BUFFER MANAGEMENT IN THE FUTURE INTERNET

Divya Balakrishna Pillai

20044234

A dissertation submitted in fulfilment of the requirements for the degree of

Master of Science (Computer Science)

Department of Computer Science, Faculty of Science and Agriculture, University of Zululand

2007

DECLARATION

I, Divya Balakrishna Pillai, declare that this dissertation represents the authors' work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from published or unpublished work of others has been acknowledged in the text and a list of references is given.

[illegible]

Signature of student

DEDICATION

I dedicate this work to my loving family, for believing in and encouraging me through the difficult times. None of this would be possible without their love, support and encouragement.

ACKNOWLEDGMENTS

I thank the Lord for all His blessings bestowed on me in seeing this research work through; without which this research could not have become a reality.

I would like to extend my sincere thanks to my supervisor Dr S.S Xulu for his advice, guidance and fatherly talks. He gave me courage to face the challenges in research.

I extend my special thanks to my co-supervisor, Mr. G.E Ojong, who willingly assisted and guided me right through my research work, and for being very patient with me. Thank you for believing in me and giving me the courage to face up to my uncertainties.

I would also like to thank Prof. M. O. Adigun for giving us fatherly advice and guidance in making us become better researchers.

I would also like to thank my family, my colleagues, and my most valuable friends, for being there for me and for supporting me during this time of my life.

I am indebted to all who have helped me in making this research work a great success in all aspects.

ABSTRACT

With the rapid development of new applications, comes the drive to modify the current Internet to accommodate real-time multimedia applications. The current Internet uses a single queue per output port to buffer packets destined for that port. This often causes congestion leading to packet loss and delay. Real-time applications are delay and loss sensitive. Therefore, there is a need to develop a buffer management system that will effectively accommodate both real-time and non-real-time applications.

In an attempt to efficiently allocate and manage output buffers of a router in the future Internet, in this research we have developed a buffer management scheme known as Dynamic Threshold Buffer Allocation Scheme (DTBAS). This scheme uses complete sharing with virtual partitioning. Pre-emption (i.e. removal of queued packets) is used to minimise congestion of high priority packets. Dynamic thresholds are used to determine the start and end of pre-emption. To further alleviate the congestion of high priority (real-time) packets, high priority OUT-packets are randomly dropped during the pre-emption period. To add some fairness to the scheme, low priority (non-real-time) packets are assigned a minimum buffer volume.

Simulation was conducted to evaluate the performance of the proposed scheme. The scheme was also compared with the Complete Sharing and Complete Partitioning schemes. It was found that DTBAS had the lowest average packet loss rate for real-time applications compared to other schemes. It was also found that DTBAS efficiently utilises its buffer space.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1 OVERVIEW

The Internet is playing an important role in our daily lives and the number of Internet users is increasing rapidly on a daily basis. Currently, many new applications such as real-time multimedia applications, are being developed to offer more and more interesting services to users. Bandwidth is one of the resources that is negatively affected by these new services as these applications require a large amount of bandwidth to handle them. Bandwidth and buffers are the main network resources that need to be considered in the transmission of flows. There has been considerable development in memory technology in such a way that memory has become far cheaper than in previous years. Nevertheless, buffer space will continue to be an important constraint on the Internet, as the future Internet would need to accommodate real-time resource-hungry applications. Many real-time applications require large capacities of buffer space to store packets effectively. A lot of congestion is created if bandwidth and buffer are not properly managed within the hosts or the routers of the network. Hence the focus of our research is to efficiently manage the buffers by allocating buffer space for both real-time and non-real-time applications to reduce the packet loss rate as well as to increase the utilization of buffer space.

This chapter is structured as follows: In the next two sections we briefly introduce issues around the following concepts: the current Internet, and buffer management. This is then followed by the presentation of our research problem, rational, goal, objectives, and methodology. We conclude the chapter with a brief discussion of the structure of the dissertation.

1.2 THE INTERNET

The current Internet uses a Best Effort service model. This model does not assure any guarantees as to timeliness or actual delivery of packets, since these packets are all treated in the same way. The focus of today's Internet is more on trying to get the packets to their destination. Little emphasis is placed on when the packets will arrive at the destination. This was acceptable during the early days because applications usually did not demand real-time services. The network was mainly used for information sharing; such as electronic mail, web access, and file transfer.

The actual delivery of packets in the current Internet is not guaranteed and, there is a possibility of high packet loss during periods of congestion. It is very important to avoid high packet losses, especially when real-time applications are involved. In this case, when a packet is dropped before it reaches its destination; it is not transmitted since there is a set time that packets have within which they should arrive at their destinations. When packets are lost, the quality of real-time applications is affected adversely. Therefore, appropriate congestion control mechanisms are needed to alleviate the congestion at the buffers, while maintaining high network efficiency. In the next subsections, we introduce concepts of congestion control and Quality of service (QoS). We also give a brief discussion of efforts that have been made to introduce *QoS* on the Internet.

1.2.1 Congestion Control

Congestion is a common problem on shared networks where multiple users contend for access to the same resource such as bandwidth or buffers. Congestion occurs in a network if the load on the network is greater than the capacity of the network. New Internet applications, such as real-time multimedia applications, are badly affected by congestion because it causes excessive dropping of packets [13]. The dropping of packets affects the quality of real-time applications more than non-real-time applications. This is because real-time applications are sensitive to packet losses and delay.

Congestion control refers to the mechanisms directed at managing the congestion and keeping the load on the network below the network capacity. In packet-switched networks, packets move in and out of buffers of switching devices as they traverse the network. A characteristic of this network is that packets may arrive in bursts from one or more sources. Buffers help routers absorb bursts until routers can catch up. If traffic is excessive then buffers fill up and new incoming packets are dropped. Increasing the size of the buffer is not a solution, because excessive buffer size can lead to excessive delay. In order to solve this problem, appropriate buffer management mechanisms must be implemented that can manage the allocation of packets within the buffers and also control the delay and packet loss rate. Hence, the need for Quality of Service (QoS) and we next introduce this concept as well as QoS requirements for different applications.

1.2.2 Quality of Service

Quality of service (QoS) is the capability of a network to provide better service to selected network traffic or users. These services can be specified using certain characteristics such as bandwidth, loss rate, delay and delay jitter. With the introduction of new applications on the Internet, there is a demand for the current network infrastructure to be modified in order to provide QoS to these applications. Below, is a brief discussion of QoS requirements for different types of applications and we also consider the QoS challenges for real-time applications.

a) QoS requirements for different types of applications

The Internet carries all kinds of traffic: both real-time and non-real time. Each of these traffic kinds have different requirements for QoS characteristics. Delay is one of the characteristics to be considered when transmitting real-time applications. A suitable network should have the following characteristics:

- There should be minimum loss of packets during packet delivery.
- Packet sequence should be the same, both at source and destination.
- There should be minimum delay experienced while delivering packets.

The two main QoS characteristics that affect real-time packet delivery are delay and packet loss. Minimising delay and packet loss rate will improve the QoS of traffic (real-time and non-real-time applications) over the internet. Applications can therefore be broadly categorised as sensitive to packet delays and/or sensitive to packet loss.

i). Applications sensitive to packet delays

Olifer and Olifer [37] categorised applications according to their sensitivity to packet delays as follows:

⇒ *Asynchronous applications*: These applications are elastic in nature and are not sensitive to packet delay. E-mail is an example of such an application.

⇒ *Interactive applications*: These applications are less sensitive to packet delay. A text editor accessing a remote file is an example of such an application.

⇒ *Isochronous applications*: These applications can tolerate delay to an extent, but if delay reaches a maximum, it affects the functionality of the application. An example of such applications is the transmission of voice. The quality of the voice playback is affected if the delay variation exceeds 100-150 msec.

⇒ *Applications oversensitive to delay*: These applications are affected by delays during data delivery. Thus it reduces the application's functionality and quality. Real-time applications are an example of applications that are affected severely by delays.

ii). Applications sensitive to packet corruption or loss

Applications can be classified, according to their sensitivity to packet loss, into the following two groups:

⇒ *Applications sensitive to loss*: Applications transmitting alphanumeric data such as text, source code, etc are highly sensitive to packet loss. For these applications a single loss in the delivery of packet can change the quality of the application.

⇒ *Applications tolerant to data loss*: These applications are tolerant to packet loss. Loss of a few packets does not significantly affect the quality of the application at the receiving end. This group of applications include most applications working with multimedia traffic (such as audio and video applications).

Not all multimedia traffic is tolerant to data loss. For example, compressed voice or video are very sensitive to loss. This research mainly deals with the effective transmission of delay sensitive real-time multimedia applications. Thus in the next section we look at some of the QoS challenges faced by these applications.

b) QoS Challenges in Real-time applications

High-speed networks are capable of carrying many types of applications such as voice, data, images, and video. The variety of traffic types in distributed multimedia environments introduces various problems on the network. The volume and inconsistency of multimedia data requires sophisticated protocols and techniques. Video or any other multimedia applications are sensitive to delay, delay variation, bit errors and packet loss. These applications have different requirements in terms of bandwidth, packet loss, delay, and delay variations.

i). Bandwidth

Bandwidth is one of the resources that need to be considered when transmitting varieties of applications over the Internet. Applications vary in terms of the bandwidth that they require. Multimedia applications require large amount of bandwidth. For example, a video application requires a minimum of 384 kbps bandwidth to support a call that is considered for instruction. Bandwidth limitation is the reason why video calls attempted over the internet are usually unsatisfactory.

ii). Packet Loss Rate

Packet loss occurs when congestion in the network increases. When traffic is excessive, buffers fill up and new incoming packets are dropped. In a network, during congestion, the average packet loss rate can be very high.

iii). Delay

Delay plays an important role in the transmission of applications. Some applications, such as audio conferencing, telephony, video conferencing, are very sensitive to delay. These applications need minimum delay in order to obtain their usable quality. An interactive real-time application program cannot tolerate delay. Internet telephone services, for example, cannot be used if there are long delays in the conversation. Some applications, such as email, are not sensitive to delay. These applications can wait for a few seconds or even hours for packet delivery.

iv). Delay variations

Delay variation or jitter is experienced while transmitting applications. If multimedia data is not delivered in a timely manner, it affects the quality of the application. Thus data delivered late is useless. In the case of video and audio, timely delivery means delivering data as they are produced, in the same order that they are produced, and without much delay.

1.2.3 Internet QoS Architecture

Efforts have been made to introduce *QoS* on the Internet. Researchers have proposed new architectures for the Internet. These QoS architectures are designed to make network components QoS aware. The Internet Engineering Task force has been the driving force behind finding a suitable QoS architecture. Listed here are some of the QoS models proposed by the IETF and their derivatives:

(1) *Integrated Service* (IntServ) [12], (2) *Differentiated Service* (DiffServ) [10], (3) *IntServ over DiffServ* [43], and (4) *Rainbow Services* [36].

A common feature in all the proposed QoS models is that, there is the need to allocate resources, either implicitly or explicitly. Appropriate *resource allocation* is one way of providing QoS in a network. Buffer management techniques involve scheduling of resources to applications, packet dropping and buffer allocation. Scheduling is used to move packets out of the buffers in a specified order and to manage delay within the buffers. Packet dropping is used to maintain the buffer size, in order to efficiently handle both real-time and non-real time packets. Buffer allocation is used to allocate buffer space for different types of flows.

Even though a lot of research work ([1], [7] , [21], [26], [34], [35], [48], [54]) has focused on buffer management strategies for non-real-time applications in communication networks, there are still no effective buffer management strategies that allow efficient management of buffers for high bandwidth applications such as real-time applications. The high bandwidth required by these applications easily causes congestion and hence results in packet loss, if an appropriate strategy is not applied. Thus, it is crucial to find a suitable buffer management strategy to overcome this problem.

1.3 BUFFER MANAGEMENT

An Internet architecture, that handles real-time multimedia applications has to include techniques to improve and provide QoS. These techniques include the management of buffers and the scheduling of packets. Buffer management, aimed at an efficient handling of congestion within the network, involves buffer allocation, and the dropping of packets. In this section we briefly discuss buffer allocation, packet dropping, and scheduling.

1.3.1 Buffer allocation

Buffer allocation constitute one aspect of buffer management that focuses on managing buffers by allocating buffer volume to class flows (streams of packets). These mechanisms are meant to improve delay as well as throughput of various applications. Only, limited research has so far been done on buffer allocation techniques. Most research on buffer management schemes concentrated on packet dropping techniques. Our research mainly focuses on buffer allocation of various applications over the internet. We further discuss buffer allocation in Section 2.3.1.

1.3.2 Packet dropping

Congestion in the network causes packets to be dropped, thus suitable packet dropping algorithms are required to manage this problem. Packet dropping schemes focus on getting an appropriate packet dropping technique to manage the dropping of packets from the buffers. In Section 2.3.2 we take a look at a number of packet dropping mechanisms.

1.3.3 Scheduling

Scheduling establishes the priority of the packet to be sent out of a buffer. Scheduling determines the bandwidth for class flows. This mechanism can reduce packet delay, packet loss and delay jitter for high priority traffic. In chapter 2 we look at scheduling in detail.

1.4 STATEMENT OF THE PROBLEM

The current Internet model is a best-effort service model which provides no service guarantees for transmitting real-time applications. The best-effort service is adequate for some applications that can tolerate large delay variation and packet losses, such as file transfer and email. On the other hand, many new applications are real-time multimedia applications and would require the underlying network to provide delay and packet loss guarantees. The future Internet would have to accommodate these real-time multimedia applications. This will obviously require service guarantees in order to avoid high packet loss rates and high delays. Since most of the delays and packet losses are experienced in the queues of network elements, one of the best ways to minimize these constraints is to manage the queues efficiently. This therefore necessitates the introduction of buffer management schemes that give preferential treatment to certain classes of packets. There is hence a need to create suitable buffer allocation mechanisms to avoid congestion in the network.

Previously proposed buffer management schemes are not capable of efficiently managing buffers for real-time multimedia applications, as they were developed with non-real-time applications in mind. The new high bandwidth applications are more likely to cause congestion, resulting in packet losses and high delays before packets reach their destinations. We propose a suitable buffer management mechanism for the future Internet aimed at reducing packet loss and delay to a minimum so as to preserve the agreed QoS promised by the network to its applications.

1.5 RATIONALE FOR THE RESEARCH

Real-time Multimedia applications are being introduced on the Internet at a very high rate. These applications require large amounts of buffers in order for packets to be transmitted effectively. There is a high probability of losing packets while transmitting them, causing wastage of valuable resources consumed at the transmission links already traversed. Currently, researchers are busy developing suitable buffer management strategies that could be deployed in the future Internet in order for this network to efficiently accommodate real-time applications.

The University of Zululand's Computer Science department has a research area that focuses on the development of distributed mobile e-services and on QoS based mechanisms for transmitting real-time multimedia applications over the Internet. We believe this contribution would add to the bulk of knowledge that would be needed to make the Internet accommodate real-time services in future.

1.6 RESEARCH GOAL AND OBJECTIVES

1.6.1 Goal

The aim of this research is to develop a buffer management strategy that will effectively allocate and manage buffers, so as to accomodate real-time multimedia applications over the future Internet.

1.6.2 Objectives

The specific objectives of the proposed research study are:

- To survey the buffer management strategies currently employed in the Internet.
- To develop a suitable buffer allocation strategy for the future Internet.
- To optimize the buffer allocation scheme by associating it with suitable packet dropping and scheduling schemes that can function in a multi-queue environment with service differentiation.

1.7 ORGANIZATION OF THE THESIS

In this chapter we have previewed the current Internet. We looked briefly at how buffer is managed on the Internet and highlighted its associated problems. We then introduced the research undertaken in this study. The rest of the dissertation is organized as follows:

Chapter 2 presents the background concepts for QoS in a network and also discusses buffer management schemes that exists.

Chapter 3 presents the related work for a number of existing buffer allocation, scheduling and packet discarding schemes used in networks. The chapter concludes with an overview of our proposed scheme.

Chapter 4 describes in detail the development and simulation of our proposed buffer management strategy DTBAS. The chapter begins by discussing DTBAS design goals and principles and concludes by providing a procedure for carrying out buffer allocation of real-time and non-real-time applications in the output buffers of the router.

Chapter 5 gives the details of the simulation implementation of the proposed buffer management model. It discusses the simulation environment and also introduces the performance metrics and simulation parameters. This chapter concludes with discussing the simulation results and their analysis.

Chapter 6 provides the conclusion of the dissertation. The chapter presents the work conducted and described in this dissertation. Some short-comings related to the research are also outlined. The chapter concludes by suggesting some views and directions for future work.

Chapter 2

BACKGROUND

2.1 INTRODUCTION

The introduction of real-time applications, such as video conferencing and Internet telephony, brought to light the limitations of the current Best Effort service model of the Internet. These limitations include: no guarantee of packet delivery, high packet loss rate due to congestion, and insufficient bandwidth. While the future Internet is expected to carry a lot more data (both real-time and non-real-time applications), it is also expected to provide its users with better QoS in transmitting these applications.

In this chapter, we explain the background concepts related to the Internet and buffer management issues. We start by briefly discussing the architecture of the Internet and a generic router. The second part of our discussion involves buffer management issues. Mechanisms that could be used to improve the transmission of various applications in the future Internet are also discussed.

2.2 THE INTERNET FRAMEWORK

The growth and development of new features in Internet applications increases the demand for QoS in applications among users. In this section, we first present the Internet architecture and briefly explain how it functions. We also discuss the structure of a generic router and look at the role a router plays in the transmission of packets within the network.

