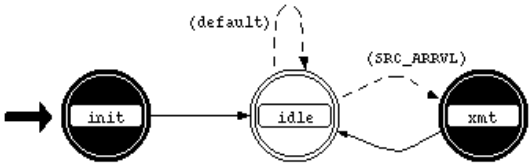


Figure 3.8 The state transition diagram (STD) of **encap** module.

Figure 3.9 shows how an FHC(3) is modeled in OPNET. The eight nodes are named from node_0 to node_7 as shown. The collct_stat node is used to collect the statistics and performance parameters, such as delay through the switch, average distance, throughput, etc.,

during and after the simulation. The double-headed arrows represent the *i*-links and the *c*-links. As mentioned above, the external input and output lines are modeled within each node. The link capacity is set to one cell per time unit where each cell is defined to be 53 bytes long. This cell length is chosen since 53 bytes is the length of the the ATM cells defined by CCITT.

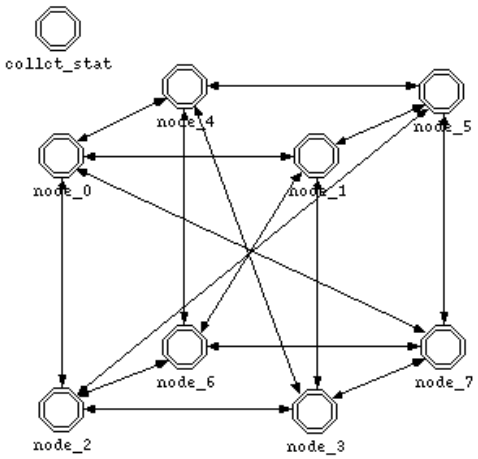


Figure 3.9 A FHC(3) modeled in OPNET.

The node and network models for the different sizes of the FHC are generated by using the External Model Access (EMA) package of OPNET. The EMA package allows an OPNET user to access a model without using the opnet graphical editors [OPN99b]. Accessing a model means creating a model, modifying a model, and extracting data from a model. One of many usuages of EMA package is creating large models that requires many objects and interconnections. Since EMA package allows algorithmic specification using a programmatic interface, it is especially useful when large models have regular structure like the FHC or hypercube. Two EMA application programs are written to create node and network models of the FHC and hypercube of various sizes, from FHC(3), 8 nodes, to FHC(10), 1024 nodes.

3.3.2 Simulation Results

In simulating the FHC message traffic, the uniform distribution is used to generate destination addresses to study the network performance under different conditions. If the distribution of destination addresses is not uniform, some parts of the network will be more congested than others, contributing to a decrease in throughput. Also, the packets are generated by

N independent random processes, one for each of *N* nodes. These assumptions lead to an analysis of the maximum performance that a network can support [DiJ81].

In addition to the uniform distribution, the normal distribution is used to generate the destination addresses. Even though the uniform destination address distribution gives the maximum throughput of a network, it is common in real world applications where one or few destinations may be more favored by the sources. To analyze the network performance in a more practical situation, the normal distribution is also used to generate destination addresses or hot-spots. Hot-spots cause a disproportionate number of packets to be routed to a subset of the destinations [DeC89, PfN85]. This is done by generating packets over a range of destination addresses with a preferential concentration around a mean destination address.

For the simulations with hot-spots, packet destinations are chosen using a normal distribution with a specified mean and standard deviation. The mean is chosen randomly, but the value chosen for the standard deviation is one-quarter of the network size, based on the study by [Pa94]. When the standard deviation was chosen to be less than one-quarter of the network size, some destinations were never chosen. When the standard deviation was chosen to be much greater than one-quarter of the network size, the range of the destination addresses favored by the address generator were too wide and did not qualify as a good model of hot-spots. When the standard deviation was about one-quarter of the network size, the address generator created a distribution that covered the whole range of addresses, from 0 to $N-1$, with a concentration narrow enough to be modeled as a hot-spot.

The simulations are done for eight network sizes (n=3 to n=10) at eight different loading factors. At 100% load, one cell is generated at every time unit in every node. The simulation results are similar to the analytical results of [LaE89, ElL91]. The average distance figures of the analytical and simulation results are compared in Table 3.1. As shown in Figure 3.10, the average distance in the FHC(3) is about 20% less than that of the same size hypercube for both uniform and normal distributions. The difference increases to about 22% as the network size grows to n=10. The average distance is given in terms of number of hops ~~tr~~aversing a link from a node to an adjacent node equals to one hop.

Table 3.1. Average Distance Figures of Analytical and Simulation Results.

	Dim (*n*)	3	4	5	6	7	8	9	10
[LaE89, ElL91]	Cube	1.6	2.1	2.6	3.1	3.6	4.1	4.5	5.0
	FHC	1.4	1.6	2.1	2.5	2.9	3.3	3.9	4.2
Simulation	Cube	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
	FHC	1.2	1.6	2.1	2.4	2.9	3.3	3.8	4.1

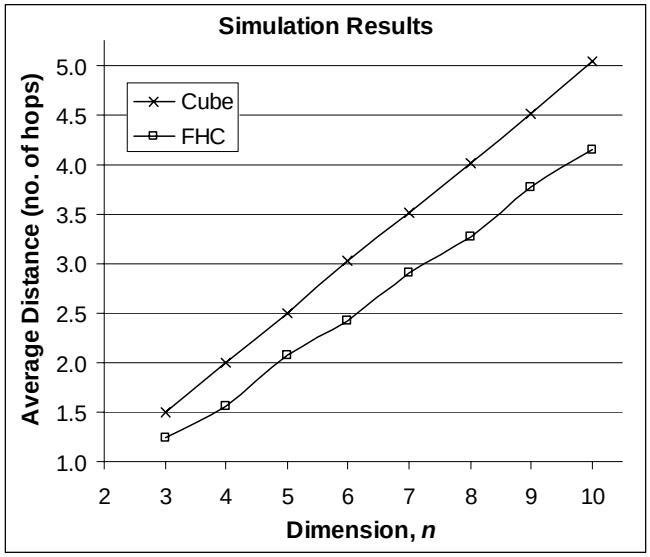


Figure 3.10 FHC vs. Hypercube – average distance.

Figure 3.11 shows the end-to-end (ETE) delay comparison between the FHC and the hypercube. ETE delays are given in terms of unit time, the time it takes for a cell to traverse a link. The results are for n=10 and under uniform distribution. As it can be seen in the figure, the cells in the FHC experience 20% lower ETE delay than the cells in the hypercube until the networks start to saturate at around 80% load. As the network saturates, the performance advantage of the FHC diminishes.

The performance advantage of the FHC increases under the normal destination distribution. As shown in Figure 3.12, the cells of the FHC experience about 20% (10% load) to 26% (80% load) lower ETE delay compared to the cells of the hypercube. Also, the FHC saturates at a higher load than the hypercube. The hypercube has begun to saturate around 80% load with an ETE delay of 14.9, but the ETE delay of the FHC is still about 14.6 at 90% load. Other network sizes have similar results. These results show that the FHC can handle heavy traffic much better than the hypercube, and that the FHC would be a good infrastructure for new ATM switches.

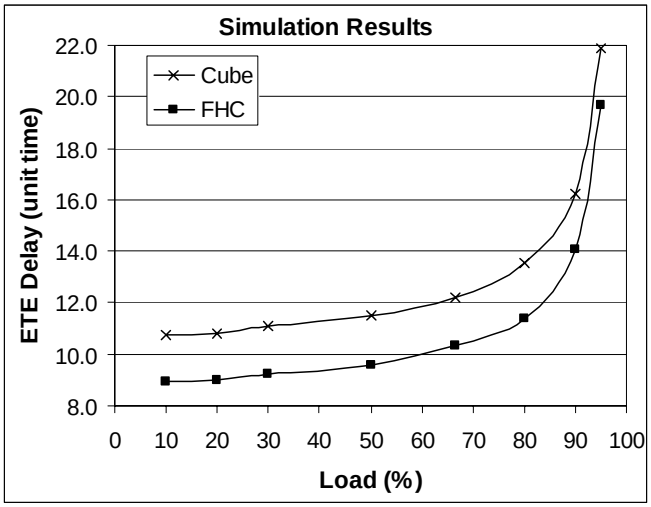


Figure 3.11 FHC vs. Hypercube – ETE delay (*n*=10; uniform).

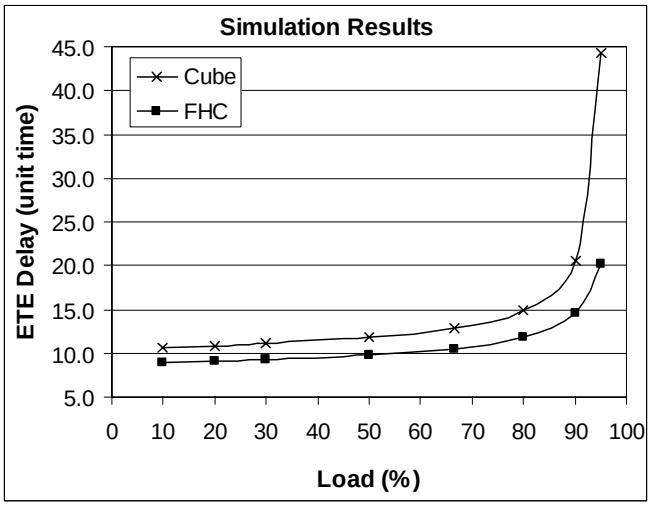


Figure 3.12 FHC vs. Hypercube – ETE delay (*n*=10; normal).

3.4 Summary

The first section of this chapter discussed two ATM switches that are based on the hypercube network. These two switches have started a new direction in the design of the interconnection network based space-division ATM switches. Then, three different variations of the hypercube networks were given. They were the twisted cube, folded hypercube (FHC) and the bisectional interconnection network (BIN). Many different designs of the twisted cube have reduced the diameter of the hypercubes without any extra hardware. However, in doing so, the twisted cube lost the symmetry of the hypercube network. The FHC and BIN also have smaller

diameters than the hypercube, but still preserve the symmetry characteristic. The FHC is chosen as the switching fabric for three new ATM switches that are introduced in the next chapter.

Before the FHC is converted into an ATM switch, a simulation model was built and validated. This section also discussed how the model was developed for the FHC using OPNET Modeler, and gave the analysis of the simulation results. For comparison, the hypercube was also modeled and simulated. The simulation results indicate that the FHC performs better than the hypercube for all network sizes under different loads. The performance advantages of the FHC were higher under the normal destination distribution. The simulation results and the advantages of the FHC over the hypercube agree with the published analytical results.

Two hypercube based ATM switches discussed in this section were shown to have better or at least comparable performance relative to other ATM switches in the literature, but have low hardware complexity. Since the FHC outperforms the hypercube, the FHC appears to be an excellent choice as an infrastructure for new ATM switches. Finally, the agreement to the published analytical results indicates that the OPNET model of the FHC can be used as a valid backbone model that can be extended to create new ATM switch models. These switches contribute another new direction in the design of the interconnection network based ATM switches.

4. New ATM Switches Based on the Folded Hypercube

In this chapter, I propose three new ATM switch architectures that use the folded hypercube (FHC) [ElL91, LaE89] as the switching fabric. [Ho92] has shown that bisectional interconnection network of 2^n nodes for any even n, is isomorphic to the FHC(n). However, the FHC was chosen over the Bisectional Interconnection Network (BIN) [GhB89] because, as discussed in Section 3.2.2, FHC has simpler routing algorithm than that of the BIN and the BIN is valid only for odd degree –only switch sizes of 4, 16, 64, 256, etc. can be implemented with the BIN. The following sections describe the three architectures, discuss the assumptions and routing functions that are used by all three architectures, and give the goals of this study.

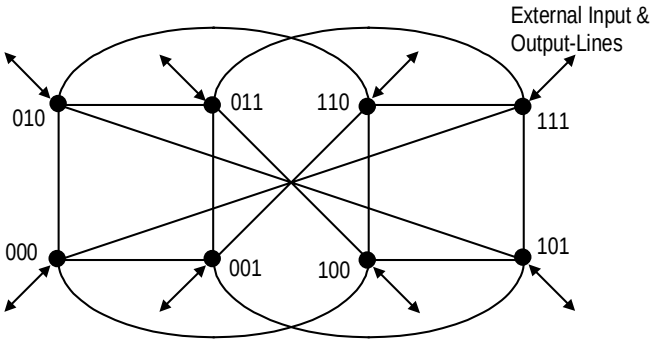


Figure 4.1 An 8×8 FAS-1.

4.1 General Architecture

The first architecture that I propose, called FAS-1 (FHC ATM Switch-1), uses a FHC(n) to implement an $N \times N$ switch where $N = 2^n$ (Figure 4.1). Each node has a bi-directional external link connecting to other ATM switches in addition to its n bi-directional i-links and one bi-directional c-link. Therefore, each node is connected to n+2 bi-directional links. My second proposed architecture, called FAS-2 (FHC ATM Switch-2), uses a FHC(n+1) to implement an $N \times N$ switch where $N = 2^n$ (Figure 4.2). The nodes with addresses from 0 to N–1 are called *input-nodes*, and the nodes with addresses from N to $2N$–1 are called *output-nodes*. The set of input-nodes (output-nodes) is called the *input-cube* (*output-cube*). Each input-node (output-

node) has a unidirectional external input-link (output-link) connecting to other ATM switches in addition to its n bi-directional i-links and one bi-directional c-link.

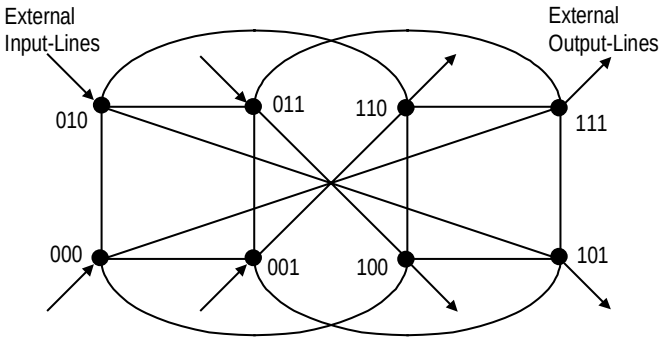


Figure 4.2 A 4×4 FAS-2.

The third architecture, called FAS-3 (FHC ATM Switch-3), uses a *modified* FHC(n+1) to implement an $N\times N$ switch. Like FAS-2, the nodes with addresses from 0 to $N-1$ are called input-nodes, and the nodes with addresses from N to $2N-1$ are called output-nodes. Also, the set of input-nodes (output-nodes) is called the *input-cube* (*output-cube*). Each input-node (output-node) has a unidirectional external input-link (output-link). Each output-node has an extra link, called the c'-link, that connects to a node at Hamming distance $n-1$. Let $a'(u) = (u_{n-2}u_{n-3}\cdots u_1u_0)$, the address bits of $a(u)$ with the most significant bit removed. FAS-3 tries to keep the cells within the output-cube if possible keeping the cells closer to their destinations. Figure 4.3 shows the third switch architecture.

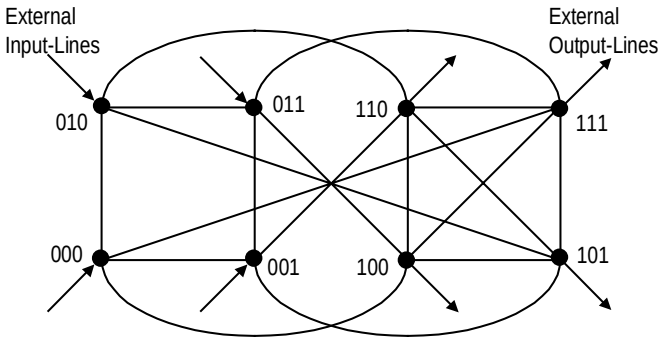


Figure 4.3 A 4×4 FAS-3.

For each input link there is a buffer to store an arriving cell while the router selects link for the cell. For an $N\times N$ FAS-1, there are a total of n+2 buffers as shown in Figure 4.4. All the

nodes of FAS-1 and the input-nodes of FAS-2 and FAS3 have input buffers that store line cells that were not chosen during the link selection process by the router. There are also output buffers at the external output line for the cells that reached their destination. The output buffer is needed since more than one cell can arrive at a cell time. Also, re-sequencing of the cells is performed while the cells are queued in the output buffers. All the nodes of FAS-1 and the input-nodes of FAS-2 and FAS3 have SAF buffers that temporarily store the cells from the links and the external line while they wait to be routed. Finally, all the nodes of FAS-1 and the output-nodes of FAS-2 and FAS3 have the output buffers.

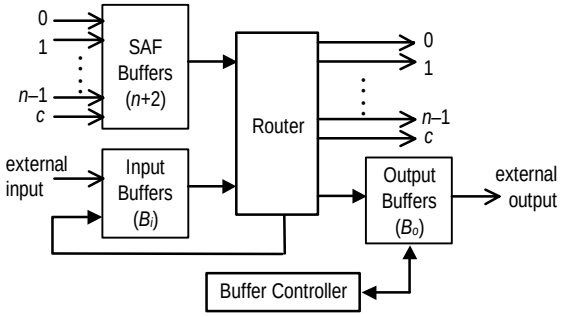


Figure 4.4 Structure of a node in FAS-1.

4.2 Priority Schemes

All the links, external and internal, of all three switch architectures are assumed to run at the same speed. The three switches use the Dynamic Distributed Routing Algorithm (DDRA) to route the cells. DDRA incorporates *deflection routing* to minimize the buffers needed in the network. [GrH82, VaB94] have studied and shown that the deflection routing performs well in hypercube networks for small as well as large networks and for a whole range of light to heavy loading. In fact, as the network size increases, the performance of the deflection routing becomes optimal [GrH82]. In deflection routing, congestion causes packets admitted to the network to be temporarily misrouted rather than buffered or dropped. This allows good performance with a smaller number of buffers compared to non-deflection routing. Furthermore, deflection routing can handle a moderate amount of traffic without any buffer at each node. Deflection routing can also handle heavy traffic and hot spots of non-uniform traffic by re-distributing and sharing the load with other nodes in the network with fewer buffers. It has

performed well when [Ma93, PaC95] have used it in their switch designs. The Dynamic Distributed Routing Algorithm is discussed in more detail in later sections.

I implemented a three-level priority scheme for all three switches. The highest priority level is the *class-priority*. An ATM network has two general classes of traffic, time-insensitive (Class-0) and time-sensitive (Class-1). A Class-1 cell will have a higher priority than a Class-0 cell during the link selection process. This is to guarantee Quality of Service (QoS) for the time sensitive ATM traffic. Also, as discussed later, the Class-1 cells have higher priority when there is contention for space in the output buffer. The next priority level is the *age-priority*. The oldest cell among all the cells requesting a particular link will win during the link selection process. Each node will use the time-stamp tagged on all new cells entering the switch. Giving a higher priority to the older cell is one of two implementations that will try to preserve the cell sequence as much as possible since keeping the cell sequence is as important as providing high throughput, low loss switching in ATM environment. Furthermore, the destination node will use cell numbers to re-order the cells before they are transmitted out to the external output link–older cells will receive lower cell numbers. This is the second cell sequencing implementation.

The lowest priority used is called *dist-priority*, and the cell closer to its destination will have a higher priority when class- and age-priority levels do not resolve the link contention. The third priority is used because a cell closest to its destination has the fewest possible candidate links and is most likely to encounter a link contention. For example, in an FHC(4) a new cell that enters the switch at node '0001' destined for node '1111' has four possible candidate links to nodes '0011,' '0101,' '1001,' and '1110.' However, when that cell reaches node '0111,' leaving only one hop to go, it has only one possible candidate link. Since deflection routing is used, no cells are dropped once they are in the network. A cell will only be dropped when the external input-link buffer overflows and when a cell reaches its destination and its time-stamp is older than that of the last cell transmitted to the external output-link. These two methods, deflection routing and the three-level priority scheme, should keep the cell loss rate to a minimum.

Each ATM cell entering the switch will be tagged with a header shown in Figure 4.5. The first field is the *class field* that denotes time sensitivity of the cell. The next two fields in the header are the cell's desination node address and the external input port address in binary.

The *assigned link field* (AL) field, the *birth-time* (BT), and the *cell number* (CN), are the rest of the header fields. When a new cell enters the switch, the current simulation time will be stored in the BT field[5]. Whenever it is needed, the router or the buffer controller can calculate the *age* of a cell by subtracting the BT field from the current time. Each switch maintains N cell numbers, CN_d where $0 \le d < N-1$, one per output nodes. When a cell enters the switch, the input node checks the cells destination, assigns the current CN_d to the cell, and increments CN_d by one. Once the cells reach their destinations, the output node will use the cells' CN to resequence them. The distance to destination (DD) of a cell is also calculated by the router using the destination address of the cell and the address of the current node. The router will sort all the cells that requested links using the class, age, and DD fields as the odering parameters during the link selection process. More details of the functions of the router, including the link selection process and the AL field, are given in later sections.

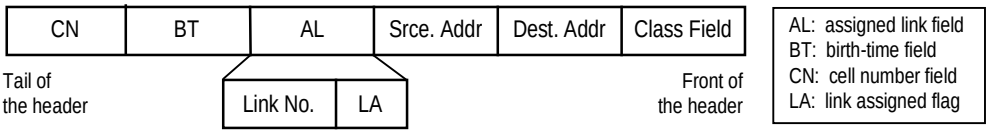


Figure 4.5 Cell header of the FAS-1, 2, or 3 switch.

4.3 Assumptions and Notations

I have modified the *Dynamic One-to-One Algorithm* given in Chapter 3 for the three proposed switches. First, before the new routing algorithm is given, changes in the assumptions and notations are given here. Let N be the size of the FHC used, and $n = \log_2 N$ be the dimension of the FHC. Let s be the binary address of the node that is performing the routing, and d be the binary address of the destination of the cell that is being routed. Rather than using the n-bit routing tag $a = s \oplus d = (a_{n-1} \cdots a_1 a_0)$, the new algorithm uses a $(n+1)$-bit routing tag $r = 0 \| a = (0, a_{n-1} \cdots a_1 a_0)$ where $\|$ denotes concatenation. Let $\|r\|$ be the Hamming weight of r. This new routing tag r always has a leading 0 followed by $s \oplus d$. Using r makes more sense since an FHC(n) has a degree of $n+1$ which means that each node is connected to $n+1$ links. It further

[5] Cells are generated at a maximum rate of one cell per simulation time unit.

simplifies the routing algorithm that will be discussed later. Links with labels corresponding to the bit positions that are 1's in r will be called *1-links*, and similarly, links with labels corresponding to the bit positions that are 0's in r will be called *0-links*. The c-link that was mentioned earlier corresponds to the most significant bit position, and since this bit is always a 0 in r, a c-link will be treated as a 0-link in the routing algorithm. Finally, the link that corresponds to the bit position i is called link-i. For example, a link that corresponds to the bit position 3 is called link-3.

The property of an FHC given in Equations (3.1) and (3.2) needs to be modified as followings when r is used instead of a.

$$DD = \|r\| \qquad \text{for } \|r\| \le \lceil n/2 \rceil \tag{4.1}$$

$$DD = n - \|r\| + 1 \qquad \text{for } \|r\| > \lceil n/2 \rceil \tag{4.2}$$

Here, the notation DD (distance to destination) is used instead of dist(u,v). In a distributed dynamic routing algorithm, distance to destination of a cell from a node is more meaningful than a distance between any two pair of nodes. This same DD value is used in the dist-priority scheme mentioned earlier.

4.4 Routing Algorithms

The following gives the routing algorithm that will be used by FAS-1 and FAS-2. FAS-3 needs a little modification and this will be presented later.

- ***Distributed Dynamic Routing Algorithm for FAS-1 & FAS-2*:**

  input: s, d;

  Begin

  $r = s \oplus d$;

  If $\|r\| \le \lceil n/2 \rceil$ **Then**

  Send the cell via any available preferred 1-link; if none is available, deflect via a 0-link.

  Else

  Send the cell via any available preferred 0-link; if none is available, deflect via a 1-link.

  Endif.

  End.

The routing algorithm is modified as follows for FAS-3. The preferred links are the same for a cell whether it is in FAS-1, FAS-2, or FAS-3. The main difference occurs when selecting among non-preferred links. To keep the cells within the output-cube, links other than links-(n–1) and c-links are considered first. If none of these links are available, then the cell is routed via either link-(n–1) or c-link. Let $r' = (r_{n-1}r_{n-2}\cdots r_1r_0)$, the bits of r without the most significant bit.

- ***Distributed Dynamic Routing Algorithm for FAS-3*:**

  input: s, d;

  Begin

  $r = s \oplus d$;

  If $\|r'\| \leq \lceil (n-1)/2 \rceil$ **Then**

  Send the cell via any available preferred 1-link; if none is available, deflect via a 0-link within the output-cube.

  Else

  Send the cell via any available preferred 0-link; if none is available, deflect via a 1-link within the output-cube.

  Endif.

  If no link is available, select either link-(n–1) or c-link randomly.

  End.

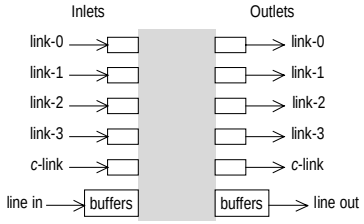
When a node in FAS-1 receives cells at the beginning of a cell cycle, the node's router performs the following link selection process in the given order. The AL field in the header of the cell contains two subfields, *link-assigned flag* (LA), and *link-number* (LN) subfields as shown in Figure 4.5. All the arriving link cells at each node have LA set to 0.

1. Place all arriving link cells in the SAF buffers and find $\|r\|$ of these cells.
2. Place the new arriving line cell in the tail of the input buffer. If the buffer is full, drop the cell. Sort all the cells in the input buffer using the priority rule. Find $\|r\|$ of the cell.
3. If a cell's $\|r\| = 0$, then it has reached its destination. Send these cells to output buffer. The output buffer controller will handle these cells.
4. Sort the link cells according to the priority rule.
5. Assign a link to each link cell using the *Distributed Dynamic Routing Algorithm*. Deflection links are not decided until all the preferred links are assigned after examining

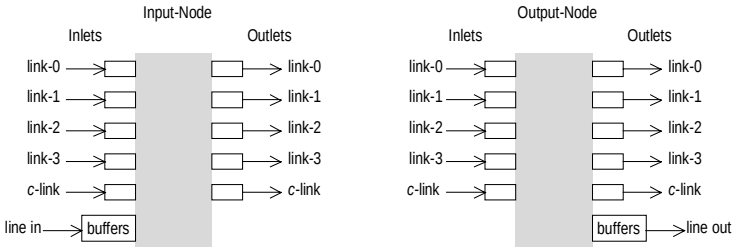
all the cells and links. When a cell is assigned with a link, its LA is set to 1, and stores the link number in the LN sub-field of the cell's header.

6. If there are *k* number of links still available, select *k* cells from the head of the input buffer and assign a link to each cell.
7. Transmit the cells via their assigned links after setting LA = 0.

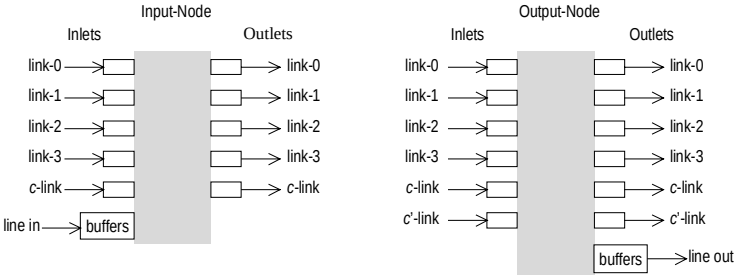
For nodes in FAS-2 and FAS-3, a few modifications are needed in the above steps. Each input-node skips Steps 3, while each output-node skips Step 2 and 4.



(a) FAS-1



(b) FAS-2



(c) FAS-3

Figure 4.6 Input and output links of a node in each of the three switches (n = 4).

When links are assigned, link cells are considered before the line cells for the following reasons. It is impossible to guarantee an output link for every cell in a node of FAS-1 and input nodes of FAS-2 and FAS-3 since up to m+1 cells may arrive at a time when there are only m links to choose from in these nodes (Figure 4.6). However, since the number of input and output

inter-node links is equal, we can guarantee at least one output link for each link cell if the link cells are routed before the line cells. A new line cell is dropped if the input buffer is full, so the switches should have enough input buffers for input cells that experienced link blocking for all the preferred and detour links. Different input buffer sizes are studied to find a buffer size that gives the best cost and cell loss trade off. Under normal traffic and an unsaturated load level, however, the probability that at least one cell has reached its destination is high. Therefore, a link can be guaranteed for a line cell as well.

indices	0	1	2	• • • • •	n–1	n
elements	i_0	i_1	i_2		i_{n-1}	i_n

Each element indicates whether or not link-k has been assigned to a cell.
If i_k = 0, the link is not yet assigned; if i_k = 1, the link is assigned to a cell.

Figure 4.7 Assigned-link array

During the link assignment process, the links are considered and selected in a cyclically increasing order. The starting bit position changes every cell cycle. For example, if links are considered in the order of bit positions 2, 3, 4, 5, 0, 1 during one cell cycle (n = 5), the links are considered in the order of bit positions 3, 4, 5, 0, 1, 2 during the next cell cycle. Note that these bits are from r, and that the most significant bit position, n, corresponds to the c-link. This cyclic selection process introduces fairness to the *Distributed Dynamic Routing Algorithm*. The router uses the *assigned-link* array to maintain whether or not a link has been assigned to a cell (Figure 4.7). There are n+1 (n+2 for FAS-3) elements in the array, one for each outgoing link. At the start of every link selection process, the router initializes all the elements to '0.' When link-k is assigned to a cell, then a '1' is stored in the k-th element. In other words, when the j-th element contains a '0', this means that link-j is free to be assigned to any cell while a '1' in the j-th element means that link-j has already been assigned to a cell. The above mentioned bit positions are the indices of the assigned-link array, and the starting index number moves cyclically every cell cycle.

4.5 Output Buffer Controller

Each node of FAS-1 and each output node of FAS-2 and FAS-3 has an *output buffer controller* (OBC) that manages the cells queued in the output buffer. The OBC maintains *N* virtual queues for cells from each of *N* external input ports. Each cell is placed in the corresponding virtual queue by checking the external input port address of the cell's header. I have chosen a shared buffering strategy over a non-shared strategy since the shared buffering strategy was shown to require a smaller total number of buffers than physically separated (partitioned) buffers [ChH96, CoK96, Li94]. One of the two functions of the OBC is monitoring whether the buffers are shared in a fair manner. This function is needed since without a good, fair buffer controller, the shared buffers may perform worse than the non-shared buffers [Li94]. There are many buffer sharing strategies, such as the *completely sharing* (CS), *partial sharing* (PS), *sharing with minimum allocation* (SMA), *sharing with maximum queue length* (SMXQ), and *fair sharing* (FS). Among these strategies, the CS and the FS are shown to have the best performance [CoK96]. However, unlike the FS, the CS is unfair since cells from one particular input port may take up all the buffer space and block cells from other input ports. Therefore, I have implemented the OBC with the FS strategy. The FS determines which virtual queue is using up the most space of the output buffer. If a new arriving cell is for this offending queue, the cell is dropped. However, if the cell is for one of non-offending queues, the cell at the end of the offending queue is dropped to make room for the new cell. The FS for the OBC is modified since in FAS-1, 2, and 3, cells are deflected instead of being dropped. The modification is given in the discussion of the OBC's second task.

The second task of the OBC is re-sequencing the cells that originated from the same external input port before they are transmitted through the external output line. For each virtual queue, the OBC maintains two numbers, one for Class-1 cells and the other for Class-0 cells. Each number, CN_{next}, represents the next cell number that should be transmitted. Every time a new cell is added to a virtual queue, the cells are sorted, from lowest to highest, based on their CN fields. If a cell's CN is lower than CN_{next}, that cell is dropped. If a cell's CN is the same as CN_{next}, the OBC transmits that cell. If a cell's CN is higher than CN_{next}, the OBC stores the cell. To avoid any unfair usage of buffer space by a virtual queue, the OBC transmits the current oldest cell in the queue if that cell has been in the queue for *Time_Limit* amount of time. The

optimum value of *Time_Limit* that gives the best switch performance is four times the switch's diameter. This value is obtained from the pilot simulation studies, and it gives a cell to arrive at its destination in time to be transmitted via the external link in sequence even if it has been deflected. When the output buffer becomes full and a new Class-1 cell arrives, the OBC removes a Class-0 cell from the virtual queue, and stores the new Class-1 cell. Whenever a cell is transmitted, the transmitted cell's CN plus one becomes the new CN_{next}.