2.2.1 Internet Architecture

The architecture of the Internet comprises of *Access Networks (AN)* and *Core Networks (CN)*. *AN* consists of hosts and routers (or switches) while *CN* consists mainly of routers and switches. A router connects two or more networks and forwards packets from one network to another. Figure 2.1 gives an example architecture for node-to-node communication. This architecture consists of small groups of networks, where each network consists of hosts and routers. The routers connect these hosts together using links. The networks are then interconnected to each other using gateways or routers. During communication, when a packet is sent from Network A to Network C (for example), these packets travel through various links and routers to reach Network C. The routers on the path will direct the packet to move from one router to another. This is achieved by checking the packet's header information. The router will take the easiest path to send the packet to its appropriate destination.

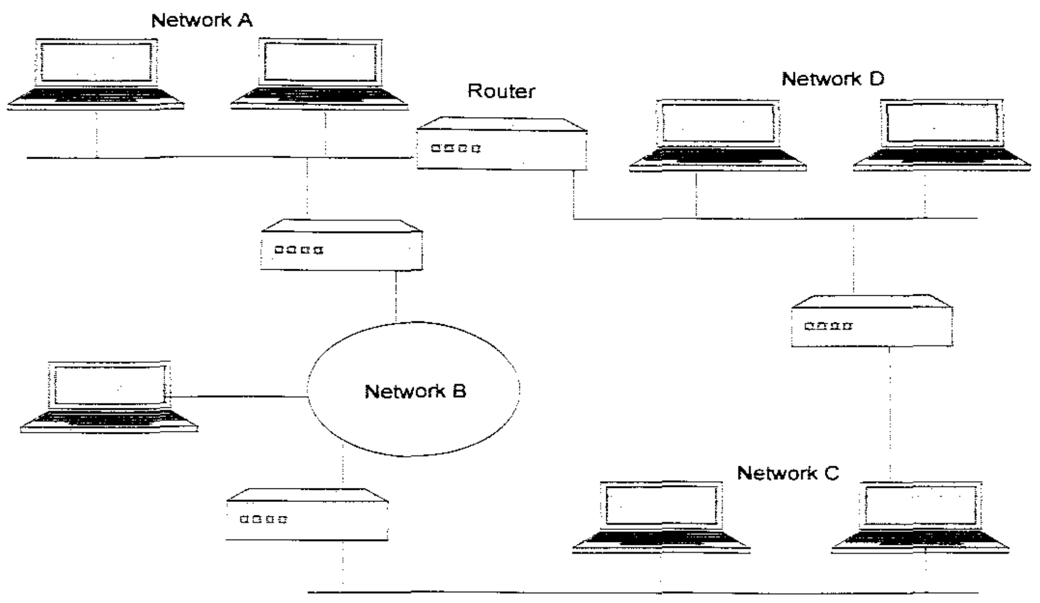


Figure 2.1: Node-to-node communications

In the section below we discuss the structure of a generic router and look at the role a router plays in the transmission of packets within the network.

2.2.2 A Generic Router

A generic Best Effort router, as shown in Figure 2.2, consists of three basic components: output buffers, input buffers, and routing processor [20].

Input buffers and *output buffers* perform the physical and data link functions of the router. The outgoing packets are queued in the output buffer before they are removed from the router. The *routing processor* processes the incoming packets by first checking the IP address and then using this address to allocate packets to appropriate output ports. In a Best Effort network, a router uses a single queue for each output port. Thus when different applications enter the single queue, congestion is caused and packets get dropped.

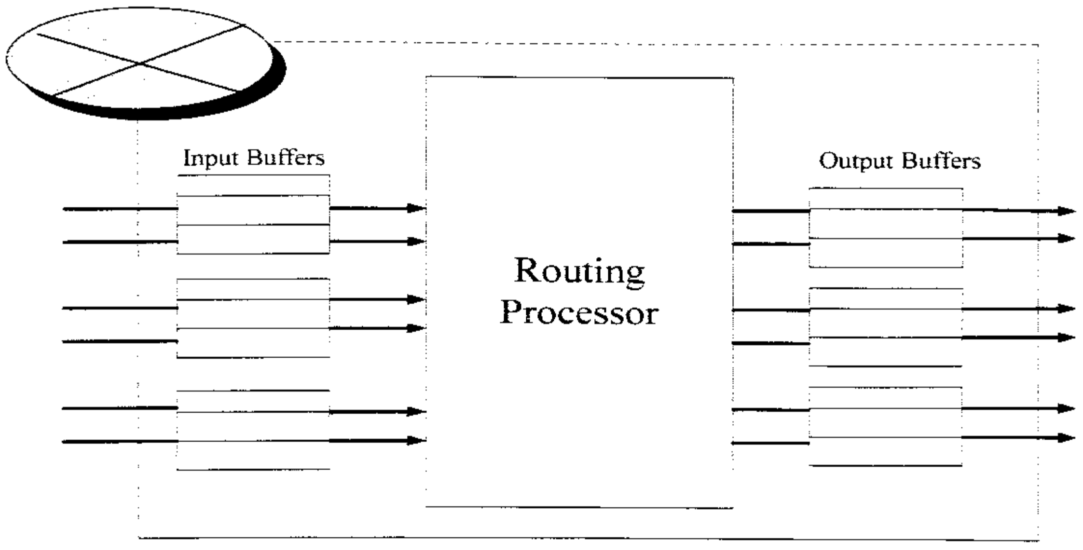


Figure 2.2: A generic Best Effort IP router

To overcome this problem, we use a router model, as shown in Figure 2.3, which has multiple queues at an output port [41]. This router model can handle various types of packets. Each class type has its own buffer area and may have its own packet dropping mechanism.

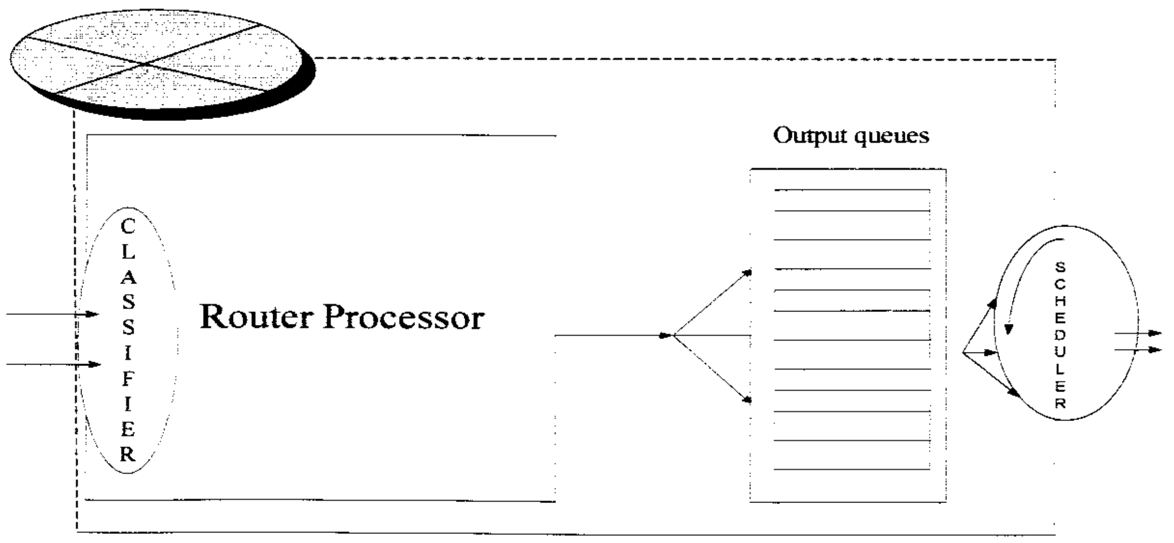


Figure 2.3: Function of the router

When the packets arrive at the router, these packets are classified using the *classifier* and sent to the input buffer. The *input buffer* receives these packets and checks for the information of each packet on the packet header. According to the information obtained from the packet header, the packets are queued in the appropriate buffer space. Each queue within the buffer performs its own process to maintain its flows. These packets are then sent out to the *routing processor*. In the routing processor, the packets are classified according to the header information, and sent out to the appropriate output buffers according to the destination address. In the *output buffer*, the outgoing packets are queued according to the header information. These packets are processed and maintained according to the buffer specifications. Then the packet is sent out of the buffer towards its destination, using a *scheduler*.

In this research study, we mainly deal with output buffers since that is the point where congestion most often occurs. Thus a suitable buffer management strategy is needed to tackle this problem. In the section below we discuss buffer management and some proposed buffer management strategies.

2.3 BUFFER MANAGEMENT

Buffer management is the process of managing the queues within the router to control and minimise delay and packet loss rate. Buffer management can be subdivided into three categories: *buffer allocation*, *packet dropping*, and *scheduling* mechanisms [35]. A buffer allocation mechanism is concerned with the allocation of bandwidth for various application flows to appropriate buffer space, so that an application does not struggle for buffer space. A packet dropping mechanism is an algorithm used within the router to manage the dropping of packets during congestion. A scheduling mechanism is used to manage and reduce delay by determining the type of packet that should be removed from the buffer; how fast, and when the packet should be removed. To make efficient use of network resources, it is necessary for different types of applications to share the network resources, such as transmission capacity and buffer space. We now look at each of the three buffer management strategies in turn.

2.3.1 Buffer Allocation Strategies

A buffer allocation strategy is part of a buffer management strategy to manage buffer resource among different flows, according to certain policies and at the same time controlling congestion within the buffer. Buffer allocation schemes can be categorized into three policy-based schemes namely: complete sharing schemes, complete partitioning schemes and partial sharing (or partitioning) schemes. Each of these schemes is explained in detail below.

*(a) **Complete partitioning (CP)***: This is a simple buffer resource allocation policy to statically partition the whole buffer space into different classes [13]. An example is as shown in Figure 2.4. In this example, the buffer resource is shared among three classes, namely: Class A, Class B and Class C. A class is a category of packets according to the application type in the Internet architecture. The packets entering the buffer are classified according to the class type and sent to their allocated buffer space. Each of the classes is assigned a buffer volume; a certain amount of buffer space which each flow can use. This scheme uses multiple queues; each assigned to a single class. Each queue is managed separately from the others.

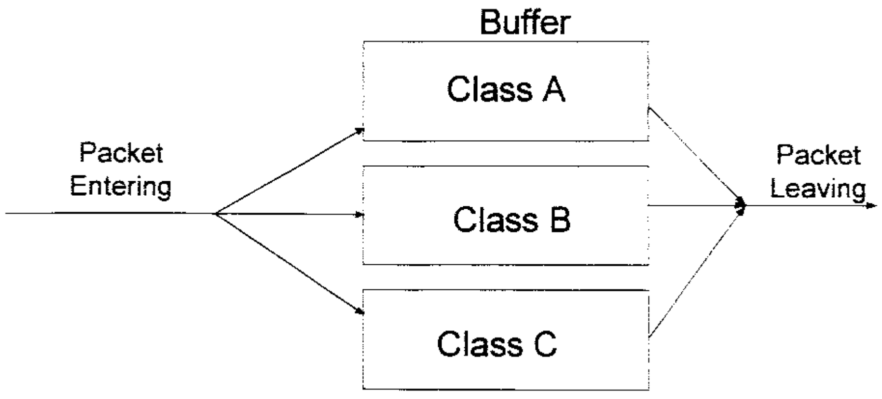


Figure 2.4: Complete partitioning

CP may reduce buffer utilization while increasing the overall packet loss rate. This is because arriving packets may be discarded even when there are buffer resources available in the other queues.

(b) Complete sharing (CS): In the complete sharing scheme [13], a packet is allocated to a class. Packets entering the buffer are stored in any available space (See Figure 2.5). The problem occurs when there are lots of non-real time packets occupying a greater share of the space. In this case, real-time applications may be delayed or dropped. This policy can achieve high buffer utilization because the buffer is always occupied, unless during period of low traffic volume.

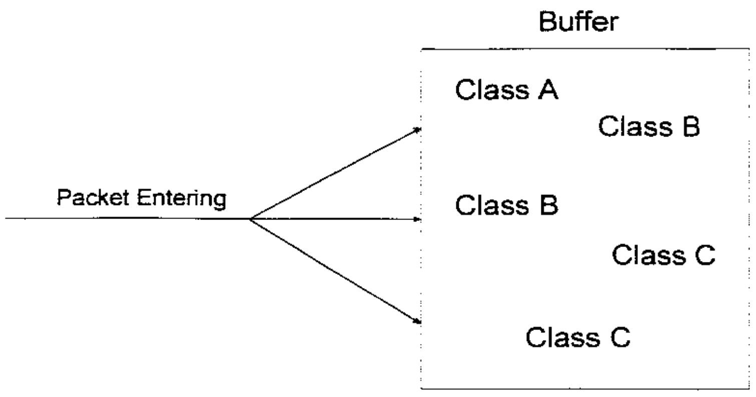


Figure 2.5: Complete Sharing.

(c) Partial Sharing: In this scheme [13], [15], packet loss is controlled based on a threshold on the buffer size. Buffer volume is allocated to a class of flows. As shown in Figure 2.6, when buffer load is low for Class A and some buffer space is available, Class B or Class C can use the remaining buffer capacity, regardless of the allocated basic buffer volume. This scheme is similar to complete sharing with virtual partitioning.

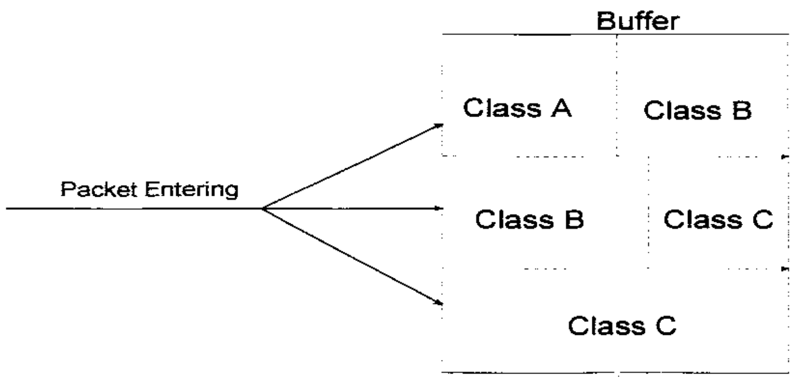


Figure 2.6: Partial Sharing

When a packet arrives and finds a full buffer, if the basic volume of that packet class has not been fully occupied by packets of that class, the arriving packet will push out packets from another class that are occupying its basic volume. The disadvantage of this scheme is that the computing complexity increases with large number of service classes.

We have looked at the generic classification of buffer allocation schemes. In Chapter 3, we will look at additional buffer allocation schemes, in detail. In the section below, we now look at packet dropping policies, and investigate how each policy plays a role in reducing congestion during transmission.

2.3.2 Packet Dropping

Buffers in network devices are managed using various queuing techniques. Properly managed queues can minimize the number of dropped packets and prevent network congestion, as well as improve network performance. Simple Drop, Selective Drop, and Active Queue Management (AQM) are a few of the dropping mechanisms that are discussed below:

***a) Simple Drop*:**

The most basic dropping technique is the simple drop. The two dropping techniques that fall under Simple drop are *Front dropping* and *Tail dropping* techniques [20]. As the packets arrive at the queue, the router first checks whether the queue is full or has available space to accommodate more packets. This is determined by detecting whether the buffer is full or not. Figure 2.7 below, depicts the Front dropping scheme. In this scheme, if it detects a full buffer, the arriving packet will be dropped or else the packet will be queued. In front dropping scheme the packet at the front of the queue is the one that is dropped and the arriving packets are added at the back of the queue.

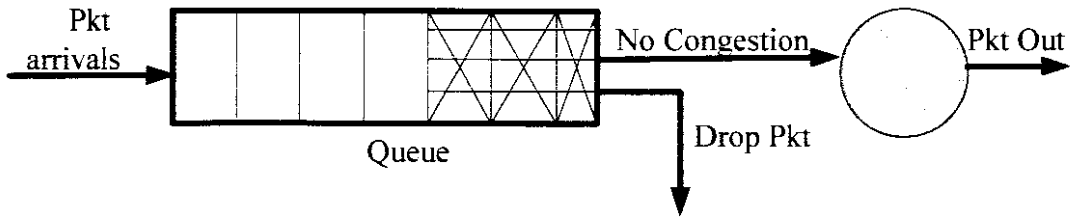


Figure 2.7: Front Dropping Scheme (DfF)

Figure 2.8 shows the *Tail dropping scheme*. In this scheme, if buffer is full, the arriving packets are dropped. Both front dropping and tail dropping schemes drop packets whenever buffer space is unavailable.

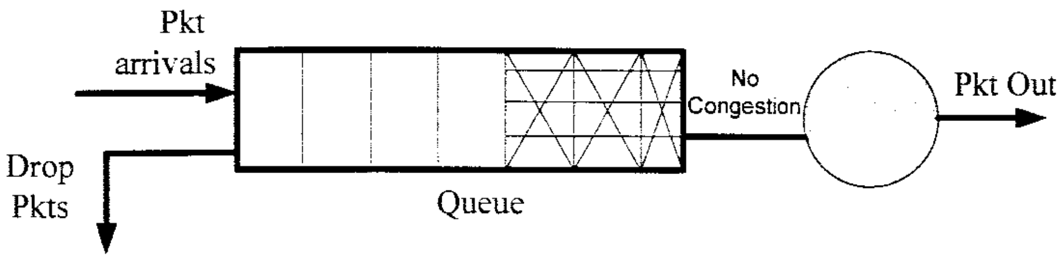


Figure 2.8: Tail Dropping Scheme

In the simple drop scheme, packets of low and high priorities are treated equally. This is a limitation when delay sensitive real-time applications are mixed with non-real-time applications. Selective drop schemes are introduced to overcome the limitations of the Simple drop strategies.

b) Selective Drop

In the selective drop scheme [20], low priority packets are discarded in a FIFO (or LIFO) manner to provide a higher degree of QoS for high priority packets. Figure 2.9 shows an example of a selective dropping mechanism using a FIFO approach to handling packets in the buffer. In the figure, we see two types of packets entering the buffer; low priority packets and high priority packets. Here, the high and the low priority packets share the same queue. When the queue is full and the arriving packet is a high priority packet, the entire queue is searched for a low priority packet to be dropped.

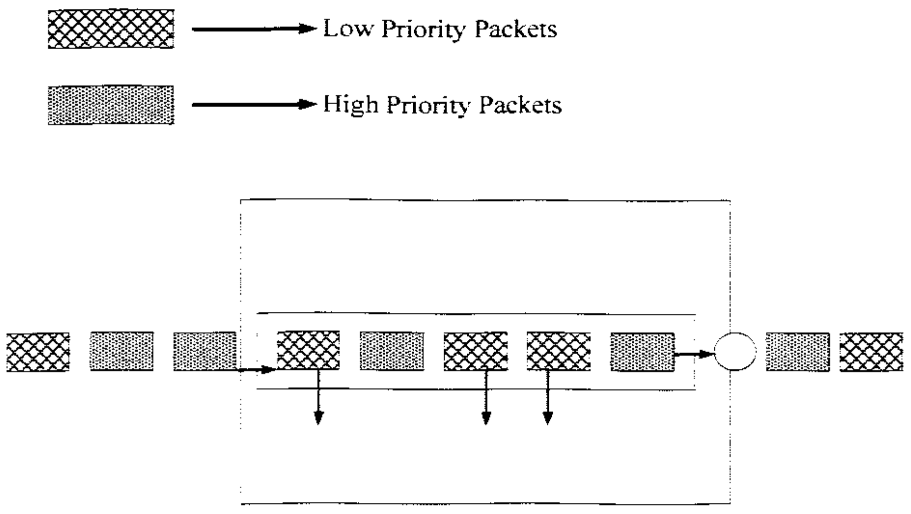


Figure 2.9: Selective Dropping algorithm

The problem faced by this scheme is the high cost of these algorithms, due to the frequent scanning of the buffer to determine the cell to be discarded.

Dropping packets is inefficient. If a host is sending packets in burst and congestion occurs, a lot of packets will be lost. It is therefore preferable to detect impending congestion conditions and actively manage congestion before it gets out of hand. Thus the network can take an active role in managing congestion. This is the role of Active Queue Management.

c) Active Queue Management (AQM) packet dropping mechanisms

Active queue management (AQM) is a technique in which routers actively drop packets from queues as a signal to senders that they should slow down. AQM uses thresholds and there could be one or two thresholds (shown in Figure 2.10). The Random Early Detection (RED) [18] algorithm is an example of an active queue management technique. RED has been proposed for IP networks with the objective of avoiding congestion and maintaining average queue size at low levels. RED does not keep any state information about a connection.

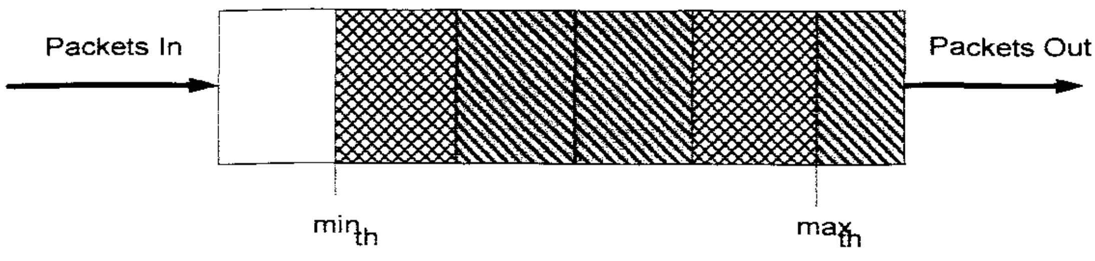


Figure 2.10: Buffer using an AQM scheme with two thresholds

This scheme retains the buffer size by monitoring and managing maximum and minimum thresholds. Loss of packet is determined by keeping track of the average buffer size and adjusting the threshold. If the packets fill up more than the maximum threshold, (a drop probability of 1) then all arriving packets will be dropped. This scheme was recommended by IETF to reduce delay and packet loss in the BE Internet during congestion.

Dropping some packets alleviates congestion to an extent. This alone cannot suffice. The type of scheduling mechanism used is also important. In the next section we look at scheduling.

2.3.3 Scheduling

Scheduling is a mechanism used to decide which packets to send first from queues and to treat those packets in a fair and appropriate manner. There are several scheduling techniques. Below we describe a few of them.

a) FIFO

First-in-first-out (FIFO) [45] is a common queuing technique and it is the one which is currently used in the BE Internet. Packets wait in a queue until the router is ready to process them. If the average arrival rate is higher than the average scheduling rate, the queue will fill up and new arriving packets will be discarded.

b) Round Robin

Round Robin (RR) scheduling [40] is one of the oldest, simplest and fairest scheduling algorithms designed especially for time-sharing systems. This scheme eliminates starvation of bandwidth among flows. In Best Effort packet switching networks, Round Robin scheduling can be used as an alternative to FIFO queuing. A router using a Round Robin scheduling uses a separate queue for every class type. This algorithm lets every queue to take turns in transferring packets on a shared channel in a periodically repeated order. If a queue is out of packet, then the next data queue will take its place.

c) *Priority Queue*

In priority queuing, packets are first assigned a priority class. Each priority class has its own queue. Packets in the highest-priority queue are processed first. Packets in the lowest-priority queue are processed last. Within each queue in a Priority Queuing (PQ) [40], packets are processed in a FIFO manner. Priority queuing can provide better QoS than FIFO queuing because higher priority traffic, such as multimedia, can reach the destination with less delay. The drawback of this scheme is that if there is continuous flow in a high-priority queue, the packets in the lower-priority queues will never have the chance to be processed.

d) *Weighted Round Robin*

In Weighted Round Robin (WRR) [53], packets are first classified into various classes (for example, real-time, or non-real-time applications) and then assigned to a queue that is specifically dedicated to that class. Each of the queues is serviced in a round-robin fashion. WRR is also called Class based queuing (CBQ). This allows the allocation of various amounts of bandwidth to different classes either by allowing a high bandwidth queue to send more than one packet out, or allowing each queue to send one packet out and to allow high bandwidth queues to send multiple times in a single round trip. WRR queuing ensures that all classes have access to at least some amount of bandwidth to avoid bandwidth starvation.

e) Fair Queuing

This scheme was introduced to ensure that each flow has a fair access to network resources and to prevent a bursty flow from consuming more than its fair share of bandwidth. In Fair Queuing (FQ) [40], packets are first classified into flows by the system and then assigned to a queue that is specifically dedicated to a flow. The packets are handled in a round robin fashion and empty queues are skipped. FQ is known as flow-based queuing. FQ fails when handling large number of queues which increases complexity and impacts the scalability of FQ in large IP networks.

f) Weighted Fair Queuing

When compared to the above schemes, a better scheduling method is weighted fair queuing. In this technique, the packets are also assigned to different classes as in PQ. Weighted Fair Queuing (WFQ) [40] supports the fair distribution of bandwidth for various packet types. The queues are weighted based on the priority of the queues; higher priority means a higher weight. WFQ uses the same technique as WRR; each queue is given a certain weight depending on the class characteristic. Each packet may be weighted according to the priority of the queue. The scheduler uses this weighted information to remove the packet out of the queue.

In this chapter we have looked at the architecture of the Internet. We focused mainly on the router, as this study is based on buffers (in routers) and the applications of various buffer management techniques to improve QoS. In the next chapter we provide related works that have been proposed.

Chapter 3

LITERATURE REVIEW

3.1 INTRODUCTION

Buffer Management plays an important role in congestion control and is a vital part of QoS provisioning mechanisms for the transmission of real-time applications. It is a combination of buffer allocation, packet dropping and scheduling mechanisms. As already indicated, the aim of this research is to develop a buffer management strategy that will effectively allocate and manage buffers. In this chapter we, therefore, review some of the existing buffer management schemes. Our discussion focuses mainly on buffer allocation strategies in the output buffers of routers on the Internet. We also take a look at packet dropping and scheduling as associated mechanisms. We conclude the chapter by providing desirable features for our proposed scheme.

3.2 Buffer Allocation Strategies

Buffer allocation is aimed at providing an efficient and fair use of available buffer spaces. Many buffer allocation mechanisms have been proposed for ATM and IP networks. In the next subsections, we discuss the previously proposed buffer allocation schemes. We categorize these buffer allocation schemes under the following headings: 1) hierarchical vs. linear 2) Class based vs. Flow based 3) Sharing vs. Non-sharing.

3.2.1 Hierarchical vs. Linear

Buffer volume can be allocated using a linear or a hierarchical structure. In a linear structure, the flows passing through each allocated port are assigned a certain amount of volume. In a hierarchical structure, buffer volume is assigned to a class which belongs to a particular flow that passes through a port to reach the buffer. So the buffer volume is allocated to classes, a certain amount of volume is allocated to flows and also to each ports. The classes can be allocated a certain volume, either equally or according to the priority of the classes. This is illustrated in Figure 3.1.

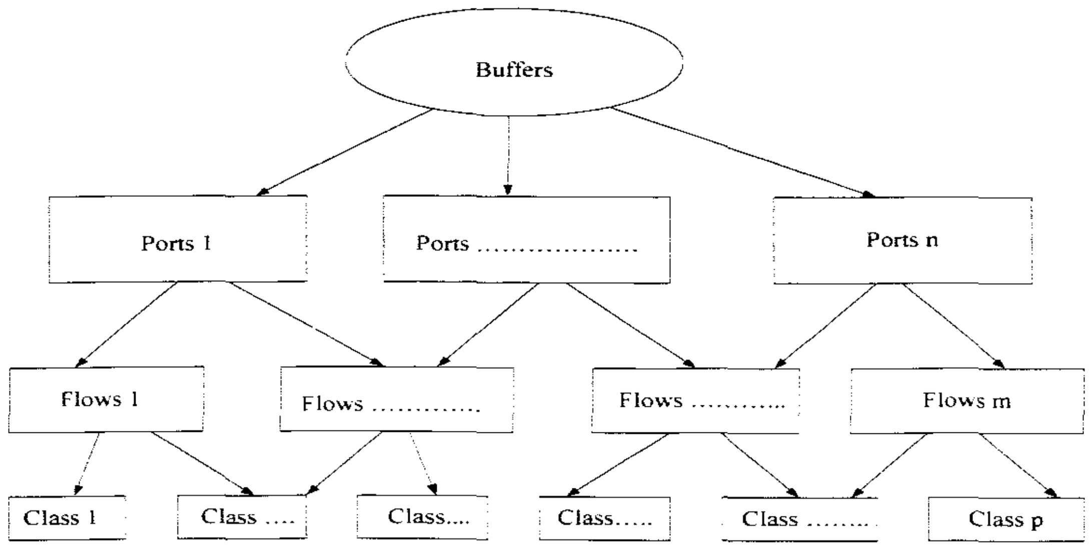


Figure 3.1: Hierarchical sharing of buffer volume

The scheme proposed by Yaprack *et al.* [50], is one example of this type of scheme. This scheme uses a hierarchical structure of buffer allocation which consists of switches, and packets that pass through these switches. Both switches and various packets are dynamically allocated a certain buffer volume.

The class-based QoS control scheme by Minami *et al.* [34], uses a single buffer. This scheme uses a hierarchical structure in the order of *ports, classes* and *flows*. In this scheme the total buffer size is shared by output ports according to the ratio of each output link speed. A basic buffer volume VOL_{pi} is allocated to Port *i*. For each class *j* within the port, a basic volume VOL_{cj} is allocated.

In the scheme proposed by Aweya *et al.* [7], buffer volume is shared hierarchically among various classes. Each port is assigned a fare share of buffer volume which is distributed among various flows in the port. The flows passing through these ports are classified into different classes and each of these classes is assigned a level of buffer volume according to their priority. In this scheme, buffer volume is logically divided into multiple queues within the buffer.

Other similar schemes that hierarchically allocate buffer volume are those proposed by Lai and Mu-Rung-Shiu, [29] and Chuang, [13]. In both of these schemes buffer volume is initially allocated to the ports. The buffer volume is then allocated to various classes according to the priority in the class hierarchy.

The scheme proposed by Garcia-Marcias *et al.* [21], shares buffer in a linear manner. In this scheme class packets enter into the output buffer and each of these class packets are allocated buffer volume. Here the buffer space is divided into three main classes and then further divided into subclasses. The buffer volume is divided among the classes in a hierarchical fashion, according to the priority of classes in the hierarchy.

Zhou and Sethu [48], proposed a scheme, where the buffer volume is linearly shared among switches. The buffer volume is then equally allocated among the flows that pass through these switches. The switches use a single shared buffer with multiple output links.

The schemes proposed in [41] and also in [44], both use a linear allocation approach. These schemes have been proposed for ATM wireless networks and so use cells. These cells are grouped into various classes. Here the buffers have a finite size and each of the classes is accommodated according to the rate of allocation.

3.2.2 Class based vs. Flow based

The allocation of buffer volume can either be *class based* or *flow based.* In class based buffer allocation, the buffer volume is allocated to different classes and the ports that they pass through. In flow based buffer allocation, buffer volume is allocated to flows; here flows are not broken down into different classes. So the buffer volume is allocated to flows and the ports they use.

Buffers can be partitioned according to classes depending on the type of architecture used. One such example is [34] that uses flows which are further classified into different class types. The scheme proposed by Kamal and Hassanein [25], is another example of a class-based scheme. It uses a class-based architecture based on the classes in the DiffServ architecture.

The scheme proposed by Garcia-Macias [21], can be considered as a class based mechanism as it is based on the DiffServ architecture. In this scheme the buffers are completely partitioned and each of the main classes allocated its own buffer volume. For example, Expedited Forwarding (EF) classes are considered to be delivered with high assurance and are allocated a higher volume when compared to Assured Forwarding (AF) and Best Effort classes.

The Multi-level RED scheme proposed by Aweya [7], is another example of class based allocation. This scheme divides the buffer among K classes. Each class is completely partitioned in the sense that each has its own buffer volume. Also there is no sharing of buffer volumes among the classes. In this scheme the buffer volume is highly underutilized and packet loss rate is considered high. This limits the efficient transmission of real-time multimedia applications.

In the scheme, proposed by Sidahmed and Ali [41], the cells are classified according to different classes. It uses a class based buffer management scheme to improve the transmission of multimedia applications over the cellular IP network. The per-class buffers are given a finite size, which is selected by the network operator. The maximum amount of traffic that can be held by the per-class buffers can be measured according to the rate of cell arrival and scheduling. Each per-class buffer handles and manages its cells separately in a complete partitioning manner. Thus the priority is given to multimedia real-time applications so as to improve the performance of these applications in the wireless internet.

The scheme proposed by Zhou and Sethu [48], uses a flow based allocation scheme where the buffer volume is equally allocated among the flows that pass through the switches.

The proposed scheme by Aweya *et al.* [6], uses a flow based allocation method. In this scheme a minimum amount of buffer volume is allocated to flows depending on the capacity of the buffer. In the next section we look at sharing and non-sharing allocation schemes.

3.2.3 Sharing vs. Non-sharing vs. Hybrid

Buffers can be shared in three basic ways *Sharing, Non-sharing* and *Hybrid.* In sharing strategy, buffers are shared among classes. Sharing was discussed in Chapter 2 under Complete Sharing (CS) scheme. In non-sharing strategy, buffer volume is partitioned among classes; they do not share their buffer volume when in need. Another name for it is Complete Partitioning (CP) scheme which we discussed in Chapter 2. Hybrid allocation scheme combines complete partitioning and also sharing. This forms the Partial Sharing (PS) scheme which we discussed in Chapter 2. Some examples of previously proposed schemes that fall under hybrid strategies are schemes by Wu and Mark [44], and also by Chuang and Li [13].

Complete Sharing using Virtual Partition (CSVP) scheme by Wu and Mark [44], is another example of a hybrid strategy (as shown in Figure 3.2). This scheme was introduced to manage multi-traffic flows in ATM networks. In the CSVP scheme, if there are u numbers of users and the available buffer space is B, the B buffer spaces are virtually partitioned into s segments corresponding to the u number of users. The whole buffer is used by n types of cells. The u number of users access n types of cells. When the buffer is not full, cells of any type are accepted upon arrival. When the buffer is full for a particular class (n_i). then this class can share its buffer space with another available class buffer (n_j), until its cells arrive. The scheme admits a newly arriving cell by pushing out a different type of cell that is borrowing that buffer volume. When a new cell arrives, it will be rejected.

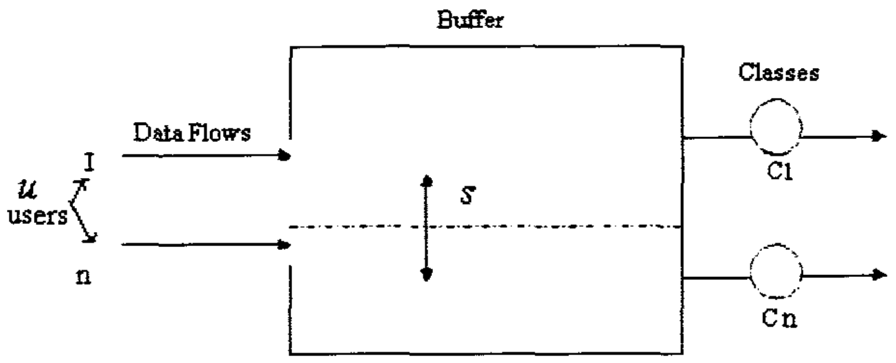


Figure 3.2: System model with two data Streams (Wu et al, 1995)

The Class-based QoS control scheme proposed by Minami et *al.* [34], is an example of CSVP allocation scheme. In this scheme, flows of various classes share buffer volume for that particular port. The one limitation is that the algorithm is a bit complex even though there is high utilization of the buffer space.