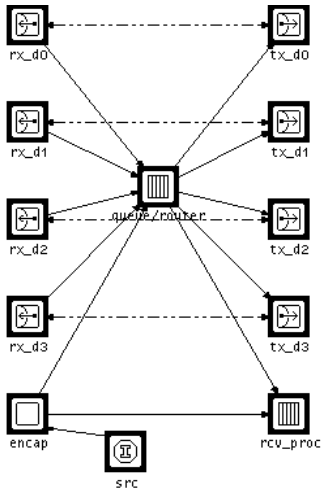
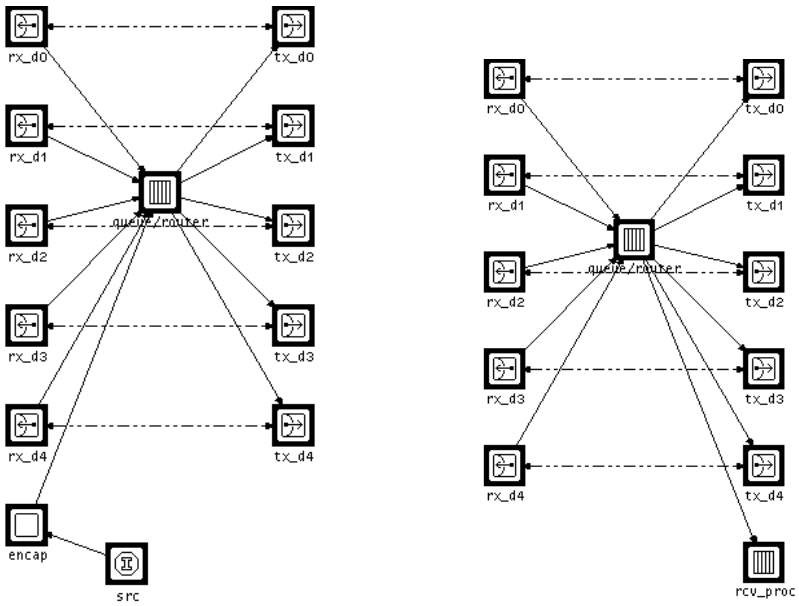


Figure 4.8 A node of 8×8 FAS-1 modeled in OPNET

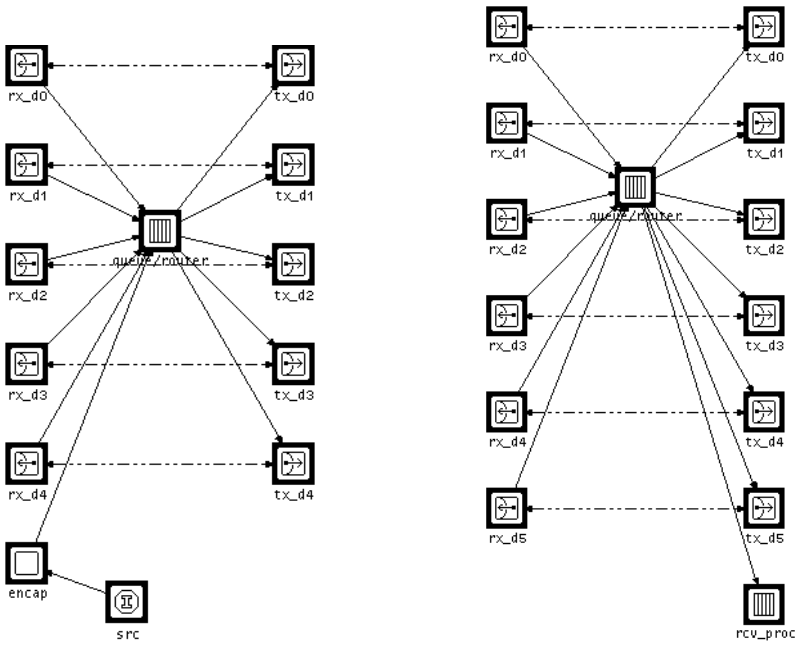
4.6 Modeling the Switches

The three proposed switch architectures are modeled using OPNET Modeler. Figure 4.8 shows how a node of 8×8 FAS-1 is modeled in OPNET. rx_d0 to rx_d2 (tx_d0 to tx_d2) modules are receivers (transmitters) that are connected to *i*-links. rx_d3 (tx_d3) module is a receiver (transmitter) that is connected to a *c*-link. src module is a cell generator that generates cells using the given interarrival time, *p*, and the uniform probabilty function. It can also be modified to generate bursty traffic. encap module generates destination addresses using either the uniform or normal distribution and tags the generated address to each cell received from src module. If a cell is tagged with an address that is the same as the particular node's address, it is passed on to rcv_proc module. If not, it is sent to queue/router module. These two modules, src and encap, model an external input line connected to each node.



(a) Input Node (b) Output Node

Figure 4.9 Nodes of 8×8 FAS-2 modeled in OPNET.



(a) Input Node (b) Output Node

Figure 4.10 Nodes of 8×8 FAS-3 Modeled in OPNET.

The input buffer is modeled in queue/router module. The output buffer and the external output line is modeled by rcv_proc module. This module queues the cells that have reached their destination, resequences them, and destroys them to free up the memory. This destroying function is equivalent to sending the cells via the external output line. Finally, queue/router module represents the SAF buffers and the router that performs the Distributed Dynamic Routing Algorithm. The nodes of FAS-1 and FAS-2 will be modeled similarly as shown in Figure 4.9 and Figure 4.10. The input nodes of FAS-2 and FAS-3 do not have rcv_proc module, and the output nodes of FAS-2 and FAS-3 do not have encap module. Also, the output node of FAS-3 has an extra set of receiver and transmitter modules to model the extra *c'*-link.

The functions or processes of encap, queue/router, and rcv_proc modules are written in *Proto-C* of OPNET Modeler. Proto-C is based on a combination of state transition diagrams (STD's), a library of high level commands known as *kernel procedures*, and the general facilities of the C programming language. Figure 4.11 shows an example of STD's, the STD of encap module. When a simulation starts, init state is invoked, and it initializes the distribution function which is used to generate destination addresses. The process waits in the idle state until a cell arrives from src module. When a cell arrives, xmt state is invoked. In this state, a destination address is generated and tagged to the received cell. For FAS-1 if a cell's destination happens to be the particular node, it is passed on to rcv_proc module. If not, it is sent to queue/router module to be queued and routed toward its destination. The process returns to idle state and waits for another cell.

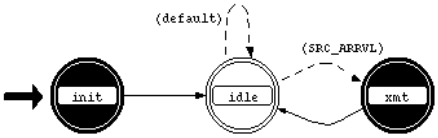


Figure 4.11 The state transition diagram (STD) of **encap** process module.

Figure 4.12 shows router process model used by queue/router module to route the cells. At the beginning of a cell cycle, the get_cells() function retrieves the arrived cells on all the links and stores them in the corresponding SAF buffers. rte_cells state first sorts the cells based on the priority scheme, and routes them by using the routing algorithm.

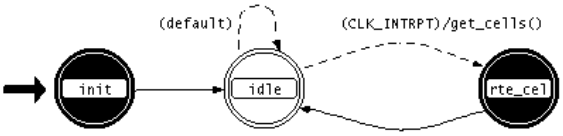


Figure 4.12 The state transition diagram (STD) of **router** process module.

The rcv process module used by the rcv_proc module is shown in Figure 4.13. At the beginning of every unit time, the update_queue function places the arriving cells in the output buffer based on the algorithm explained in Section 4.5. proc_cell state resequences the cells and selects the next cell that needs to be transmitted to the external line according to the algorithm explained in Section 4.5.

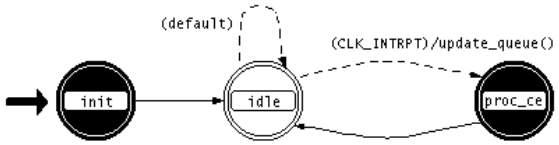


Figure 4.13 The state transition diagram (STD) of **rcv** process module.

Figure 4.14 shows how an (8 × 8) FAS-1 will be modeled in OPNET. The eight nodes are named from node_0 to node_7 as shown. The collct_stat node is used to collect the statistics and performance parameters, such as throughput, delay through the switch, etc., during and after the simulation. The double-headed arrows represent the *i*-links and the *c*-links. As mentioned above the external input and output lines are modeled within each node. FAS-2 and FAS-3 are modeled similarly.

The node and network models, of different sizes, FAS-1, FAS-2, and FAS-3 are generated using the External Model Access (EMA) package of OPNET. One of many usages of EMA package is creating large models that require many objects and interconnections. Since the EMA package allows algorithmic specification using a programmatic interface, it is especially useful when large models have regular structure like the FHC or hypercube. Three sets of two EMA application programs—one for node model and the other for network model—are written for FAS-1, FAS-2 and FAS-3. These programs are used to generate switches of various sizes—8 ×8 to 1024×1024.

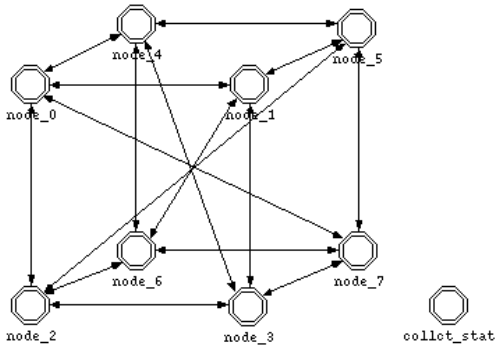


Figure 4.14 An 8×8 FAS-1 modeled in OPNET.

4.7 Summary

In this chapter, three new ATM switch architectures based on the FHC are proposed. First, the general architectures of the three proposed switches were given along with the three level priority schemes used by the switches. Then, the assumptions and notations used in implementing the three switches were listed. The routing algorithms and the output controller used by the switches were explained. The discussion on how the switches are modeled in OPNET Modeler was also given.

5. Analysis of Simulation Results

This chapter presents the analysis of simulation results of the three switches. The models of the switches were simulated using OPNET Modeler. The simulations were done under three different traffic patterns: uniform, normal, and bursty traffic. Two versions of bursty traffic patterns were used. One with a mean of five active duration and the other with a mean of ten active duration. Implementation of bursty traffic is discussed in detail in Section 5.3. For each traffic pattern, the ratio between time-sensitive Class-1 cells to time-insensitive Class-0 cells was 3:7. All simulations were done for eight different switch sizes, 8×8 to 1024×1024, and seven different loads, 0.1 to 0.9[6]. The simulations were run until about 100,000 cells were submitted into the switch by each input node to run. Performance figures were collected after each output node had received 1,000 cells to ensure the switch had reached a steady state. The length of simulation and starting point of data collection were chosen after several pilot studies. So, for a 32×32 switch, a total of 3,200,000 cells were injected into the switch. This chapter presents results for 32×32 switches with two input buffers and 32 output buffers per each node. The results of other switch sizes are given in Appendix A.

The two major performance parameters that were measured were the cell loss rate and end-to-end (ETE) cell time delay through the switch. The cell loss rate is defined as the percentage of cells that are dropped because either they arrived at their destination out of sequence or the buffer was full. The ETE delay is given in terms of unit time, the time it takes for a cell to traverse a link. To maintain the quality of service, especially for real-time audio and video cells, an ATM switch needs to have low cell loss rate. The ETE delay through the switch is an important parameter since the trend of the world is for a faster network. Low ETE delay will allow the time-sensitive cells to go in and out of a switch faster and reach their destinations faster. The ETE delays through the three new switches were expected to be small compared to switches based on the hypercube or even the Banyan networks. This is because an $N \times N$ hypercube or Bayan network has a diameter of n (= $\log_2 N$) while an FHC of the same size has a diameter of $\lceil n/2 \rceil$. Furthermore, many high performance ATM switches based on the Banyan type network can have much higher

[6] At 1.0 (100%) load, one cell is generated at every time unit in every node.

ETE delay since they have more than *n* stages. In fact, in some Banyan network based switches like the sorting based switches, a cell has to go through a multiple number of Banyan networks.

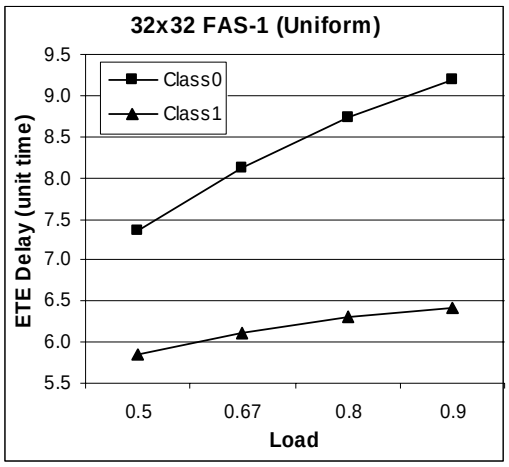
Section 5.1 gives the results for uniform traffic pattern. Results for normal traffic pattern are given in Section 5.2. The results from two bursty traffic patterns are given in Section 5.3. The three switches performances are compared to that of the two hypercube based switches by [Ma93] and [PaC95] in Section 5.4. Section 5.5 presents results from different Class-1 to Class-0 cell ratios. The summary of the analysis is given in Section 5.6.

5.1 Uniform Traffic Pattern

The first set of simulations used the uniform distribution to generate destination addresses. The uniform distribution is a good way to analyze the maximum or best-case performance that a network can support [DiJ81]. If the distribution of destination addresses is not uniform, some parts of the network will be more congested than others, contributing to a decrease in throughput. The cells were generated by *N* independent random processes, one for each of the *N* nodes.

Figure 5.1 through Figure 5.3 show the ETE delay and cell loss rate of the three switches. As can be seen in the graphs, the time-sensitive Class-1 cells experienced a much lower ETE delay as they traversed through the switch. More importantly, the cell loss rate of the Class-1 cell was very low even at high loads. FAS-1 achieved a cell loss rate of 3.23×10^{-5} for the Class-1 cells at 0.9 loading. The cell loss rate for the Class-0 cells was a respectable 4.28×10^{-4}, resulting in the overall cell loss rate of 3.09×10^{-4}.

FAS-2 and FAS-3 did not loose any Class-1 cells even at 90% loading. FAS-2 achieved a cell loss rate of 1.34×10^{-5} for the Class-0 cells while FAS-3 achieved still lower cell loss rate of 8.92×10^{-7} for the Class-0 cells at 90% loading. The overall cell loss rate was 9.38×10^{-6} for FAS-2 and 6.25×10^{-7} for FAS-3. FAS-3 achieved the best performance by keeping the cells within the output cube when possible. This makes the cells remain close to their destinations, even when they are deflected due to link contentions. From these results, it can be concluded that cells in FAS-3 experienced the fewest cases of link contention, which allowed them to be routed quickly. As mentioned above, the input buffer size and the output buffer size per node were set to two and 32, respectively. However, FAS-1 needed two input and 27 output buffers, FAS-2 needed one input and 26 output buffers, and FAS-3 needed one input and 22 output buffers per node. The ETE delay values and cell loss rate of all three switches at different loads are given in Table 5.1 and Table 5.2.



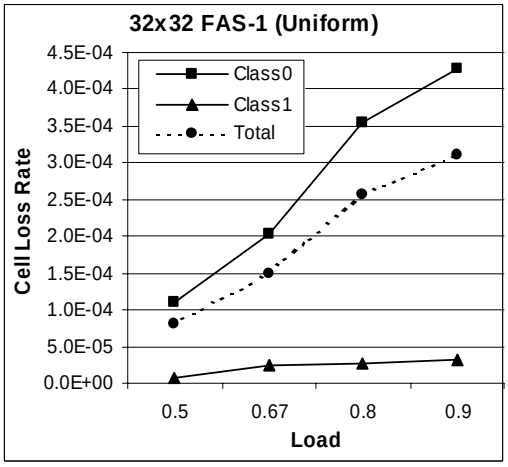
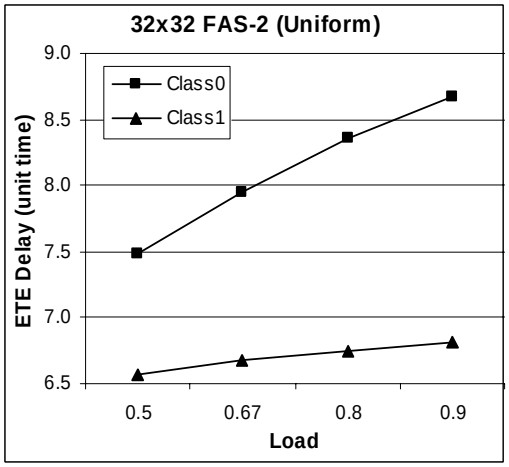


Figure 5.1 ETE Delay and cell loss rate of 32x32 FAS-1 under uniform traffic pattern.



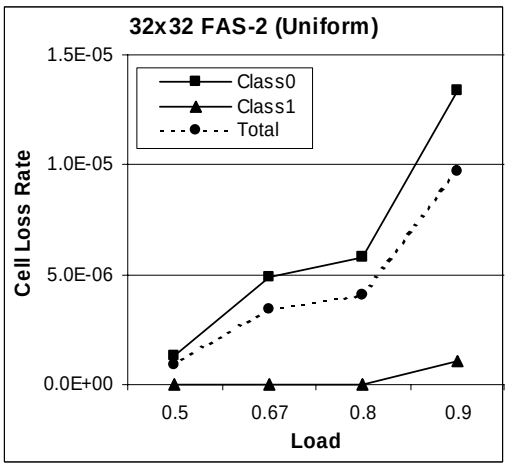
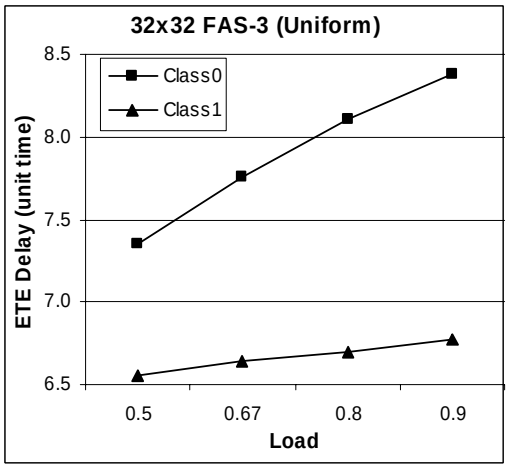


Figure 5.2 ETE Delay and cell loss rate of 32x32 FAS-2 under uniform traffic pattern.



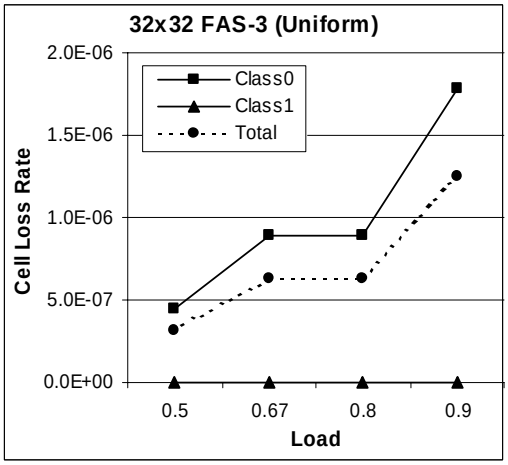


Figure 5.3 ETE Delay and cell loss rate of 32x32 FAS-3 under uniform traffic pattern.

Table 5.1. ETE Delay of 32x32 FAS-1/2/3 Under Uniform Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.58	6.01	6.61	7.36	8.13	8.73	9.19
	Class-1	5.28	5.63	5.84	6.10	6.32	6.42	6.50
FAS-2	Class-0	6.42	6.66	7.03	7.48	7.94	8.36	8.68
	Class-1	6.23	6.33	6.45	6.56	6.67	6.45	6.82
FAS-3	Class-0	6.40	6.62	6.96	7.35	7.75	8.11	8.38
	Class-1	6.23	6.32	6.42	6.55	6.64	6.70	6.77

Table 5.2. Cell Loss Rate of 32x32 FAS-1/2/3 Under Uniform Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.92E-07	1.07E-05	3.03E-05	1.11E-04	2.02E-04	3.55E-04	4.28E-04
	Class-1	0.00E+00	0.00E+00	4.17E-06	8.34E-06	2.40E-05	2.71E-05	3.23E-05
	Total	6.25E-07	7.50E-07	2.25E-05	8.03E-05	1.49E-04	2.57E-04	3.10E-04
FAS-2	Class-0	0.00E+00	0.00E+00	4.46E-07	1.34E-06	4.91E-06	5.80E-06	1.34E-05
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.04E-06
	Total	0.00E+00	0.00E+00	3.13E-07	9.38E-07	3.44E-06	4.06E-06	9.69E-06
FAS-3	Class-0	0.00E+00	0.00E+00	0.00E+00	4.46E-07	8.92E-07	8.93E-07	1.79E-06
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	3.13E-07	6.25E-07	6.25E-07	1.25E-06

5.2 Normal Traffic Pattern

The normal distribution was used to generate the destination addresses in the second simulation series. Even though the uniform destination address distribution gives the maximum throughput of a network, it is common in real world applications that one or few destinations may be more favored by the sources. To analyze the network performance in a more practical situation, the normal distribution was also used to generate destination addresses or hot-spots. Hot-spots cause a disproportionate number of packets to be routed to a subset of the destinations [DeC89, PfN85]. This is done by generating packets over a range of destination addresses with a preferential concentration around a mean destination address.

For the simulations with hot-spots, packet destinations were chosen using a normal distribution with a specified mean and standard deviation. The mean was chosen randomly, but the value chosen for the standard deviation is one-quarter of the network size, based on the study by [Pa94]. When the standard deviation was chosen to be less than one-quarter of the network size, some destinations were never chosen. When the standard deviation was chosen to be much

greater than one-quarter of the network size, the range of the destination addresses favored by the address generator were too wide and did not qualify as a good model of hot-spots. When the standard deviation was about one-quarter of the network size, the address generator created a distribution that covered the whole range of addresses, from 0 to $N-1$, with a concentration narrow enough to be modeled as a hot-spot.

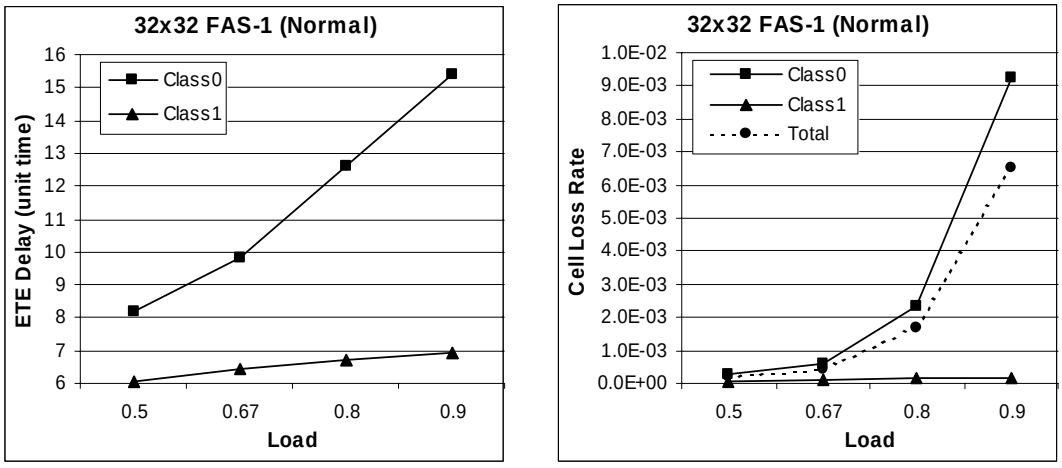


Figure 5.4 ETE Delay and cell loss rate of 32x32 FAS-1 under normal traffic pattern.

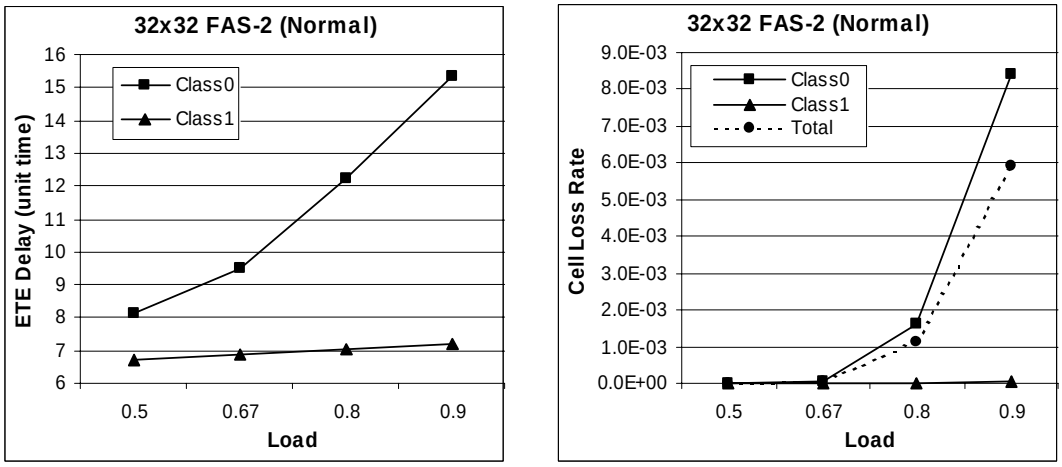


Figure 5.5 ETE Delay and cell loss rate of 32x32 FAS-2 under normal traffic pattern.

Figure 5.4 through Figure 5.6 show the simulation results under normal traffic pattern. The overall cell loss rate figures were higher than for the uniform traffic case, as expected, but the Class-1 cell loss rate figures increased only slightly. The priority scheme used by the routing algorithm and the output buffer controller was able to route the time-sensitive Class-1 cells faster

to obtain respectable cell loss rate figures even with a heavily congested normal distribution traffic pattern. The performance figures for the three switches under seven different loading levels are summarized in Table 5.3 and Table 5.4.

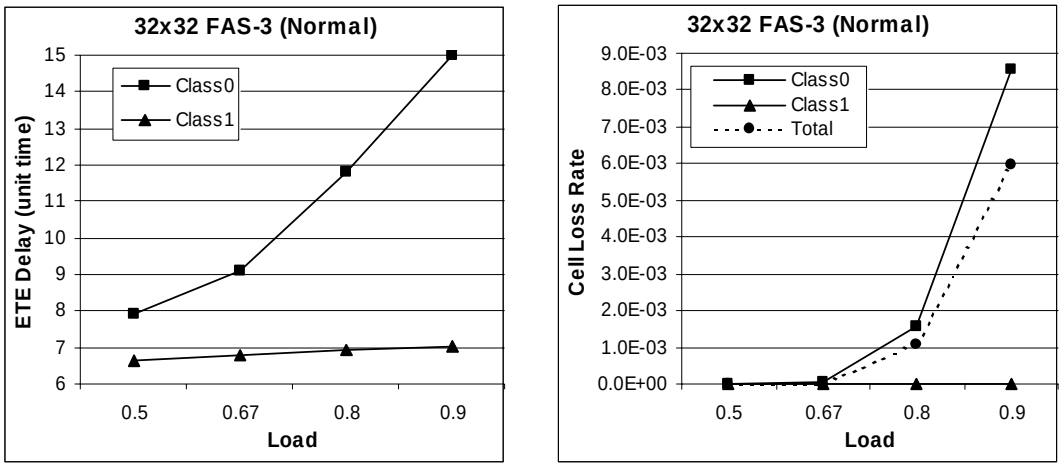


Figure 5.6 ETE Delay and cell loss rate of 32x32 FAS-2 under normal traffic pattern.

Table 5.3. ETE Delay of 32x32 FAS-1/2/3 Under Normal Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.63	6.22	7.03	8.19	9.82	12.60	15.41
	Class-1	5.30	5.48	5.75	6.08	6.43	6.73	6.92
FAS-2	Class-0	6.45	6.79	7.30	8.12	9.49	12.26	15.34
	Class-1	6.24	6.36	6.49	6.69	6.87	7.05	7.20
FAS-3	Class-0	6.43	6.74	7.19	7.90	9.12	11.79	14.99
	Class-1	6.24	6.35	6.47	6.63	6.80	6.94	7.03

Table 5.4. Cell Loss Rate of 32x32 FAS-1/2/3 Under Normal Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.34E-06	8.86E-05	8.30E-05	2.96E-04	6.06E-04	2.36E-03	9.26E-03
	Class-1	0.00E+00	1.04E-06	7.30E-06	3.23E-05	8.55E-05	1.38E-04	1.85E-04
	Total	9.38E-07	2.03E-05	6.03E-05	2.17E-04	4.50E-04	1.69E-03	6.54E-03
FAS-2	Class-0	0.00E+00	0.00E+00	1.79E-06	6.69E-06	5.27E-05	1.61E-03	8.40E-03
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.13E-06	1.77E-05	5.53E-05
	Total	0.00E+00	0.00E+00	1.25E-06	4.69E-06	3.78E-05	1.14E-03	5.90E-03
FAS-3	Class-0	0.00E+00	0.00E+00	4.46E-07	4.46E-07	2.77E-05	1.54E-03	8.55E-03
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.06E-06
	Total	0.00E+00	0.00E+00	3.13E-07	3.13E-07	1.94E-05	1.08E-03	5.99E-03

5.3 Bursty Traffic Pattern

The third traffic pattern used in simulating the three new switches was a bursty traffic pattern. Liew has shown that, for any buffering strategy, bursty traffic generally requires larger buffers to obtain the same P_{loss} compared to uniform traffic [Li90]. Also, the switch needs to operate at a load much lower than the maximum throughput load to achieve small P_{loss} under bursty traffic. For bursty traffic, the traffic is modeled by an *on-off* source at each input node [BaM94]. The number of cells generated during a burst is geometrically distributed over two means. Two mean active durations (m_A) are used: m_A=5 (Bur-5) and m_A=10 (Bur-10). Cells belonging to the same burst were generated in consecutive time units and were tagged with the same destination address. The destination addresses for different bursts were uniformly distributed. This model was used by Pao and Chau [PaC95], and was chosen for this research to do performance comparison among the switch by Pao and Chau [PaC95] and the three new switches.

First, the simulation results under the Bur-5 traffic pattern are given in Figure 5.7 through Figure 5.9 and also in Table 5.5 and Table 5.6. The average ETE delay figures were higher than those of the uniform traffic pattern. The additional delay experienced by the cells was mostly due to time spent in the output buffer once they arrived at their destinations and not the delay the cells experienced while traversing the switch. This was because cells of the same burst were destined to the same destination. Therefore, an OBC could receive one or a few cells during a time unit for several consecutive time units. This will overload the OBC causing the cells to wait longer to be re-sequenced and transmitted to the external output line. The cell loss rate figures were higher because more cells are dropped by the OBC. As mentioned in Chapter 4, a cell was dropped if it was queued in the buffer longer than *Time_Limit*, even if it arrived in-sequence.

FAS-2 performed the worst among the three switches. The switch used twice the number of nodes compared to FAS-1 requiring the cells to travel longer, especially when a cell was deflected. In other words, FAS-2 had the longest deflection paths among the three switches. So, when a cell was deflected, it could arrive at the destination out-of-sequence, causing it to be dropped. FAS-3 did not suffer from this problem and it achieved the best performance. This was because, as mentioned earlier, FAS-3 kept the cells within the output cube, keeping the cells close to their destinations, even when they were deflected. FAS-1 performed worse than FAS-3 since cells in FAS-3 experienced less link contention, as mentioned in Section 5.1.

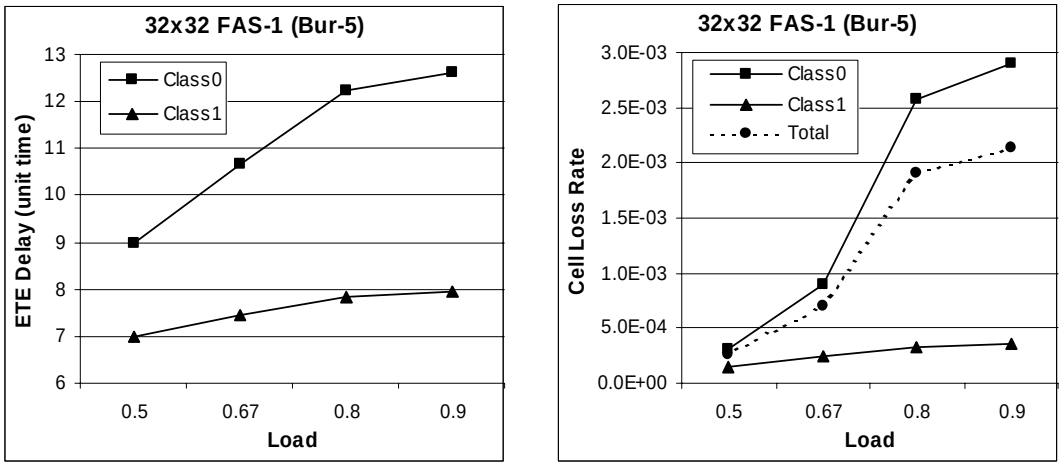


Figure 5.7 ETE Delay and cell loss rate of 32x32 FAS-1 under Bur-5 traffic pattern.

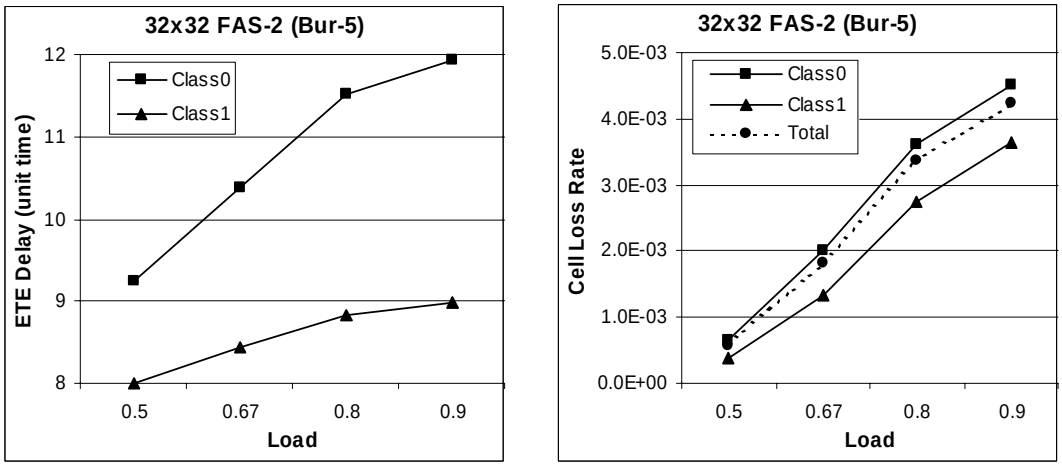


Figure 5.8 ETE Delay and cell loss rate of 32x32 FAS-2 under Bur-5 traffic pattern.