The scheme proposed by Yaprack *et al.* [50], also uses hybrid buffer architecture with a threshold, based on virtual partitioning of buffers among the output ports. These thresholds are updated based on the characteristics of the traffic (that is, real-time or non-real-time applications) on the outgoing link, to adapt the traffic load. When the buffer is not full, any incoming cell is accepted. Otherwise a selected cell is accepted or dropped according to the priority of the cell.

The Dynamic Partial Buffer Sharing (DPBS) scheme proposed by Chuang and Li [13], was introduced to solve the limitations faced by static sharing of buffer space; as Partial Sharing scheme mainly focuses on sharing the buffer partially by controlling the threshold. The thresholds in this scheme can be dynamically adjusted to reach an optimal value and meet the needs of the network traffic. These thresholds are dynamically adjusted at run-time, based on packet loss behaviour.

The scheme proposed by Kamal and Hassanein [25], uses a sharing mechanism to increase the throughput of High priority traffic. In this scheme both high priority (HPC) and low priority classes (LPC) have reserved buffer spaces. Also there is a shared buffer, which can be occupied by both LPC and HPC traffic. The LPC traffic is given a lower priority in using the shared buffer. This traffic uses its allocated buffer volume first, if the buffer volume is completely occupied, only then is the shared buffer used. But if there is available space in its buffer space and the flows were using the shared buffer, then they will be forced to move out of the shared buffer. The HPC traffic always uses the shared buffer before using its allocated buffer volume. If the HPC sees LPC traffic in the shared buffer then it will push out the LPC traffic to occupy the shared buffer. The LPC traffic is guaranteed a minimum space in the shared buffer and if it is occupying beyond that, the HPC traffic will push out the LPC packet.

The scheme proposed by Aweya *et al.* [7], uses dynamic sharing of buffers among TCP flows. In this scheme the buffers are logically organised into multiple queues and a dynamic buffer threshold is used to ensure efficient and fair usage of buffer memory among the queues. In the dynamic threshold scheme, a common buffer threshold is dynamically adjusted according to the changing traffic load. When there is congestion in the queues, buffers with fewer loads can access flows as long as these buffers use their fair share of the volume allocated to them. This helps in improving the overall buffer utilization, while ensuring fair share of buffer memory among all the queues.

Most schemes that have been discussed mainly focused on throughput. A few dwelled on packet loss as an aspect of QoS. Delay which is a major aspect of QoS for the transmission of multimedia real-time applications has largely been neglected. Most multimedia applications are very sensitive to delay and packet loss rate. When these two aspects of QoS fail, the quality of the application is affected and hence the throughput of these applications is also affected.

Our research focus is mainly on the use of buffer allocation to minimize both delay and loss rate. Buffer allocation alone cannot help the efficiency of transmission. It is a proper combination of buffer allocation, packet dropping and scheduling mechanisms that would produce an optimal strategy. This study uses basic packet dropping and scheduling mechanisms so that the effect of the buffer allocation strategy will be largely felt. In the next section some of the dropping techniques that have been proposed are briefly discussed.

3.3 Packet Dropping (PD) Strategies

This section discusses congestion control strategies based on selective dropping of packets as well as Active Queue Management (AQM) techniques that have been previously proposed to manage congestion. In this section, packet dropping schemes are classified according to whether they are Single-threshold or Multi-threshold dropping schemes.

3.3.1 Single-threshold dropping schemes

Most dropping policies use a single-threshold as a sign of congestion. This usually determines the point to start dropping packets. Simple drop schemes such as tail drop, front drop and selective dropping, all use a single threshold to detect congestion and immediately start dropping packets.

An example of an AQM scheme that uses single threshold dropping is BLUE proposed by Feng [17]. The BLUE mechanism is a variation of RED AQM scheme. The BLUE mechanism controls congestion by packet loss and link utilization history, instead of queue occupancy. A single probability, *Pm* is maintained, to mark (or drop) packets. If the queue is continually dropping packets due to buffer overflow, *Pm* is increased. This increases the rate at which congestion notification is sent back or packets are dropped. Conversely, if the queue becomes empty or if the link is idle, *Pm* is decreased. This effectively allows the rate at which congestion notifications, are sent back, to be correctly maintained. The disadvantage of the BLUE mechanism is that, the drop probability doesn't react fast enough, thus leading to buffer overflows and underflows.

The Dynamic Random Early Detection (DRED) algorithm proposed by Aweya [6], is another variation of RED and uses a simple feedback control approach to randomly discard packets in the queue. The objective of the DRED algorithm (shown in Figure 3.3) is to stabilize the actual queue size $q(n)$ [where n is the number of packets] at a predetermined target queue size T, independent of the network traffic load. The actual queue size is sampled every Δt units of time and used to produce an error signal $e(n) = q(n) - T$. This error signal is then used to adapt the drop probability Pd, so that $e(n)$ can be kept as small as possible.

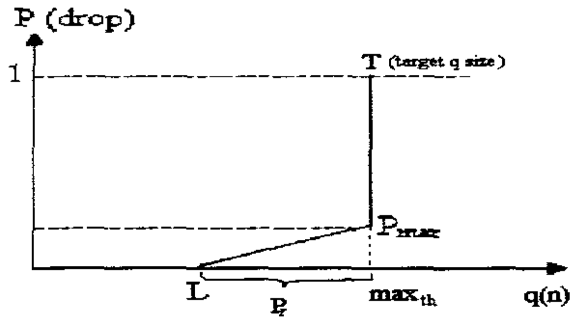


Figure 3.3: Dynamic RED (DRED) (Aweya .J et al,2001)

The multilevel AQM with dynamic thresholds scheme proposed by Aweya [7], uses a DRED threshold method. This scheme randomly discards packets with a load-dependent probability when a buffer in a network gets congested.

A technique proposed in [6], enhances the effectiveness of the RED scheme by dynamically changing the threshold settings of a controller as the number of connections increases or decreases. The drop controller adjusts according to the number of connections, by monitoring the current value of the packet loss rate. When the number of connections decreases, the threshold is brought back down to prevent excessive build-up in the queue. The threshold is adjusted to keep the loss rate at a minimum value. This scheme is implemented when packet loss rate is low, so that TCP can maintain its sending rate. This scheme only helps with isolated losses and not with bursty losses.

3.3.2 Multi-threshold dropping schemes

The Multi-threshold scheme uses different thresholds to mark the dropping probability of packets. This is sometimes used when differentiating traffic types, so that each type will have its own dropping probability to maintain the packet loss rate.

In the Weighted RED (WRED) scheme [14], packets are differentiated as marked or unmarked. This scheme is shown in Figure 3.4. Marked packets (low priority) can get through the queue when the network is relatively congestion free. This scheme helps in the efficient transmission of high priority packets as they are sensitive to delay during congestion. The main aim of this scheme is to remove out-of-profile (marked) packets before in-profile (unmarked) packets, when congestion is tight. Unmarked (high priority) packets use RED, with $min1_{th}$ as its lower threshold, $max1_{th}$ as its upper threshold, and max_p as the peak drop probability before the function jumps to 1.

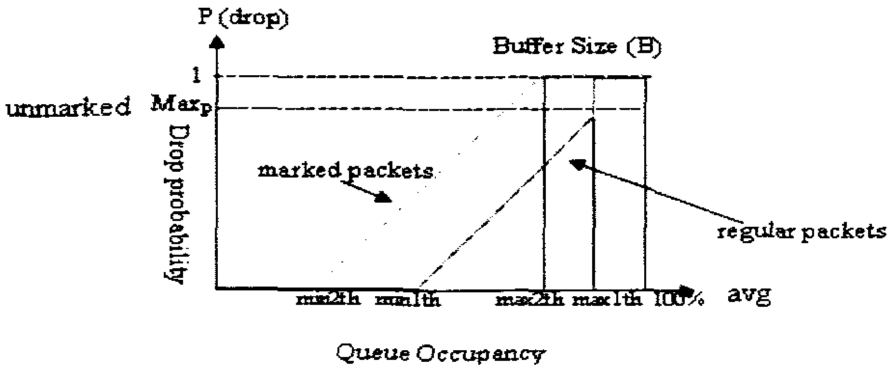


Figure 3.4: Weighted RED (WRED)(Cisco Specifications,2001)

Marked packets use a more aggressive curve in which random drops begin at a lower average occupancy level of $min2_{th}$, increasing rapidly to 1 at $max2_{th}$. This results in a higher drop-out rate than the unmarked packets. The marking of packets results in better transmission of unmarked packets.

The scheme proposed by Li and Liu [51], uses a novel adaptive queue management scheme with Resilient Packet Ring (RPR) feedback information in IEEE 802.17 Resilient Packet Ring

(RPR) optical networks. This scheme uses an AQM scheme in RPR networks by modifying the RED algorithm slightly. The max_{th} threshold and min_{th} threshold is set as fixed constant value. The RED parameters are dynamically adjusted, based on the state of the network. One shortcoming of this scheme is that the threshold range causes similar queuing delays among all flow types in RPR networks.

The adaptive RED algorithm proposed by Floyd [19], is a variation of RED algorithm. In this scheme, maximum drop probability (P_{max}) is adapted to keep the average queue size between min_{th} and max_{th} threshold and also a target range, which is half-way between min_{th} and max_{th}. The demerit of this algorithm is that during bursty traffic conditions, this algorithm fails to monitor the target queue size and thus the delay of traffic in the queues increases.

In RED with an IN/OUT bit (RIO), Clark and Fang [15], proposed a relative of RED. This also uses packet marking to modify the algorithm on a packet-by-packet basis. RIO assumes that packets have already passed through an upstream marker, and a single bit in the packet header indicates whether the marker found the packet to be IN or OUT profile. RIO also discards OUT-profile packets during times of congestion. RIO differs from WRED in the aspect of queue average as RIO uses two separate queue averages, avg_{IN} for IN-packets and $\mathrm{avg}_{\mathrm{OUT}}$ for OUT-packets. Similar to WRED, RIO also has three sets of thresholds; min_{th}, maxth and $\mathrm{max}_{\mathrm{p}}$, where min_{th} and max_{th} are lower for OUT-packets than IN-packets, whereas, max_p is higher for OUT-packets than for IN-packets. When calculating a drop probability for IN-packets, the queue occupancy is taken from avg_{IN}, whereas for OUT-packets the average queue occupancy is taken from $\mathrm{avg}_{\mathrm{OUT}}$. But avg_{OUT} is based on the average total queue occupancy of both IN-packets and OUT-packets. The number of OUT-packets passing through the queue does not affect the drop probability of IN-packets.

3.4 Scheduling Strategies

Scheduling is an important part of buffer management which also plays an important role in providing QoS to application types. The delay aspect of QoS of various application types can also be controlled and maintained by an appropriate scheduling mechanism. Many scheduling schemes proposed as discussed in [40], [52] and [53]. In this section, the Weighted Round-Robin [40], [53], is discussed because we have used it in this study.

The Weighted Round-Robin (WRR) scheme proposed by Semeria [40], [53], is designed to efficiently handle flows with different priorities. Each flow class is assigned a weight; an integer value that indicates the processing capacity. A Class with higher weight is processed before a class with lower weight. For example, real time applications will be assigned higher weight than non-real time applications. In weighted round-robin scheduling, a scheduling series will be generated according to the class weights. If all the classes have equal weights, WRR becomes RR scheme. In the example shown in Figure 2.11, if the weight of the queue is 3 then, three packets will be removed out of that queue. The same applies to the rest of the queues. If there is no weight imposed on the priority of a class, all will be treated equally.

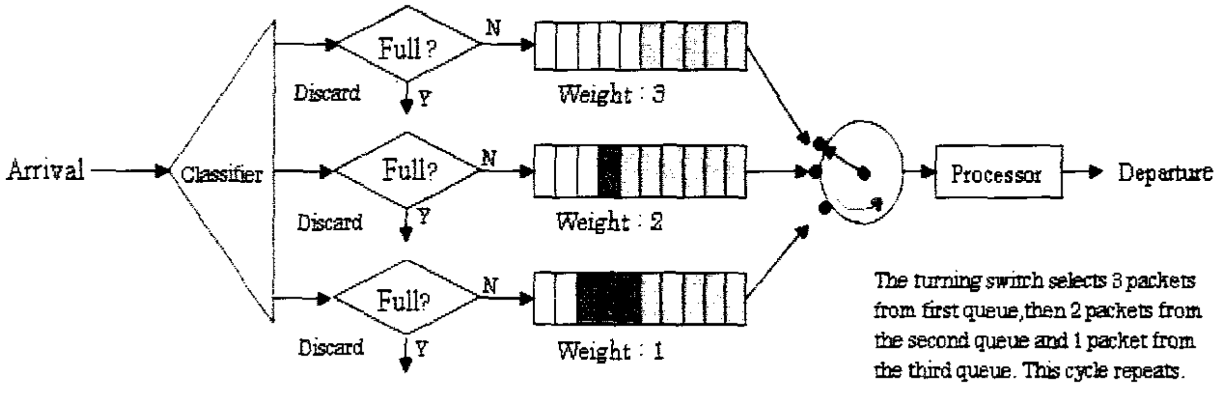

Arrival
Classifier
Full ?
Full?
Full?
Discard
Discard
Discard
N
Y
Weight : 3
Weight : 2
Weight : 1
Processor
Departure
The turning switch selects 3 packets from first queue,then 2 packets from the second queue and 1 packet from the third queue. This cycle repeats.

Figure 2.11: Weighted Round Robin

In the scheme proposed in this study, Weighted Round Robin (WRR) is used, as it is a simple but efficient scheme for scheduling real-time applications with service differentiation.

3.5 Summary of limitations of previously proposed schemes

The previously proposed buffer management schemes focused mainly on packet dropping and scheduling strategies than the buffer allocation part of the management. Even the buffer allocation strategies that have been proposed, favour the transmission of non-real-time flows than the real-time applications. The previously proposed buffer allocation schemes presented in the above section show a static allocation of buffer volume. Due to static allocation, if there is congestion in the network then large amount of packets will be dropped. These schemes that have been discussed above have focussed more on throughput of applications than the packet loss rate, delay and buffer utilisation factors of QoS. These factors affect the quality of real-time multimedia applications.

The limitations faced by packet dropping algorithms are that they do not consider delay and loss rate of real-time applications. Due to the congestion in the network, real-time packets get dropped and the quality of the packets becomes poor. A suitable packet dropping scheme also plays an important part in the management of real-time traffic. Thus overall a suitable buffer management strategy needs to be developed which looks into improving the QoS factors of real-time applications.

In the next section we give a brief description of the characterisitics of the model proposed in this study.

3.6 Proposed Scheme

The proposed buffer management scheme consists of the following characteristics: It is a hierarchical class based architecture, which is structured into a hierarchy of ports and classes. The scheme uses complete sharing with virtual partitioning plus pre-emption to allocate packets in the buffer. The total buffer volume is virtually partitioned into two and each partitioned class buffer uses pre-emption to discard packets. A threshold-based packet discarding mechanism is used. Each class can access the other class' buffer volume. Class 1 flows have an IN-profile and OUT-profile packets, so during sharing it is mostly the OUT-profile packets that are dropped, so as to minimise the loss rate. During sharing, the high-priority packets can pre-empt the low priority packets.

Chapter 4

MODEL DEVELOPMENT

4.1 Introduction

In the previous chapter we presented a number of existing buffer management schemes that were proposed in the literature, to provide QoS guarantees. These schemes offer limited guaranteed QoS to user applications; resulting in the inability of the current Internet to transmit successfully real-time multimedia applications.

To address this limitation we propose a Dynamic Threshold Buffer Allocation Scheme (DTBAS) that uses complete sharing with virtual partitioning; as a buffer allocation mechanism for future Internet. DTBAS also uses pre-emption and marked IN/OUT packets.

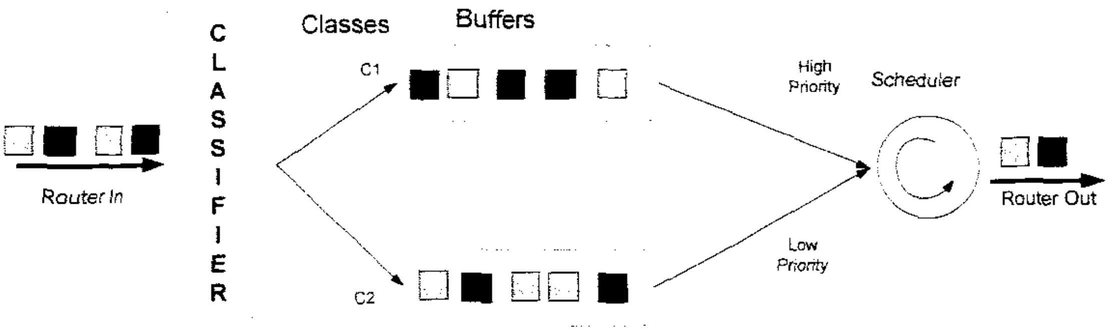


Figure 4.1: Architectural model of DTBAS

The architecture model of DTBAS is shown in Figure 4.1. DTBAS'S architecture is hierarchical and class-based. A router's output buffer consists of ports with Multiple queues per port. Each queue is assigned to a class of traffic. Hence ports consist of classes and classes consist of packets. In this study case only two classes are used. This may be extended to more than 2 in future. These classes are named: Class 1 and Class 2. Class 1 has a higher priority than Class 2; Class 1 is for real-time applications while class 2 is for non-real-time applications.

The design goals and principles of DTBAS are presented in section 4.2. In section 4.3 we discuss the proposed buffer allocation scheme. We refer to this scheme as the Dynamic Threshold Buffer Allocation scheme (DTBAS). The algorithms used by the router to allocate buffer space are given in section 4.4.