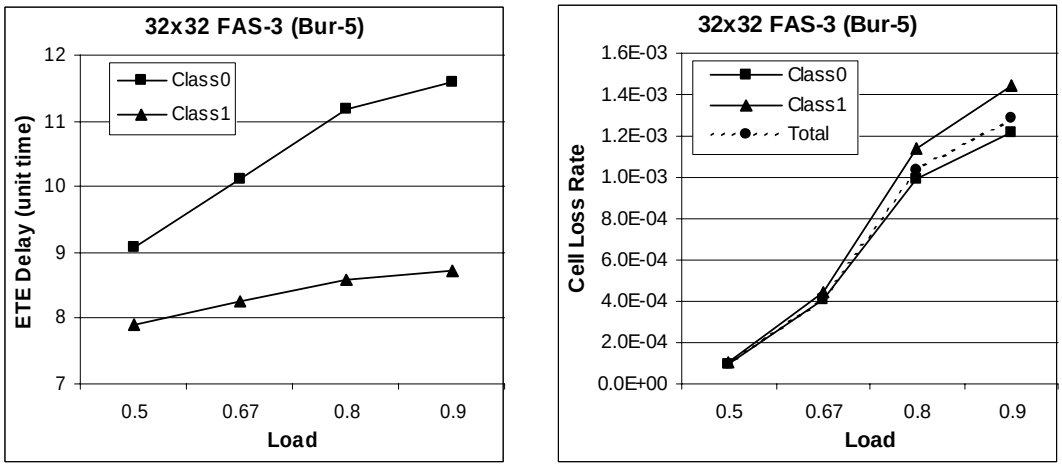


Figure 5.9 ETE Delay and cell loss rate of 32x32 FAS-3 under Bur-5 traffic pattern.

However, the Class-1 cells of FAS-3 experienced slightly higher cell loss rate than the Class-0 cells. The values of *Time_Limit* was set to four times the diameter of the backbone FHC based on the pilot study, this value worked well for a sample of switch sizes under different load and traffic pattern. It turned out that *Time_Limit* of twelve used for 32×32 FAS-3 was not large enough under Bur-5 traffic pattern. Extra simulation results show that by increasing *Time_Limit* by two to 14, the Class-1 cell loss rate dropped by almost fifty percent and was twenty percent lower than the Class-0 cell loss rate.

Table 5.5. ETE Delay of 32x32 FAS-1/2/3 Under Bur-5 Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	6.53	7.03	7.84	8.99	10.66	12.23	12.62
	Class-1	6.23	6.42	6.70	7.00	7.44	7.84	7.93
FAS-2	Class-0	7.43	7.81	8.35	9.25	10.39	11.51	11.94
	Class-1	7.25	7.41	7.65	8.00	8.43	8.84	8.98
FAS-3	Class-0	7.41	7.77	8.26	9.08	10.10	11.18	11.60
	Class-1	7.24	7.58	7.60	7.91	8.25	8.59	8.72

Table 5.6. Cell Loss Rate of 32x32 FAS-1/2/3 Under Bur-5 Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	3.57E-06	1.61E-05	6.96E-05	3.12E-04	8.98E-04	2.58E-03	2.91E-03
	Class-1	3.13E-06	1.15E-05	5.84E-05	1.49E-04	2.37E-04	3.28E-04	3.57E-04
	Total	3.44E-06	1.47E-05	6.63E-05	2.63E-04	7.00E-04	1.90E-03	2.14E-03
FAS-2	Class-0	8.93E-07	4.06E-05	1.78E-04	6.53E-04	2.02E-03	3.61E-03	4.51E-03
	Class-1	1.04E-06	7.29E-06	1.19E-04	3.76E-04	1.34E-03	2.76E-03	3.63E-03
	Total	9.38E-07	3.06E-05	1.60E-04	5.70E-04	1.82E-03	3.36E-03	4.25E-03
FAS-3	Class-0	0.00E+00	0.00E+00	2.50E-05	9.73E-07	4.07E-04	9.89E-04	1.22E-03
	Class-1	0.00E+00	1.79E-06	2.81E-05	1.01E-04	4.45E-04	1.14E-03	1.45E-03
	Total	0.00E+00	1.25E-06	2.59E-05	9.85E-05	4.18E-04	1.04E-03	1.29E-03

The simulation results under the Bur-10 traffic pattern show a similar pattern as shown in Figure 5.10 through Figure 5.12 and also in Table 5.7 and Table 5.8. However, the average ETE delay and cell loss figures were higher that those of Bur-5. This was as expected since, on average, twice the number of cells belonged to the same burst destined for the same node. More cells arriving at a node meant more cells could be dropped because they were in the buffer longer than *Time_Limit* or because the output buffer was full. FAS-2 gave the worst performance while FAS-3 achieved the best performance, as in the Bur-5 traffic case. As mentioned above, FAS-2 had the longest

deflection paths causing many cells to be dropped because they arrived at their destination too late or out-of-sequence. FAS-1 and FAS-3 had shorter deflection paths, but since FAS-3 routed cells faster than FAS-1, FAS-3 achieved the best cell loss rate.

Just as FAS-3 did not provide a lower cell loss rate to the Class-1 cells under Bur-5 traffic pattern, the Class-1 cells of FAS-2 and FAS-3 did not have significant cell loss rate advantage over the Class-0 cells. As mentioned above, this problem was solved when the value of *Time_Limit* increased by one or two. With this, the cell loss rate of Class-1 cells were significantly lower in FAS-2 and FAS-3

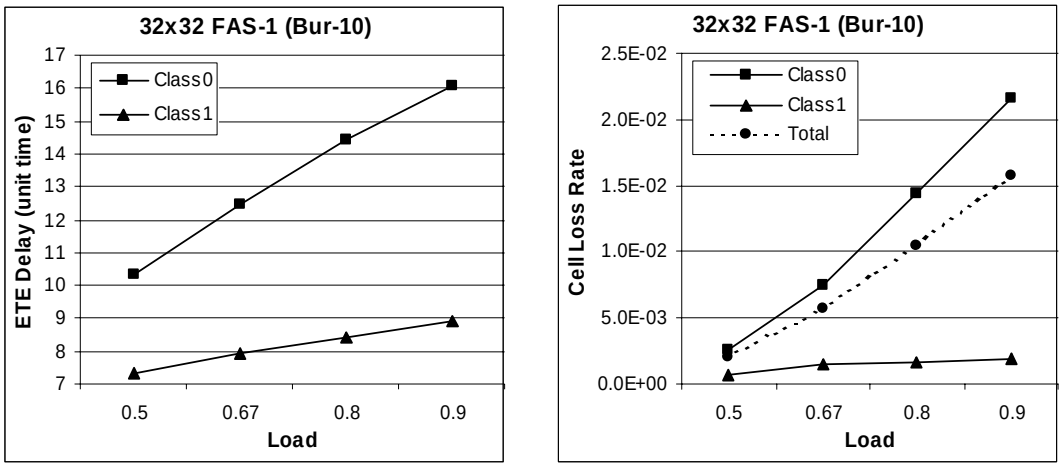


Figure 5.10 ETE Delay and cell loss rate of 32x32 FAS-1 under Bur-10 traffic pattern.

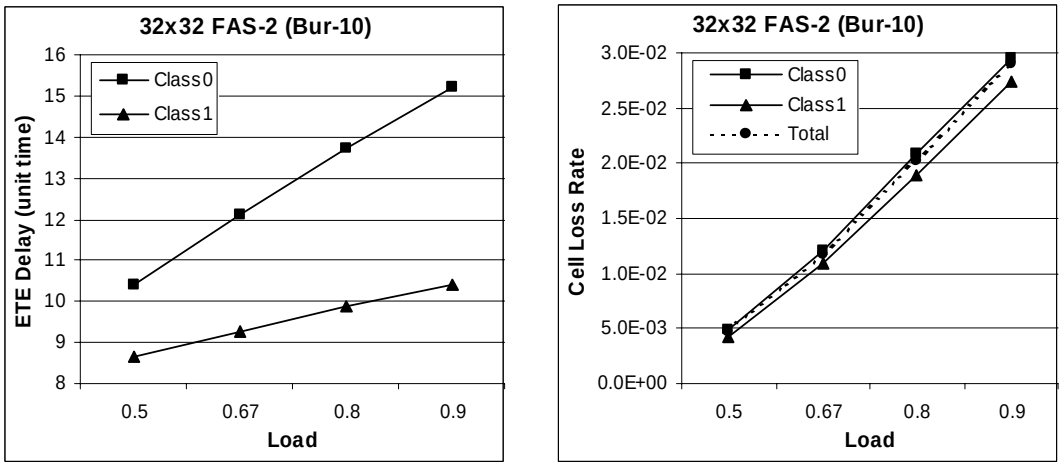


Figure 5.11 ETE Delay and cell loss rate of 32x32 FAS-2 under Bur-10 traffic pattern.

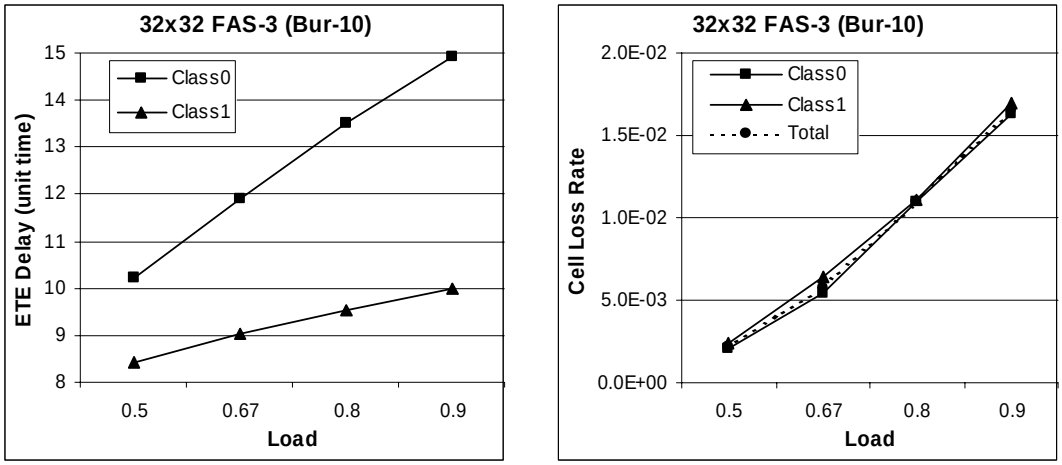


Figure 5.12 ETE Delay and cell loss rate of 32x32 FAS-3 under Bur-10 traffic pattern.

Table 5.7. ETE Delay of 32x32 FAS-1/2/3 Under Bur-10 Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	6.71	7.47	8.65	10.31	12.48	14.45	16.08
	Class-1	6.30	6.56	6.89	7.34	7.95	8.44	8.93
FAS-2	Class-0	7.62	8.17	9.07	10.42	12.12	13.74	15.23
	Class-1	7.36	7.60	8.04	8.65	9.28	9.88	10.40
FAS-3	Class-0	7.59	8.10	8.96	10.20	11.90	13.49	14.93
	Class-1	7.32	7.56	7.92	8.43	9.03	9.54	9.99

Table 5.8. Cell Loss Rate of 32x32 FAS-1/2/3 Under Bur-10 Traffic Pattern.

	Loads	0.1	0.2	0.33	0.5	0.67	0.8	0.9
FAS-1	Class-0	6.65E-05	2.49E-04	8.72E-04	2.54E-03	7.51E-03	1.43E-02	2.16E-02
	Class-1	7.40E-05	1.83E-04	3.98E-04	7.29E-04	1.50E-03	1.59E-03	1.92E-03
	Total	6.88E-05	2.29E-04	7.30E-04	2.00E-03	5.71E-03	1.05E-02	1.57E-02
FAS-2	Class-0	1.17E-04	4.63E-04	1.75E-03	4.88E-03	1.20E-02	2.08E-02	2.95E-02
	Class-1	7.72E-05	3.41E-04	1.37E-03	4.27E-03	1.09E-02	1.89E-02	2.75E-02
	Total	1.05E-04	4.27E-04	1.64E-03	4.70E-03	1.17E-02	2.02E-02	2.90E-02
FAS-3	Class-0	1.52E-05	1.50E-04	6.42E-04	2.04E-03	5.49E-03	1.09E-02	1.64E-02
	Class-1	1.98E-05	1.45E-04	7.10E-04	2.36E-03	6.43E-03	1.11E-02	1.70E-02
	Total	1.66E-05	1.49E-04	6.62E-04	2.14E-03	5.77E-03	1.10E-02	1.65E-02

5.4 Performance Comparison with [Ma93] and [PaC95]

The switch designed by [Ma93] used an n-dimensional hypercube to implement an $N \times N$ switch ($N = 2^n$). [Ma93] gave simulation results for a 128×128 switch (7-dimensional hypercube, a total of 128 nodes) with five different routing algorithms. The switch by [Ma93] used two input

buffers and eight output buffers per node. The best cell loss rate of [Ma93] is compared with those of FAS-1, FAS-2 and FAS-3 in Table 5.9. [Ma93] gave two cell loss rate figures, one due to input buffer overflow and the other due to output buffer overflow. The cell loss rate due to the input buffer overflow (1.5×10^{-5}) is negligible compared to the cell loss rate due to the output buffer overflow (2.2×10^{-1}). As can be seen in Table 5.9, all three new switches outperformed the switch by [Ma93]. They achieved better cell loss rate, and needed only one input buffer per node. As expected, the best performance was given by FAS-3. Due to the priority scheme used in the routing algorithm, the three new switches were still able to achieve very low cell loss rate for the time-sensitive Class-1 cells. The Class-1 cell loss rate figures with only eight output buffers were only slightly higher compared to the Class-1 cell loss rate figures with 32 output buffers per node.

Table 5.9. Cell Loss Rate Comparison of FAS-1/2/3 with [Ma93]
128x128 switch under uniform traffic and 0.9 loading.
I.B.L: Cell loss rate due to input buffer overflow
O.B.L: Cell loss rate due to output buffer overflow

	[Ma93]		FAS-1	FAS-2	FAS-3
	I.B.L.	O.B.L.			
Total	1.5×10^{-5}	2.2×10^{-1}	1.6×10^{-1}	8.8×10^{-2}	7.6×10^{-2}
Class 0			2.2×10^{-1}	1.3×10^{-1}	1.1×10^{-1}
Class 1			2.9×10^{-5}	4.9×10^{-6}	3.4×10^{-6}

The switch designed by [PaC95] used an n+1-dimensional hypercube to implement an $N\times N$ switch ($N = 2^n$). [PaC95] gave simulation results for a 32×32 switch (5-dimensional hypercube, a total of 64 nodes). The performance comparison among FAS-1, FAS-2, FAS-3 and [PaC95] is summarized in Table 5.10. The cell loss rate of FAS-1 was lower than that of the switch by [PaC95] with fewer total nodes (32 nodes) and smaller output buffers. Also, FAS-1 was able to keep the cell loss rate and the average cell delay (or ETE delay) of the time sensitive cells (Class-1) very low. FAS-2 also had a total of 64 nodes, but gave much lower total cell loss rate by a factor of 100 with smaller sized output buffers. The cells of the switch by [PaC95] experienced delay of about 25 time units as they traversed through the switch from input to output under uniform traffic pattern and 0.95 loading. However, the cells of the FAS-1 experienced average delay of about 8.5 time units the Class 1 cells experienced only about 6.5 time units. The cells of FAS-2 experienced lower end-to-end delay as the cells of FAS-1. FAS-3 gave the best performance again with zero cell loss for the Class-1 cells, and a very low overall cell loss rate.

The three new switches achieved these performance figures with fewer total buffers per node. The switch by [PaC95] needed 32 output buffers, but the three switches needed 27 or fewer output buffers per node.

Table 5.10. Comparison of FAS-1/2/3 with [PaC95] Under Uniform Traffic.
32x32 switch under 0.95 loading.
CLR: Cell Loss Rate; AED: Average End-to-End Delay.
IBS: Input Buffer Size; OBS: Output Buffer Size.

		FAS-1			FAS-2			FAS-3		
	[PaC95]	Total	Class 0	Class 1	Total	Class 0	Class 1	Total	Class 0	Class 1
CLR	7×10^{-3}	3.8×10^{-4}	5.1×10^{-4}	5.5×10^{-5}	1.1×10^{-5}	1.4×10^{-5}	2.1×10^{-6}	2.2×10^{-6}	3.1×10^{-6}	0
AED	25	8.54	9.43	6.50	8.25	8.85	6.86	8.02	8.55	6.79
IBS	2	2			1			1		
OBS	32	27			26			22		

[PaC95] also gave simulation results under bursty traffic pattern with m_A=5 (Bur-5) and m_A=10 (Bur-10). Table 5.11 and Table 5.12 show the comparison among FAS-1, FAS-2, FAS-3 and the switch by [PaC95] under these two traffic patterns. As seen in the tables, FAS-1 achieved similar overall cell loss rate, but better Class-1 cell loss rate compared to that of [PaC95]. FAS-3, which had the best performance in all three traffic patterns, achieved lower cell loss rate compared to that of [PaC95].

Table 5.11. Cell Loss Rate Comparison of FAS-1/2/3 with [PaC95] Under Bur-5.
32x32 switch under 0.8 loading; 2 input & 32 output buffers per node.

	[PaC95]	FAS-1	FAS-2	FAS-3
Total	2×10^{-3}	1.9×10^{-3}	3.4×10^{-3}	1.0×10^{-3}
Class 0		2.6×10^{-3}	3.6×10^{-3}	9.9×10^{-4}
Class 1		3.3×10^{-4}	2.8×10^{-3}	1.1×10^{-3}

Table 5.12. Cell Loss Rate Comparison of FAS-1/2/3 with [PaC95] Under Bur-10.
32x32 switch under 0.67 loading; 2 input & 32 output buffers per node.

	[PaC95]	FAS-1	FAS-2	FAS-3
Total	6×10^{-3}	7.3×10^{-3}	1.2×10^{-2}	5.8×10^{-3}
Class 0		7.5×10^{-3}	1.2×10^{-2}	5.5×10^{-3}
Class 1		1.5×10^{-3}	1.1×10^{-2}	6.4×10^{-3}

5.5 Other Class-0 to Class-1 Ratios

The simulation results presented so far were obtained with 70% of the cells being Class-0 and the remaining 30% cells to be Class-1 giving 7:3 Class-0 to Class-1 ratio. To see how the switches performed under different mixes of Class-0 and Class-1 cells, additional simulations were done. The ratios simulated and analyzed were 5:5 and 3:7. Table 5.13 through Table 5.16 show the simulation results of FAS-1 with different Class-0 cell to Class-1 cell ratios under uniform and normal traffic patterns. The ETE delay experienced by the Class-1 cells increased as the percentage of those cells in the traffic increased. This was because the queueing delay the cells experience in the output buffer increased. As more cells arrive at the destination, it took longer for each cell to be selected by the OBC and to be transmitted via the external output line.

Table 5.13. ETE Delay of 32x32 FAS-1 Under Uniform Traffic Pattern with Different Class-0 to Class-1 Ratios.

C0:C1	Loads	0.5	0.67	0.8	0.9
7:3	Class-0	7.36	8.13	8.73	9.19
	Class-1	5.84	6.10	6.32	6.42
5:5	Class-0	6.99	7.59	8.02	8.34
	Class-1	6.33	6.72	7.03	7.26
3:7	Class-0	6.63	7.04	7.42	7.67
	Class-1	6.88	7.50	8.03	8.44

Table 5.14. Cell Loss Rate of 32x32 FAS-1 Under Uniform Traffic Pattern with Different Class-0 to Class-1 Ratios.

C0:C1	Loads	0.5	0.67	0.8	0.9
7:3	Class-0	1.11E-04	2.02E-04	3.55E-04	4.28E-04
	Class-1	8.34E-06	2.40E-05	2.71E-05	3.23E-05
	Total	8.03E-05	1.49E-04	2.57E-04	3.10E-04
5:5	Class-0	1.02E-04	2.09E-04	3.24E-04	4.63E-04
	Class-1	2.25E-05	4.13E-05	7.50E-05	1.12E-04
	Total	6.25E-05	1.25E-04	1.99E-04	2.87E-04
3:7	Class-0	6.87E-05	1.53E-04	2.64E-04	3.24E-04
	Class-1	4.11E-05	8.57E-05	1.39E-04	1.68E-04
	Total	4.94E-05	1.06E-04	1.76E-04	2.15E-04

The Class-1 cell loss rate figures got higher as the percentage of the Class-1 cell increased. However, they were still lower than the Class-0 cell loss rate figures for all three different Class-0

to Class-1 ratios. This was because the OBC gave higher priority to the Class-1 cells. When a Class-1 cell arrived and the buffer was full, the OBC made room by removing a Class-0 cell. Under the uniform traffic pattern, the overall, Class-1, and Class-0 cell loss rate figures were similar for all three ratios. However, the best overall, Class-1, and Class-0 cell loss rate figures were obtained when equal number of Class-0 and Class-1 cells was injected into the switch.

Table 5.15. ETE Delay of 32x32 FAS-1 Under Normal Traffic Pattern with Different Class-0 to Class-1 Ratios.

C0:C1	Loads	0.5	0.67	0.8	0.9
7:3	Class-0	8.19	9.82	12.60	15.41
	Class-1	6.08	6.43	6.73	6.92
5:5	Class-0	7.52	8.34	8.97	9.45
	Class-1	6.74	7.35	7.87	8.26
3:7	Class-0	7.00	7.62	8.11	8.83
	Class-1	7.60	9.10	12.00	15.34

Table 5.16. Cell Loss Rate of 32x32 FAS-1 Under Normal Traffic Pattern with Different Class-0 to Class-1 Ratios.

C0:C1	Loads	0.5	0.67	0.8	0.9
7:3	Class-0	2.96E-04	6.06E-04	2.36E-03	9.26E-03
	Class-1	3.23E-05	8.55E-05	1.38E-04	1.85E-04
	Total	2.17E-04	4.50E-04	1.69E-03	6.54E-03
5:5	Class-0	2.91E-04	6.09E-04	9.60E-04	1.27E-03
	Class-1	7.50E-05	1.79E-04	2.65E-04	3.67E-04
	Total	1.83E-04	3.94E-04	6.12E-04	8.20E-04
3:7	Class-0	1.95E-04	6.24E-04	9.94E-03	4.40E-02
	Class-1	1.33E-04	2.63E-04	2.46E-03	1.26E-02
	Total	1.51E-04	3.71E-04	4.70E-03	2.20E-02

5.6 Summary

The three new ATM switches based on the FHC network, FAS-1, FAS-2 and FAS-3, were simulated and their performances were analyzed in this chapter. Four different traffic patterns were used in the simulation study. Analysis of the simulation results showed the three switches to have good performances in terms of end-to-end cell delay and cell loss rate. Also, the performance of the three switches were favorable when compared to two hypercube switches in the literature.

FAS-1, FAS-2, and FAS-3 outperformed the switches in [Ma93, PaC95] with only slight increase in hardware cost. While the switches by [Ma93, PaC95] did not implement any time-based priority, FAS-1, FAS-2, and FAS-3 were implemented with two-level time-based priority giving higher priority to time-sensitive cells. The benefit of this priority scheme was shown in the simulation results. The cells loss rate and the average cell delay of the time-sensitive cells were much lower than those of the time-insensitive cells. Even when the loading neared 100%, all three switches still managed to achieve good cell loss rate for the time-sensitive Class-1 cells.

When different Class-0 to Class-1 ratios were simulated, the three switches still performed well giving better performance for Class-1 cells. This showed that the priority scheme used by the routing algorithm and the OBC performed well even when the majority of the traffic was Class-1 cells. These results indicate that FAS-1, FAS-2 and FAS-3 are good candidates for new high-speed ATM switches. In particular, FAS-3 is the best candidate since it gave the lowest ETE delay and the lowest cell loss rate for all traffic patterns under low to high loads.

6. Conclusions

As outlined in Chapter 1, the motivation for this research was to design new ATM switches that can provide high performance with low hardware complexity. There are many ATM switches proposed in the literature. Most of these are space-division switches based on multistage interconnection networks (MINs). These switches have high performance, but suffer from high hardware complexity. The three new switches proposed and studied in this research were based on the hypercube network, a single-stage interconnection network. By taking advantage of the self-routing and multi-path properties of the FHC, the new switches were shown to achieve high performance in terms of cell delay and cell loss rate while having low hardware complexity than most of the MIN-based switches.

6.1 Summary of the Research

Chapter 2 gave an overview of space-division ATM switches proposed in the literature. First, different buffering strategies were discussed. The output buffering has a higher throughput than the input buffering under any traffic model. This disadvantage of the input buffering is due to Head of Line (HOL) blocking. Then, various switch architectures were presented. Based on the switching fabrics used, the switches can be categorized into three different groups: crossbar-based, disjoint-path-based, and Banyan network-based switches. Different architectures have different performance levels and complexity, and the switches with better throughput or cell loss performance tend to be more complex to implement.

Most of the crossbar-based switches are limited to small switch sizes due to their square growth of complexity. They also require a large number of buffers for high performance and suffer from output blocking. There are two types of path-disjoint switches: path-disjoint in time and path-disjoint in space. The main disadvantage of the switches that are path-disjoint in time is that the switching fabrics need to operate N times faster (for an $N \times N$ switch) than the external lines. Like the crossbar-based switches, this disadvantage makes building large switches impractical. The switches that are path-disjoint in space, such as the Knockout switch [YeH87], do not need switching fabrics that operate faster than the external line. However,

they are also impractical for large switches due to their high complexity. Some variations of the Knockout switches, like the GAUSS switch [DeV90] and the Cylinder switch [MoP90] are less complex, but they still require the shift register or the shift register ring to run faster than the external lines.

As can be seen in Chapter 2, the majority of the space-division ATM switches in the literature are based on Banyan networks (or multistage interconnection networks). This is because the Banyan networks have several attractive properties that were given in Chapter 2. However, they still have major disadvantages, such as internal link blocking and the output port blocking. To overcome these problems and to improve the performance of the Banyan network, most of the switches use multiple Banyan networks or extra redundant stages of switching elements. Using extra stages or multiple Banyan networks also results in higher delay through the switches. Therefore, there is a need for switch designs with low hardware complexity and high performance. Recently, in order to design less complex switches, there has been a move toward switch designs based on another type of interconnection network, specifically, the hypercube network. This can be attributed to the low complexity, large bandwidth, and inherent self-routing capability of the hypercube.

In Chapter 3, two ATM switches based on the hypercube networks were discussed. These switches were shown to have comparable performance but have lower hardware complexity when compared to many Banyan-based switches. Different variations of the hypercube were also given in Chapter 3. These variations further improve the characteristics and performance of the hypercube. One of these variations, the folded hypercube (FHC) was shown to have better performance than the hypercube with minimal increase in the hardware complexity. Chapter 3 also presented the validation of the backbone simulation model of the FHC. The modeling validation was done using OPNET Modeler. The simulation results agreed with published analytical results.

Chapter 4 proposed three new ATM switches, FAS-1, FAS-2, and FAS-3, based on the FHC. A detailed discussion of the architecture of the three new switches was given. This was the first time that an FHC is used as a basis for ATM switch. The main advantage of the hypercube variations over the regular hypercube is their reduced network diameter. However, the FHC and BIN keep the symmetric characteristic of the hypercube while the twisted

hypercubes are asymmetric. The FHC was chosen over the BIN due to its simpler routing algorithm. As shown in Chapter 3, the FHC was shown to have better performance than the hypercube with minimal increase in the hardware complexity.

Chapter 5 gave the simulation results of the three new switches. The three switches were modeled and simulated using OPNET Modeler. The two major performance parameters measured were the cell loss rate and cell delay through the switch. These parameters were measured under uniform, normal, and bursty traffic patterns. Compared to the ATM switches based on the multistage interconnection networks, the three proposed switch architectures had lower hardware complexity while having equivalent or better switching performance. The cell delays through the three new switches were smaller when compared to the two switches based on the hypercube or the Banyan type networks.

6.2 Summary of the Results

Three different traffic patterns were used in the simulation study: uniform, normal, and bursty traffic. For bursty traffic, the traffic was modeled by an *on-off* source at each input node [BaM94]. The number of cells generated during a burst was geometrically distributed over two means. Two mean active durations (m_A) were used: m_A=5 (Bur-5) and m_A=10 (Bur-10). Analysis of the simulation results showed the three switches to have very good performance in terms of average cell or end-to-end (ETE) delay and cell loss rate. Also, the performance of the three switches was favorable when compared to other space-division switches in the literature.

When compared to some of the other space-division switches in the literature, the three new switches gave comparable or better performance figures with lower hardware complexity. For N=128 and 0.5 loading of uniform traffic, the Multi Single-Stage Shuffling Switch (MS4) and Modified Shuffle-Exchange Network (MSN) needed 24 stages to achieve a cell loss rate of 10^{-5}. That is a lot of extra stages considering that a regular 128×128 Banyan network has only 7 stages. Under the same condition, the total cell loss rate was 6.9×10^{-6} for FAS-1 and zero for FAS-2 and FAS-3. For N=1024 and 0.5 loading of normal traffic, the Tandem-Banyan Switch (TB) and MS4 needed about 100 stages and 43 stages, respectively, to achieve a cell loss rate of 10^{-6}. Again, considering that a regular 1024×1024 Banyan network has only 10 stages, TB and MS4 needed four to ten times more stages. Under the same condition, the total cell loss rate

was 5.5×10^{-6} for FAS-1, 1.3×10^{-7} for FAS-2, and 3.9×10^{-8} for FAS-3. Other high performance Banyan-based switches, such as the Starlite, Sunshine, and Batcher-Banyan switches, use multiple copies of a Banyan network in their architecture. However, only one copy of an $N\times N$ FHC is needed to implement an $N\times N$ FAS-1, only one copy of a $2N\times 2N$ FHC is needed to implement an $N\times N$ FAS-2, and only one copy of an $N\times N$ FHC and $\log_2 N$ additional links are needed to implement an $N\times N$ FAS-3.

Also, FAS-1, FAS-2, and FAS-3 outperformed the two hypercube-based switches proposed by Matsunaga [Ma93], and Pao and Chau [PaC95]. For N=128 and 0.9 loading of uniform traffic with two input and eight output buffers per node, the cell loss rate of [Ma93] was 0.22 and the total cell loss rate of FAS-1 was 0.16. Matsunaga used a 128×128 hypercube and FAS-1 used a 128×128 FHC. FAS-1 needed only seven extra links to provide better performance. The total cell loss rates of FAS-2 and FAS-3 were 0.088 and 0.076, respectively, but needed more hardware since both used a 256×256 FHC. For N=32 and 0.95 loading of uniform traffic, the cell loss rate of [PaC95] was 7×10^{-3} with two input and 32 output buffers per node. Under the same condition, the total cell loss rate of FAS-1, FAS-2 and FAS-3 were 3.8×10^{-4}, 1.1×10^{-5}, and 2.2×10^{-6}, respectively. Pao and Chau used a 64×64 hypercube, FAS-1 used a 32×32 FHC, and FAS-2 and FAS-3 used a 64×64 FHC. FAS-2 and FAS-3 needed several extra links compared to [PaC95], but gave much better performance. Furthermore, FAS-1 used almost half the hardware, but outperformed the switch by Pao and Chau.

Another advantage of the three new switches besides the low ETE delay, cell loss rate, and hardware complexity, is that they give higher priority time-sensitive Class-1 cells, such as audio and video cells, over the non-time-sensitive (Class-0) cells. Most of the switches in the literature, including the two switches by [Ma93, PaC95], do not give any priority to time-sensitive cells. FAS-1, FAS-2, and FAS-3 implemented this priority scheme in the routing algorithm and the output buffer controllers (OBC). This priority scheme allowed the Class-1 cells to be switched faster than the Class-0 cells. The benefit of this priority scheme was shown in the simulation results. The cell loss rate and the average ETE delay of the Class-1 cells were much lower than those of the Class-0 cells. Even when the loading neared 100%, all three switches still managed to achieve good cell loss rates for Class-1 cells.

The three new switches performed well when different Class-0 to Class-1 ratios were simulated, giving better performance figures to Class-1 cells. This showed that the priority scheme used by the routing algorithm and the OBC performed well even when the majority of the traffic was Class-1 cells. These results indicate that FAS-1, FAS-2 and FAS-3 are good candidates as new high-speed ATM switches. In particular, FAS-3 is the best candidate since it gave the lowest ETE delay, and the lowest cell loss rate for all traffic patterns under low to high loads.

6.3 Contributions

This study showed a new approach in designing ATM switches by using the FHC as the switching fabrics for the first time instead of using the crossbar, multi-path, or Banyan based switching fabrics. Three new high performance ATM switches based on the FHC were proposed. A new routing algorithm, Dynamic Distributed Routing Algorithm, was developed for the new switches. The algorithm incorporated deflection routing and three-level priority scheme to route the cells quickly through the switch while giving priority to real-time cells.

To manage the output buffer space efficiently, output buffer controller (OBC) was developed. The OBC maintained fair sharing of the available buffer space, and in addition, the OBC re-sequenced the cells before they were transmitted via the external output lines. The results discussed in Chapter 5 and Appendix showed the effectiveness of the routing algorithm and the output buffer controller. Finally, the routing algorithm and the OBC can easily modified and updated to handle different types of traffic for future studies that are outlined in the next section.

6.4 Possible Future Work

One extension of this research can be studying the performance under heterogeneous traffic. This research only studied homogeneous traffic, one type of traffic pattern throughout the simulation whether it was uniform, normal, Bur-5 or Bur-10. Simulation study can be done with mixed traffic of uniform, normal, Bur-5 and Bur-10 during the same simulation. During a simulation a subset of input nodes can be set to create cells of a particular traffic. Also, the input nodes can be modified to change traffic pattern at random times during a

simulation.Lately, the need for broadcast and multicast over the high-speed networks is increasing. And therefore, good broadcast and multicast switches are in demand. A second possibility of future work is the implementation of broadcast and multicast in the three new switches. The FHC has the broadcast capability and the designers of the FHC have given an algorithm to broadcast cells over the FHC in their papers [ElL91, LaE89]. Adding multicast capability to the new switches by modifying the current routing algorithm will make the switches more attractive.

Another extension of this research can be modifying the new switches for networks with non-constant packet length. The current design is for an ATM environment where all cells are at constant 53 bytes. One possible way of accomplishing this is to add a mechanism that divides different length packets to multiple 53 bytes cells before admitting them to the switch. This division of packets will lead burst of cells. In other words, even if packets entering the switch are uniformly distributed, once they are divided into cells, they will become bursty traffic pattern within the switch. Performance may or may not be degraded depending on the traffic pattern, the value of *Time_Limit*, and the number of buffers used.

The second way of handling the varying packet length is to modify the switches to handle them. The link selection process within the routing algorithm and the output buffer controller need to be modified to handle variable-length packets. The length of the packets will be another factor in link selection process besides the class, age, and distance to destination variables. Like the first method, performance of the switches should depend on the traffic pattern, the value of *Time_Limit*, and the number of buffers.

For both methods, a distribution needs to be used to randomly generate the lengths of the packets. After the modification and analyses of the simulation results, the research can be further extended by studying the effects of heterogeneous traffic of variable-length packets.

References

[Aav92] E. Aanen, J. L. van den Berg, and R. J. F. De Vries, "Cell loss performance of the GAUSS ATM switch," *Proceedings of IEEE INFOCOM'92*, pp. 717-726, 1992.

[AbP89] S. Abraham and K. Padmanabhan, "An analysis of the twisted cube topology," *International Conference on Parallel Processing*, vol. 1, pp. 116-120, 1989.

[AhD89] H. Ahmadi and W. E. Denzel, "A survey of modern high-performance switching techniques," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 7, pp. 1091-1103, Sept. 1989.

[AlK99a] M. Al-Mouhamed and M. Kaleemuddin, "Evaluation of pipelined Banyan switch architecture for ATM networks," *Proceedings of IEEE Symposium on Computers & Communications*, pp. 266-272, 1999.

[AlK99b] M. Al-Mouhamed and M. Kaleemuddin, "Evaluation of pipelined dilated Banyan switch architecture for ATM networks," *IEEE/ACM Transactions on Networking*, vol. 7, no. 5, pp. 724-740, Oct. 1999.

[AlY91] M. K. M. Ali and M. Youssefi, "The performance of an input access scheme in a high-speed packet switch," *Proceedings of IEEE INFOCOM'91*, pp. 454-161, 1991.

[AtA94] M. Atiquzzaman and M. S. Akhtar, "Effect of nonuniform traffic on the performance of multistage interconnection networks," *IEE Proceedings-Computers and Digital Techniques*, vol. 141, no. 3, pp. 169-176, May 1994.

[AwM94a] R. Y. Awdeh and H. T. Mouftah, "A comparative study of unbuffered interconnection networks based on crosspoint complexity," *Proceedings of 6th IEEE International Conference on Computing and Information*, pp. 604-621, 1994.

[AwM94b] R. Y. Awdeh and H. T. Mouftah, "A contention resolution algorithm for input-buffered Batcher-Banyan networks," *International Journal on Communication Systems*, vol. 7, no. 1, pp. 33-38, Jan.-Mar. 1994.

[AwM95a] R. Y. Awdeh and H. T. Mouftah, "MS4-A high performance output buffering ATM switch," *Computer Communications*, vol. 18, no. 9, pp. 631-644, Sept. 1995.

[AwM95b] R. Y. Awdeh and H. T. Mouftah, "Survey of ATM switch architectures," *Computer Networks and ISDN Systems*, vol. 27, no. 12, pp. 1567-1613, Nov. 1995.

[Ba68] K. E. Batcher, "Sorting networks and their applications," *Proceedings of AFIPS Spring Joint Comp. Conf.*, vol. 32, pp. 307-314, 1968.

[BaD92] S. Bassi, M. Decina, and A. Pattavina, "Performance analysis of the ATM Shuffleout switching architecture under nonuniform traffic patterns," *Proceedings of IEEE INFOCOM'92*, pp. 734-742, 1992.

[BaM91] H. F. Badran and H. T. Mouftah, "Head of line arbitration in ATM switches with input-output buffering and backpressure control," *Proceedings of IEEE GLOBECOM'91*, pp. 347-351, 1991.

[BaM93] H. F. Badran and H. T. Mouftah, "Input-output-buffered broad-band packet-switch architectures with correlated input traffic," *Canadian Journal of Electrical and Computer Engineering*, vol. 18, no. 3, pp. 133-139, 1993.

[BaM94] H. F. Badran and H. T. Mouftah, "ATM Switch architecture with input-output-buffering: Effect of input traffic correlation, contention resolution policies, buffer allocation strategies and delay in backpressure signal," *Computer Networks and ISDN Systems*, vol. 26, no. 9, pp. 1187-1213, 1994.

[Be64] V. E. Benes, "Optimal rearrangeable multistage connecting networks," *Bell Systems Technical Journal*, vol. 43, pp. 1641-1656, July 1964.

[BeA99] T. Bearly and J. P. Agrawal, "Split input Shunshine switch architecture," *International Journal of Communication Systems*, vol. 2, no. 5-6. pp. 427-438, Sept.-Dec. 1999.

[BhA84] L. N. Bhuyan and D. P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Transactions on Computers*, vol. 33, no. 4, pp. 323-333, April 1984.

[BhY89] L. N. Bhuyan, Q. Yang, and D. P. Agrawal, "Performance of multiprocessor interconnection networks," *IEEE Computer*, vol. 22, no. 2, pp. 25-37, Feb. 1989.

[BiP96] G. Bianchi and A. Pattavina, "Architecture and performance of non-blocking ATM switches with shared internal queueing," *Computer Networks and ISDN Systems*, vol. 28, no. 6, pp. 835-853, April 1996.

[BrH83] G. Broomell and J. R. Heath, "Classification categories and historical development of circuit switching topologies," *Computing Surveys*, vol. 15, no. 2, pp. 95-133, June 1983.

[BuT89] R. S. Bubenik and J. S. Turner, "Performance of a broadcast packet switch," *IEEE Transactions on Communications*, vol. 37, no. 1, pp. 60-69, Jan. 1989.

[Ça99] H. Çam, "A high-performance ATM switch based on modified shuffle-exchange network," *Computer Communications*, vol. 22, no. 2, pp. 110-119, Jan. 1999.

[CaP91] P. Campoli and A. Pattavina, "An ATM switch with folded shuffle-topology and distributed access," *Proceedings of IEEE ICC'91*, pp. 1021-1027, 1991.

[Ch90] H. J. Chao, "A distributed modular tera-bit/sec ATM switch," *Proceedings of IEEE GLOBECOM'90*, pp. 1594-1601, 1990.

[Ch91] X. Chen, "A survey of multistage interconnection networks for fast packet switches," *International Journal on Digital and Analog Communication Systems*, vol. 4, pp. 33-59, 1991.

[Ch95] F. B. Chedid, "On the generalized twisted cube," *Information Processing Letters*, vol. 55, no. 1, pp. 49-52, July 7, 1995.

[Ch00] H. J. Chao, "Saturn: A terabit packet switch using dual round-robin," *Proceedings of IEEE GLOBECOM'2000*, pp. 487-495, 2000.

[ChC93] F. B. Chedid and R. B. Chedid, "A new variation on hypercubes with smaller diameter," *Information Processing Letters*, vol. 46, no. 6, pp. 275-280, July 26, 1993.

[ChH96] A. K. Choudhury and E. L. Hahne, "Dynamic queue length thresholds in a shared memory ATM switch," *Proceedings of IEEE INFOCOM'96*, pp. 679-685, 1996.

[ChM93] D. X. Chen and J. W. Mark, "SCOQ: a fast packet switch with shared concentration and output queueing," *IEEE/ACM Transactions on Networking*, vol. 1, no. 1, pp. 142-151, Feb. 1993.

[CoK96] B. R. Collier and H. S. Kim, "Efficient analysis of shared buffer management strategies in ATM networks under non-uniform bursty traffic," *IEEE INFOCOM'96*, pp. 671-678, 1996.

[CuL91] P. Cull and S. Larson, "The Möbius cubes," *Distributed Memory Computing Conference*, pp. 699-702. 1991.

[CuL92] P. Cull and S. Larson, "The Möbius cubes: Improved cubelike networks for parallel computation," *International Parallel Processing Symposium*, pp. 610-613, 1992.

[CuL93] P. Cull and S. M. Larson, "Static and dynamic performance of the Möbius cubes (Short Version)," *PARLE'93: Parallel Architectures and Languages Europe*, pp. 92-103, 1993.

[CuL95] P. Cull and S. M. Larson, "On generalized twisted cubes," *Information Processing Letters*, vol. 55, no. 1, pp. 53-55, July 7, 1995.

[De90] R. J. F. De Vries, "GAUSS: a single-stage ATM switch with output buffering," *Proceedings of IEE International Conference on Integrated Broadband Services and Networks*, pp. 248-252, 1990.

[De91] M. De Prycker, *Asynchronous Transfer Mode: Solution for Broadband ISDN*, EllisHorwood, Chichester, 1991.

[DeF93] E. Del Re and R. Fantacci, "Performance evaluation of input and output queueing techniques in ATM switching systems," *IEEE Transactions on Communications*, vol. 41, no. 10, pp. 1565-1575, Oct. 1993.

[DeG91a] M. Decina, P. Giacomazzi, A. Pattavina, and E. Tombolini, "Shuffle interconnection networks with deflection routing for ATM switching: the Open-Loop Shuffleout," *Proceedings of ITC-13*, pp. 27-34, 1991.

[DeG91b] M. Decina, P. Giacomazzi, A. Pattavina, and E. Tombolini, "Shuffle interconnection networks with deflection routing for ATM switching: the Closed-Loop Shuffleout," *Proceedings of ITC-13*, pp. 1256-1263, 1991.

[DeJ86] S. R. Deshpande and R. M. Jenevein, "Scalability of a binary tree on a hypercube," *International Conference on Parallel Processing*, pp. 661-668, 1986.

[DiJ81a] D. M. Dias and J. R. Jump, "Analysis and simulation of buffered delta networks," *IEEE Transactions on Computers*, vol. 30, no. 4, pp. 273-282, Apr. 1981.

[DiJ84] D. M. Dias and J. R. Jump, "Packet switching in NlogN multistage networks," *Proceedings of IEEE GLOBECOM'84*, pp. 114-120, 1984.

[Ef89] K. Efe, "Programming the twisted-cube architectures," *International Conference on Distributed Computing Systems*, pp. 254-262, 1989.

[Ef91] K. Efe, "A variation on the hypercube with lower diameter," *IEEE Transactions on Computers*, vol. 40, no. 11, pp. 1312-1316, Nov. 1991.

[EfB88] K. Efe, P. Blackwell, T. Shiau, and W. Slough, "A reduced diameter interconnection network," *Symposium on the Frontiers of Massively Parallel Computers*, pp. 471-474, 1988.

[ElL91] A. El-Amawy and S. Latifi, "Properties and performance of folded hypercubes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 2, no. 1, pp. 31-42, Jan. 1991.

[EsN88] A. Esfahanian, L. M. Ni, and B. E. Sagan, "On enhancing hypercube multiprocessors," *International Conference on Parallel Processing*, vol. 1, pp. 86-89, 1988.

[EsN91] A. Esfahanian, L. M. Ni, and B. E. Sagan, "The twisted *N*-cube with application to multiprocessing," *IEEE Transactions on Computers*, vol. 40, no. 1, pp. 88-93, Jan. 1991.

[Fe81] T.-Y. Feng, "A survey of interconnection networks," *IEEE Computer*, vol. 14, no. 12, pp. 12-27, Dec. 1981.

[GhB89] A. Ghafoor, T. R. Bashkow, and I. Ghafoor, "Bisectional fault-tolerant communication architecture for supercomputer systems," *IEEE Transactions on Computers*, vol. 38, no. 10, pp. 1425-1446, Oct. 1989.

[GhD91] D. Ghosh and J. C. Daly, "Delta networks with multiple links and shared output buffers: a high performance architecture for packet switching," *Proceedings of IEEE GLOBECOM'91*, pp. 949-953, Dec. 1991.

[GiH91] J. N. Giacopelli, J. J. Hickey, W. S. Marcus, W. D. Sincoskie, and M. Littlewood, "Sunshine: a high-performance self-routing broadband packet switch architecture," *IEEE Journal on Selected Areas of Communications*, vol. 9, no. 8, pp. 1289-1298, Oct. 1991.

[GoL73] L. R. Goke and G. J. Kipovski, "Banyan networks for partitioning multiprocessor systems," *Proceedings of 1st Annual Symposium on Computer Architecture*, pp. 21-28, 1973.

[GrH82] A. G. Greenberg and B. Hajek, "Deflection routing in hypercube networks," *IEEE Transactions on Communications*, vol. 40, no. 6, pp. 1070-1081, June 1982.

[GuB91] A. K. Gupta, L. O. Barbosa, and N. D Georganas, "A 16×16 limited intermediate buffer switch module for ATM networks, *Proceedings of IEEE GLOBECOM'91*, pp. 939-943, 1991.

[HiK87] P. A. J. Hilbers, M. R. J. Koopman, and J. L. A. van de Snepscheut, "The twisted cube," *PARLE: Parallel Architecture and Languages Europe, Vol. 1: Parallel Architectures*, pp. 152-158, 1987.

[HlK88] M. G. Hluchyj and M. J. Karol, "Queueing in high-performance packet switching," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1587-1597, Dec. 1988.

[Ho92] C. Ho, "An observation on the bisectional interconnection networks," *IEEE Transactions on Computers*, vol. 41, no. 7, pp. 873-877, July 1992.

[HuA87] J. Y. Hui and E. Arthurs, "A broadband packet switch for integrated transport," *IEEE Journal on Selected Areas of Communications*, vol. 5, no. 8, pp. 1264-1273, Oct. 1987.

[HuK84] A. Huang and S. Knauer, "Starlite: a wideband digital switch," *Proceedings of IEEE GLOBECOM'84*, pp. 121-125, 1984.

[ImU88] H. Imagawa, S. Urushidani, and K. Hagishima, "A new self-routing switch driven with input-to-output adress difference," *Proceedings of IEEE GLOBECOM'88*, pp. 1607-1611, 1988.

[IyE89] A. Iyengar and M. El Zarki, "Switching prioritized packets," *Proceedings of IEEE GLOBECOM'89*, pp. 1181-1186, 1989.

[KaH87] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input versus output queueing on a space-division packet switch," *IEEE Transactions on Communications*, vol. 28, no. 7, pp. 992-1003, Dec. 1980.

[Kr00] B. Kraimeche, "Design and analysis of the Stacked-Banyan ATM switch fabric," *Computer Networks*, vol. 32, no. 2, pp. 171-184, 2000.

[KrS83] C. P. Kruskal and M. Snir, "The performance of multistage interconnection networks for multiprocessors," *IEEE Transactions on Computers*, vol. 32, no. 12, pp. 1091-1098, Dec. 1983.

[KuJ86] M. Kumar and J. R. Jump, "Performance of unbuffered shuffle-exchange networks," *IEEE Transactions on Computers*, vol. 35, no. 6, pp. 573-577, June 1986.

[La75] D. Lawrie, "Access and alignment of data in an array processor," *IEEE Transactions on Computers*, vol. 24, no. 12, pp. 1145-1155, Dec. 1975.

[La90] S. Latifi, "Fault-tolerant hypercube multiprocessors," *IEEE Transactions on Reliability*, vol. 29, no. 3, pp. 361-368, Aug. 1990.

[LaE89] S. Latifi and A. El-Amawy, "On folded hypercubes," *International Conference on Parallel Processing*, vol. 1, pp. 180-187, 1989.

[LaL90] P. S. Y. Lau and A. Leon-Garcia, "Design and performance analysis of a buffer subsystem for the Batcher-Banyan switch," *Proceedings of IEEE GLOBECOM'90*, pp. 1926-1930, 1990.

[Le88] T. T. Lee, "Nonblocking copy networks for multicast packet switching," *IEEE Journal on Selected Areas of Communications*, vol. 6, no. 9, pp. 1455-1467, Dec. 1988.

[Le90a] C.-L. Lea, "Design and performance evaluation of unbuffered self-routing networks for wide band packet switching," *Proceedings of IEEE INFOCOM'90*, pp. 148-156, 1990.

[Le90b] T. T. Lee, "A modular architecture for very large packet switches," *IEEE Transactions on Communications*, vol. 38, no. 7, pp. 1097-1106, July 1990.

[LeL92] T. T. Lee and S. C. Liew, "Broadband packet switches based on dilated interconnection networks," *Proceedings of IEEE ICC'92*, pp. 255-261, 1992.

[Li90] S. C. Liew, "Performance of input-buffered and output buffered ATM switches under bursty traffic: simulation study," *Proceedings of IEEE GLOBECOM'90*, pp. 1919-1925, 1990.

[Li94] S. C. Liew, "Performance of various input-buffered and output-buffered ATM switch design principles under bursty traffic: simulation study," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 1371-1379, 1994.

[LiL92] S. C. Liew and T. T. Lee, "NlogN dual shuffle-exchange network with error correcting routing," *Proceedings of IEEE ICC'92*, pp. 262-268, 1992.

[Ma93] T. Matsunaga, "Sorting-based routing algorithms of a photonic ATM cell switch: HiPower," *IEEE Transactions on Communications*, vol. 41, no. 9, pp. 1356-1363, Sept. 1993.

[McM84] R. J. McMillen, "A survey of interconnection networks," *Proceedings of IEEE GLOBECOM'84*, pp. 105-113, Nov. 1984.

[MoP90] B. Monderer, G. Pacifici, and C. Zukowski, "The cylinder switch: an architecture for a manageable VLSI giga-cell switch," *Proceedings IEEE ICC'90*, pp. 567-571, 1990.

[Na88] M. Narashimha, "The Batcher-Banyan self-routing network: universality and simplification, *IEEE Transactions on Communications*, vol. 36, no. 10, pp. 1175-1181, Oct. 1988.

[Ne88] P. Newman, "A fast packet switch for the integrated services backbone network," *IEEE Journal on Selected Areas of Communications*, vol. 6, no. 9, pp. 1468-1479, Dec. 1988.

[NiS00] Y. Nishino and I. Sasase, "Knockout ATM switch with two speedup factors under non-uniform traffic with variable hot-spot ports," *IEICE Transactions on Fundamentals of Electronics Communications & Computer Sciences*, no. 10, pp. 1936-1944, Oct. 2000.

[NoT87] S. Nojima, E. Tsutsui, H. Fukuda, and M. Hashimoto, "Integrated services packet network using bus-matrix switch," *IEEE Journal on Selected Areas of Communications*, vol. 5, no. 10, pp. 1284-1292, Oct. 1987.

[OkÇ97] S. F. Oktuğ and M. U. Çağlayan, "Design and performance evaluation of a Banyan network based interconnection structure for ATM switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 5, pp. 807-816, June 1997.

[Ok01] S. F. Oktuğ, "A Multicast ATM switch Based on PIPN," *Proceedings of International Conference on Networking'01*, pp. 390-398, 2001.

[OP99a] OPNET Modeling Manual Vol. 1, OPNET Version 6.0, OPNET Technologies, Inc., 1999.

[OP99b] OPNET External Interfaces Manual, OPNET Version 6.0, OPNET Technologies, Inc., 1999.

[Pa81] J. H. Patel, "Performance of processor-memory interconnection network," *IEEE Transactions on Computers*, vol. 30, no. 10 pp. 771-780, Oct. 1981.

[Pa89] A. Pattavina, "Fairness in a broadband packet switch," *Proceedings of IEEE ICC'89*, pp. 404-409, 1989.

[Pa90a] A. Pattavina, "A broadband packet switch with input and output queueing," *Proceedings of International Switching Symposium*, pp. 11-16, 1990.

[Pa90b] A. Pattavina, "Design and performance evaluation of a packet switch for broadband central offices," *Proceedings of IEEE INFOCOM'90*, pp. 1252-1259, 1990.

[Pa94] J. S. Park, *Performance analysis of partitioned multistage cube network and adaptive routed single-stage cube network*, Master's Thesis, Virginia Polytechnic Institute and State University, 1994.

[PaC95] D. C. W. Pao and W. N. Chau, "Design of ATM switch using hypercube with distributed shared input buffers and dedicated output buffer," *International Conference on Network Protocols*, pp. 92-99, 1995.

[PaD99] J. S. Park and N. J. Davis IV, "Modeling the Folded Hypercube Network with OPNET," *The Proceedings of OPNETWORK '99*, Washington DC, August 1999.

[PaY99] J. Park, H. Yoon, and H. Lee, "The deflection self-routing Banyan network: A large-scale ATM switch using the fully adaptive self-routing and its performance analyses," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 588-604, Aug. 1999.

[Pe77] M. C. Pease, "The Indirect Binary n-Cube multiprocessor array," *IEEE Transactions on Computers*, vol. 26, no. 5, pp. 458-473, May 1977.

[PrV81] F. P. Preparata and J. Vuillemin, "The cube-connected cycles: A versatile network for parallel computation," *Communications of ACM*, vol. 24, no. 5, pp. 300-309, May 1981.

[SaH99] K. Sakamoto, S. Hiyama, Y. Nishino, and I. Sasase, "A large scalable knockout dilated Banyan multicast ATM switch," *Proceedings of IEEE GLOBECOM'99*, pp. 1418-1422, 1999.

[SaS88] Y. Saad and M. H. Schultz, "Topological properties of hypercubes," *IEEE Transactions on Computers*, vol. 37, no. 7, pp. 867-872, July 1988.

[Si85] H. J. Siegel, *Interconnection Networks for Large Parallel Processing*. Lexington, MA: Lexington Books, 1985.

[SiG95] N. K. Singhvi and K. Ghose, "The Mcube: A symmetrical cube based network with twisted links," *International Parallel Processing Symposium*, pp. 11-16, 1995.

[SiZ93] S. Sibal and J. Zhang, "On a class of Banyan networks and tandem Banyan switching fabrics," *Proceedings of IEEE INFOCOM'93*, pp. 481-488, 1993.

[St71] H. S. Stone, "Parallel processing with the perfect shuffle," *IEEE Transactions on Computer*, vol. 20, no. 2, pp. 153-161, Feb. 1971.

[SzS89] T. Szymanski and S. Shaikh, "Markov chain analysis of packet-switched Banyans with arbitrary switch sizes, queue sizes, link multiplicities and speedup," *Proceedings of IEEE INFOCOM'89*, pp. 960-971, 1989.

[To90] F. A. Tobagi, "Fast packet switch architectures for broadband integrated services digital networks," *Proceedings of IEEE*, vol. 78, no. 1, pp. 133-166, Jan. 1990.

[ToK91] F. A. Tobagi, T. Kwok, and F. M. Chiussi, "Architecture, performance, and implementation of the tandem-Banyan fast packet switch," *IEEE Journal on Selected Area of Communications*, vol. 9, no. 8, pp. 1173-1193, Oct. 1991.

[Tu88] J. S. Turner, "Design of a broadcast packet switching network," *IEEE Transactions on Communications*, vol. 36, no. 6, pp. 734-743, June 1988.

[TzP99] N. Tzeng, K. Ponnuru, and K. Vibhatavanij, "A cost-effective design for ATM switching fabrics," *Proceedings of IEEE International Conference on Communications*, pp. 1468-1472, 1999.

[VaB94] E. A. Varvarigos and D. P. Bertsekas, "Performance of hypercube routing schemes with or without buffering," *IEEE/ACM Transactions on Networking*, vol. 2, no. 3, pp. 299-311, June 1994.

[WuF80] C.-L. Wu and T.-Y. Feng, "On a class of multistage interconnection networks," *IEEE Transactions on Computers*, vol. 29, no. 8, pp. 694-702, Aug. 1980.

[YeH87] Y. S. Yeh, M. G. Hluchyj, and A. S. Acampora, "The Knockout switch: A simple, modular, architecture for high-performance packet switching," *IEEE Journal on Selected Areas Communications*, vol. 5, no. 8, pp. 1274-1283, Oct. 1987.

[YoE99] M. A. Youssef, M. N. El-Derini, and H. H. Aly, "Structure and performance evaluation of a replicated Banyan network based ATM switch," *Proceedings of IEEE Symposium on Computers & Communications*, pp 258-265. 1999.

[YoL88] H. Yoon, M. T. Liu, and K. Y. Lee, "The Knockout switch under nonuniform traffic," *Proceedings of IEEE GLOBECOM'88*, pp. 1628-1634, 1988.

[YoU93] Y. S. Youn and C. K. Un, "Performance of dilated Banyan network with recirculation," *Electronics Letters*, vol. 29, no. 1, pp. 62-63, Jan. 1993.

[ZaM93a] R. Zarour and H. T. Mouftah, "Bridged Shuffle-Exchange Network: a high performance self-routing ATM switch," *Proceedings of IEEE ICC'93*, pp. 696-700, 1993.

[ZaM93b] R. Zarour and H. T. Mouftah, "The Closed-Loop Bridged Shuffle-Exchange Network: a high performance self-routing ATM switch," *Proceedings of IEEE GLOBECOM'93*, pp. 1164-1168, 1993.

Appendix. Simulation Results

This appendix contains simulation results for FAS-1, FAS-2, and FAS-3 for switch sizes of 8×8, 16×16, 64×64, 128×128, 256×256, 512×512, and 1024×1024, under uniform, normal, and bursty traffic. For all results, the ratio between the non-real-time Class-0 cells to real-time Class-1 cells is 7:3, and each node has two input buffers and eight output buffers. The results are only for 0.5, 0.67, 0.8, and 0.9 loading since the results for lower loading values are trivial.

A.1 Size 8x8

A.1.1 Uniform Traffic

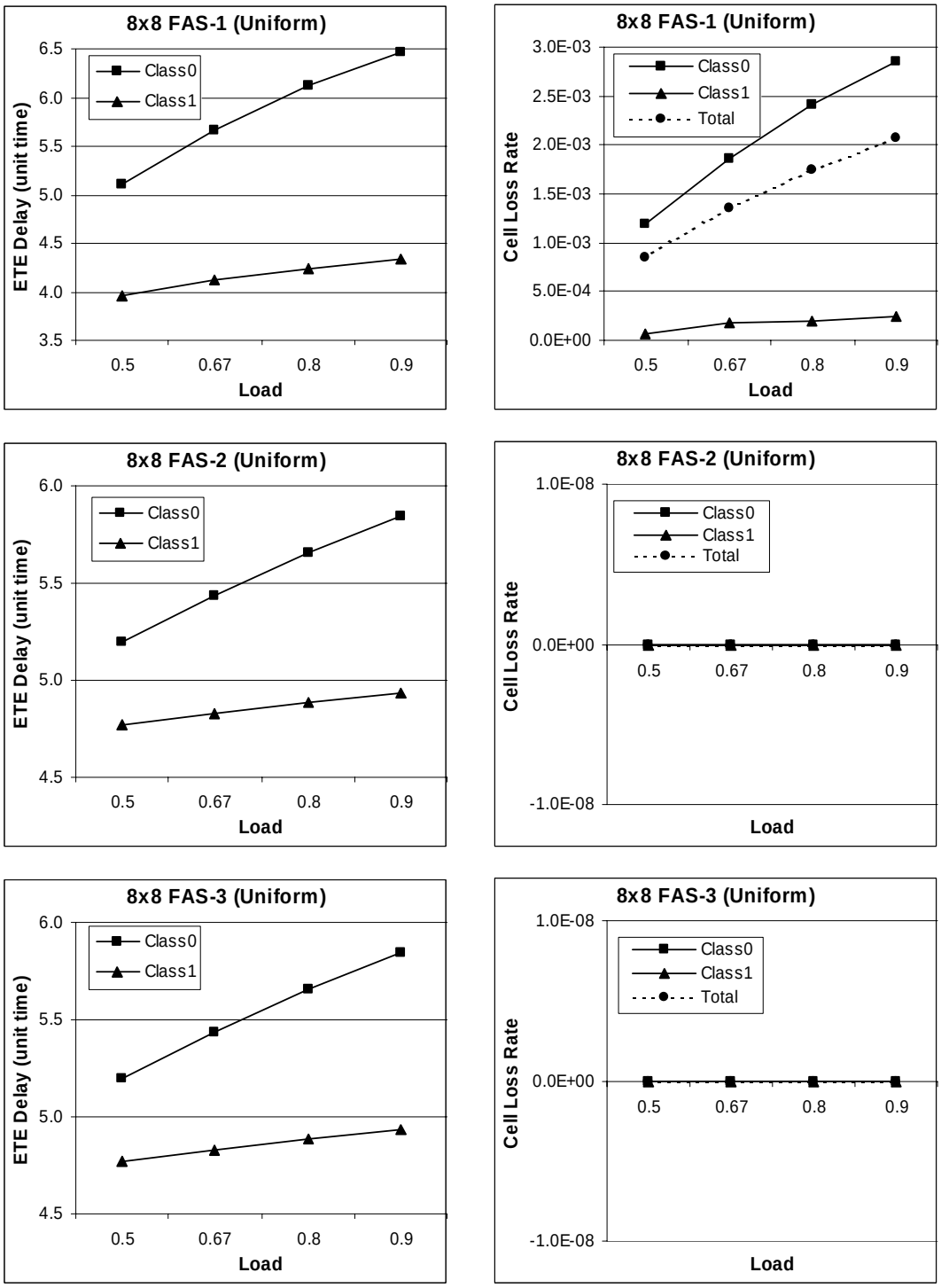


Figure A.1 ETE Delay and cell loss rate of 8x8 FAS-1/2/3 under uniform traffic.

Table A.1. ETE Delay and Cell Loss Rate of 8x8 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.11	5.66	6.12	6.47
	Class-1	3.95	4.12	4.24	4.34
FAS-2	Class-0	5.20	5.43	5.66	5.84
	Class-1	4.77	4.83	4.89	4.93
FAS-3	Class-0	5.20	5.43	5.66	5.84
	Class-1	4.77	4.83	4.89	4.93

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.18E-03	1.86E-03	2.41E-03	2.86E-03
	Class-1	6.67E-05	1.75E-04	1.92E-04	2.42E-04
	Total	8.48E-04	1.36E-03	1.74E-03	2.07E-03
FAS-2	Class-0	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	0.00E+00
FAS-3	Class-0	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	0.00E+00

A.1.2 Normal Traffic

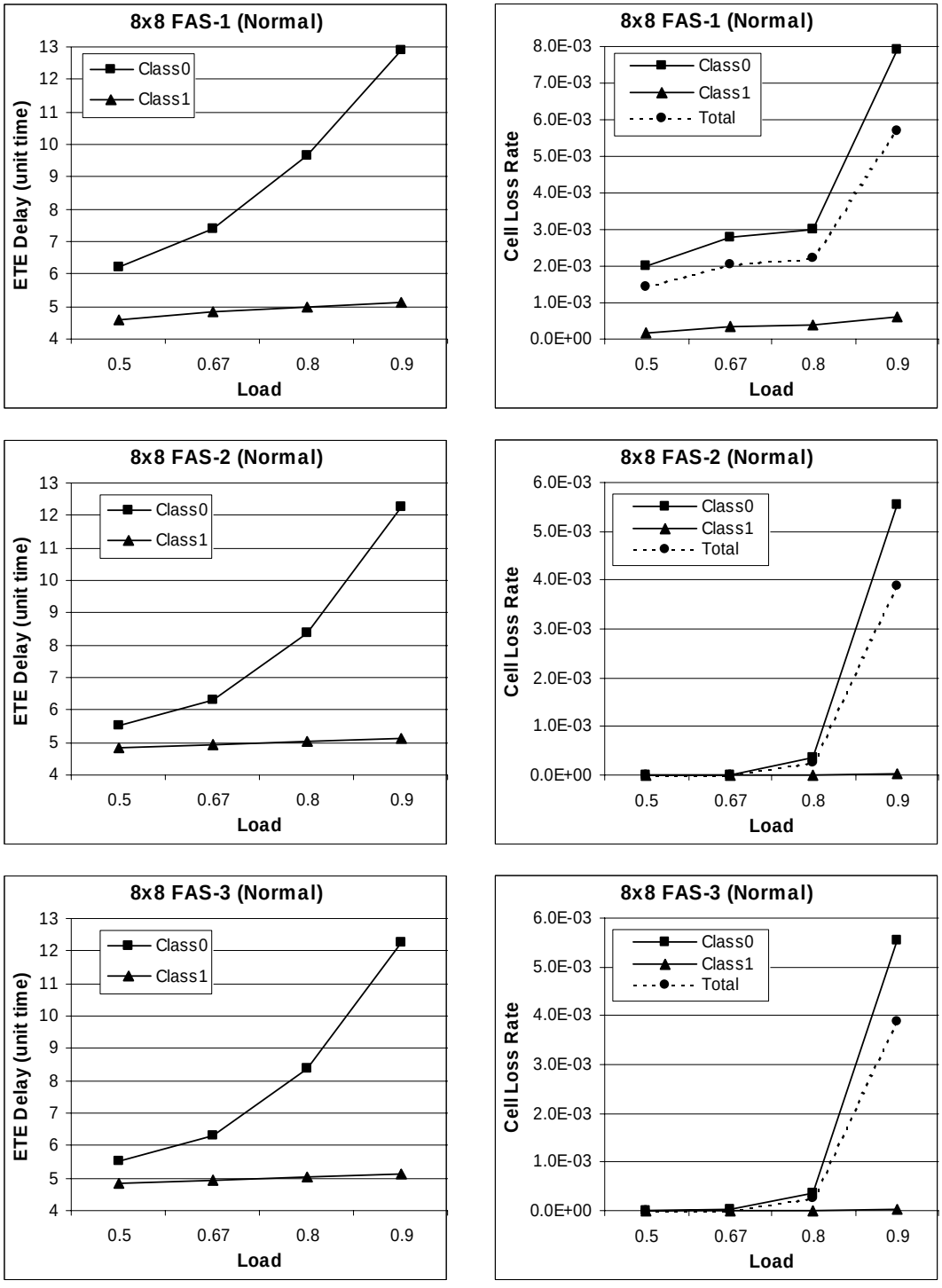


Figure A.2 ETE Delay and cell loss rate of 8x8 FAS-1/2/3 under normal traffic.