4.2 DTBAS Design goals and Principles

The design goals are to: 1) minimise packet loss rate 2) accommodate bandwidth hungry delay-sensitive applications 3) provide efficient buffer utilization and 4) minimise delay. A brief description of each of the mentioned design goal is given below:

4.2.1 Minimise packet loss rate

Real-time multimedia applications are very sensitive to delay and packet loss. To minimise this, we use a buffer management scheme that consists of a suitable allocation and packet dropping algorithm. Hence, a suitable allocation mechanism helps in reducing the packet loss rate and guarantees packet delivery for real-time application flows.

4.2.2 Accommodate bandwidth hungry delay sensitive applications

The current Internet with its Best Effort service model is unable to accommodate and provide high QoS to real-time applications, since routers use a single output buffer to manage and hold all types of applications. Applications statistically share the bandwidth resources; therefore there is no explicit resource allocation. Packets of different classes may come in burst. This may cause delay sensitive applications to be dropped, if there is little space to accommodate them.

To overcome this problem, we use two separate queues within a single output buffer space in the router, so that traffic of each class type flows through its own queue. Thus we can separate highly delay sensitive flows from least delay sensitive ones and manage their delay and packet loss rate separately.

4.2.3 Efficient buffer utilization

The main problem faced by certain high bandwidth applications is encountering no buffer space within the router. If a complete partitioning or partial sharing buffer allocation scheme is used, then there could be a situation where, at a certain period of time, certain applications are starved of buffer volume even though there is buffer space. When a packet enters a full buffer, the arriving packets are forced to drop, if there is no sharing involved. Thus packet loss rate will be high; hence decreasing buffer utilisation. This shows that proper utilization of buffer can lead to a decrease in packet loss rate.

To efficiently use the buffer space, we use complete buffer sharing with virtual partitioning (CSVP) allocation strategy. CSVP reduces the packet loss rate of real-time applications to a minimum.

4.2.4 Minimise Delay

Proper buffer management should lead to high link utilization for delay sensitive real-time multimedia applications. When delay in the router is minimized, the throughput will gradually increase.

To minimize delay for real-time applications, we use a scheduling algorithm that manages the sending rate of real-time and non-real-time applications using a weighted round robin algorithm.

In the section above we have discussed the design goals and objectives of the proposed scheme. In the section below, we look at the proposed model approach.

4.3 Proposed Buffer Management Strategy

A buffer management strategy is used to manage the efficient storage of packets at routers or intermediate systems. Our focus is on the Output buffer of a node. This proposed scheme is referred to as the Dynamic Threshold Buffer Allocation scheme (DTBAS). The allocation strategy used in our scheme is a Complete Sharing with Virtual Partitioning (CSVP) mechanism plus Pre-emption. In this scheme two thresholds are used as minimum threshold (min_{th}) and maximum threshold (max_{th}); as shown in Figure 4.2 below.

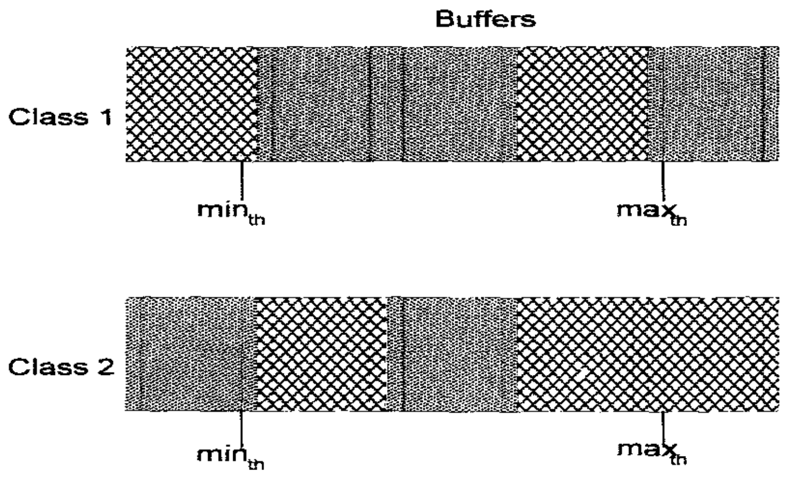


Figure 4.2: Buffer allocation structure

In DTBAS, the allocation strategy used is a complete sharing with virtual partitioning (CSVP) plus pre-emption. Here, two classes are used; Class1 and Class 2. Each class is allocated a certain volume. Two types of applications enter this buffer i.e.: high priority real-time applications and low priority non-real-time applications for Class 1 and Class 2 respectively. These applications use their own buffer space within the buffer. This scheme uses a virtual partition to separate the space for each of the classes. When a high-priority packet enters its designated buffer space and finds it full, then the next step is to look for an available space to borrow. If there is an available space within the low-priority flows, the available space will be borrowed by the high priority class. The space can be borrowed until the minimum threshold of the low priority packets is reached. At this point the high priority packets will start dropping its arriving IN-packets.

The components used in the proposed model are packet classes, classifier, buffer and scheduler. Each of these components is discussed below, on the role each of them plays within the DTBAS model.

4.3.1 Router Model

The proposed router architecture is as shown in Figure 4.3. It is composed of a classifier, two queues and a scheduler. It is assumed that there is no input buffer, since this study is concerned with what happens at the output buffers. This structure has also been used in [21].

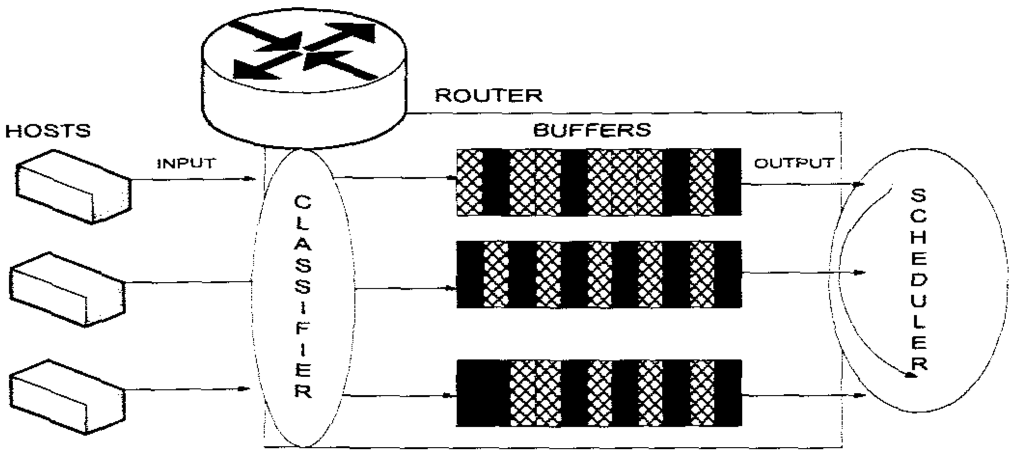


Figure 4.3: Router Model

i) *Classifier*

The router uses a classifier to sort the packets into classes using the Type of Service (ToS) field found in the header of the packet. The sorted packets are sent to their appropriate queues.

ii) *Packet Classes*

In this scheme, application flows are divided into two classes (See Table 4.1).

- Class 1 – real-time multimedia traffic, such as video conferencing and voice conferencing. It is the high priority class.
- Class 2 – non real-time data traffic, such as email and web applications. It is the low priority class.

Table 4.1: Classification of applications

Priorities	Classes	Packet dropping policy
High	Class1- Real-time applications	Drop Tail
Low	Class2-Non-real time applications	Drop Tail

iii) ***Buffer***

In our model the output buffer of a port uses a fixed buffer size. The buffer is divided into two separate queues each having its own basic volume. A buffer volume is the amount of initial buffer assigned to a class. The buffers are dynamically adjusted, so that classes can share the buffer space; thus increasing the utilisation of the buffer space.

iv) ***Scheduler***

A Scheduler removes the packets out of the buffers and transmits it to the next output link. The Weighted Round Robin Scheduler which performs priority-based scheduling is used in this scheme. High priority (Class 1) packets are moved out faster than low priority (Class 2) packets. The weight assigned to each queue determines the number of packets that could be transmitted at any time interval, for each class.

Above we have discussed the router components and the role they play in the proposed architecture. In the next section, we discuss in detail the proposed model approach.

4.3.2 Proposed Buffer Allocation Scheme

We use a hierarchical structure for allocation based on the proposed scheme by Minami, [34]. The buffer space is share among flow classes i.e. Class 1 and Class 2. Each class type is given a buffer volume from the total buffer space. We use tail dropping strategy to drop packets of each class if the allocated space is used up. This is in order to obtain the independent effect of the buffer allocation mechanism. The buffer allocation method used in this scheme is the complete sharing with virtual partitioning plus pre-emption of Class 2. In this scheme, Class 1 packets can pre-empt Class 2 packets, if the basic volume of Class 1 is used up and more Class 1 packets arrive. By doing this we can control the traffic flow and manage low-priority best effort flows, trying to dominate the buffer space.

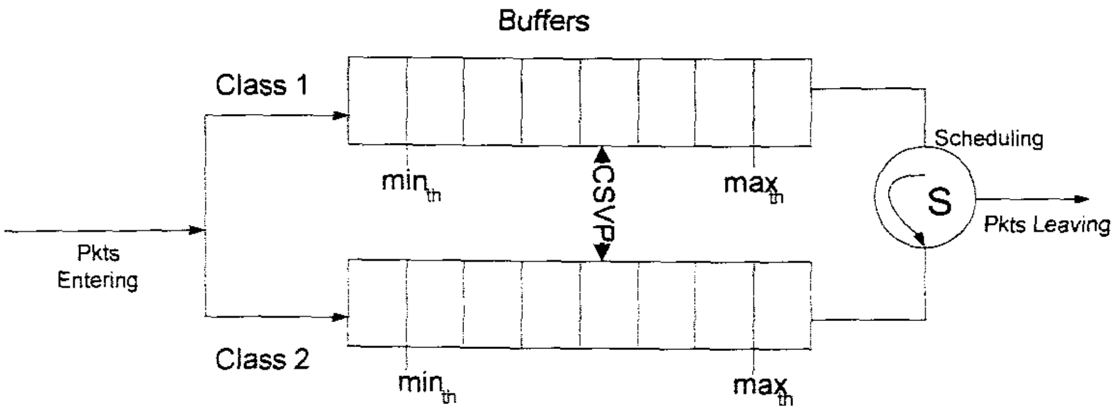


Figure 4.4: Complete Sharing with virtual partitioning CSVP buffer allocation scheme

In this scheme, we use two types of flows, represented as Class 1 which is high priority class transmitting real-time applications and Class 2 which is a low priority class transmitting non-real-time applications. Class 1 has two types of packets: IN-profile and OUT-profile. The IN-profile packets are delay and loss sensitive, while OUT-profile flows are delay sensitive but loss insensitive. Next we look at the buffer allocation and dropping of packets in Class 1 and Class 2.

i) *Class 1 queue:*

Class 1 buffer holds high priority packets. These packets are very sensitive to delay and packet loss. Each buffer has two thresholds: minimum threshold (min_{th}) and maximum threshold (max_{th}). Min_{th} determines the point where congestion is likely to start and packets need to start to drop, while max_{th} shows the peak point of congestion and arriving packets need to be dropped. When an arriving high priority packet finds the buffer space for that class fully occupied, (see in Figure 4.5) space will be sorted in the next queue i.e. Class 2 queue. This available space will be borrowed until the min_{th} threshold is reached. But in the case where a Class 1 packet looks for available space and finds a full Class 2 buffer, then some of the Class 2 packets will be pre-empted and dropped.

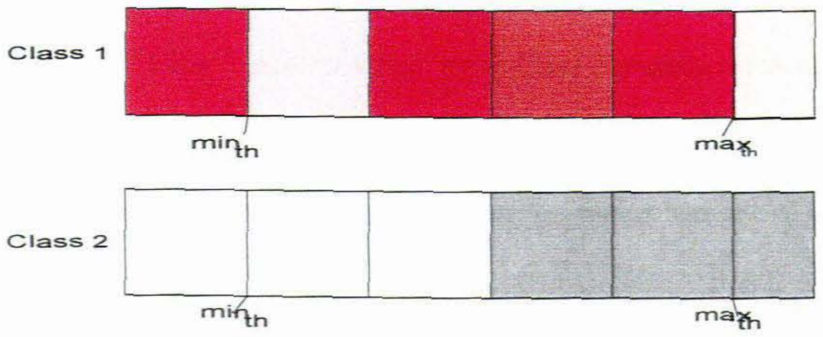


Figure 4.5: Buffer Management for Class 1 traffic

This space borrowed by Class 1 flows is available for a short period of time, but when the Class 2 packets arrive in bulk the Class 1 queue have to slowly release the borrowed buffer space. This avoids wastage of buffer space and also minimises the packet loss of high priority Class 1 flows. Class 1 flows can use Class 2 buffer space until Class 2 is left only with min_{th}. If packets of Class 1 arrive at this time, they are dropped. Dropping OUT-profile packets does not change the quality of the high priority application. If an OUT-profile packet arrives, this packet will be dropped to accommodate the IN-profile packet. This helps in minimising the loss rate of Class 1 traffic and also improves the utilisation of the buffer.

ii) *Class 2 buffer:*

The Class 2 queue holds low priority non-real-time applications and these applications are not sensitive to delay or packet loss. The loss of a Class 2 packet does not affect the quality of these applications. When a Class 2 packet arrives in a full Class 2 buffer (see Figure 4.6), it seeks available space in Class 1 buffer. If there is no available space in Class 1, the Class 2 packets will be forced to drop. This is because Class 1 packets have high priority over Class 2 packets. In the case where there is available space in the Class 1 buffer, this space will be used by the Class 2 packets. If Class 1 packets arrive while Class 2 packets are occupying Class 1 buffer space and no other space is available, the Class 2 packets will be pre-empted from the Class 1 buffer space. By using a pre-emption mechanism the packet loss rate of Class 1 packets will be kept to a minimum.

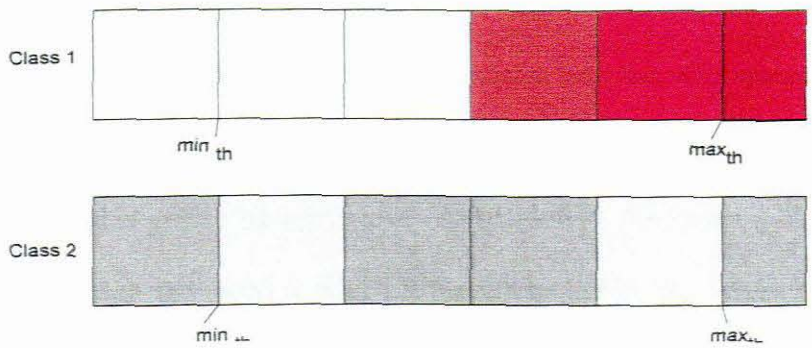


Figure 4.6: Buffer management for Class 2 traffic

4.3.2 Packet discarding and Scheduling Mechanisms

i The packet dropping mechanism

The packet dropping mechanism used in this scheme is a threshold based dropping mechanism. We use Drop tail with dynamic threshold to drop the packets within Class 1 and Class 2 flows. When a Class 1 high priority packet meets a full class 1 buffer space, it will start dropping its OUT-packets. During the sharing of buffer volume, when the excess space is being occupied by Class 1 flows, the Class 2 flows will pre-empt Class 1 flows only if the borrowed time is exceeded. This is by dropping Class 1 packets until Class 2 buffer space reaches its maximum threshold. If Class 2 borrows Class 1`s excess buffer space, then Class 2 packets will be pre-empted when Class 1 packets arrive, even if the borrowed time has not been exceeded. This is because Class 1 flows are high priority real-time applications. These flows are sensitive to loss.

ii Scheduling mechanism

The scheduling mechanism used in this scheme is the Weighted Round Robin [53] mechanism, as shown in Figure 4.7. In this scheme, each class queue is assigned a certain weight, and the one with the highest weight sends out more packets than the other queue. This is followed in the order of the weights that are allocated to the class buffers. By using this method we can control the speed at which packets move out of each of the buffers. If the packet's arriving speed is higher than the speed at which it leaves the buffer, then there is a greater chance of delay and packet loss. So it is important to reduce the delay in the system, especially for high priority packets.

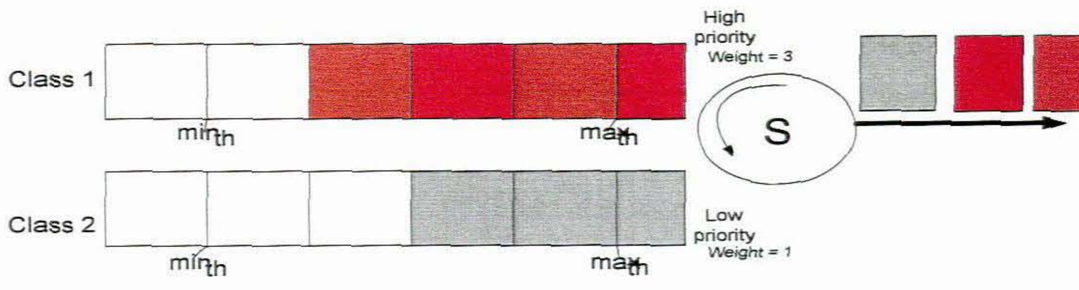


Figure 4.7: Scheduling mechanism

In the example shown in the above figure, a high priority class is assigned a priority weight of 3 and the low priority class is assigned a priority weight of 1. In this case, the class with the higher weight can send out up to 3 packets for every 1 packet which the class with the lower weight sends out.

4.4 DTBAS Algorithm

This section presents the algorithm used by the router for allocating buffer space, dropping packets to manage congestion and schedule packets out of the buffers. The notations used in these pseudo-codes are as shown in Figure 4.8.

4.4.1 The Buffer Allocation algorithm

In this section we break down the buffer allocation algorithms into 2 phases. They are as follows:

- i Buffer allocation algorithm for Class 1 flows
- ii Buffer allocation algorithm for Class 2 flows

An arriving packet is classified by the classifier. The appropriate buffer allocation algorithm is invoked depending on the class of the packet.