Table A.2. ETE Delay and Cell Loss Rate of 8x8 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.71	6.91	9.18	12.42
	Class-1	4.09	4.32	4.48	4.61
FAS-2	Class-0	5.52	6.32	8.39	12.24
	Class-1	4.85	4.95	5.05	5.15
FAS-3	Class-0	5.52	6.32	8.39	12.24
	Class-1	4.85	4.95	5.05	5.15

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	2.00E-03	2.78E-03	3.02E-03	7.90E-03
	Class-1	1.66E-04	3.28E-04	3.95E-04	6.28E-04
	Total	1.45E-03	2.04E-03	2.23E-03	5.71E-03
FAS-2	Class-0	0.00E+00	1.79E-06	3.52E-04	5.54E-03
	Class-1	0.00E+00	4.15E-06	0.00E+00	3.74E-05
	Total	0.00E+00	2.50E-06	2.46E-04	3.88E-03
FAS-3	Class-0	0.00E+00	1.79E-05	3.52E-04	5.54E-03
	Class-1	0.00E+00	4.15E-06	0.00E+00	3.74E-05
	Total	0.00E+00	2.50E-06	2.46E-04	3.88E-03

A.1.3 Bur-5 Traffic

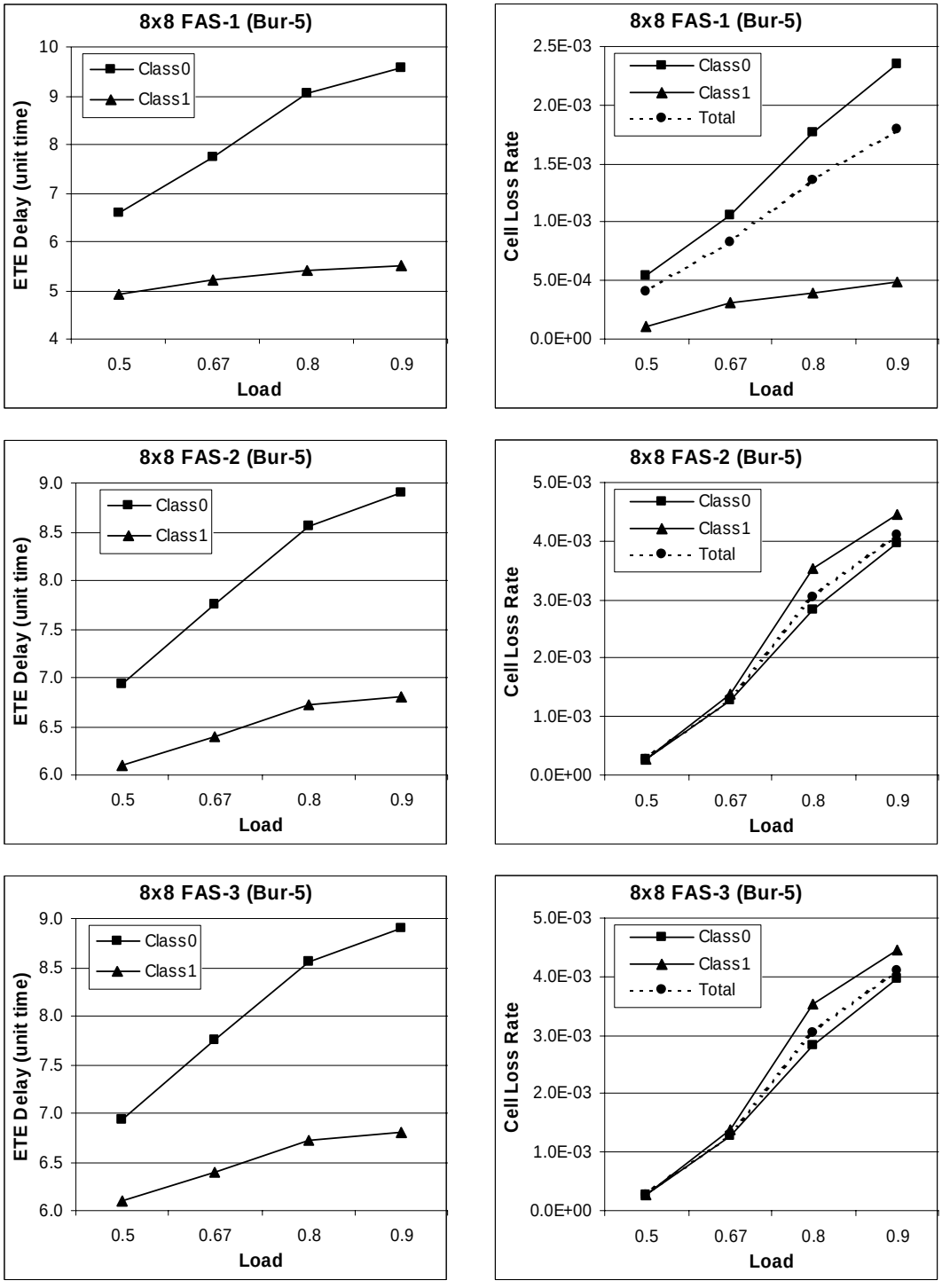


Figure A.3 ETE Delay and cell loss rate of 8x8 FAS-1/2/3 under Bur-5 traffic.

Table A.3. ETE Delay and Cell Loss Rate of 8x8 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	6.61	7.72	9.05	9.59
	Class-1	4.91	5.20	5.41	5.52
FAS-2	Class-0	6.93	7.76	8.55	8.90
	Class-1	6.10	6.40	6.72	6.81
FAS-3	Class-0	6.93	7.76	8.55	8.90
	Class-1	6.10	6.40	6.72	6.81

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.41E-04	1.06E-03	1.77E-03	2.35E-03
	Class-1	1.08E-04	3.08E-04	4.00E-04	4.95E-04
	Total	4.11E-04	8.32E-04	1.36E-03	1.79E-03
FAS-2	Class-0	2.72E-04	1.27E-03	2.84E-03	3.97E-03
	Class-1	2.66E-04	1.39E-03	3.52E-03	4.45E-03
	Total	2.70E-04	1.30E-03	3.05E-03	4.12E-03
FAS-3	Class-0	2.72E-04	1.27E-03	2.84E-03	3.97E-03
	Class-1	2.66E-04	1.39E-03	3.52E-03	4.45E-03
	Total	2.70E-04	1.30E-03	3.05E-03	4.12E-03

A.1.4 Bur-10 Traffic

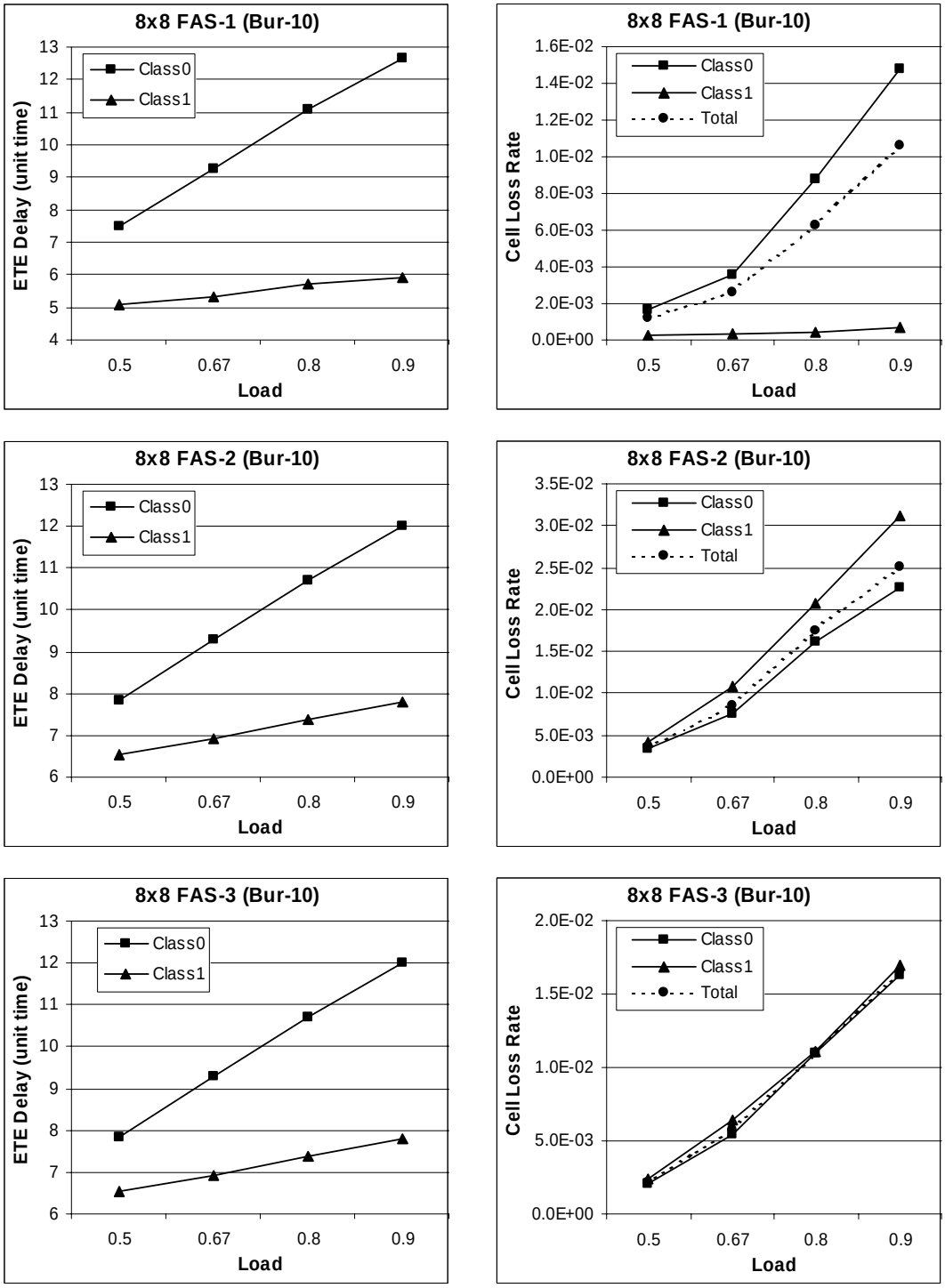


Figure A.4 ETE Delay and cell loss rate of 8x8 FAS-1/2/3 under Bur-10 traffic.

Table A.4. ETE Delay of Cell Loss Rate of 8x8 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.48	9.25	11.06	12.63
	Class-1	5.07	5.32	5.71	5.91
FAS-2	Class-0	7.85	9.29	10.69	12.02
	Class-1	6.52	6.93	7.37	7.78
FAS-3	Class-0	7.85	9.29	10.69	12.02
	Class-1	6.52	6.93	7.37	7.78

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.65E-03	3.53E-03	8.77E-03	1.48E-02
	Class-1	2.96E-04	3.59E-04	4.54E-04	7.11E-04
	Total	1.24E-03	2.58E-03	6.27E-03	1.06E-02
FAS-2	Class-0	3.42E-03	7.64E-03	1.61E-02	2.26E-02
	Class-1	4.12E-03	1.08E-02	2.07E-02	3.13E-02
	Total	3.63E-03	8.60E-03	1.75E-02	2.51E-02
FAS-3	Class-0	2.04E-03	5.49E-03	1.09E-02	1.64E-02
	Class-1	2.36E-03	6.43E-03	1.11E-02	1.70E-02
	Total	2.14E-03	5.77E-03	1.10E-02	1.65E-02

A.2 Size 16x16

A.2.1 Uniform Traffic

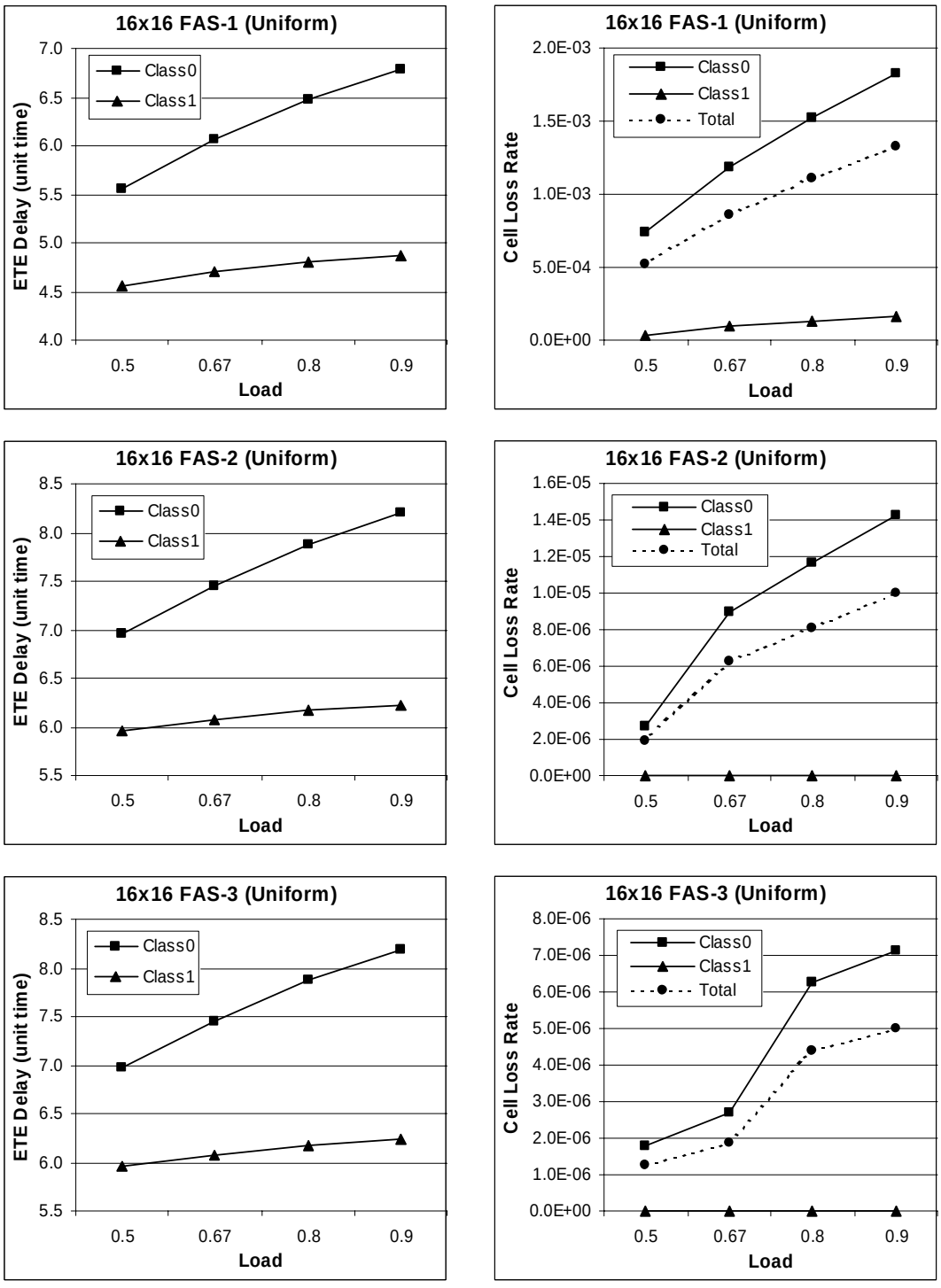


Figure A.5 ETE Delay and cell loss rate of 16x16 FAS-1/2/3 under uniform traffic.

Table A.5. ETE Delay and Cell Loss Rate of 16x16 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.56	6.07	6.48	6.79
	Class-1	4.55	4.71	4.81	4.87
FAS-2	Class-0	6.96	7.45	7.88	8.20
	Class-1	5.97	6.07	6.17	6.22
FAS-3	Class-0	6.97	7.45	7.88	8.19
	Class-1	5.96	6.07	6.17	6.23

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.36E-04	1.18E-03	1.52E-03	1.83E-03
	Class-1	3.75E-05	9.79E-05	1.27E-04	1.67E-04
	Total	5.27E-04	8.55E-04	1.10E-03	1.33E-03
FAS-2	Class-0	2.68E-06	8.93E-06	1.16E-05	1.43E-05
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	1.88E-06	6.25E-06	8.13E-06	1.00E-05
FAS-3	Class-0	1.79E-06	2.68E-06	6.25E-06	7.14E-06
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	1.25E-06	1.88E-06	4.38E-06	5.00E-06

A.2.2 Normal Traffic

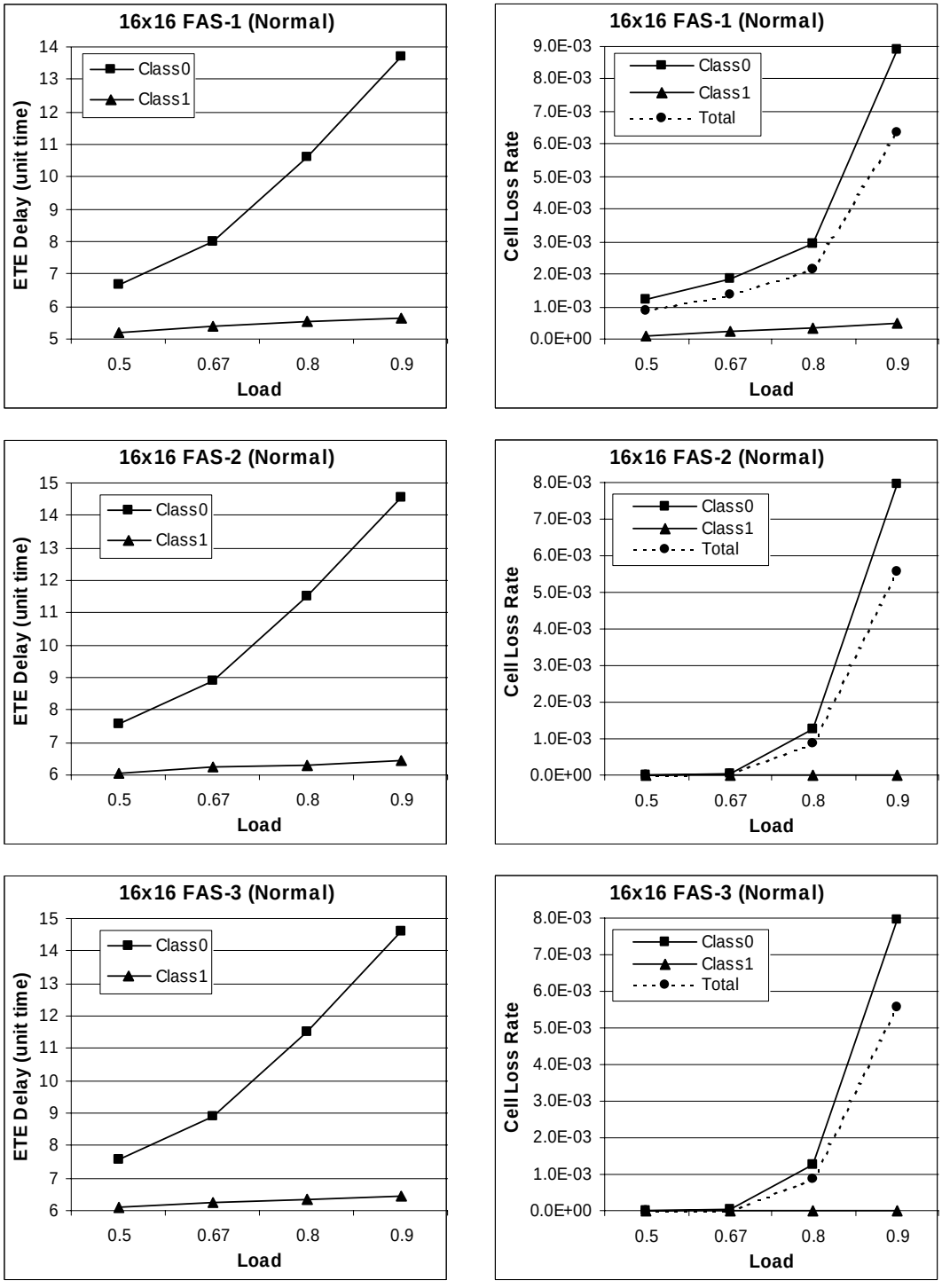


Figure A.6 ETE Delay and cell loss rate of 16x16 FAS-1/2/3 under normal traffic.

Table A.6. ETE Delay and Cell Loss Rate of 16x16 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	6.16	7.51	10.08	13.21
	Class-1	4.68	4.88	5.04	5.16
FAS-2	Class-0	7.59	8.91	11.51	14.57
	Class-1	6.06	6.23	6.31	6.42
FAS-3	Class-0	7.59	8.92	11.51	14.63
	Class-1	6.08	6.22	6.33	6.44

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.24E-03	1.86E-03	2.93E-03	8.89E-03
	Class-1	9.37E-05	2.21E-04	3.37E-04	4.83E-04
	Total	8.94E-04	1.37E-03	2.15E-03	6.36E-03
FAS-2	Class-0	4.47E-06	3.13E-05	1.27E-03	7.94E-03
	Class-1	0.00E+00	0.00E+00	0.00E+00	2.08E-06
	Total	3.13E-06	2.19E-05	8.90E-04	5.56E-03
FAS-3	Class-0	3.57E-06	2.95E-05	1.26E-03	7.95E-03
	Class-1	0.00E+00	0.00E+00	0.00E+00	2.08E-06
	Total	2.50E-06	2.06E-05	8.83E-04	5.56E-03

A.2.3 Bur-5 Traffic

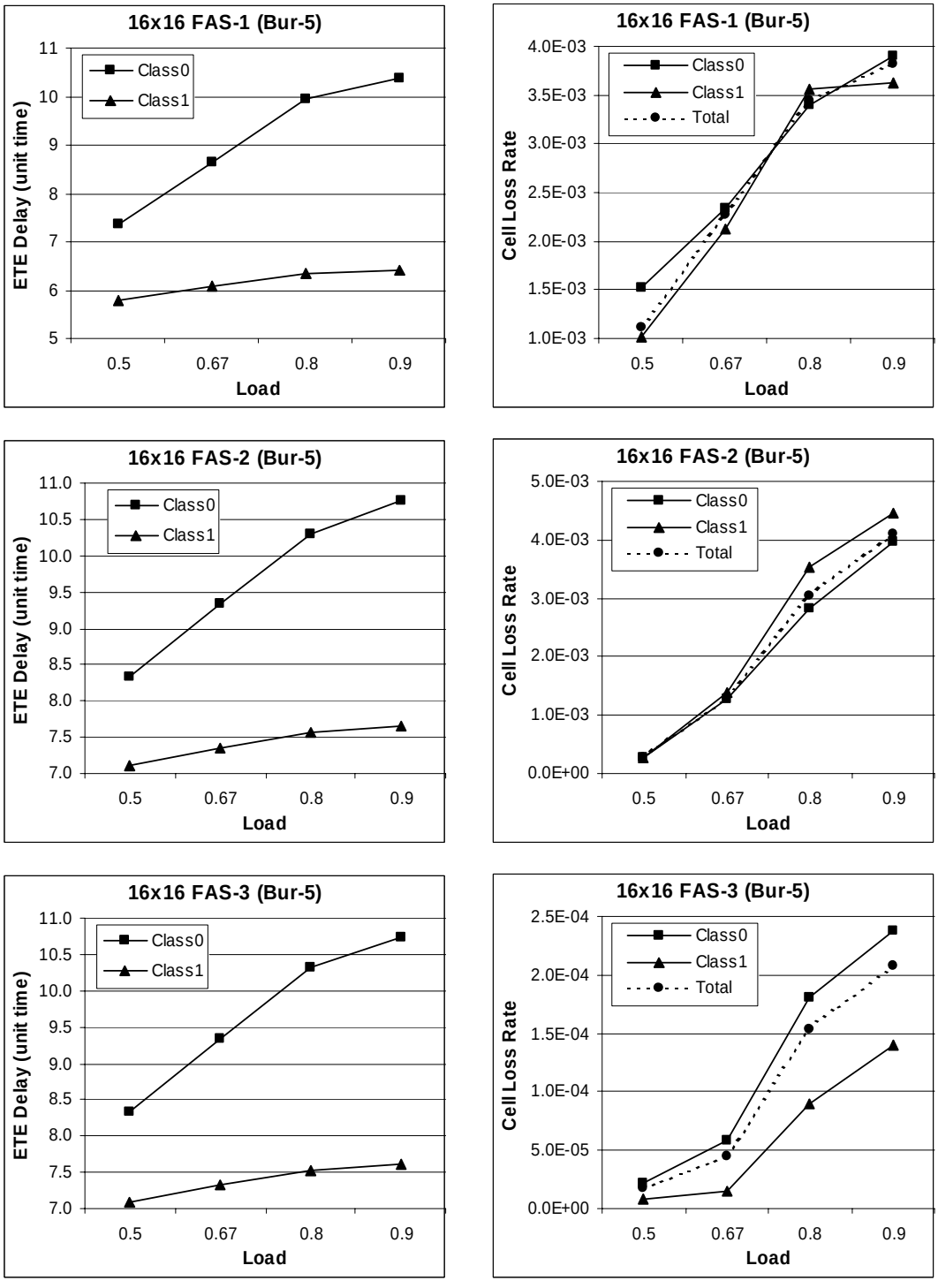


Figure A.7 ETE Delay and cell loss rate of 16x16 FAS-1/2/3 under Bur-5 traffic.

Table A.7. ETE Delay and Cell Loss Rate of 16x16 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.37	8.64	9.96	10.37
	Class-1	5.78	6.10	6.36	6.42
FAS-2	Class-0	8.33	9.35	10.31	10.75
	Class-1	7.10	7.36	7.56	7.65
FAS-3	Class-0	8.33	9.34	10.31	10.74
	Class-1	7.10	7.33	7.52	7.61

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.52E-03	2.34E-03	3.39E-03	3.91E-03
	Class-1	1.02E-03	2.12E-03	3.57E-03	3.62E-03
	Total	1.11E-03	2.27E-03	3.44E-03	3.82E-03
FAS-2	Class-0	2.72E-04	1.27E-03	2.84E-03	3.97E-03
	Class-1	2.66E-04	1.39E-03	3.52E-03	4.45E-03
	Total	2.70E-04	1.30E-03	3.05E-03	4.12E-03
FAS-3	Class-0	2.14E-05	5.81E-05	1.81E-04	2.38E-04
	Class-1	8.34E-06	1.46E-05	8.95E-05	1.39E-04
	Total	1.75E-05	4.50E-05	1.54E-04	2.08E-04

A.2.4 Bur-10 Traffic

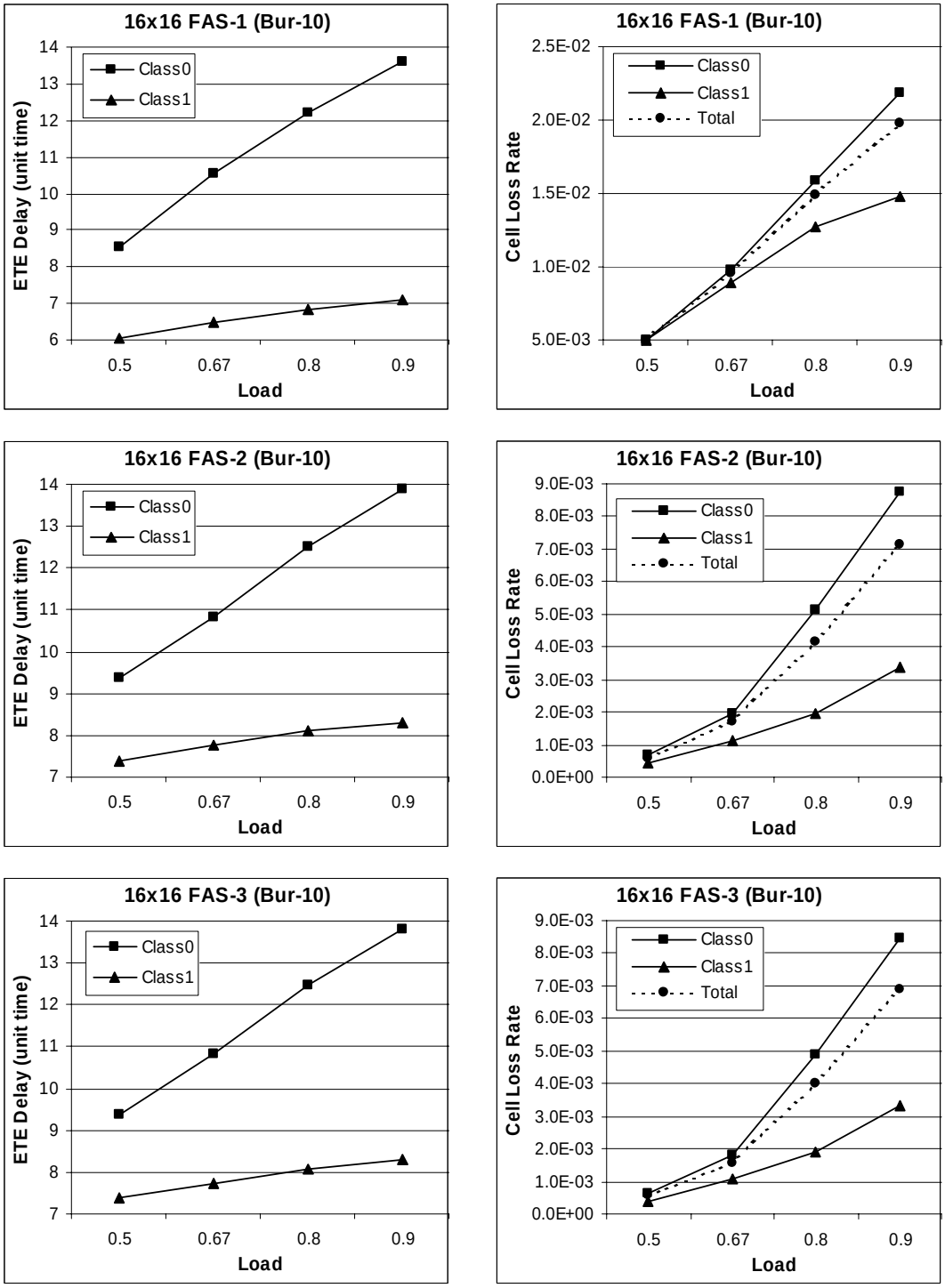


Figure A.8 ETE Delay and cell loss rate of 16x16 FAS-1/2/3 under Bur-10 traffic.

Table A.8. ETE Delay and Cell Loss Rate of 16x16 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.55	10.54	12.21	13.59
	Class-1	6.05	6.49	6.81	7.11
FAS-2	Class-0	9.38	10.82	12.49	13.89
	Class-1	7.40	7.78	8.09	8.31
FAS-3	Class-0	9.37	10.83	12.46	13.80
	Class-1	7.39	7.74	8.07	8.30

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.05E-03	9.77E-03	1.58E-02	2.18E-02
	Class-1	5.02E-03	8.95E-03	1.27E-02	1.48E-02
	Total	5.04E-03	9.52E-03	1.49E-02	1.97E-02
FAS-2	Class-0	6.63E-04	1.93E-03	5.12E-03	8.78E-03
	Class-1	4.22E-04	1.14E-03	1.95E-03	3.39E-03
	Total	5.91E-04	1.69E-03	4.17E-03	7.16E-03
FAS-3	Class-0	6.36E-04	1.79E-03	4.90E-03	8.44E-03
	Class-1	4.09E-04	1.10E-03	1.89E-03	3.35E-03
	Total	5.68E-04	1.58E-03	3.99E-03	6.91E-03

A.3 Size 64x64

A.3.1 Uniform Traffic

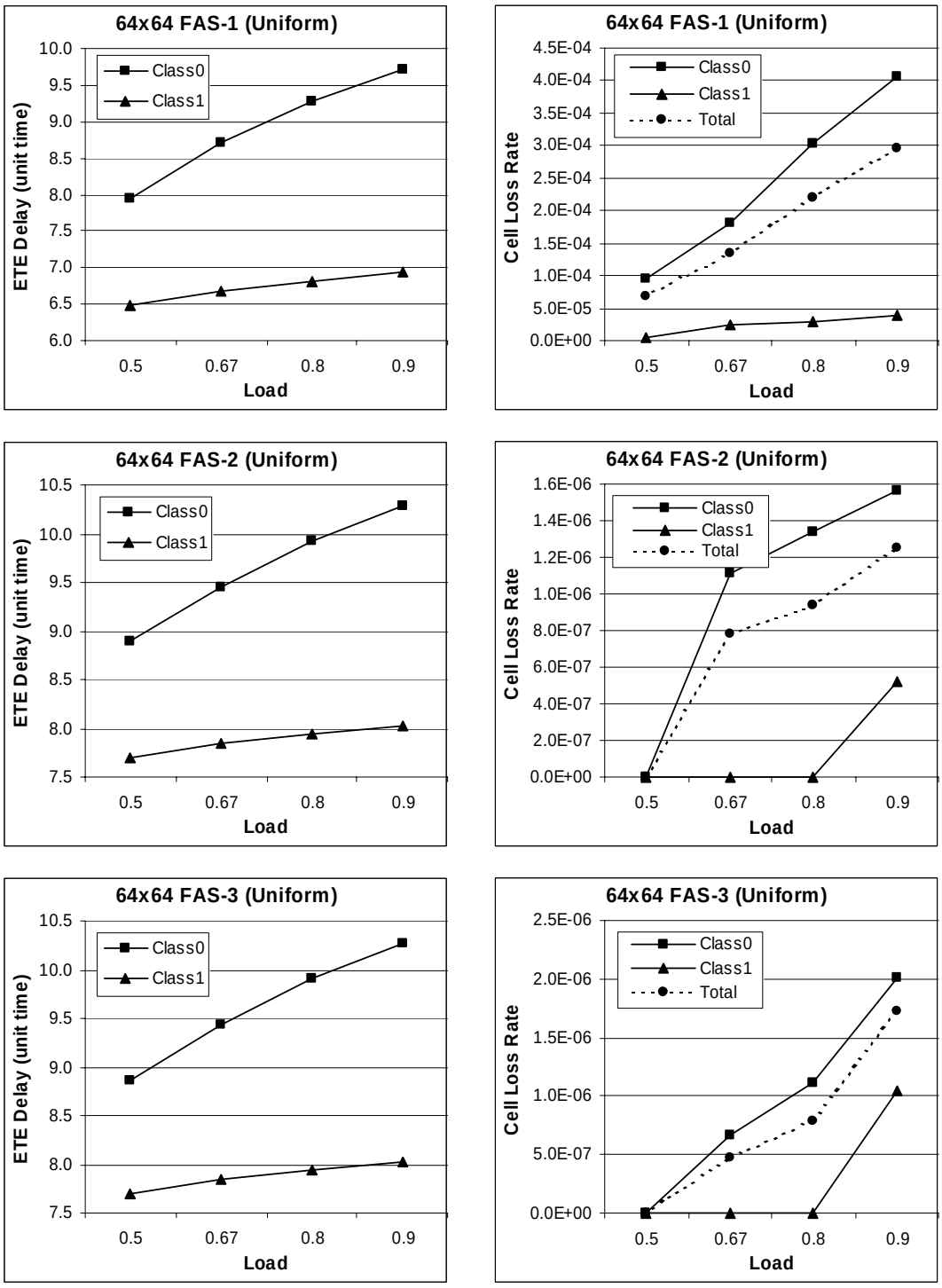


Figure A.9 ETE Delay and cell loss rate of 64x64 FAS-1/2/3 under uniform traffic.

Table A.9. ETE Delay and Cell Loss Rate of 64x64 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.94	8.70	9.27	9.71
	Class-1	6.47	6.69	6.81	6.94
FAS-2	Class-0	8.89	9.45	9.93	10.29
	Class-1	7.70	7.85	7.94	8.02
FAS-3	Class-0	8.87	9.44	9.91	10.26
	Class-1	7.69	7.84	7.94	8.02

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.53E-05	1.81E-04	3.03E-04	4.05E-04
	Class-1	5.21E-06	2.35E-05	2.81E-05	4.01E-05
	Total	6.83E-05	1.34E-04	2.20E-04	2.96E-04
FAS-2	Class-0	0.00E+00	1.12E-06	1.34E-06	1.56E-06
	Class-1	0.00E+00	0.00E+00	0.00E+00	5.21E-07
	Total	0.00E+00	7.81E-07	9.38E-07	1.25E-06
FAS-3	Class-0	0.00E+00	6.70E-07	1.12E-06	2.01E-06
	Class-1	0.00E+00	0.00E+00	0.00E+00	1.04E-06
	Total	0.00E+00	4.69E-07	7.81E-07	1.72E-06

A.3.2 Normal Traffic

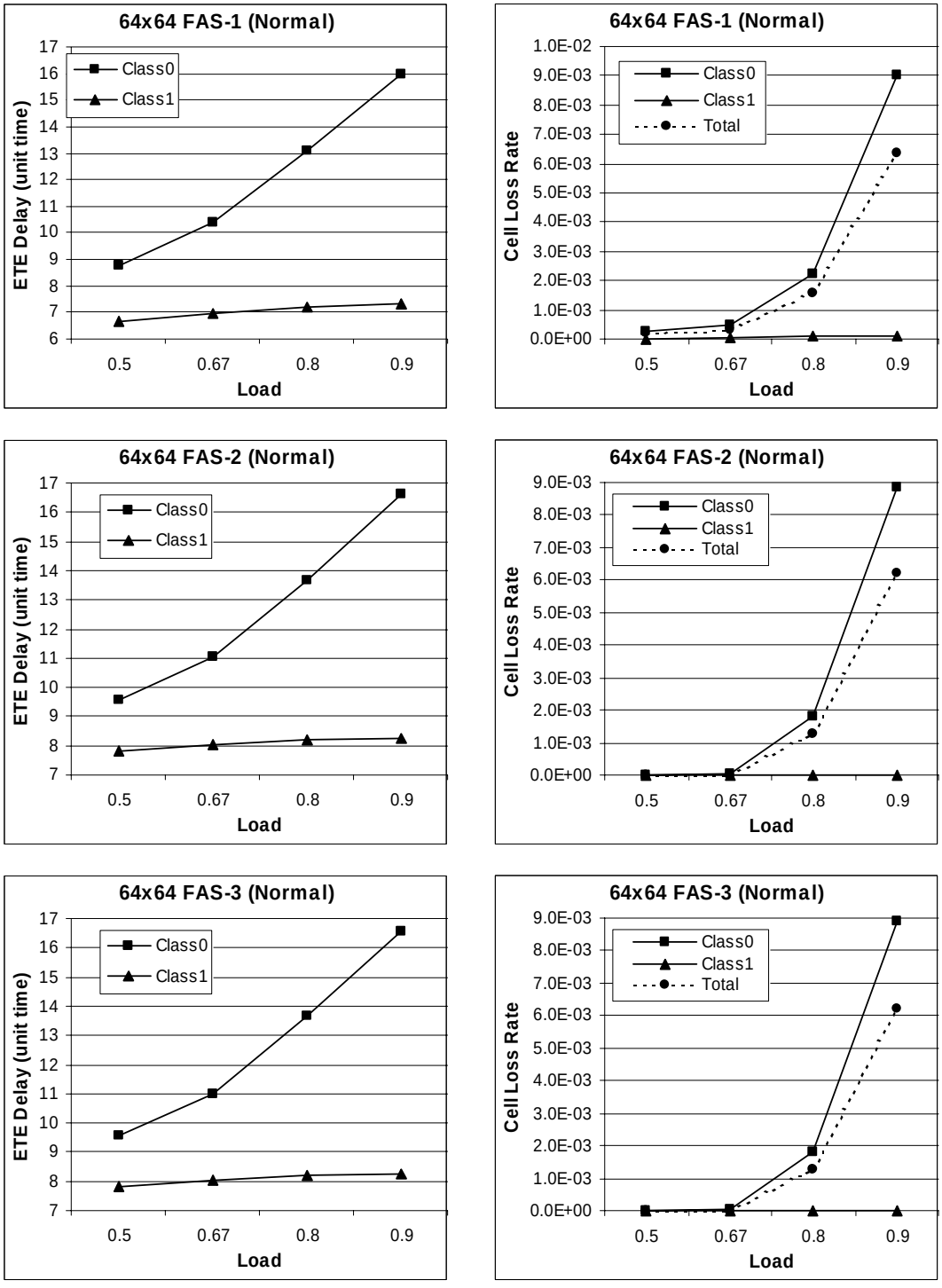


Figure A.10 ETE Delay and cell loss rate of 64x64 FAS-1/2/3 under normal traffic.

Table A.10. ETE Delay and Cell Loss Rate of 64x64 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.79	10.39	13.09	15.96
	Class-1	6.66	6.96	7.21	7.35
FAS-2	Class-0	9.59	11.02	13.67	16.61
	Class-1	7.84	8.04	8.20	8.25
FAS-3	Class-0	9.56	10.99	13.68	16.55
	Class-1	7.83	8.04	8.21	8.28

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	2.82E-04	4.66E-04	2.25E-03	9.03E-03
	Class-1	2.45E-05	6.15E-05	8.96E-05	1.20E-04
	Total	2.05E-04	3.45E-04	1.60E-03	6.36E-03
FAS-2	Class-0	1.56E-06	2.55E-05	1.80E-03	8.85E-03
	Class-1	1.04E-06	0.00E+00	3.65E-06	0.00E+00
	Total	1.41E-06	1.78E-05	1.26E-03	6.19E-03
FAS-3	Class-0	1.79E-06	2.59E-05	1.81E-03	8.90E-03
	Class-1	5.21E-07	0.00E+00	2.08E-06	0.00E+00
	Total	1.41E-06	1.81E-05	1.27E-03	6.23E-03

A.3.3 Bur-5 Traffic

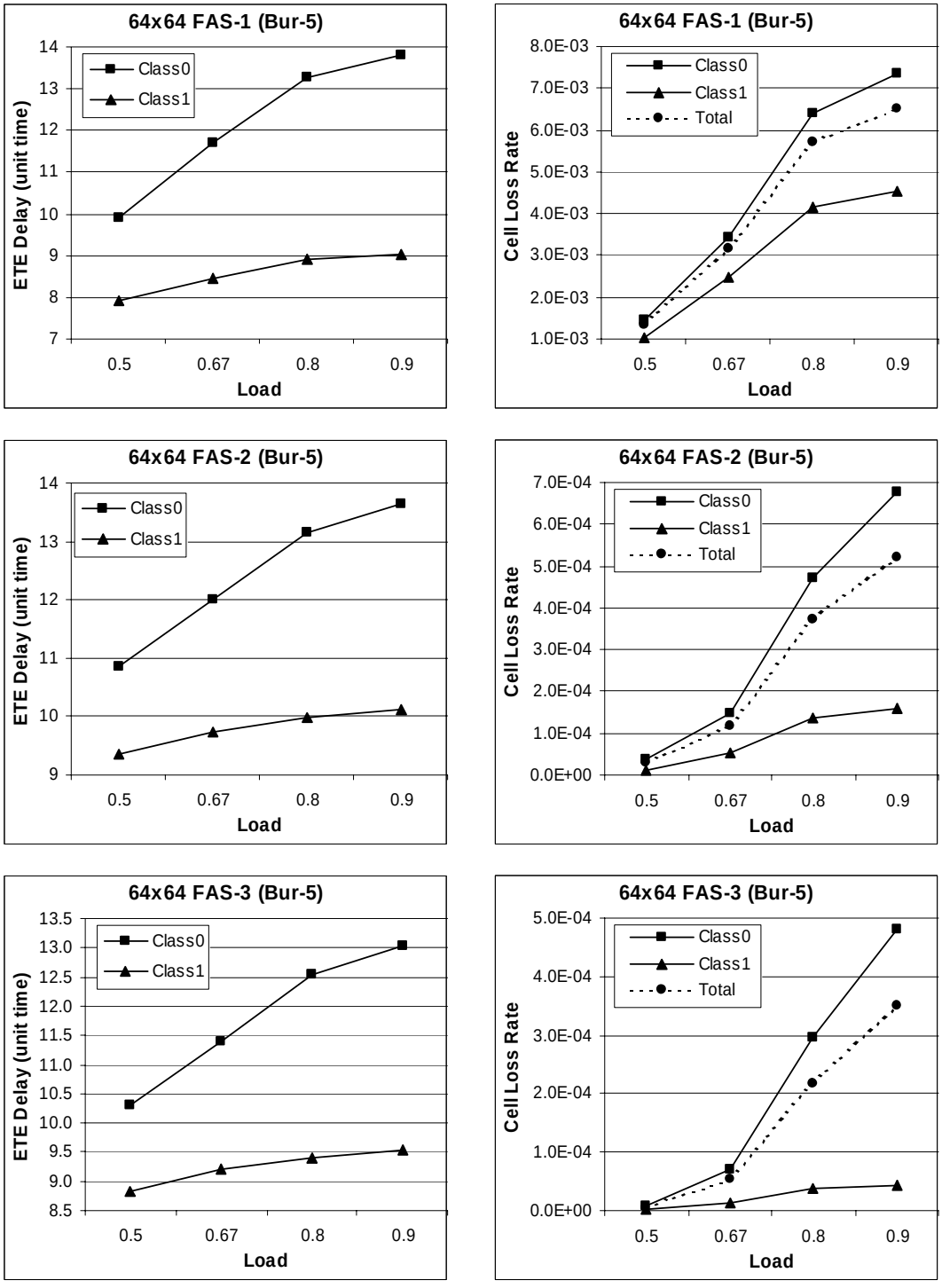


Figure A.11 ETE Delay and cell loss rate of 64x64 FAS-1/2/3 under Bur-5 traffic.

Table A.11. ETE Delay and Cell Loss Rate of 64x64 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.91	11.69	13.28	13.81
	Class-1	7.90	8.45	8.93	9.04
FAS-2	Class-0	10.35	11.52	12.64	13.14
	Class-1	8.87	9.25	9.49	9.63
FAS-3	Class-0	10.30	11.40	12.54	13.03
	Class-1	8.84	9.20	9.39	9.54

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.47E-03	3.44E-03	6.39E-03	7.35E-03
	Class-1	1.03E-03	2.48E-03	4.18E-03	4.54E-03
	Total	1.34E-03	3.15E-03	5.72E-03	6.50E-03
FAS-2	Class-0	3.84E-05	1.47E-04	4.72E-04	6.79E-04
	Class-1	1.15E-05	5.47E-05	1.37E-04	1.59E-04
	Total	3.03E-05	1.19E-04	3.72E-04	5.23E-04
FAS-3	Class-0	7.59E-06	7.05E-05	2.96E-04	4.80E-04
	Class-1	2.09E-06	1.36E-05	3.70E-05	4.43E-05
	Total	5.94E-06	5.34E-05	2.19E-04	3.49E-04

A.3.4 Bur-10 Traffic

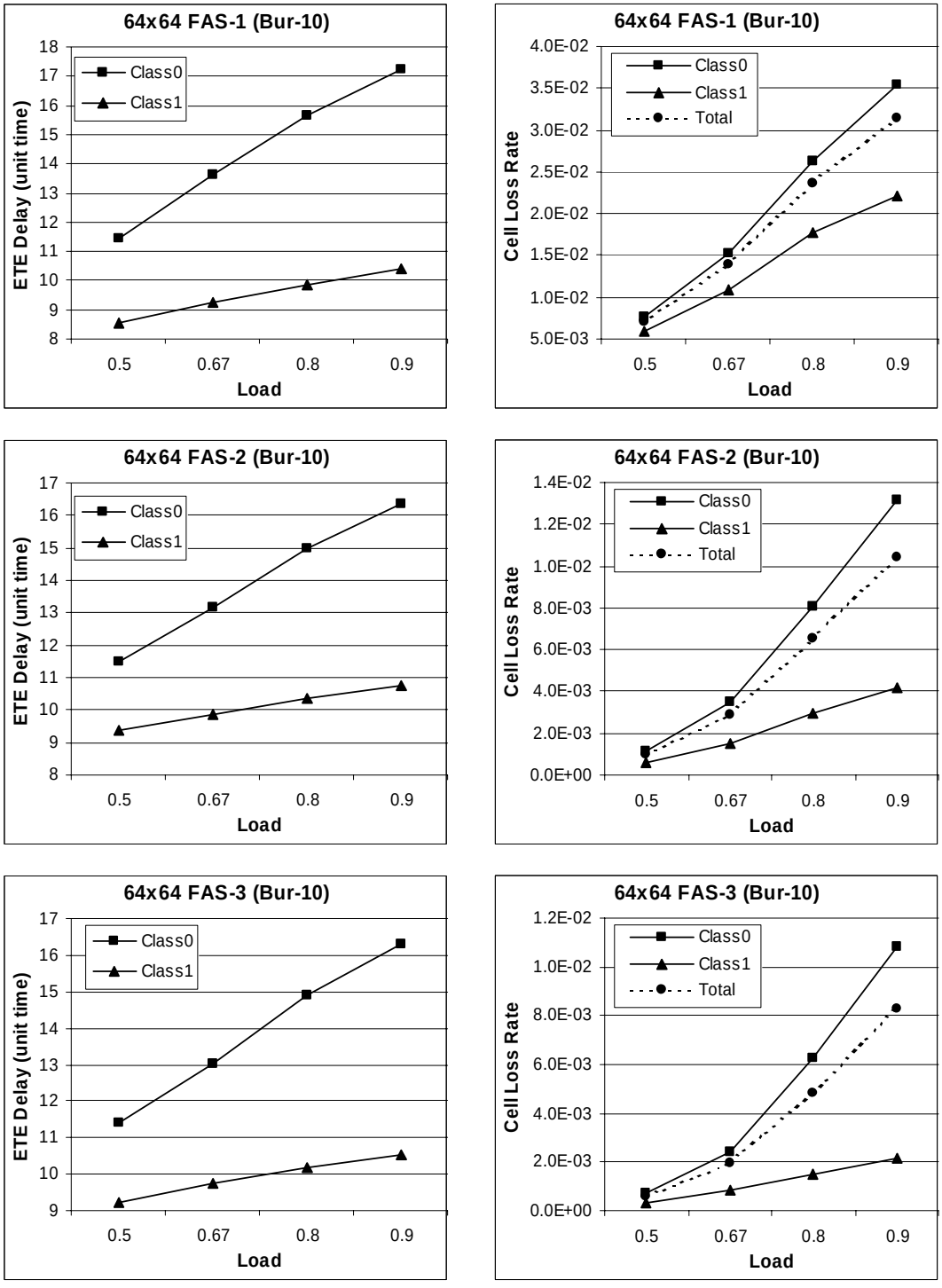


Figure A.12 ETE Delay and cell loss rate of 64x64 FAS-1/2/3 under Bur-10 traffic.

Table A.12. ETE Delay and Cell Loss Rate of 64x64 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	11.43	13.65	15.66	17.24
	Class-1	8.55	9.24	9.86	10.40
FAS-2	Class-0	11.48	13.18	14.99	16.37
	Class-1	9.36	9.87	10.38	10.75
FAS-3	Class-0	11.43	13.04	14.90	16.29
	Class-1	9.24	9.74	10.18	10.55

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.61E-03	1.53E-02	2.62E-02	3.54E-02
	Class-1	6.03E-03	1.09E-02	1.77E-02	2.22E-02
	Total	7.14E-03	1.40E-02	2.37E-02	3.15E-02
FAS-2	Class-0	1.159E-03	3.513E-03	8.058E-03	1.314E-02
	Class-1	6.465E-04	1.529E-03	2.950E-03	4.161E-03
	Total	1.005E-03	2.918E-03	6.526E-03	1.045E-02
FAS-3	Class-0	6.93E-04	2.44E-03	6.29E-03	1.09E-02
	Class-1	3.09E-04	8.19E-04	1.51E-03	2.17E-03
	Total	5.78E-04	1.95E-03	4.86E-03	8.25E-03

A.4 Size 128x128

A.4.1 Uniform Traffic

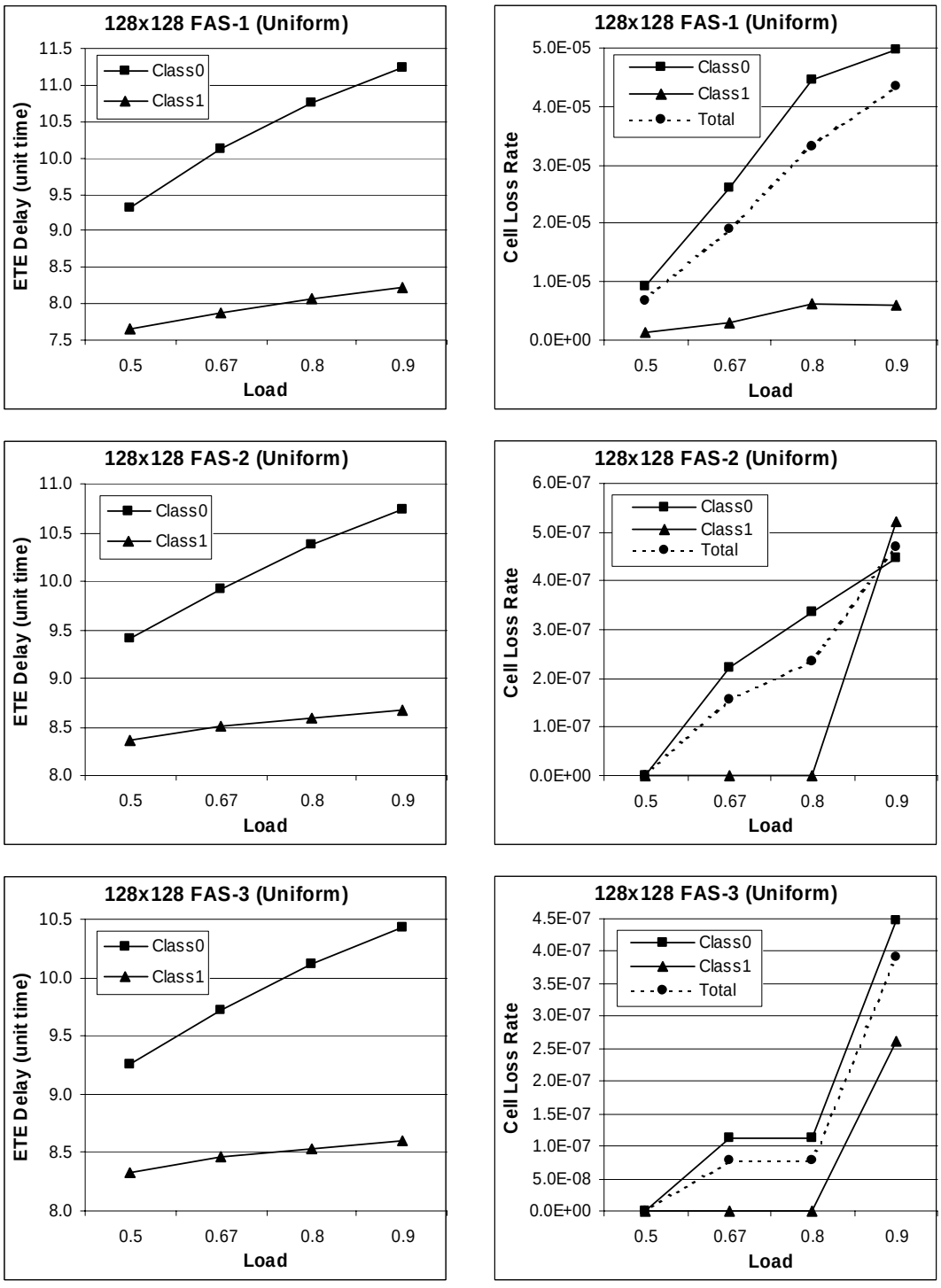


Figure A.13 ETE Delay and cell loss rate of 128x128 FAS-1/2/3 under uniform traffic.

Table A.13. ETE Delay and Cell Loss Rate of 128x128 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.32	10.13	10.77	11.24
	Class-1	7.66	7.87	8.08	8.22
FAS-2	Class-0	9.40	9.93	10.37	10.73
	Class-1	8.36	8.50	8.59	8.66
FAS-3	Class-0	9.26	9.72	10.12	10.43
	Class-1	8.33	8.46	8.53	8.60

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.26E-06	2.60E-05	4.45E-05	4.96E-05
	Class-1	1.30E-06	2.87E-06	6.25E-06	5.99E-06
	Total	6.88E-06	1.91E-05	3.31E-05	4.35E-05
FAS-2	Class-0	0.00E+00	2.23E-07	3.35E-07	4.46E-07
	Class-1	0.00E+00	0.00E+00	0.00E+00	5.21E-07
	Total	0.00E+00	1.56E-07	2.34E-07	4.69E-07
FAS-3	Class-0	0.00E+00	1.12E-07	1.12E-07	4.46E-07
	Class-1	0.00E+00	0.00E+00	0.00E+00	2.61E-07
	Total	0.00E+00	7.81E-08	7.81E-08	3.91E-07

A.4.2 Normal Traffic

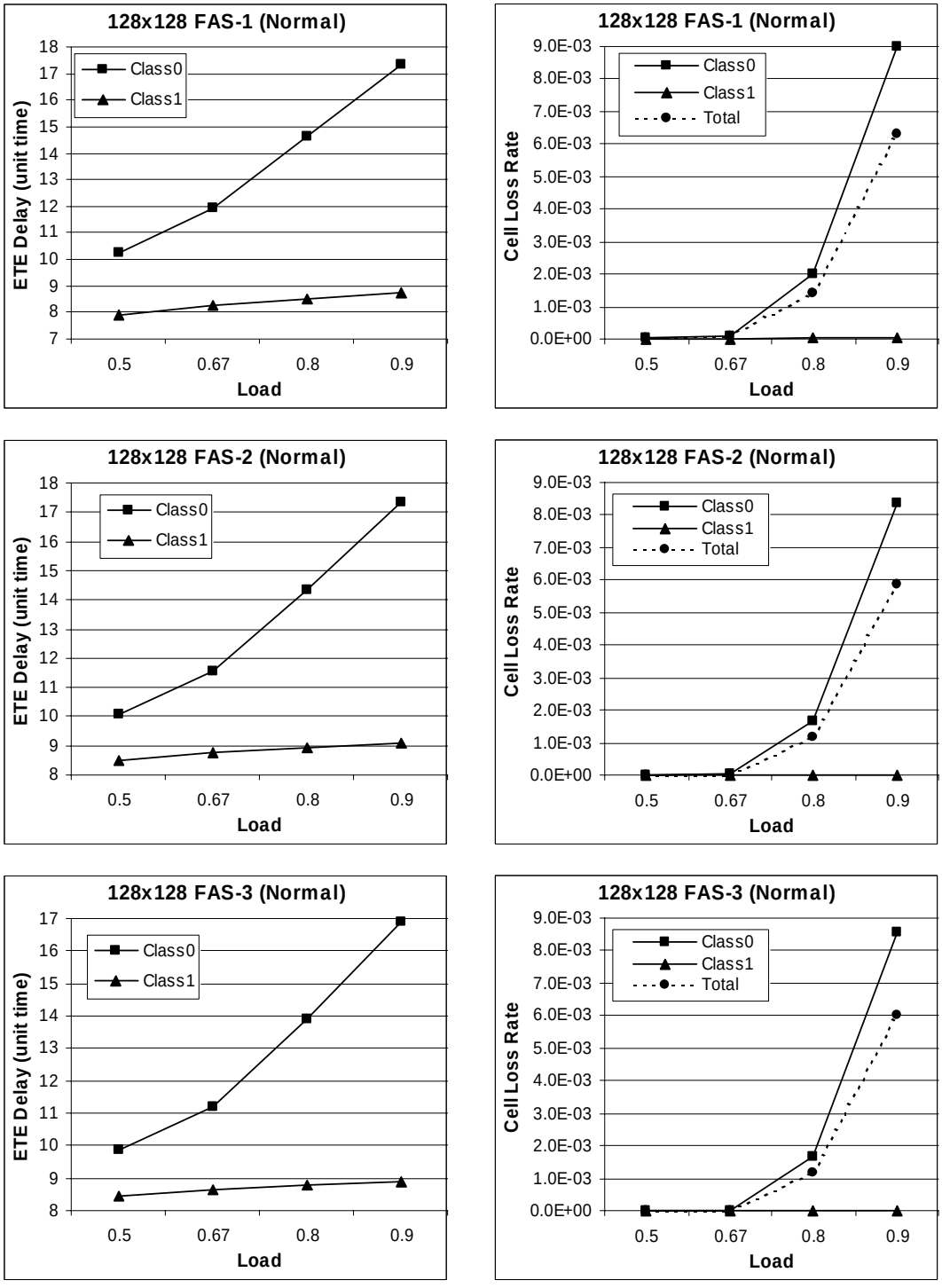


Figure A.14 ETE Delay and cell loss rate of 128x128 FAS-1/2/3 under normal traffic.

Table A.14. ETE Delay and Cell Loss Rate of 128x128 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	10.22	11.94	14.63	17.34
	Class-1	7.91	8.24	8.52	8.73
FAS-2	Class-0	10.09	11.56	14.35	17.34
	Class-1	8.49	8.74	8.93	9.10
FAS-3	Class-0	9.86	11.18	13.89	16.88
	Class-1	8.46	8.63	8.79	8.91

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	4.09E-05	1.19E-04	2.03E-03	9.00E-03
	Class-1	4.95E-06	1.30E-05	2.50E-05	2.84E-05
	Total	3.01E-05	8.69E-05	1.43E-03	6.31E-03
FAS-2	Class-0	1.23E-06	2.59E-05	1.65E-03	8.38E-03
	Class-1	0.00E+00	7.81E-07	4.69E-06	1.56E-05
	Total	8.59E-07	1.84E-05	1.16E-03	5.87E-03
FAS-3	Class-0	7.81E-07	1.74E-05	1.64E-03	8.56E-03
	Class-1	0.00E+00	0.00E+00	0.00E+00	1.04E-06
	Total	5.47E-07	1.22E-05	1.15E-03	5.99E-03

A.4.3 Bur-5 Traffic

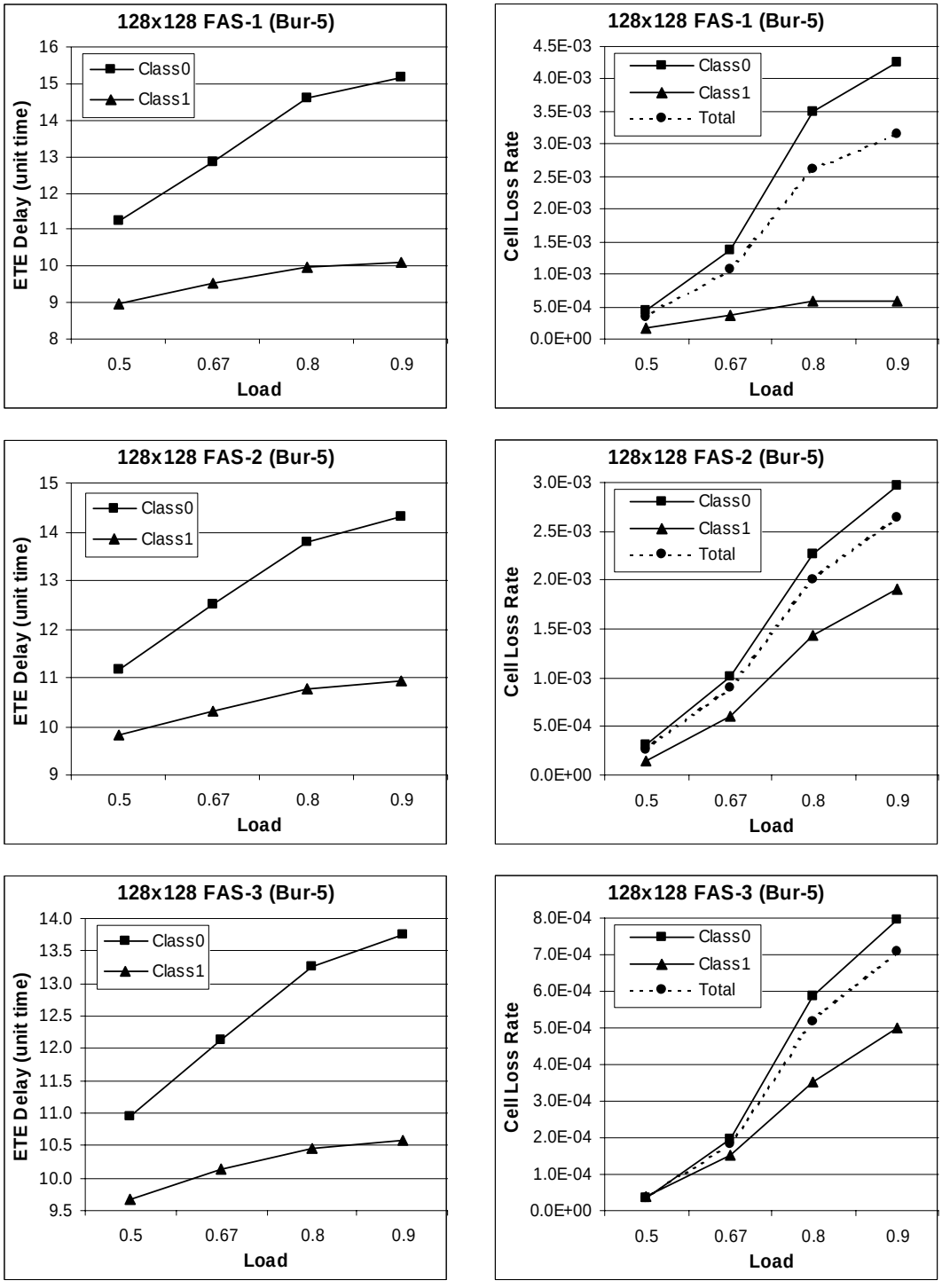


Figure A.15 ETE Delay and cell loss rate of 128x128 FAS-1/2/3 under Bur-5 traffic.