Qtot	: Total volume of the buffer
Q1tot	: buffer volume of Q1
Q2tot	: buffer volume of Q2
OutQ1qlen	: queue length of Output buffer Q1
OutQ2qlen	: queue length of Output buffer Q2
time_borrowed:	The time limit set for borrowing excess buffer volume.
$Q1max_{th}$	: Q1 maximum threshold
$Q2max_{th}$	: Q2 maximum threshold
$Q1min_{th}$	: Q1 minimum threshold
$Q2min_{th}$	: Q2 minimum threshold

Figure 4.8: Pseudo-code notations for buffer allocation and packet dropping

i ***Buffer allocation algorithm for Class 1 flows***

When a Class 1 packet arrives, queue 1 is checked if there is space available to store the packet. If there is space available, the packet is stored. If there is no space available and class 2 packets are using the space in queue 1, the Class 2 packets will be pre-empted (dropped by class 1 packets). If queue 1's length is equal to or greater than queue 1's buffer volume, then Class 2 packets in queue 2 are pre-empted. At this point OUT-profile packets of Class 1 are dropped. Pre-emption of Class 2 packets continues until Class 2's queue length drops to the lower threshold ($Q2min_{th}$). When this happens, all incoming Class 1 packets are dropped. A timer is then activated to track the duration of Class 1 packets in queue 2 at full congestion. If the timer's time equals the amount of time set for Class 1 to borrow Class 2 space, then Class 1 packets will be pre-empted from queue 2. All arriving Class 1 packets will be dropped. The algorithm is presented in Figure 4.9.

```
Pre-Condition: Packet is marked as IN- or OUT- Profile, and time_borrowed set

        If (packet class = 1) then
            If (outQ1qlen < $Q1max_{th}$ and $Q1max_{th}$ = Q1tot) then
                Q1: Accept (packet)
            Else If (outQ1qlen < $Q1max_{th}$ and $Q1max_{th}$ < Q1tot) then
                    Q1: drop (class 2 packet)  //Retrieve buffer space
                    $Q2max_{th} = Q2max_{th} - 1$
                    $Q1max_{h} = Q1max_{th} + 1$
                    Q1: Accept (arriving packet)
            Else If (timer != activated and packet_mark = IN) then
                If (outQ2qlen < $Q2max_{th}$ and outQ2qlen > $Q2min_{th}$) then
                        $Q2max_{th} = Q2max_{th} - 1$   //Borrow buffer space from Class 2
                        $Q1max_{th} = Q1max_{th} + 1$
                        Q1: Accept (arriving packet)
                Else If (outQ2qlen = $Q2max_{th}$ and outQ2qlen > $Q2min_{th}$) then
                    Q2: drop (packet)   //Preempt Class 2 packet
                    $Q2max_{th} = Q2max_{th} - 1$
                    $Q1max_{h} = Q1max_{th} + 1$
                    Q1: Accept (arriving packet)
                Else
                    Q1: drop (arriving packet)
                    Timer = activated
                End If
            Else  //timer is activated OR packet is marked as OUT
                Q1: drop (arriving packet)
            End If
        End If

Post condition: accept/reject packets
```

Figure 4.9: Pseudo-code of buffer allocation algorithm for Class 1 flows

ii Buffer allocation algorithm for Class 2 flows

When a Class 2 packet arrives, queue 2 is checked if there is space available to store the packet. If there is space available, the packet is stored. If all of queue 2 space is used up, queue 1 is checked if there is available space. If there is space available in queue 1, the Class 2 packets are stored in this space. If Class 1 packets are occupying queue 2's space and the timer's time is less than the time for Class 1 to borrow this space, then Class 2 packets are dropped. If the timer's time is equal to or greater than the time for Class 1 to borrow the space, then Class 1 packets are pre-empted from queue 2. Class 1 packets are pre-empted until there is no Class 1 packets left in queue 2. At this point, if class 2 packets arrive, they will all be dropped. The algorithm is presented in Figure 4.10.

4.4.2 Packet dropping and Scheduling algorithm

This section presents the algorithm used by DTBAS scheme when dropping and scheduling packets for both Class 1 and Class 2 flows. The dropping algorithm used in our scheme for both classes are the Drop Tail scheme with dynamic thresholds and the scheduling algorithm used to send packets out of the buffer is the weighted round robin algorithm [53].

The drop tail packet discarding mechanism with dynamic threshold minimises the drop rate of real-time multimedia applications by dropping only the out-profile packets from the router. Dropping OUT-profile packets does not affect the quality of the application. Also the low loss of in-profile packets maximises the quality of the application. The WRR scheduling mechanism schedules out real-time applications faster than the non-real-time applications as they are assigned a higher weight. This helps in reducing delay within the buffer reserved for real-time applications.

```
Pre-Condition: time_borrowed set

        If packet class = 2 then
          If (outQ2qlen < $Q2max_{th}$) then
            Q2: Accept (arriving packet)
          Else
            //Check if Class 2 Upper threshold is less than Q2 basic volume
            If ($Q2max_{th}$ < Q2tot ) then
              Q2: Drop (packet)
            Else If (outQ1qlen < $Q1max_{th}$ and $Q1max_{th}$ > 0) then
                      //Borrow buffer Space from Class 1
                      $Q1max_{h} = Q1max_{th} - 1$
                      $Q2max_{th} = Q2max_{th} + 1$
                      Q2: Accept (arriving packet)
            Else    //Q1 length > Class 1 Upper threshold
                If timer_time >= time_borrowed then
                   Q2: Push Out (Class 1 packet)
                   $Q1max_{h} = Q1max_{th} - 1$
                   $Q2max_{th} = Q2max_{th} + 1$
                   Q2: Accept (arriving packet)
                   If $Q2max_{th}$ = Q2tot then
                     Timer = deactivate
                     Reset timer
                   End If
                Else
                   Q2: Drop (arriving packet)
                End If
            End If
          End If
        End If

Post condition: accept/reject packets
```

Figure 4.10: Pseudo-code of buffer allocation algorithm for Class 2 flows

Chapter 5

SIMULATION, RESULTS AND ANALYSIS

5.1 Introduction

Simulation was conducted to evaluate the performance of the proposed model. This chapter presents the simulation, its results for tests conducted using simulation time, as the simulation parameter. The next section discusses the simulation of the DTBAS with a detailed experiment conducted to evaluate its performance.

5.2 DTBAS simulation

5.2.1 Simulation Model

To evaluate the performance of our model we conducted a simulation. The simulation design consisted of two hosts, a router and a sink, as shown in Figure 5.1. Each host generates packets and sends them to the router, where they are classified, queued and then scheduled out to the sink. We use a Two Service Class model; Class 1 (for real-time applications) and Class 2 (for non-real-time applications).

Host 1 generates packets of Class 1 while Host 2 generates packets of Class 2. The generated packets are sent to the Router. A certain amount of buffer volume is assigned to each class (as shown in Figure 5.2). The total buffer space is partitioned as follows: x for Class 1 and y for Class 2.

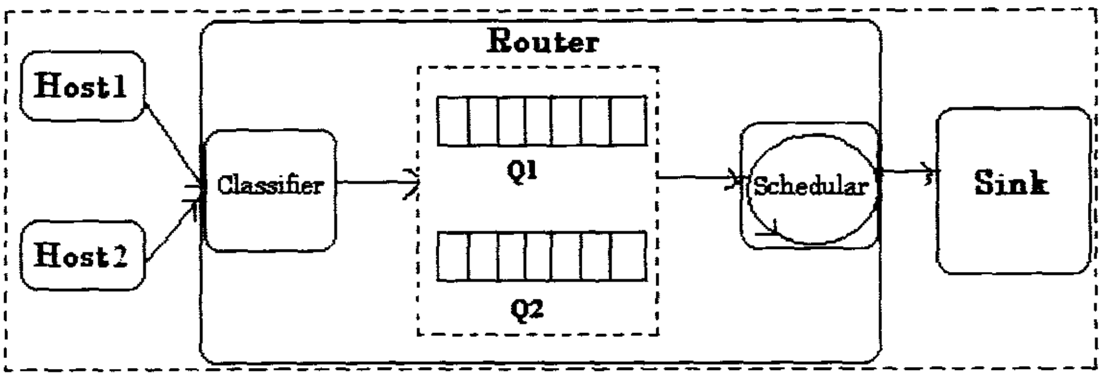


Figure 5.1: Simulation Model

As shown in Figure 5.2, *y* is the buffer volume of the High priority class; *x* is the buffer volume of the Low priority class.

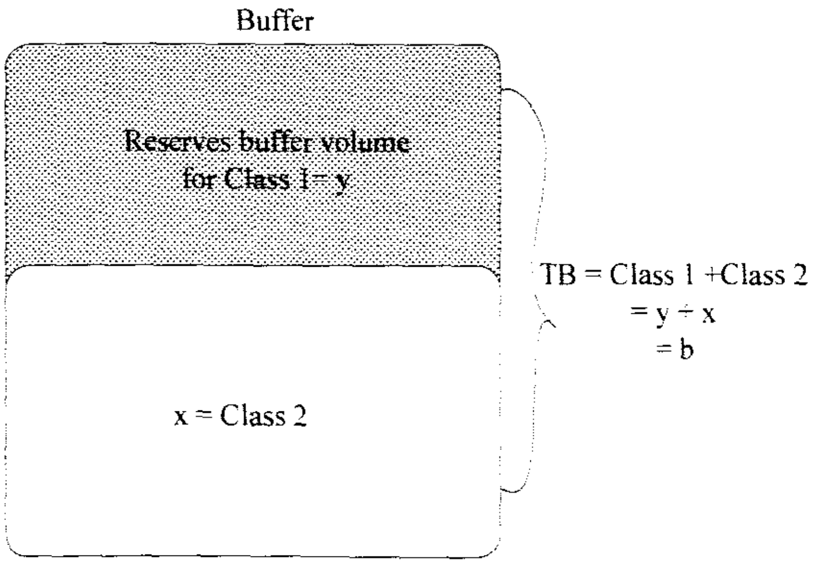


Figure 5.2: Buffer Volume is allocated among Classes as follows:

The buffer volume is calculated and distributed among the classes. If the total buffer volume is B_v then:

$B_v = y + x$ ... ...Equation 5.1 .

5.2.2 Simulation Environment

Simulation was implemented using Microsoft Visual Basic.Net on the Microsoft Visual Studio.Net 2003 environment. The .Net package was selected because it is totally Object Oriented and has simplified development efforts. This makes VB.net advantageous and easily deployable. The simulation results and statistics were stored in a text file. Microsoft Excel was used for graph plotting. The simulation was run on a PC running Microsoft Windows XP operating system with an Intel Pentium 4 processor.

5.2.3 Simulation Performance Metrics

To test the ability of the system to carry Multimedia traffic, we measure the following performance parameters: packet loss rate and buffer utilization.

- *Packet loss rate (PLR) is defined as the number of packets lost per unit time.* The formula to calculate PLR is as given in Equation 5.2.

$$PLR = \frac{Lp}{Gp} \text{..} Equation 5.2$$

Where $PLR =$ Packet loss rate

$Lp =$ Total number of lost packets

$Gp =$ Total number of generated packets

- *Buffer utilization is defined as the amount of buffer that is in use at the router, at any particular time.* The formula to calculate buffer utilization is a given in Equation 5.3.

$$\beta = \frac{b_u}{b_T} \text{..} Equation 5.3$$

Where $\beta =$ Buffer utilisation

b_u = Amount of used buffer

b_T = Total buffer volume

5.2.4 Simulation Parameters

The simulation parameters used are Simulation Time, ratio of Class basic volume, Class 1 upper threshold and total buffer volume.

5.3 *Experiments*

The goal of conducting the experiments was to observe and analyse the behaviour of the DTBAS scheme in managing buffers for both real-time and non-real-time applications. The traffic types used in this experiment were Constant Bit Rate (CBR) and Poisson traffic types. CBR traffic was used so as to observe how the schemes react towards aggressive traffic. The schemes were then subjected to Poisson traffic, widely used as the standard traffic model for simulating Internet traffic. This traffic type has also been used by most of the previously proposed schemes.

We performed several experiments using PLR and buffer utilization as performance metrics; against simulation time. Experiments were conducted to compare the packet loss rate and buffer utilization of the DTBAS scheme with previously proposed schemes such as Complete Partitioning (CP) and Complete Sharing (CS) buffer allocation schemes. All the experiments used two traffic classes; Class 1 and Class 2. Class 1 (Real-time) application is represented here as Clas 1, while Class 2 (non-real-time) application is represented as Clas 2. Drop Tail (DT) packet discarding mechanism was used to discard packets during congestion.

In the sections that follow, we discuss each experiment in detail. The details of the experiments are tabulated and are also represented graphically.

5.3.1 Measurement of Packet loss rate (PLR) using CBR traffic.

Experiments were carried out to measure the PLR of the Complete Partitioning (CP), Complete Sharing (CS) and DTBAS buffer allocation schemes. This was done to compare the performance of DTBAS with those of CP and CS buffer allocation schemes. In this experiment, the allocated buffer volume for each class is 15. We used CBR traffic. We ran the simulation for a total of 5000 seconds (sec).

a) PLR in CP buffer allocation scheme using CBR traffic

- Test

The purpose of the experiment was to check how congestion and packet loss rate in a CP system affect the performance of high priority (Class 1) class. Packets of each class passed through separate queues. An equal amount of buffer volume was allocated to each class from the total buffer volume.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.1.

Figure 5.3 shows a graphical representation of the tabulated data.

Table 5.1: CP packet loss rate using CBR traffic

	Sim_time (sec)										
PLR	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Class 1	0	0	0	0	0.05	0.19	0.29	0.33	0.39	0.43	0.47
Class 2	0	0	0	0	0.10	0.19	0.29	0.36	0.41	0.43	0.47

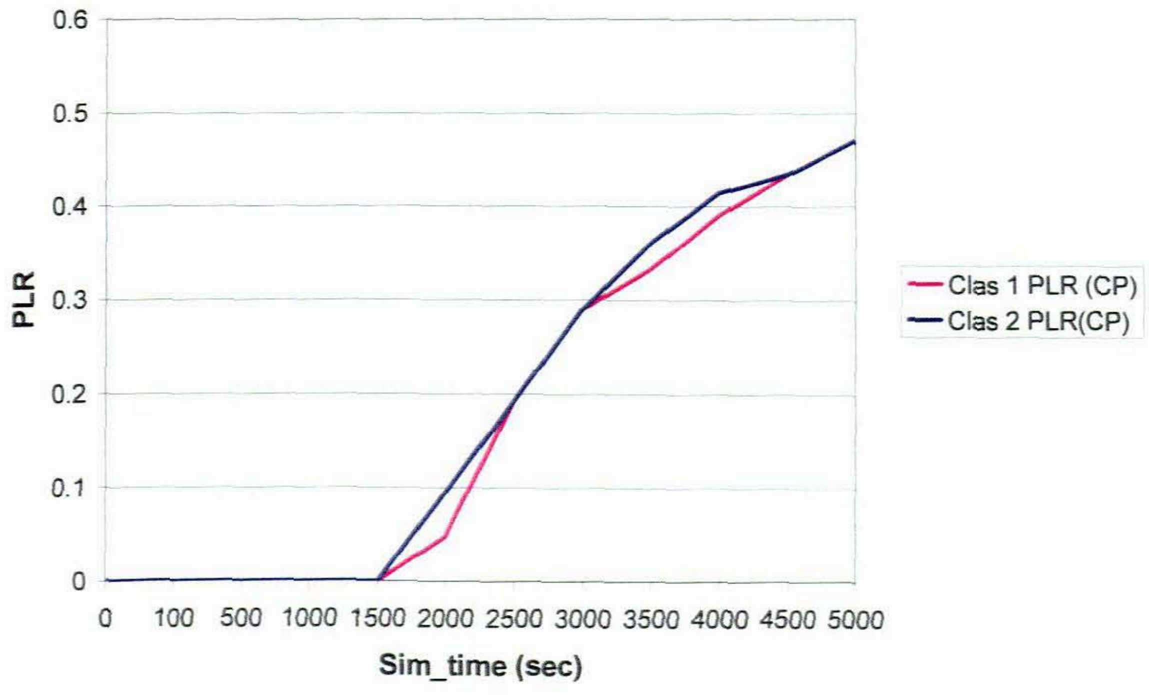


Figure 5.3: PLR vs. Sim_time (CBR, CP, DT)

➢ Analysis

From the graph in Figure 5.3, we see that the packet loss rates of the two traffic classes are approximately the same. This is due to the fact that each traffic class has a separate queue where packets of that class pass through. Since the buffer volume assigned to the two classes were approximately equal, packet loss in both queues are approximate the same.

b) PLR in CS buffer allocation scheme using CBR traffic

➢ Test

The purpose of the experiment was to check how congestion and packet loss rate in a CS system affect the performance of high priority class (Class 1). In this experiment, the output queue was shared statistically by packets of each class.

➢ Results

The data generated during simulation is presented in a tabular form, in Table 5.2. Figure 5.4 shows a graphical representation of the tabulated data.