Table A.15. ETE Delay and Cell Loss Rate of 128x128 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	11.23	12.87	14.61	15.17
	Class-1	8.96	9.55	9.98	10.11
FAS-2	Class-0	11.17	12.52	13.79	14.31
	Class-1	9.81	10.32	10.76	10.92
FAS-3	Class-0	10.95	12.13	13.27	13.76
	Class-1	9.67	10.13	10.47	10.58

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	4.30E-04	1.37E-03	3.50E-03	4.26E-03
	Class-1	1.62E-04	3.75E-04	5.79E-04	5.80E-04
	Total	3.49E-04	1.07E-03	2.62E-03	3.15E-03
FAS-2	Class-0	3.06E-04	1.02E-03	2.26E-03	2.96E-03
	Class-1	1.48E-04	5.96E-04	1.44E-03	1.91E-03
	Total	2.59E-04	8.92E-04	2.01E-03	2.65E-03
FAS-3	Class-0	3.57E-05	1.94E-04	5.86E-04	7.96E-04
	Class-1	3.75E-05	1.52E-04	3.54E-04	5.00E-04
	Total	3.63E-05	1.81E-04	5.16E-04	7.07E-04

A.4.4 Bur-10 Traffic

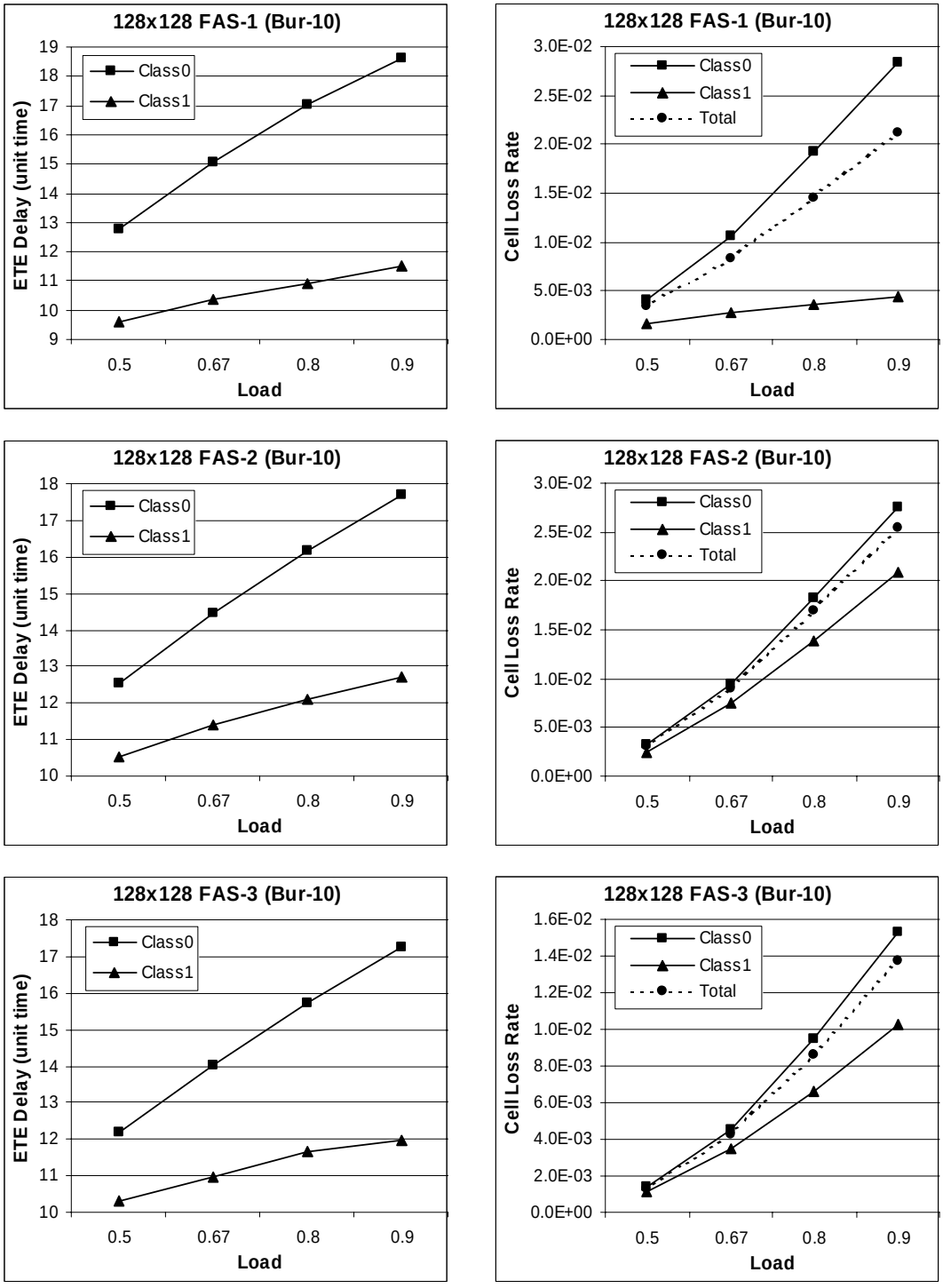


Figure A.16 ETE Delay and cell loss rate of 128x128 FAS-1/2/3 under Bur-10 traffic.

Table A.16. ETE Delay and Cell Loss Rate of 128x128 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	12.75	15.06	17.04	18.61
	Class-1	9.62	10.35	10.93	11.49
FAS-2	Class-0	12.52	14.45	16.17	17.70
	Class-1	10.52	11.39	12.08	12.69
FAS-3	Class-0	12.20	14.02	15.74	17.24
	Class-1	10.28	10.98	11.68	11.98

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	4.11E-03	1.06E-02	1.92E-02	2.84E-02
	Class-1	1.58E-03	2.74E-03	3.59E-03	4.39E-03
	Total	3.35E-03	8.26E-03	1.45E-02	2.12E-02
FAS-2	Class-0	3.29E-03	9.51E-03	1.82E-02	2.75E-02
	Class-1	2.52E-03	7.43E-03	1.38E-02	2.09E-02
	Total	3.06E-03	8.89E-03	1.69E-02	2.55E-02
FAS-3	Class-0	1.35E-03	4.52E-03	9.45E-03	1.53E-02
	Class-1	1.12E-03	3.52E-03	6.58E-03	1.03E-02
	Total	1.28E-03	4.22E-03	8.59E-03	1.38E-02

A.5 Size 256x256

A.5.1 Uniform Traffic

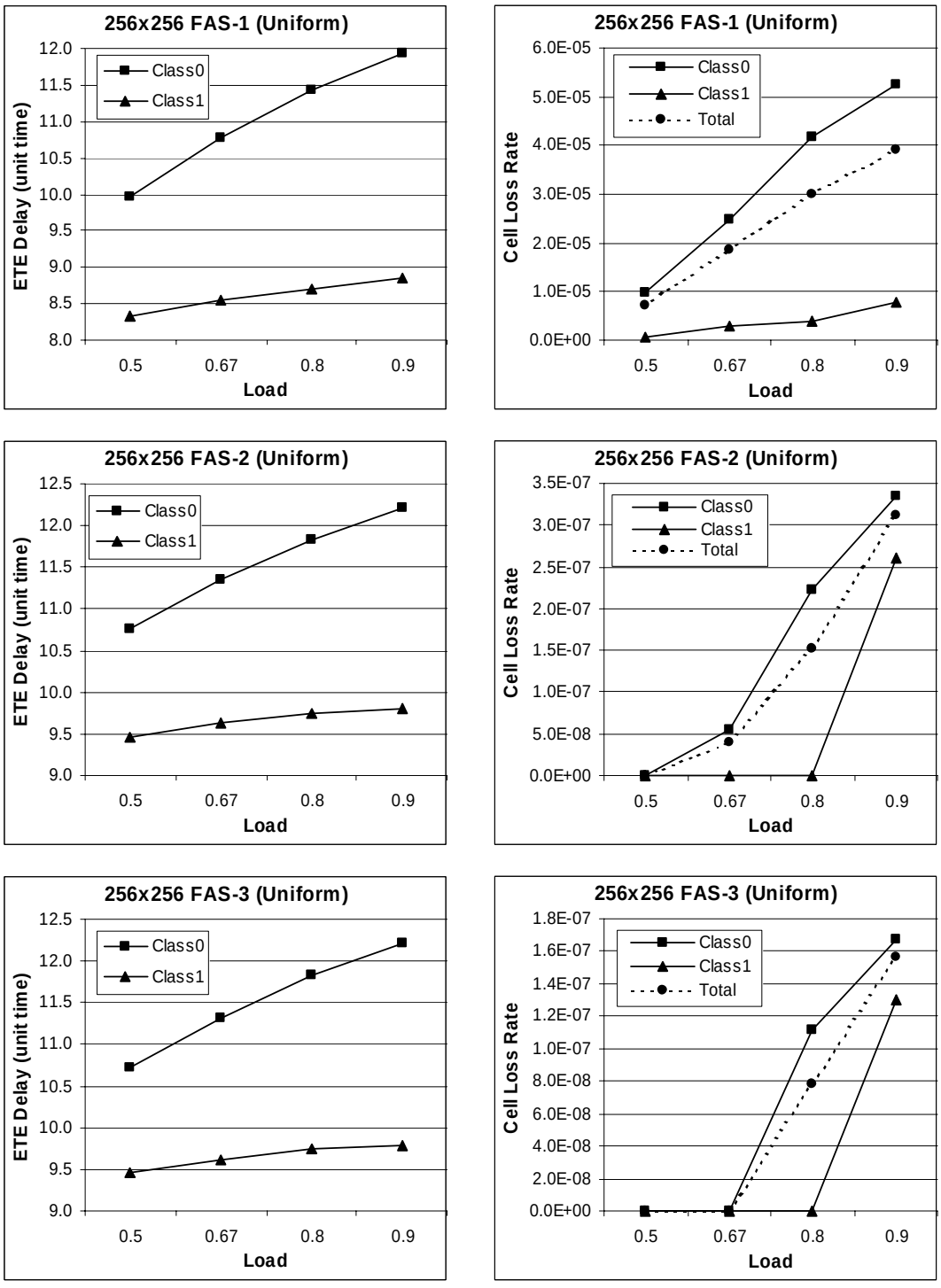


Figure A.17 ETE Delay and cell loss rate of 256x256 FAS-1/2/3 under uniform traffic.

Table A.17. ETE Delay and Cell Loss Rate of 256x256 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.97	10.78	11.43	11.93
	Class-1	8.33	8.54	8.70	8.85
FAS-2	Class-0	10.76	11.35	11.83	12.22
	Class-1	9.46	9.63	9.74	9.80
FAS-3	Class-0	10.73	11.32	11.83	12.21
	Class-1	9.46	9.62	9.74	9.79

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	9.88E-06	2.48E-05	4.17E-05	5.25E-05
	Class-1	6.51E-07	3.00E-06	3.91E-06	7.94E-06
	Total	7.11E-06	1.85E-05	3.01E-05	3.92E-05
FAS-2	Class-0	0.00E+00	5.58E-08	2.23E-07	3.35E-07
	Class-1	0.00E+00	0.00E+00	0.00E+00	2.60E-07
	Total	0.00E+00	3.91E-08	1.53E-07	3.13E-07
FAS-3	Class-0	0.00E+00	0.00E+00	1.12E-07	1.67E-07
	Class-1	0.00E+00	0.00E+00	0.00E+00	1.30E-07
	Total	0.00E+00	0.00E+00	7.81E-08	1.56E-07

A.5.2 Normal Traffic

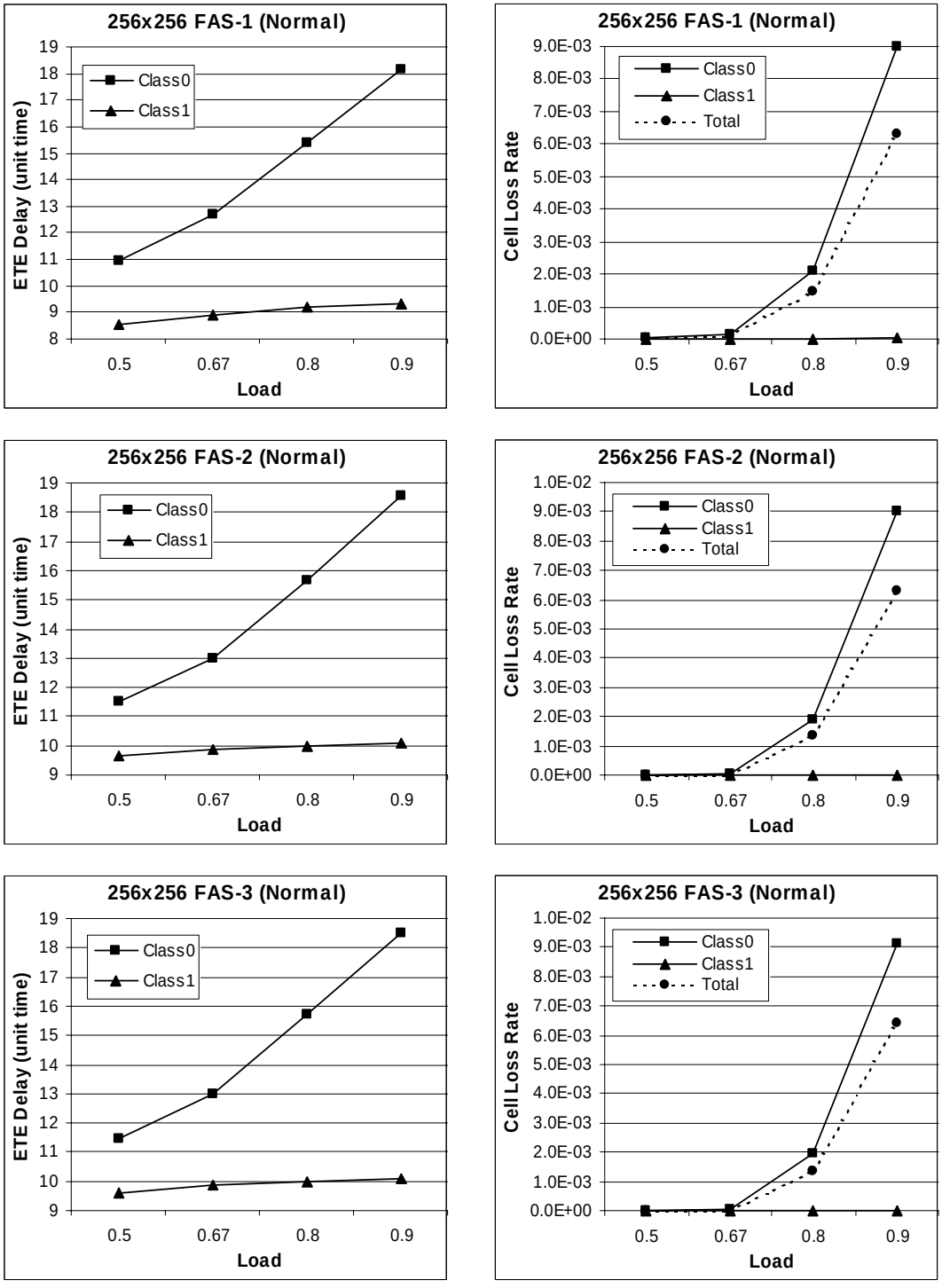


Figure A.18 ETE Delay and cell loss rate of 256x256 FAS-1/2/3 under normal traffic.

Table A.18. ETE Delay and Cell Loss Rate of 256x256 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	10.93	12.66	15.40	18.18
	Class-1	8.55	8.91	9.17	9.35
FAS-2	Class-0	11.51	12.99	15.69	18.55
	Class-1	9.63	9.86	10.00	10.12
FAS-3	Class-0	11.47	12.97	15.70	18.52
	Class-1	9.62	9.87	10.01	10.12

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	3.86E-05	1.25E-04	2.10E-03	9.01E-03
	Class-1	5.86E-06	1.42E-05	2.31E-05	2.85E-05
	Total	2.88E-05	9.19E-05	1.47E-03	6.32E-03
FAS-2	Class-0	0.00E+00	3.35E-05	1.93E-03	9.04E-03
	Class-1	0.00E+00	3.91E-07	5.21E-07	2.60E-07
	Total	0.00E+00	2.36E-05	1.35E-03	6.33E-03
FAS-3	Class-0	0.00E+00	3.54E-05	1.97E-03	9.15E-03
	Class-1	1.30E-07	1.30E-07	0.00E+00	0.00E+00
	Total	3.91E-08	2.48E-05	1.38E-03	6.40E-03

A.5.3 Bur-5 Traffic

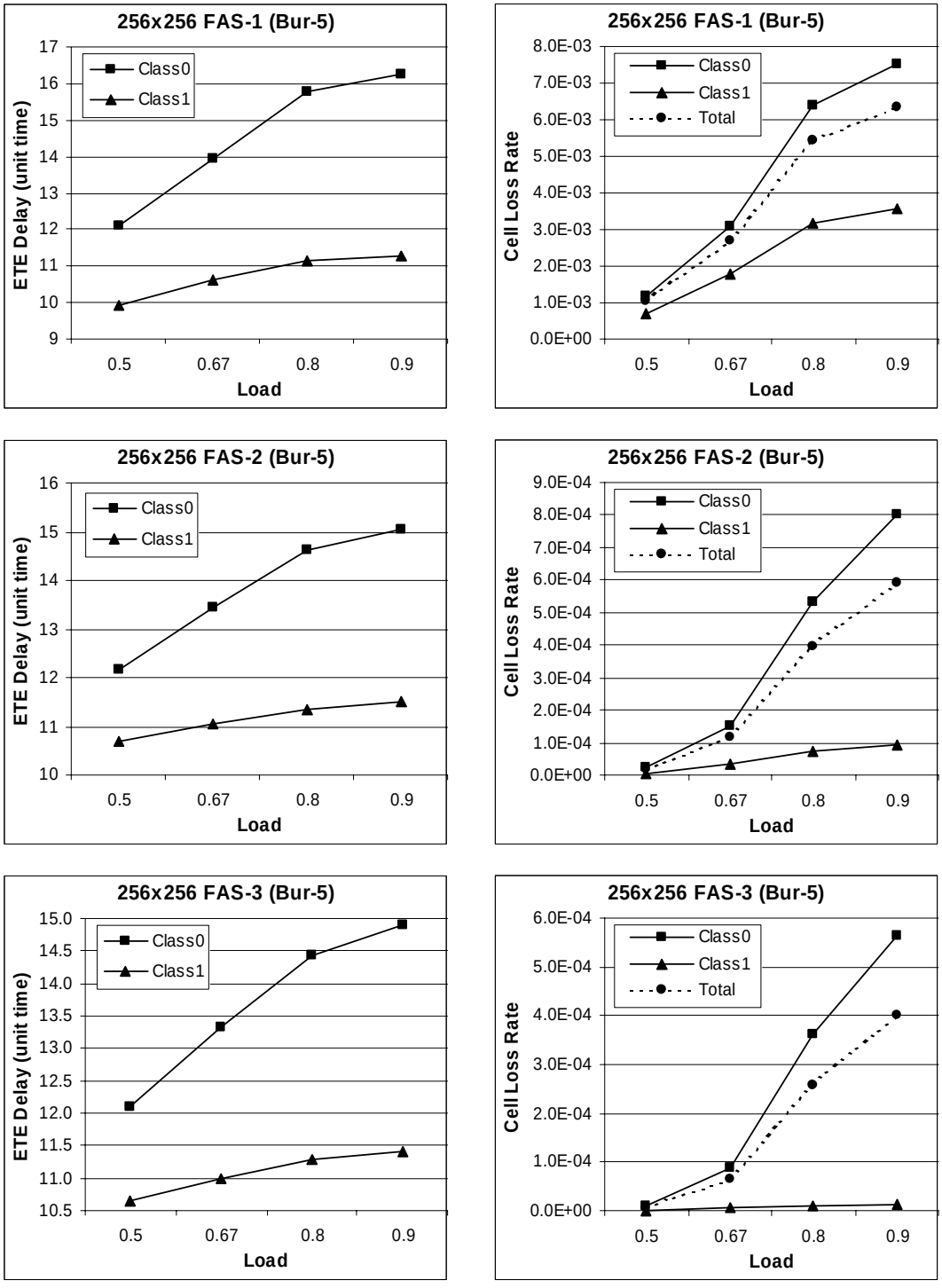


Figure A.19 ETE Delay and cell loss rate of 256x256 FAS-1/2/3 under Bur-5 traffic.

Table A.19. ETE Delay and Cell Loss Rate of 256x256 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	12.11	13.95	15.77	16.27
	Class-1	9.94	10.61	11.13	11.28
FAS-2	Class-0	12.17	13.44	14.62	15.06
	Class-1	10.70	11.05	11.33	11.52
FAS-3	Class-0	12.10	13.33	14.44	14.91
	Class-1	10.65	10.98	11.29	11.40

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.16E-03	3.10E-03	6.38E-03	7.52E-03
	Class-1	7.10E-04	1.78E-03	3.18E-03	3.56E-03
	Total	1.03E-03	2.71E-03	5.42E-03	6.33E-03
FAS-2	Class-0	2.31E-05	1.54E-04	5.35E-04	8.03E-04
	Class-1	3.91E-06	3.24E-05	7.11E-05	9.27E-05
	Total	1.77E-05	1.18E-04	3.96E-04	5.90E-04
FAS-3	Class-0	8.37E-06	8.85E-05	3.63E-04	5.63E-04
	Class-1	5.21E-07	5.73E-06	8.21E-06	1.43E-05
	Total	6.02E-06	6.37E-05	2.57E-04	4.00E-04

A.5.4 Bur-10 Traffic

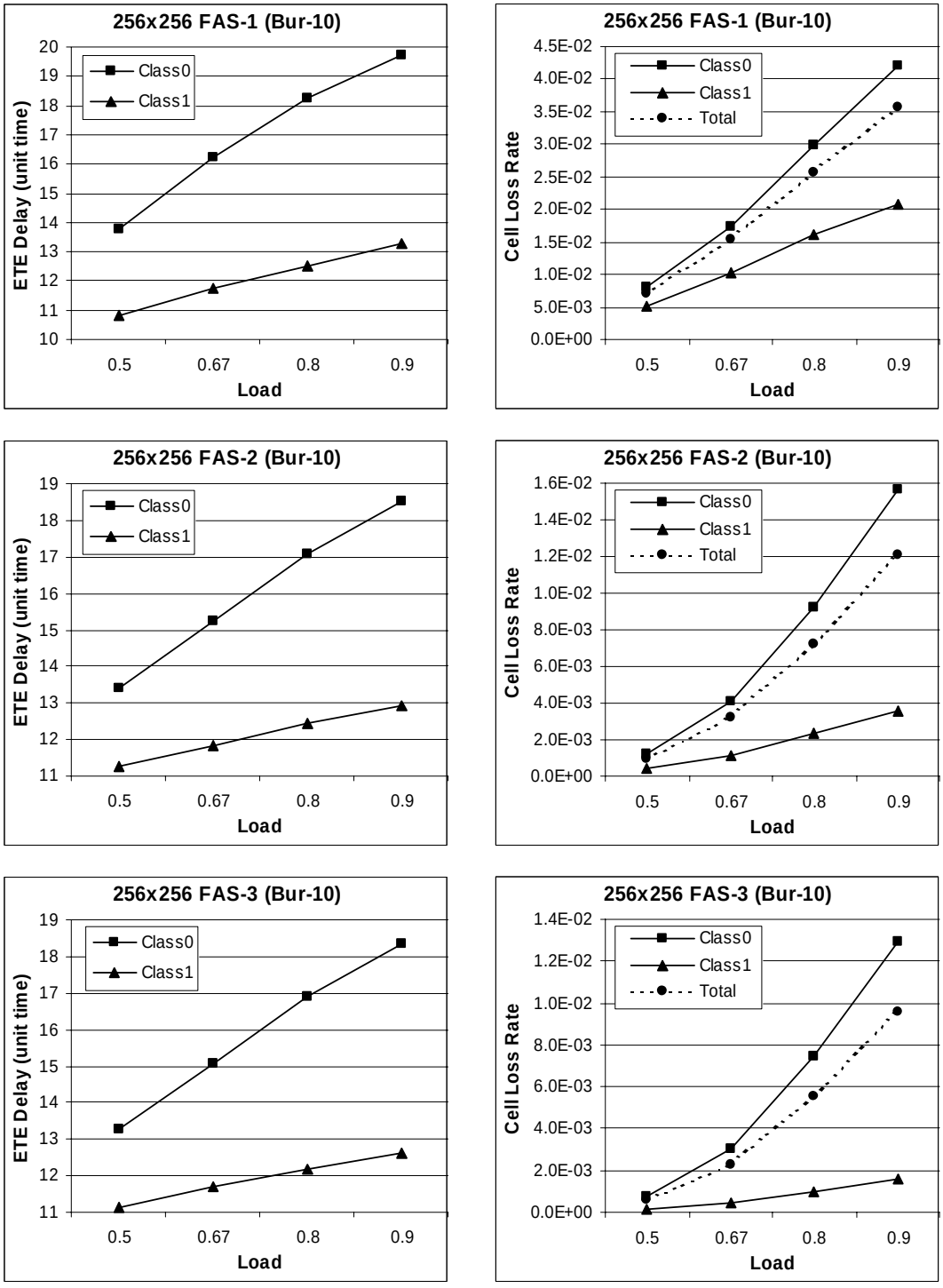


Figure A.20 ETE Delay and cell loss rate of 256x256 FAS-1/2/3 under Bur-10 traffic.

Table A.20. ETE Delay and Cell Loss Rate of 256x256 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	13.79	16.23	18.26	19.71
	Class-1	10.81	11.76	12.51	13.27
FAS-2	Class-0	13.41	15.25	17.06	18.53
	Class-1	11.27	11.85	12.44	12.92
FAS-3	Class-0	13.29	15.05	16.91	18.33
	Class-1	11.12	11.68	12.19	12.61

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.97E-03	1.75E-02	2.98E-02	4.20E-02
	Class-1	5.23E-03	1.04E-02	1.61E-02	2.09E-02
	Total	7.15E-03	1.54E-02	2.57E-02	3.57E-02
FAS-2	Class-0	1.21E-03	4.09E-03	9.26E-03	1.57E-02
	Class-1	4.63E-04	1.16E-03	2.35E-03	3.54E-03
	Total	9.84E-04	3.21E-03	7.19E-03	1.21E-02
FAS-3	Class-0	7.78E-04	3.07E-03	7.48E-03	1.30E-02
	Class-1	1.69E-04	4.76E-04	9.85E-04	1.58E-03
	Total	5.95E-04	2.29E-03	5.53E-03	9.55E-03

A.6 Size 512x512

A.6.1 Uniform Traffic

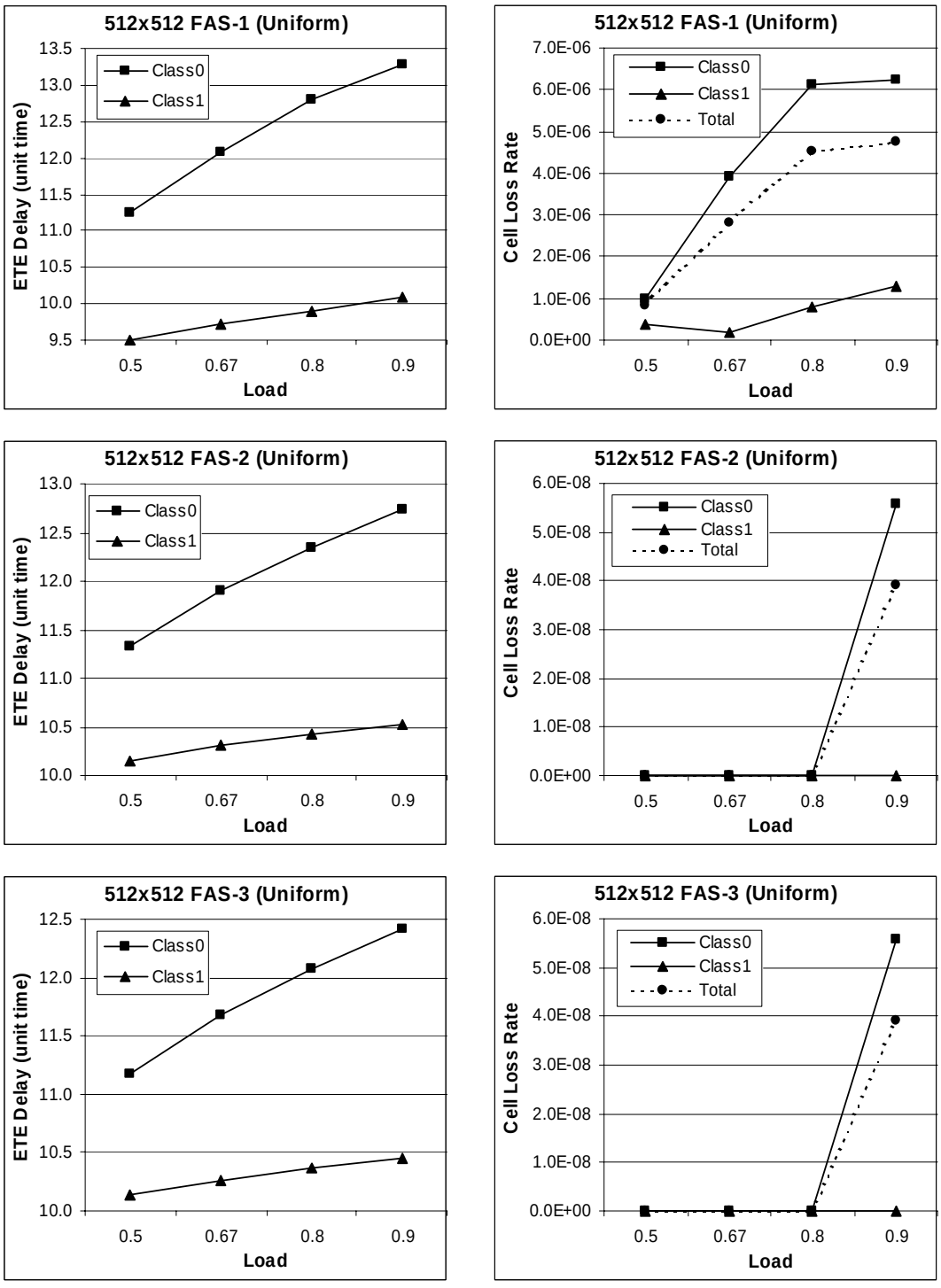


Figure A.21 ETE Delay and cell loss rate of 512x512 FAS-1/2/3 under uniform traffic.

Table A.21. ETE Delay and Cell Loss Rate of 512x512 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	11.25	12.09	12.79	13.28
	Class-1	9.49	9.71	9.89	10.08
FAS-2	Class-0	11.33	11.90	12.34	12.73
	Class-1	10.16	10.31	10.43	10.52
FAS-3	Class-0	11.18	11.67	12.08	12.41
	Class-1	10.13	10.26	10.37	10.45

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	1.00E-06	3.93E-06	6.11E-06	6.22E-06
	Class-1	3.91E-07	1.95E-07	7.81E-07	1.30E-06
	Total	8.20E-07	2.81E-06	4.51E-06	4.75E-06
FAS-2	Class-0	0.00E+00	0.00E+00	0.00E+00	5.58E-08
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	3.91E-08
FAS-3	Class-0	0.00E+00	0.00E+00	0.00E+00	5.58E-08
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	3.91E-08

A.6.2 Normal Traffic

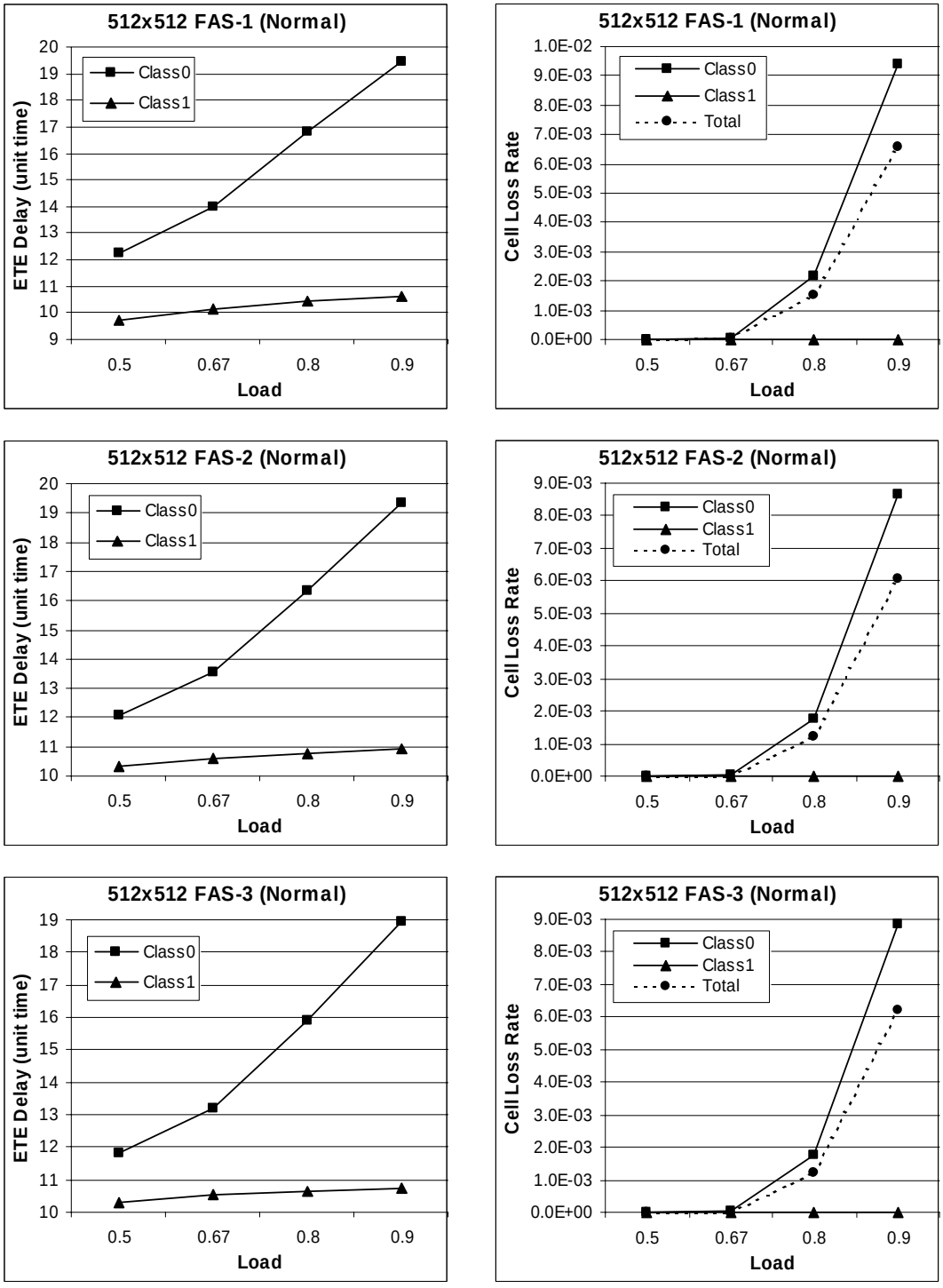


Figure A.22 ETE Delay and cell loss rate of 512x512 FAS-1/2/3 under normal traffic.

Table A.22. ETE Delay and Cell Loss Rate of 512x512 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	12.23	14.01	16.80	19.47
	Class-1	9.74	10.16	10.43	10.62
FAS-2	Class-0	12.07	13.56	16.36	19.32
	Class-1	10.34	10.60	10.78	10.95
FAS-3	Class-0	11.82	13.19	15.90	18.93
	Class-1	10.29	10.52	10.65	10.74

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.15E-06	6.77E-05	2.19E-03	9.41E-03
	Class-1	1.30E-06	2.87E-06	5.41E-06	7.42E-06
	Total	6.09E-06	4.82E-05	1.53E-03	6.59E-03
FAS-2	Class-0	1.95E-07	3.08E-05	1.78E-03	8.65E-03
	Class-1	0.00E+00	6.51E-08	1.11E-06	2.67E-06
	Total	1.37E-07	2.16E-05	1.24E-03	6.06E-03
FAS-3	Class-0	5.58E-08	2.50E-05	1.77E-03	8.87E-03
	Class-1	0.00E+00	0.00E+00	3.26E-07	2.61E-07
	Total	3.91E-08	1.75E-05	1.24E-03	6.21E-03

A.6.3 Bur-5 Traffic

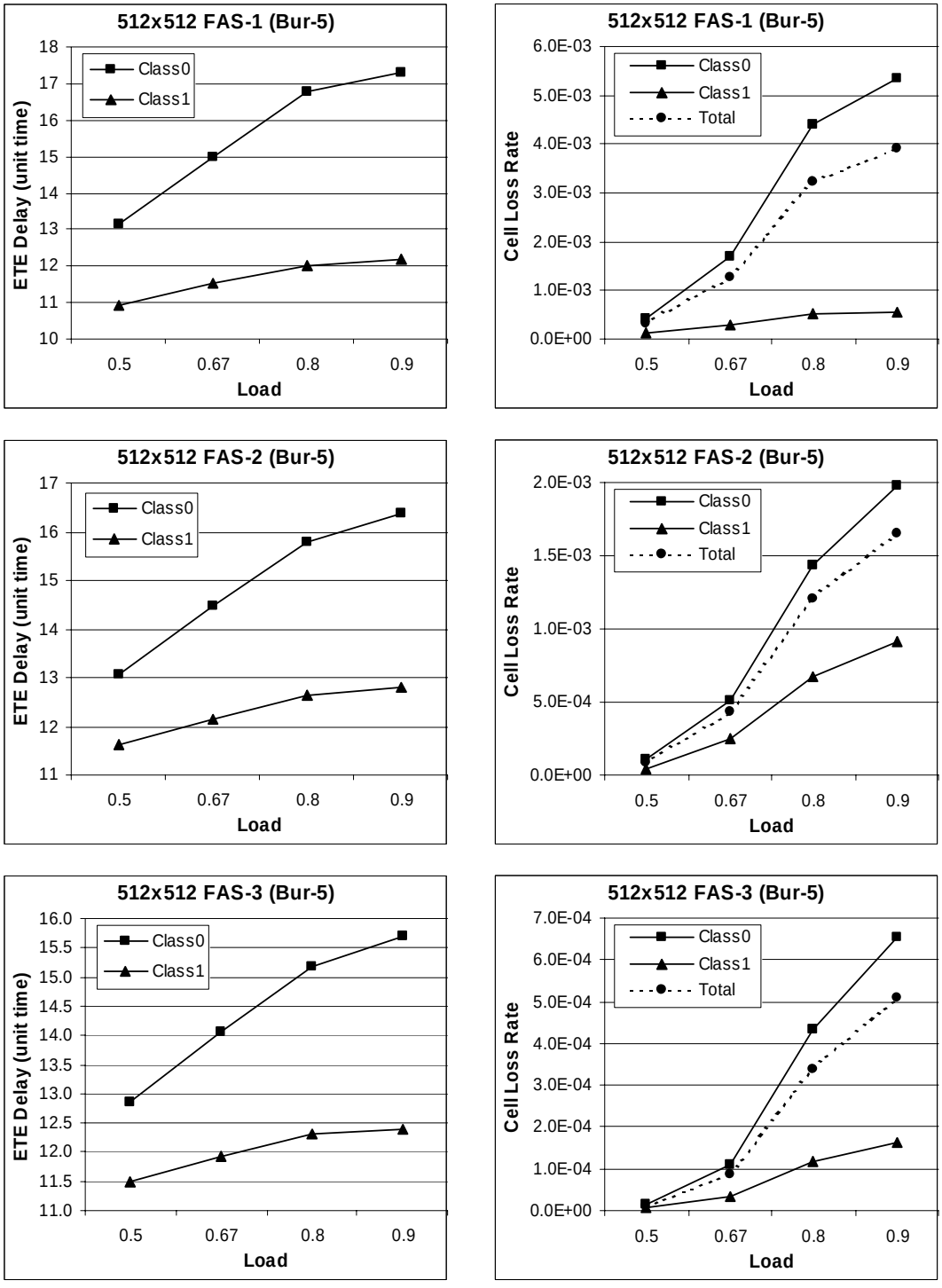


Figure A.23 ETE Delay and cell loss rate of 512x512 FAS-1/2/3 under Bur-5 traffic.

Table A.23. ETE Delay and Cell Loss Rate of 512x512 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	13.15	15.00	16.78	17.29
	Class-1	10.91	11.54	12.00	12.18
FAS-2	Class-0	13.08	14.46	15.79	16.38
	Class-1	11.62	12.15	12.63	12.79
FAS-3	Class-0	12.85	14.06	15.18	15.70
	Class-1	11.48	11.92	12.32	12.40

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	4.19E-04	1.68E-03	4.41E-03	5.34E-03
	Class-1	1.18E-04	3.01E-04	5.06E-04	5.64E-04
	Total	3.28E-04	1.27E-03	3.24E-03	3.91E-03
FAS-2	Class-0	1.05E-04	5.12E-04	1.43E-03	1.98E-03
	Class-1	4.60E-05	2.47E-04	6.72E-04	9.09E-04
	Total	8.76E-05	4.33E-04	1.21E-03	1.66E-03
FAS-3	Class-0	1.47E-05	1.10E-04	4.33E-04	6.56E-04
	Class-1	5.99E-06	3.60E-05	1.19E-04	1.63E-04
	Total	1.21E-05	8.75E-05	3.39E-04	5.08E-04

A.6.4 Bur-10 Traffic

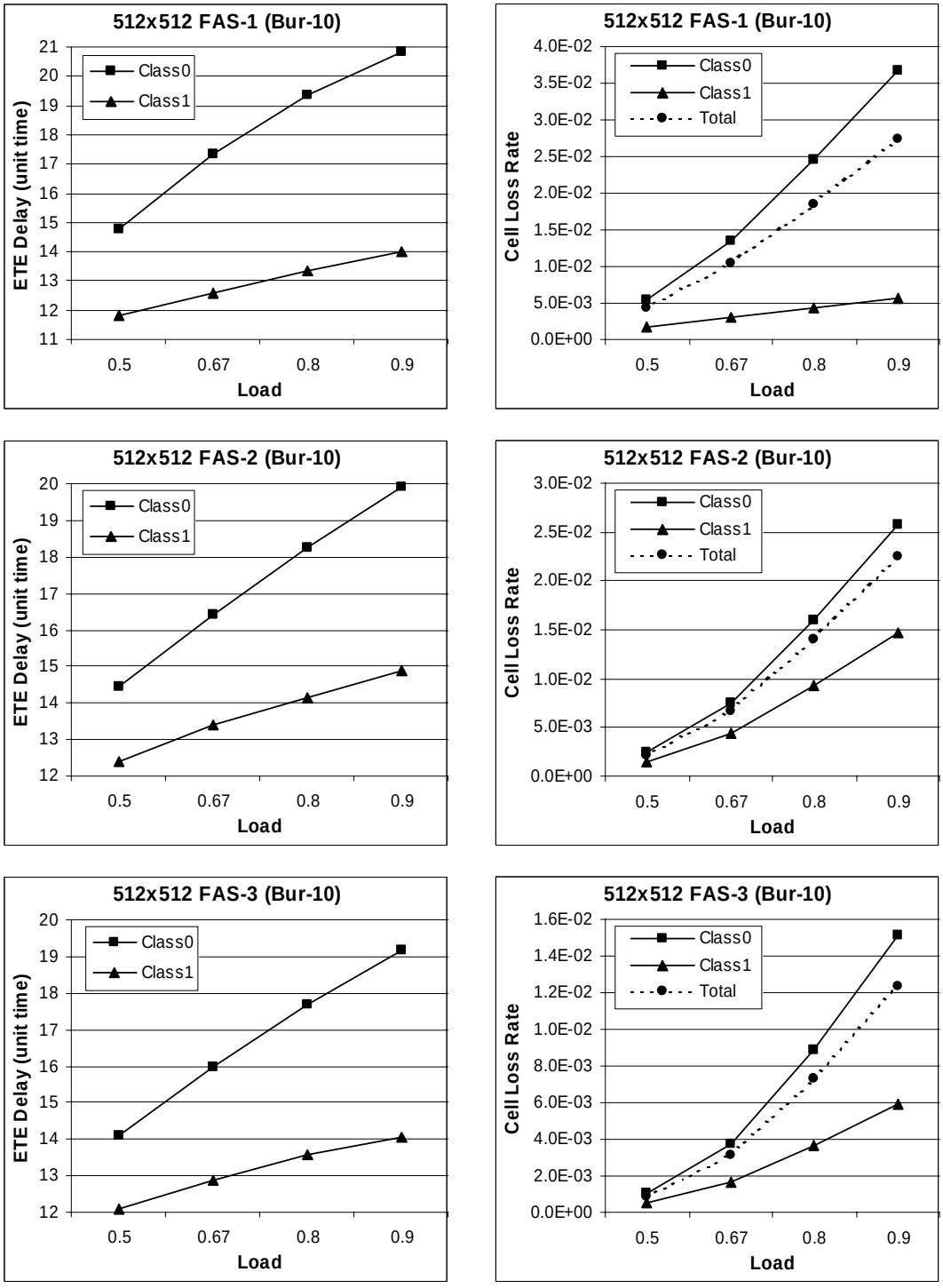


Figure A.24 ETE Delay and cell loss rate of 512x512 FAS-1/2/3 under Bur-10 traffic.

Table A.24. ETE Delay and Cell Loss Rate of 512x512 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	14.76	17.35	19.34	20.83
	Class-1	11.79	12.58	13.37	14.02
FAS-2	Class-0	14.45	16.41	18.26	19.91
	Class-1	12.39	13.38	14.13	14.90
FAS-3	Class-0	14.09	15.98	17.70	19.19
	Class-1	12.11	12.87	13.58	14.04

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	5.38E-03	1.36E-02	2.46E-02	3.68E-02
	Class-1	1.70E-03	3.15E-03	4.44E-03	5.73E-03
	Total	4.27E-03	1.04E-02	1.86E-02	2.75E-02
FAS-2	Class-0	2.43E-03	7.54E-03	1.60E-02	2.58E-02
	Class-1	1.48E-03	4.47E-03	9.26E-03	1.47E-02
	Total	2.15E-03	6.62E-03	1.40E-02	2.25E-02
FAS-3	Class-0	1.01E-03	3.73E-03	8.86E-03	1.51E-02
	Class-1	5.18E-04	1.67E-03	3.67E-03	5.94E-03
	Total	8.62E-04	3.11E-03	7.30E-03	1.23E-02

A.7 Size 1024x1024

A.7.1 Uniform Traffic

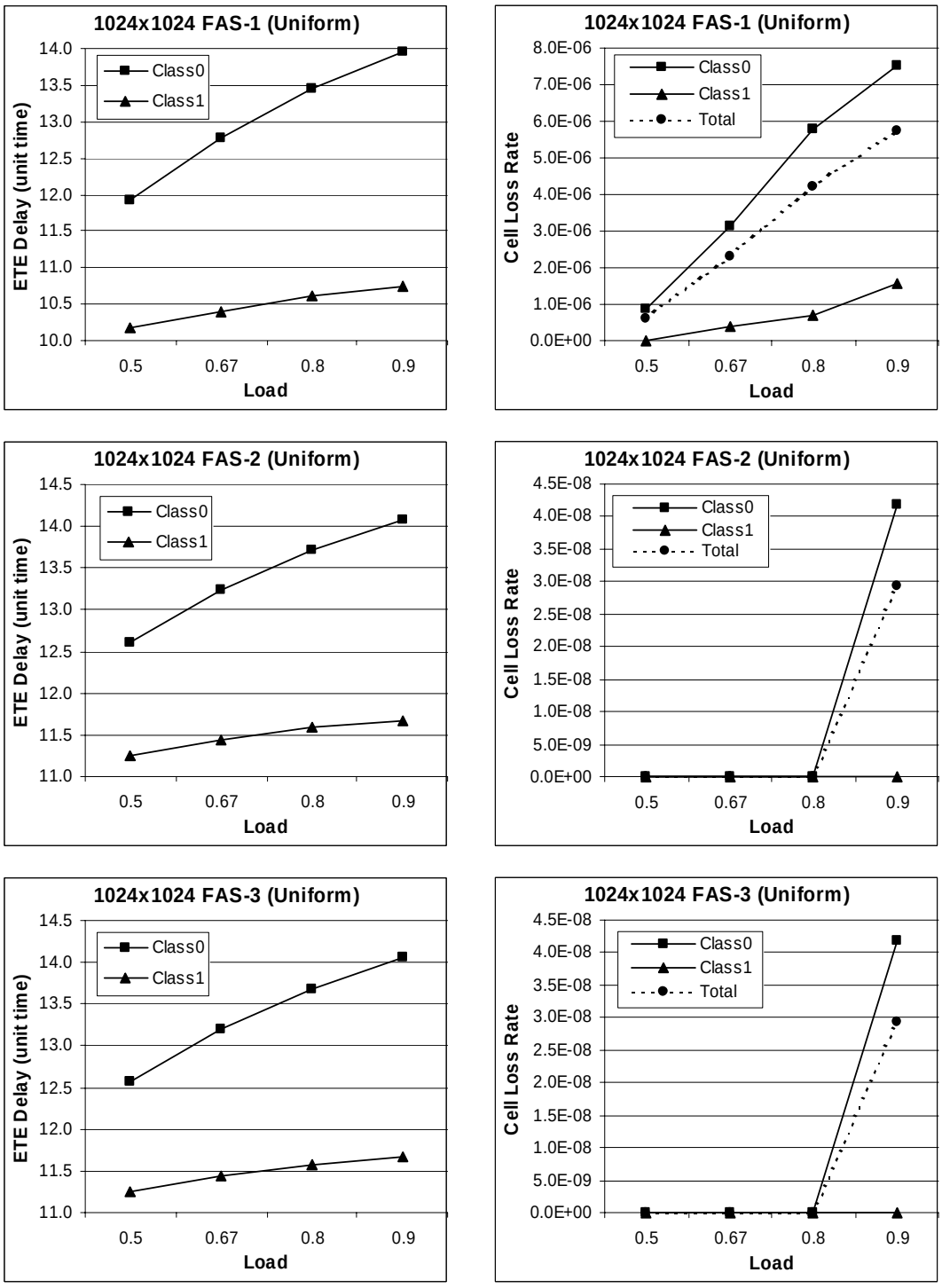


Figure A.25 ETE Delay and cell loss rate of 1024x1024 FAS-1/2/3 under uniform traffic.

Table A.25. ETE Delay and Cell Loss Rate of 1024x1024 FAS-1/2/3 Under Uniform Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	11.92	12.77	13.46	13.96
	Class-1	10.18	10.40	10.61	10.75
FAS-2	Class-0	12.62	13.24	13.72	14.08
	Class-1	11.26	11.45	11.59	11.67
FAS-3	Class-0	12.58	13.21	13.68	14.06
	Class-1	11.26	11.45	11.58	11.67

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.79E-07	3.13E-06	5.76E-06	7.52E-06
	Class-1	0.00E+00	3.91E-07	6.84E-07	1.56E-06
	Total	6.15E-07	2.31E-06	4.24E-06	5.73E-06
FAS-2	Class-0	0.00E+00	0.00E+00	0.00E+00	4.19E-08
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	2.93E-08
FAS-3	Class-0	0.00E+00	0.00E+00	0.00E+00	4.19E-08
	Class-1	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Total	0.00E+00	0.00E+00	0.00E+00	2.93E-08

A.7.2 Normal Traffic

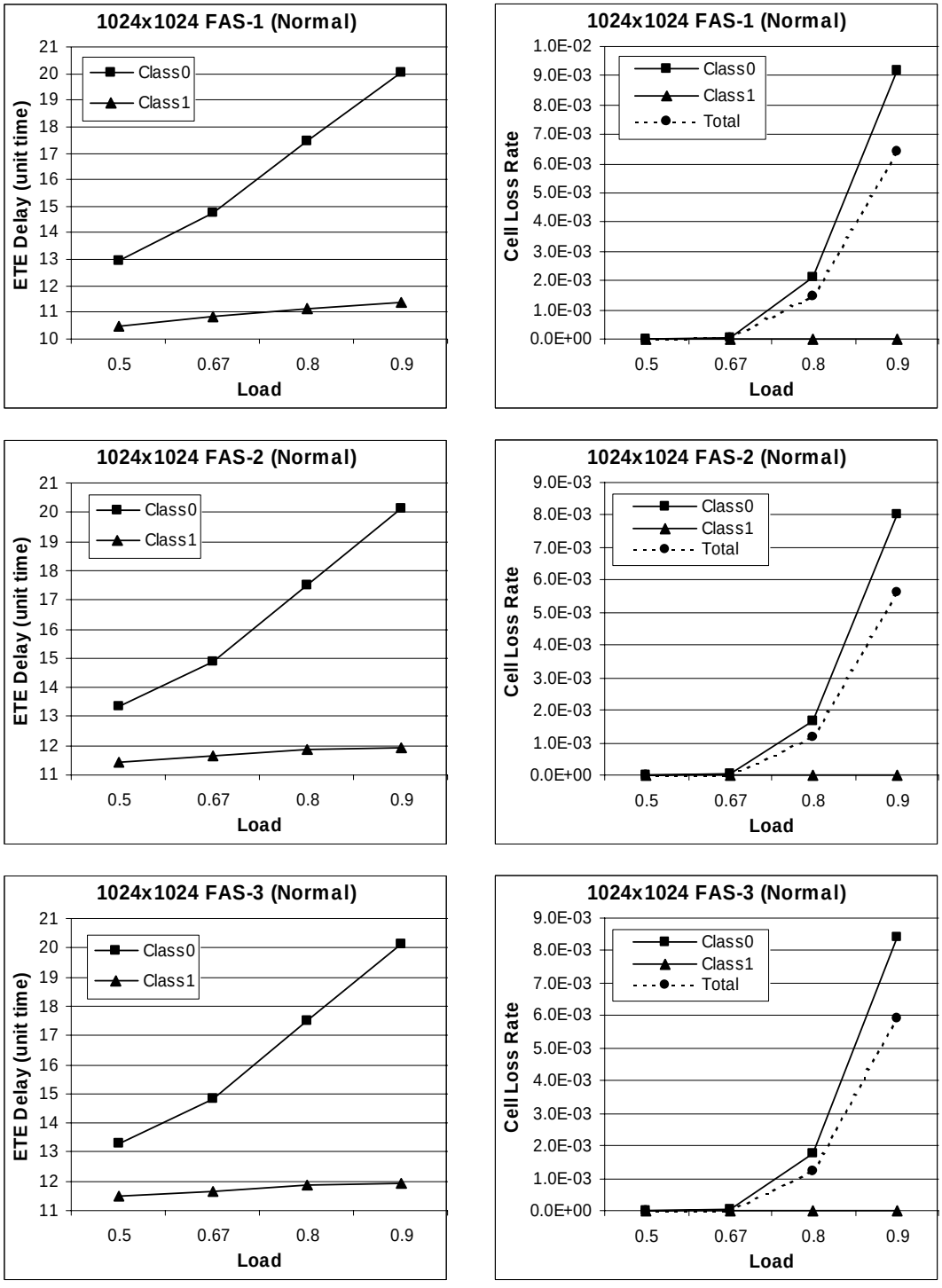


Figure A.26 ETE Delay and cell loss rate of 1024x1024 FAS-1/2/3 under normal traffic.

Table A.26. ETE Delay and Cell Loss Rate of 1024x1024 FAS-1/2/3 Under Normal Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	12.94	14.75	17.48	20.04
	Class-1	10.46	10.82	11.13	11.38
FAS-2	Class-0	13.36	14.86	17.51	20.13
	Class-1	11.45	11.67	11.86	11.95
FAS-3	Class-0	13.32	14.85	17.50	20.13
	Class-1	11.49	11.67	11.88	11.95

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.51E-06	6.68E-05	2.12E-03	9.20E-03
	Class-1	6.84E-07	4.75E-06	6.25E-06	9.02E-06
	Total	5.46E-06	4.82E-05	1.48E-03	6.44E-03
FAS-2	Class-0	1.81E-07	2.86E-05	1.65E-03	8.01E-03
	Class-1	0.00E+00	6.51E-08	9.77E-07	2.47E-06
	Total	1.27E-07	2.01E-06	1.16E-03	5.61E-03
FAS-3	Class-0	5.58E-08	2.48E-05	1.75E-03	8.42E-03
	Class-1	0.00E+00	0.00E+00	2.60E-07	3.26E-07
	Total	3.91E-08	8.69E-06	1.22E-03	5.90E-03

A.7.3 Bur-5 Traffic

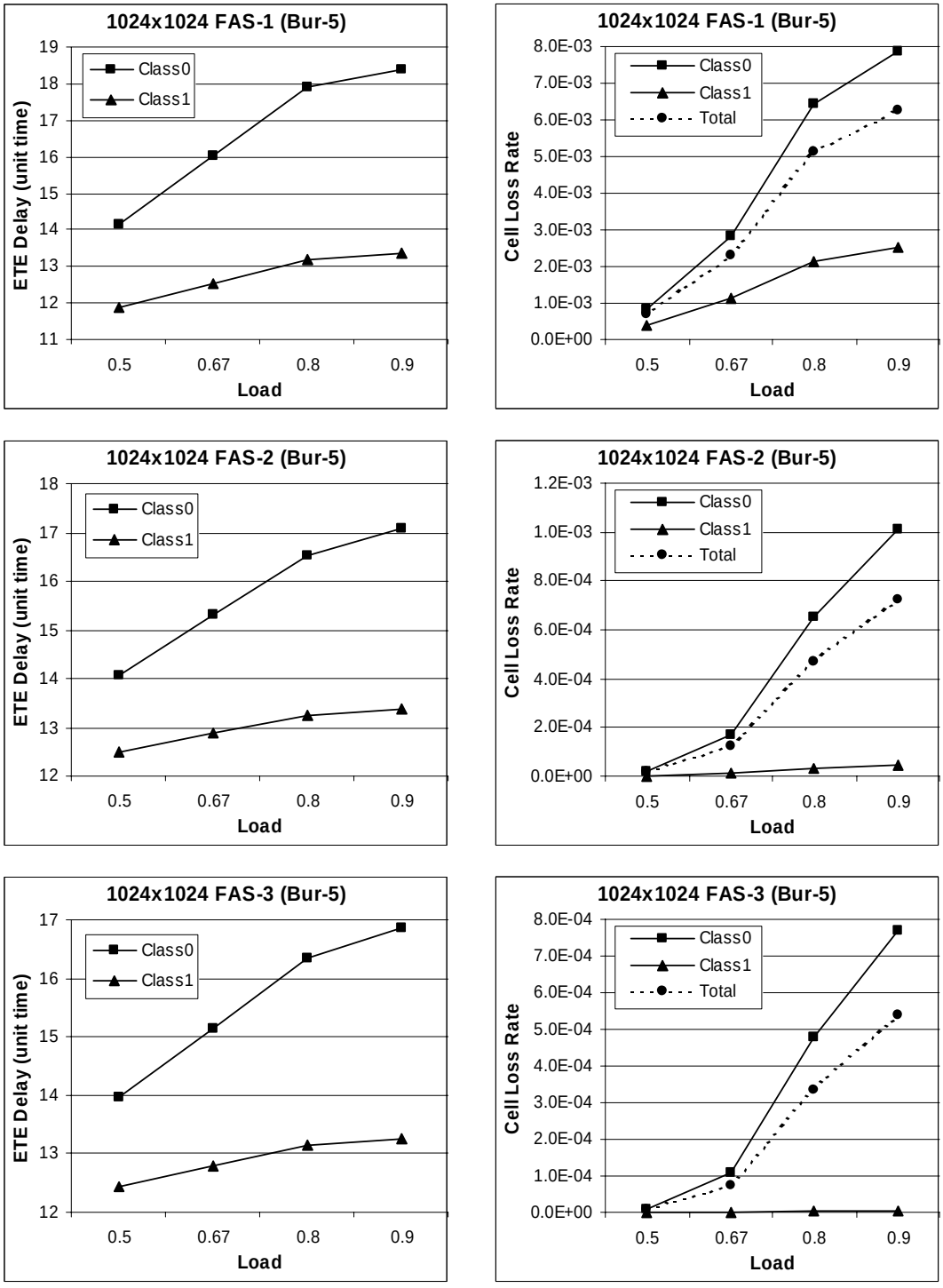


Figure A.27 ETE Delay and cell loss rate of 1024x1024 FAS-1/2/3 under Bur-5 traffic.

Table A.27. ETE Delay and Cell Loss Rate of 1024x1024 FAS-1/2/3 Under Bur-5 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	14.14	16.05	17.92	18.39
	Class-1	11.87	12.54	13.16	13.36
FAS-2	Class-0	14.06	15.31	16.53	17.09
	Class-1	12.50	12.87	13.25	13.38
FAS-3	Class-0	13.97	15.15	16.35	16.85
	Class-1	12.44	12.80	13.14	13.25

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	8.26E-04	2.82E-03	6.44E-03	7.87E-03
	Class-1	3.87E-04	1.11E-03	2.11E-03	2.54E-03
	Total	6.94E-04	2.31E-03	5.14E-03	6.27E-03
FAS-2	Class-0	2.26E-05	1.69E-04	6.55E-04	1.01E-03
	Class-1	1.86E-06	1.16E-05	3.18E-05	4.83E-05
	Total	1.64E-05	1.22E-04	4.68E-04	7.24E-04
FAS-3	Class-0	1.02E-05	1.07E-04	4.77E-04	7.70E-04
	Class-1	6.51E-08	8.79E-07	4.20E-06	5.31E-06
	Total	7.16E-06	7.53E-05	3.35E-04	5.40E-04

A.7.4 Bur-10 Traffic

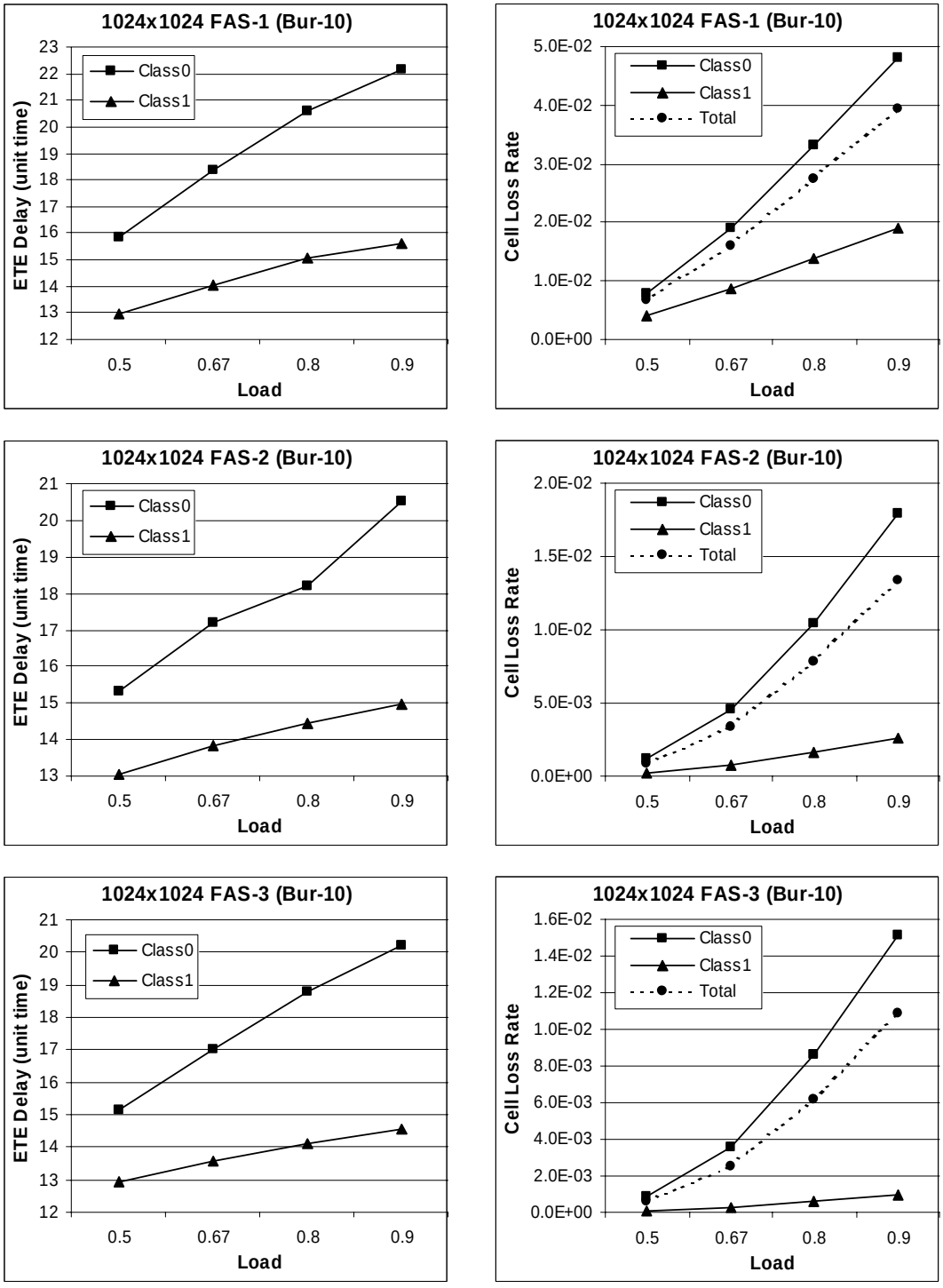


Figure A.28 ETE Delay and cell loss rate of 1024x1024 FAS-1/2/3 under Bur-10 traffic.

Table A.28. ETE Delay and Cell Loss Rate of 1024x1024 FAS-1/2/3 Under Bur-10 Traffic.

		ETE Delay			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	15.83	18.37	20.61	22.14
	Class-1	12.96	14.07	15.09	15.62
FAS-2	Class-0	15.33	17.20	18.18	20.51
	Class-1	13.05	13.85	14.44	14.95
FAS-3	Class-0	15.17	17.02	18.81	20.21
	Class-1	12.94	13.58	14.13	14.53

		Cell Loss Rate			
	Loading	0.5	0.67	0.8	0.9
FAS-1	Class-0	7.94E-03	1.90E-02	3.32E-02	4.82E-02
	Class-1	4.18E-03	8.80E-03	1.39E-02	1.89E-02
	Total	6.81E-03	1.59E-02	2.74E-02	3.94E-02
FAS-2	Class-0	1.19E-03	4.51E-03	1.05E-02	1.80E-02
	Class-1	2.57E-04	8.07E-04	1.65E-03	2.62E-03
	Total	9.11E-04	3.40E-03	7.82E-03	1.34E-02
FAS-3	Class-0	8.64E-04	3.53E-03	8.61E-03	1.51E-02
	Class-1	8.48E-05	2.77E-04	5.87E-04	9.23E-04
	Total	6.30E-04	2.55E-03	6.20E-03	1.09E-02

Vita

Jahng Sun Park

Jahng Sun Park was born on September 6, 1968 in Seoul, Korea. Because of his father who worked for Korean Overseas Information Services and worked overseas frequently, he attended three years of elementary school in Hong Kong and high school in Fairfax County, Virginia, U.S.A. He completed his remaining elementary education and middle school in Seoul, Korea. From September 1985 to June 1986, he attended Thomas Jefferson High School for Science and Technology, a Virginia Governor's magnet school. In May 1990, he received his Bachelor of Science degree in electrical engineering at Virginia Tech. He continued to work towards his Master's degree at Virginia Tech. During the period of September 1992 to April 1993, he returned to Korea to fulfill the mandatory military service. After returning to Virginia Tech, he received his Master of Science degree in electrical engineering in February 1994.

During his study for the doctorate degree, he worked as a research assistant or teaching assistant every semester before he worked as an instructor for the electrical and computer engineering department. From August 1998 to June 2001, he taught undergraduate computer engineering courses. He is an active member of the Institute of Electrical and Electronics Engineers (IEEE), IEEE Computer Society, and IEEE Communications Society.

His parents and youngest sister lives in Seoul, Korea, and his other sister lives in Santa Clara, CA with her husband and her daughter.

His publications include.

J. S. Park and N. J. Davis IV, "Modeling the Folded Hypercube ATM Switches," *The Proceedings of OPNETWORK 2001*, Washington DC, August 2001.

J. S. Park and N. J. Davis IV, "The Folded Hypercube ATM Switches," *The Proceedings of IEEE International Conference on Networking*, July 2001, Part II, pp. 370–379.

J. S. Park and N. J. Davis IV, "Modeling the Folded Hypercube Network with OPNET," *The Proceedings of OPNETWORK '99*, Washington DC, August 1999.

V. Ramachandran, R. A. Raines, J. S. Park, and N. J. Davis IV, "Performance Studies of Packet Switched Augmented Shuffle Exchange Networks," *4th Symposium on the Frontiers of Massively Parallel Computation*, October 1992, pp.566–568.