Table 5.2: CS packet loss rate using CBR traffic

	Sim_time (sec)										
PLR	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Class 1	0	0	0	0	0.05	0.12	0.19	0.22	0.27	0.28	0.31
Class 2	0	0	0	0	0.10	0.27	0.39	0.47	0.54	0.59	0.63

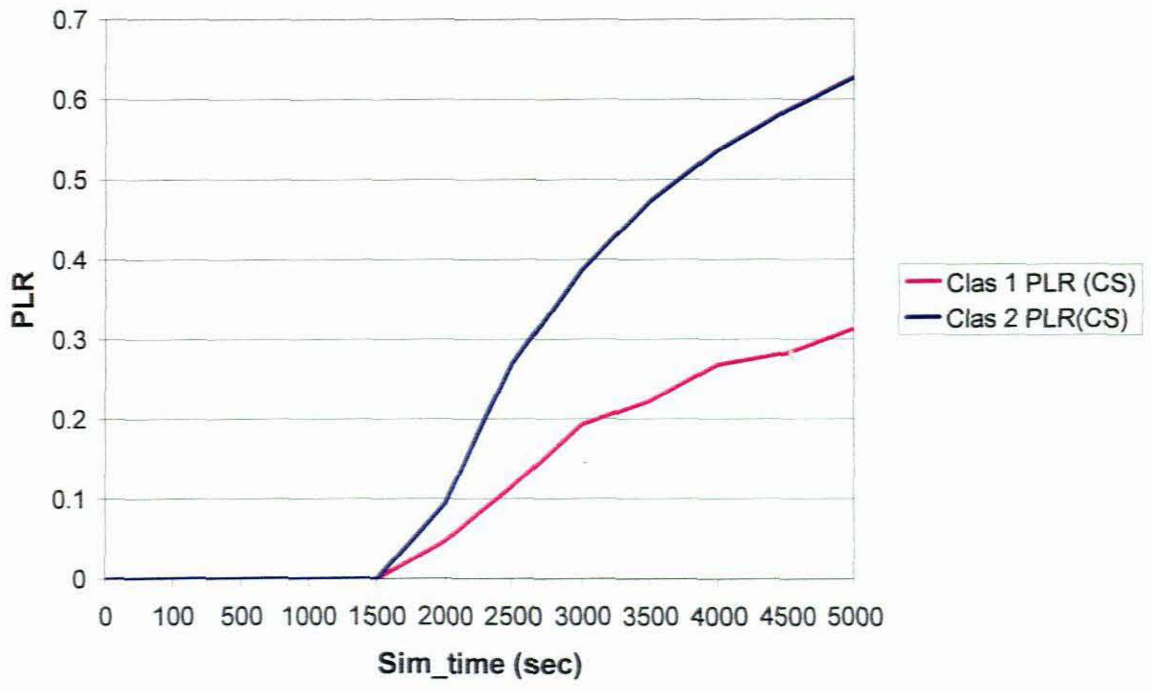


Figure 5.4: PLR vs. Sim_time (CBR, CS, DT)

➢ Analysis

Figure 5.4 shows that the PLR for high priority class is lower than that of the lower priority class. This is due to the fact that buffer space was completely shared among the two classes. When packets of the two classes arrived at the same time, packets of high priority class (Class 1) are stored first. This, quite often, leaves little room for packets of the low priority class.

c) PLR in DTBAS using CBR traffic

➢ Test

The purpose of the experiment was to check how packet loss rate in the DTBAS, affect the performance of high priority class (Class 1) packets. For the DTBAS scheme, the total buffer was virtually partitioned into two, with equal buffer volume assigned to each class. The upper thresholds were initially made equal to the buffer volume of each class. The lower threshold of the low priority class was kept at 30% of the class' buffer volume.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.3. Figure 5.5 shows a graphical representation of the tabulated data.

Table 5.3: DTBAS packet loss rate using CBR traffic

	Sim_time (sec)										
PLR	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Class 1	0	0	0	0	0	0	0	0	0.10	0.17	0.24
Class 2	0	0	0	0	0.14	0.38	0.58	0.69	0.71	0.70	0.71

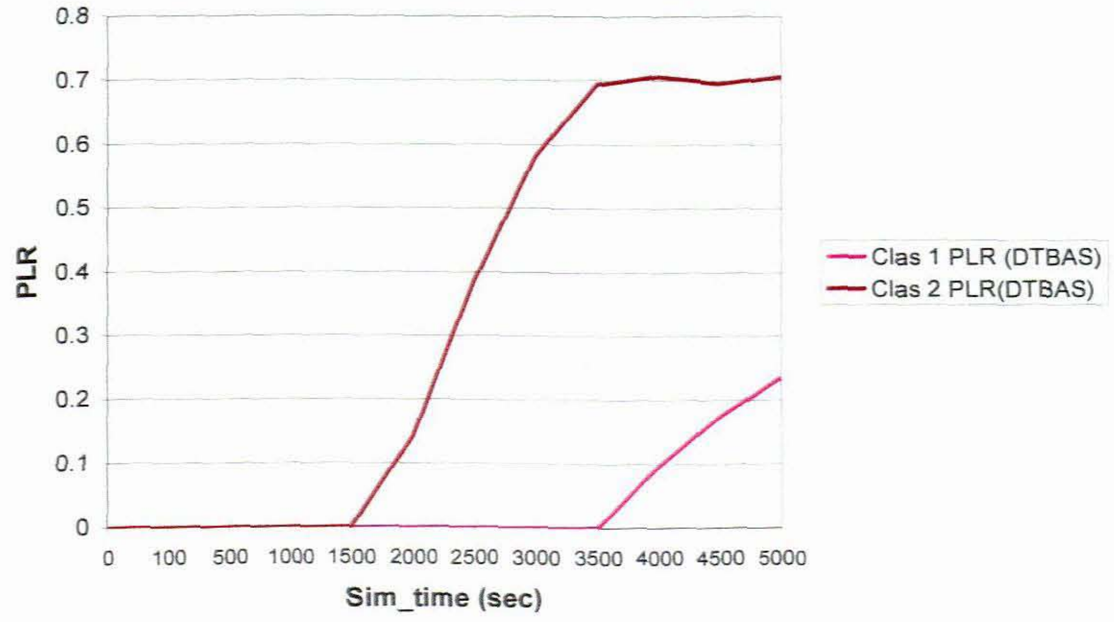


Figure 5.5: PLR vs. Sim_time (CBR, DTBAS, DT)

➢ Analysis

From the graph of Figure 5.5 we can see that low priority packets drop far earlier than high priority packets. This is due to the pre-emption policy used in DTBAS. When high priority packets arrive at full buffer volume, low priority packets are pre-empted. We also notice that the PLR of low priority packets is always higher than that of high priority packets. The scheme aggressively starts dropping high priority (Class 1) IN-packets only when low priority (Class 2) packets have been pre-empted until the lower threshold is reached. By using Complete Sharing with Pre-emption, and the dropping of high priority OUT-packets at full buffer volume, DTBAS minimises the PLR of high priority packets.

d) *Comparison of PLR in DTBAS, CP, and CS schemes using CBR traffic*

➢ Test

The purpose of the experiment was to compare the Class 1 PLR of the DTBAS with those of the CP and CS buffer allocation schemes, using CBR traffic. We used the same packet arriving rate and service rate for all three schemes.

➢ Results

The data generated during simulation is presented in a tabular form, in Table 5.4. Figure 5.6 shows a graphical representation of the tabulated data.

Table 5.4: Comparison of packet loss rate of H1 packets in CP, CS and DTBAS schemes using CBR traffic

	Sim_time (sec)										
PLR of Class 1	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
CP	0	0	0	0	0.05	0.19	0.29	0.33	0.39	0.43	0.47
CS	0	0	0	0	0.05	0.12	0.19	0.22	0.27	0.28	0.31
DTBAS	0	0	0	0	0	0	0	0	0.10	0.17	0.24

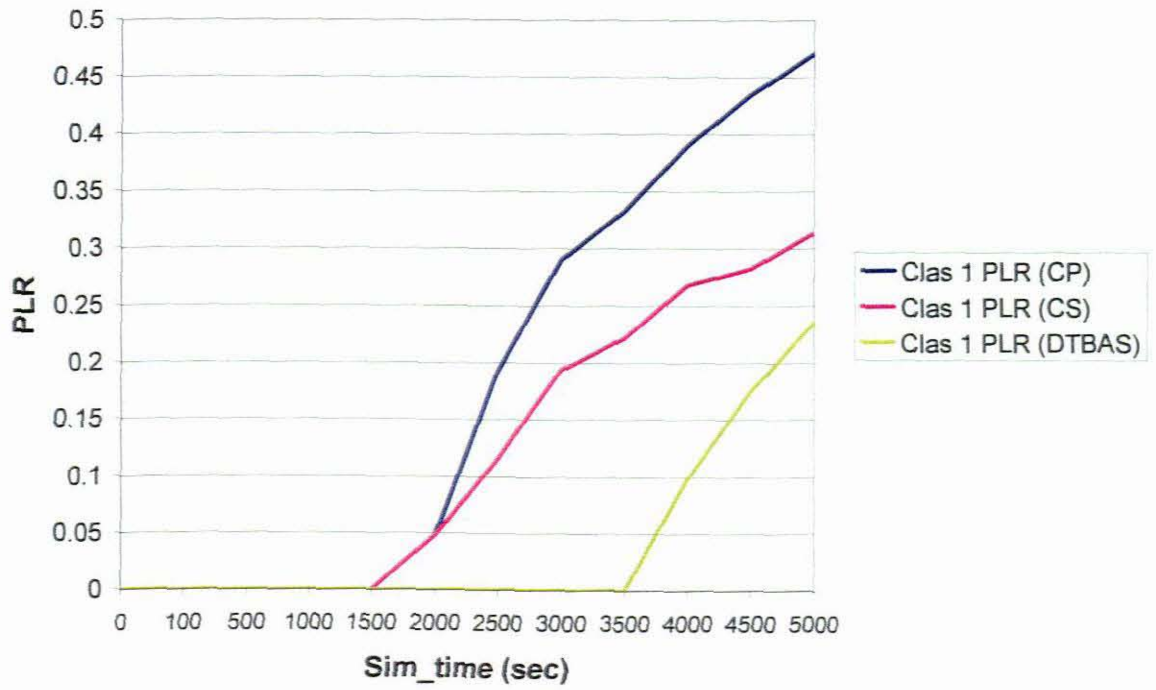


Figure 5.6: PLR vs. Sim_time (CP, CS, DTBAS) using CBR traffic for Class 1 traffic

➢ Analysis

From Figure 5.6, it can be seen that even with the aggressive CBR traffic arrival, the DTBAS takes the longest time to start dropping packets. This is due to the pre-emption policy used in DTBAS. When packets of Class 1 use up their assigned volume, pointed to by the upper threshold, they can increase the upper threshold by pre-empting low priority packets.

It can be noticed from the figure that the PLR for the DTBAS scheme is comparatively far less than those for CP and CS schemes. This is due to the additional advantage gained by dropping OUT-packets of the high priority class, before congestion occurs. Only the IN-profile packets are used to calculate the PLR.

5.3.2 Measurement of Packet loss rate (PLR) using Poisson traffic

Experiments were conducted to measure the PLR in the two common existing buffer allocation schemes i.e. Complete Partitioning (CP) and Complete Sharing (CS) buffer allocation schemes. The results obtained in these schemes were compared with the DTBAS. In all the experiments in this section, Poisson traffic was used for both arriving and departing packets. For the following experiments, two traffic classes were used. These were Class 1 and Class 2. Each class was allocated equal buffer volume. Drop Tail (DT) packet discarding mechanism was also used in all the tests. The simulation run was for a total of 5000 seconds (sec).

a) PLR in CP buffer allocation scheme using Poisson traffic

➢ Test

This experiment was conducted to test how packet loss rate would affect the quality of high priority (real-time) applications in a CP buffer allocation scheme when using Poisson traffic. The buffers were completely partitioned. Packets of each class passed through separate queues. An equal amount of buffer volume was allocated to each class from the total buffer volume.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.5. Figure 5.7 shows a graphical representation of the tabulated data.

Table 5.5: CP packet loss rate using Poisson traffic

	Sim_time (sec)										
PLR	*0*	*500*	*1000*	*1500*	*2000*	*2500*	*3000*	*3500*	*4000*	*4500*	*5000*
Class 1 (CP)	0	0	0	0.06	0.24	0.41	0.44	0.51	0.49	0.54	0.54
Class 2 (CP)	0	0	0	0.30	0.32	0.41	0.42	0.46	0.45	0.50	0.51

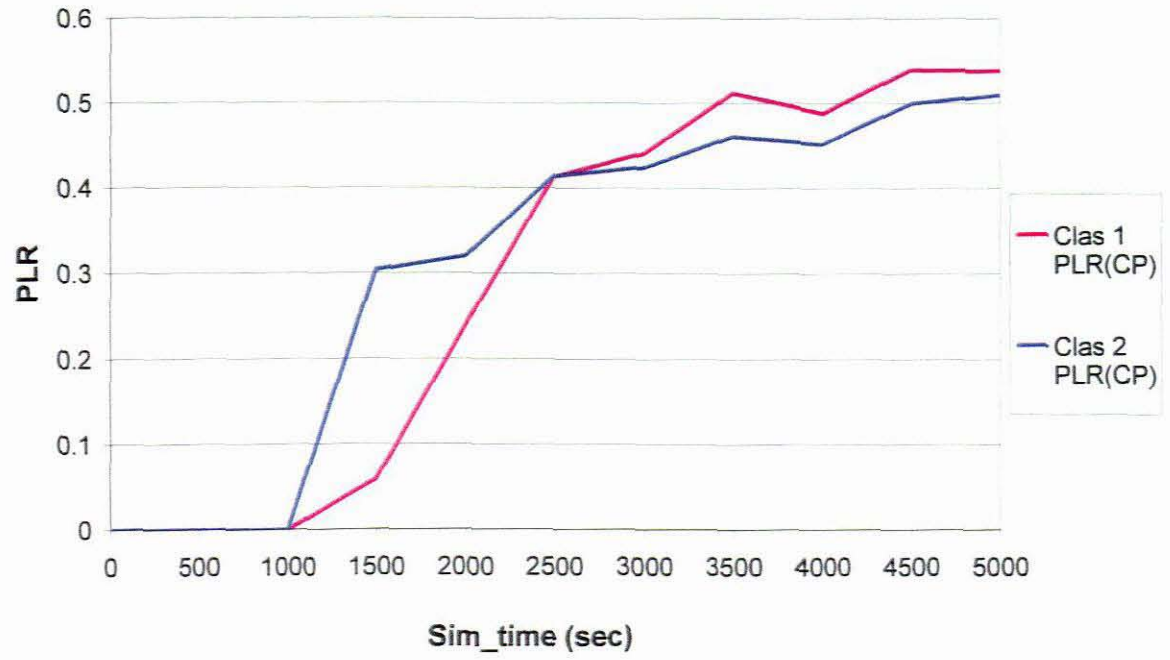


Figure 5.7: PLR vs. Sim_time (Poisson, CP, DT) for Class 1 traffic

➢ Analysis

From Figure 5.7, we notice that PLR for Class 1 moves from being lower to higher that that of Class 2. The crossover is at the 1000 sec mark. Below this point, Class 1 PLR is lower than that of Class 2. Above 2500 sec, Class 1 PLR is higher than that of Class 2. This is due to the fact that the Poisson traffic type was used for both the arriving and departing packets. This makes the packet arrival and departure unpredictable. Hence the CP buffer allocation scheme is unable to cope with this type of traffic with regards to real-time applications, since the scheme does not share its buffer space.

b) PLR in CS buffer allocation scheme using Poisson traffic

➢ Test

The purpose of the experiment was to check how congestion and packet loss rate in a CS system affect the performance of high priority class (Class 1). In this experiment, the output queue was shared statistically by packets of each class.

➢ Results

The data generated during simulation is presented in a tabular form, in Table 5.6. Figure 5.8 shows a graphical representation of the tabulated data.

Table 5.6: CS packet loss rate using Poisson traffic

	Sim_time (sec)										
PLR	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Class 1 (CS)	0	0	0	0.09	0.16	0.28	0.30	0.33	0.36	0.42	0.45
Class 2 (CS)	0	0	0	0.29	0.43	0.55	0.56	0.59	0.58	0.62	0.60

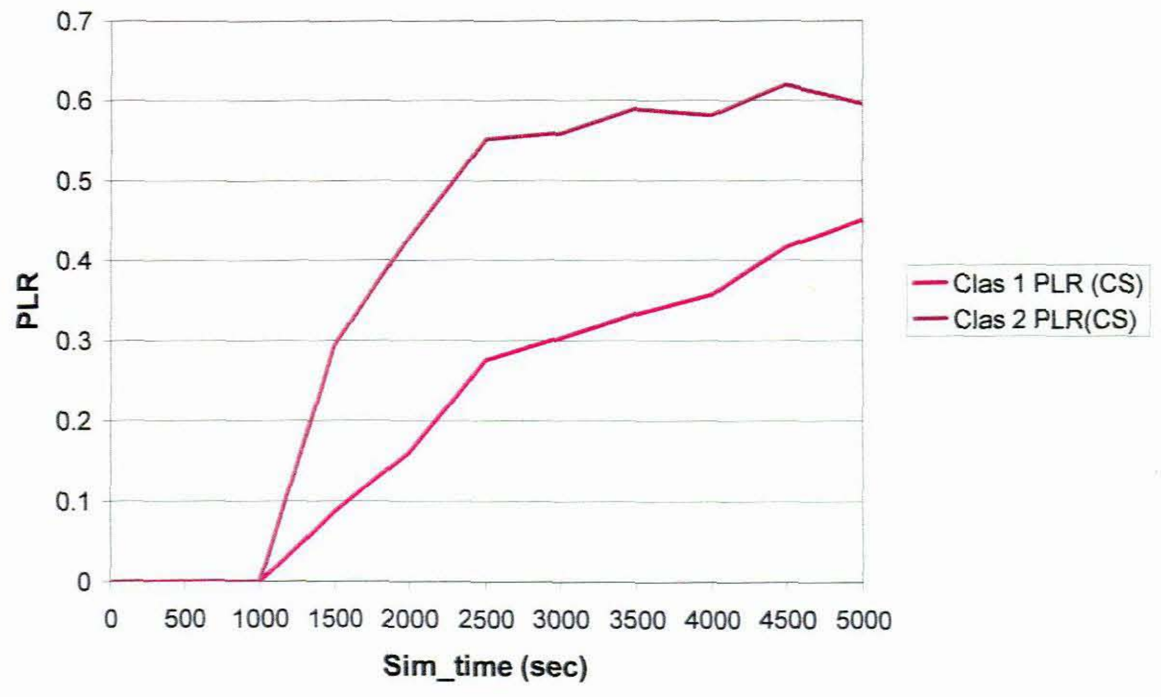


Figure 5.8: PLR vs. Sim_time (Poisson, CS, DT) for Class 1 traffic

➢ Analysis

From Figure 5.8, we notice that the PLR of Class 1 is always lower than that of Class 2. This is as expected. This is due to the fact that buffer space was completely shared among the two classes. When packets of the two classes arrive at the same time, packets of high priority class (Class 1) are stored first. This, quite often, leaves little room for packets of the low priority class.

c) *PLR in DTBAS using Poisson traffic*

- Test

The purpose of the experiment was to compare the packet loss rate of Class 1 and Class 2 flows in the DTBAS. In this scheme, Class 1 and Class 2 were assigned equal buffer volume. Each class' upper threshold was made equal to the buffer volume.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.7. Figure 5.9 shows a graphical representation of the tabulated data.

Table 5.7: DTBAS packet loss rate using Poisson traffic

	Sim_time (sec)										
PLR	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Class 1 (DTBAS)	0	0	0	0	0	0	0.09	0.21	0.21	0.30	0.31
Class 2 (DTBAS)	0	0	0	0.30	0.52	0.83	0.79	0.72	0.74	0.75	0.75

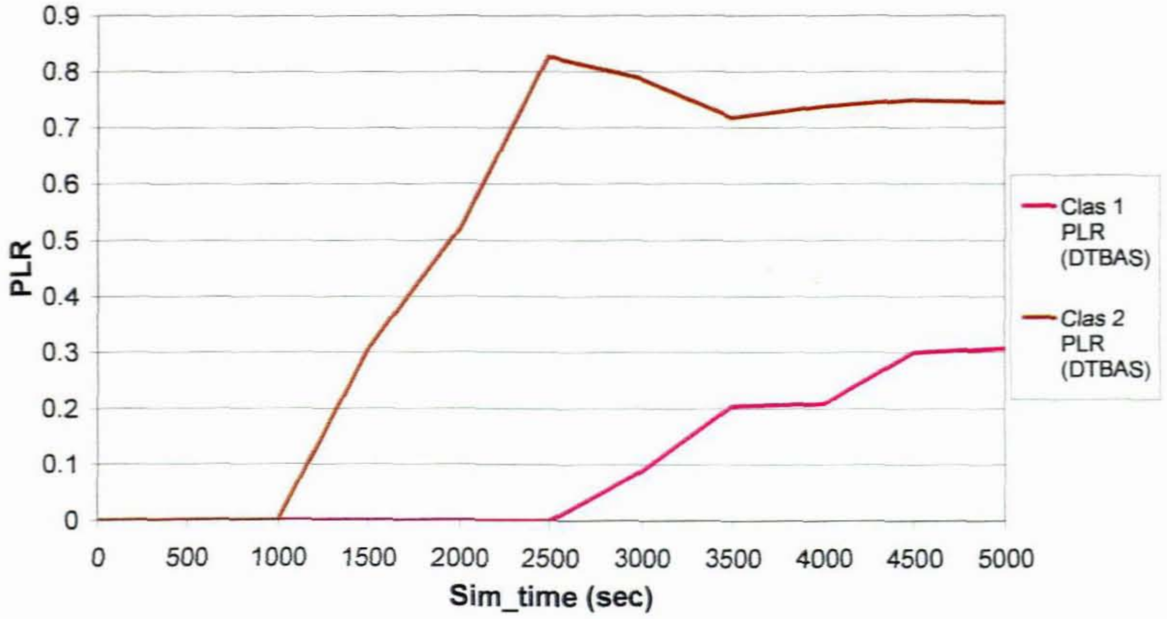


Figure 5.9: PLR vs. Sim_time (Poisson, DTBAS, DT) for Class 1 flows

➢ Analysis

From Figure 5.9, it can be noticed that Class 2 packets are dropped far earlier than packets of Class 1. It can also be seen that after 1000 sec, the packet loss rate of Class 1 is always far lower than that of Class 2. This is due to Class 1 packets being able to pre-empt Class 2 packets at full buffer volume. At this time the threshold of Class 2 is decreased until it reaches a minimum buffer volume. Correspondingly, the upper threshold of Class 1 is increased to include the buffer volume gained from Class 2. Thus, using a dynamic threshold-based sharing with pre-emption helps in reducing the loss rate of Class 1 flows.

d) Comparison of PLR of Class 1 packets of DTBAS with those of CP and CS buffer allocation schemes using Poisson traffic

➢ Test

This test was conducted to compare the packet loss rate of DTBAS with those of the CP and CS schemes. Each of the schemes was subjected to the same packet arriving and departing rate.

➢ Results

The data generated during simulation is presented in a tabular form, in Table 5.8. Figure 5.10 shows a graphical representation of the tabulated data.

Table 5.8: Comparison of PLR of Class 1 packets in CP, CS and DTBAS schemes using Poisson traffic

	Sim_time (sec)										
PLR of Class 1	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
CP	0	0	0	0.06	0.24	0.41	0.44	0.51	0.49	0.54	0.54
CS	0	0	0	0.09	0.16	0.28	0.30	0.33	0.36	0.42	0.45
DTBAS	0	0	0	0	0	0	0.09	0.21	0.21	0.30	0.31

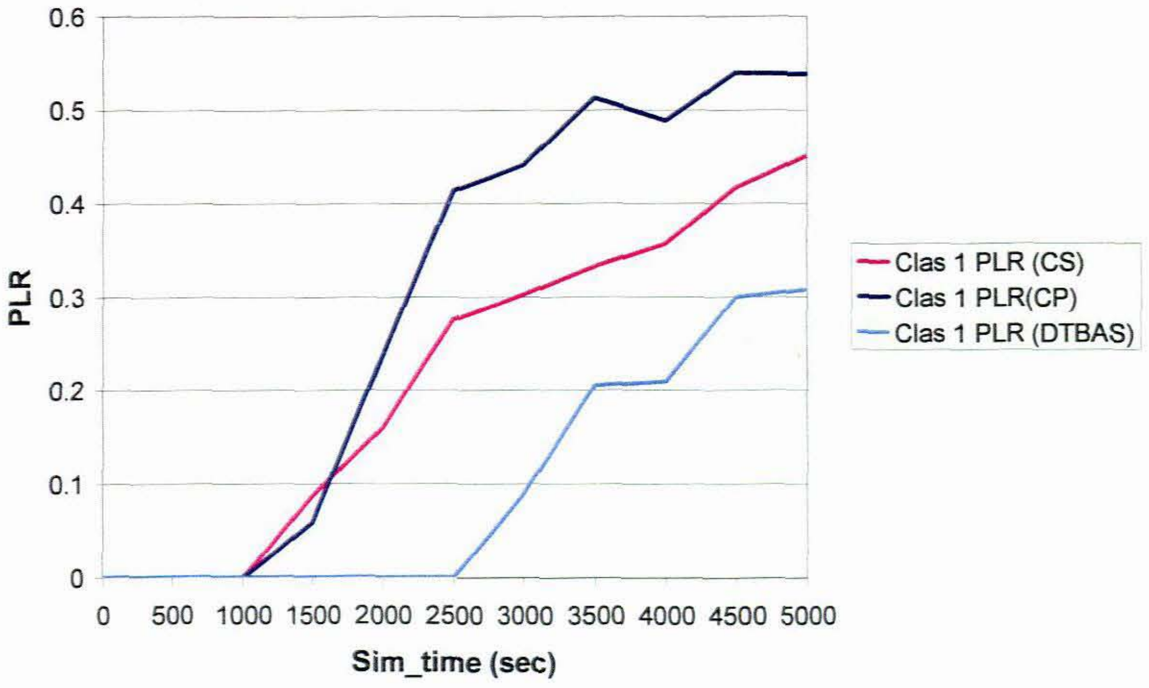


Figure 5.10: PLR vs. Sim_time (CP, CS, DTBAS) using Poisson traffic for Class 1 flows

➢ Analysis

From 5.10, it can be observed that it takes a longer time before Class 1 packets in the DTBAS start dropping, when compared to the CP and CS schemes. It can also be seen that the packet loss rate of Class 1 packets of the DTBAS is far lower than those of the CP and CS schemes. This is due to the fact that in the DTBAS, at full buffer volume, high priority packets of Class 1 pre-empt low priority packets of Class 2. Also, the system starts dropping arriving OUT-packets at the output queue, creating more space for IN-packets to be stored.

5.3.3 Measurement of Buffer Utilization using CBR traffic

Experiments were carried out to measure the buffer utilization of the DTBAS and compared with those of the CP and CS buffer allocation schemes. The aim was to check how efficient the buffers are used by the DTBAS. Two experiments were conducted, one using CBR traffic and the other using Poisson traffic. Drop Tail (DT) was used as the packet discarding mechanism. In this experiment, the total buffer volume for each scheme was 30 and the simulation ran for a total of 5000 seconds (sec).

a) *Comparison of Buffer Utilization of Class 1 packets in CP, CS and DTBAS buffer allocation schemes using CBR traffic*

- Test

In this experiment, CBR traffic was used in all three schemes. The performance of DTBAS, in terms of efficient buffer utilization, was compared with those of the CP and CS schemes.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.9. Figure 5.11 shows a graphical representation of the tabulated data.

Table 5.9: Comparison of buffer utilization in CP, CS and DTBAS schemes using CBR traffic

Buffer Utilization	**Sim_time (sec)**										
	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Tot Q1 +Q2 (CP)	0.07	0.5	0.77	0.93	0.97	1	0.97	1	0.97	1	0.97
Tot Q1 +Q2 (CS)	0.07	0.5	0.9	1	0.97	1	0.97	1	0.97	1	0.97
Tot Q1+ Q2 (DTBAS)	0.07	0.5	0.9	1	0.97	1	0.97	1	0.97	1	0.97

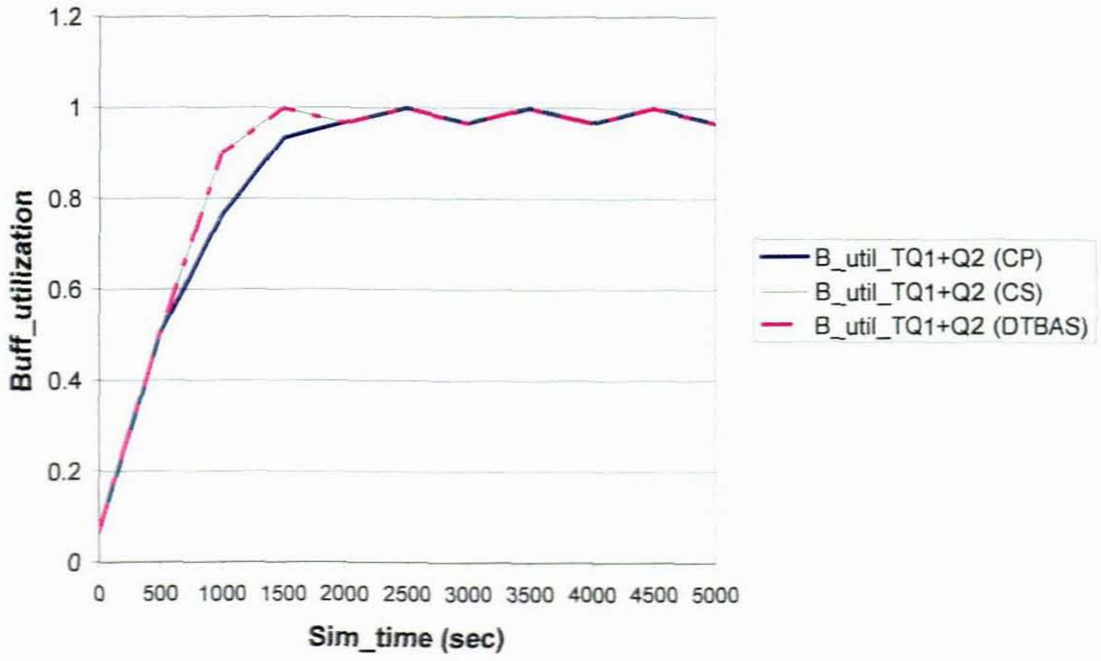


Figure 5.11: Buff_utilization vs. Sim_time for CP, CS and DTBAS systems using CBR traffic

➢ Analysis

From Figure 5.11, it can be noticed that the buffer utilization of the DTBAS scheme is similar to that of the CS scheme. This means that DTBAS also utilises its buffer very efficiently as in the CS scheme.

b) Comparison of Buffer Utilization of Class 1 packets in CP, CS and DTBAS buffer allocation schemes using Poisson traffic

- Test

In this experiment, the performance of DTBAS, in terms of efficient buffer utilization, was compared with those of the CP and CS schemes. Poisson traffic was used in all three schemes.

- Results

The data generated during simulation is presented in a tabular form, in Table 5.10. Figure 5.12 shows a graphical representation of the tabulated data.

Table 5.10: Comparison of buffer utilization in CP, CS and DTBAS schemes using Poisson traffic

Buffer Utilization	Sim_time (sec)										
	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
Tot Q1+Q2 (CP)	0.07	0.63	0.93	0.97	1	0.97	1	1	0.97	0.97	0.97
Tot Q1+Q2 (CS)	0.07	0.63	1	1	1	1	1	1	0.97	1	1
Tot Q1+Q2 (DTBAS)	0.07	0.63	1	1	1	0.97	1	1	0.97	1	0.97

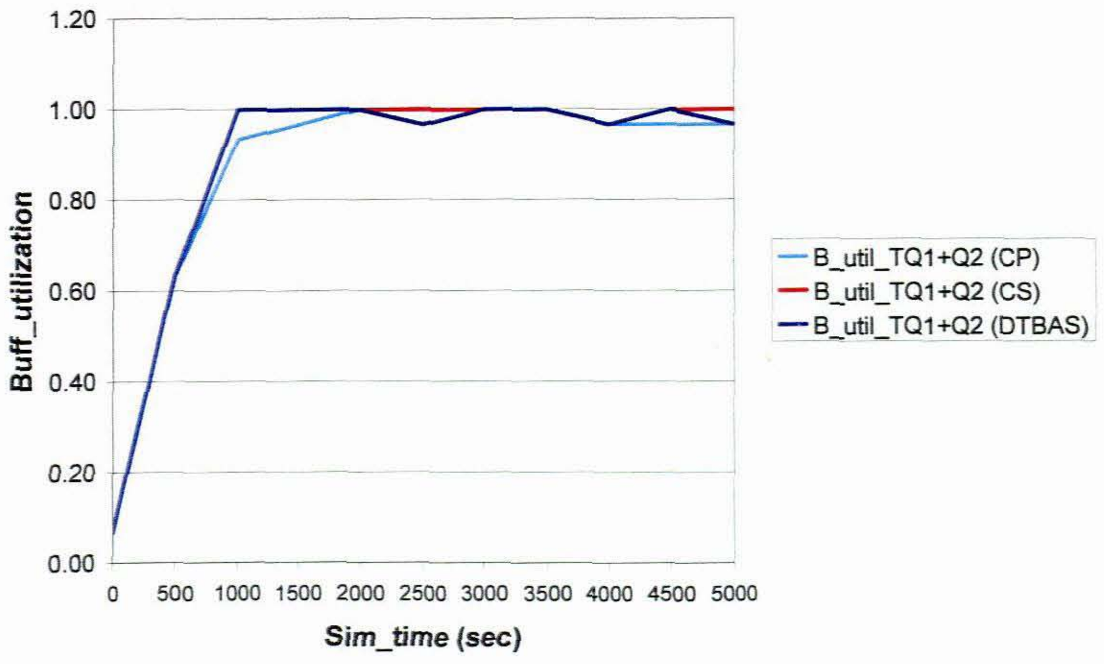


Figure 5.12: Buff_utilization vs. Sim_time for CP, CS and DTBAS using Poisson traffic

➢ Analysis

It can be observed from Figure 5.12, that the buffer utilisation of DTBAS is again comparable to that of the CS scheme. This is because of the sharing of buffer space in these two schemes, especially during periods of congestion.

5.3.5 Further experiments conducted on the DTBAS

More experiments were conducted on the DTBAS scheme to measure the packet loss rate, while varying an additional parameter, including simulation time. In all the cases, Poisson traffic was used. For these experiments, two classes of traffic were used: Class 1 and Class 2. The total allocated buffer volume was varied from 20 to 50. The simulation ran for a total of 5000 sec. The experiments were conducted by:

1. Varying the ratio of class basic volume while measuring PLR for the whole simulation period.
2. Varying the upper threshold of Class 1 queue while measuring the PLR for the whole simulation period.
3. Varying the total buffer volume while measuring the PLR for the whole simulation period.

Below we discuss in detail, each of the above mentioned experiments.

a) *Variation of the ratio of class basic volume*

➢ Test

This experiment was conducted to observe how the PLR in the DTBAS scheme varies with the ratio of class basic volume. The ratio of class basic volume is calculated by dividing the basic volume of Class 1 to that of Class 2. Ratios of 5:3, 2:1, and 3:1 were used. The simulation ran for 5000 sec.

➢ Result

The data generated during simulation is presented in a tabular form, in Table 5.11. Figure 5.13 shows a graphical representation of the tabulated data.

Table 5.11: PLR when varying the ratio of class basic volume

PLR of Class 1	**Sim_time (sec)**										
	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
ratio = 3:1	0	0	0	0	0	0.16	0.28	0.35	0.42	0.47	0.51
ratio = 2:1	0	0	0	0	0.22	0.35	0.44	0.49	0.54	0.58	0.61
ratio = 5:3	0	0	0.10	0.32	0.46	0.55	0.61	0.63	0.67	0.69	0.71

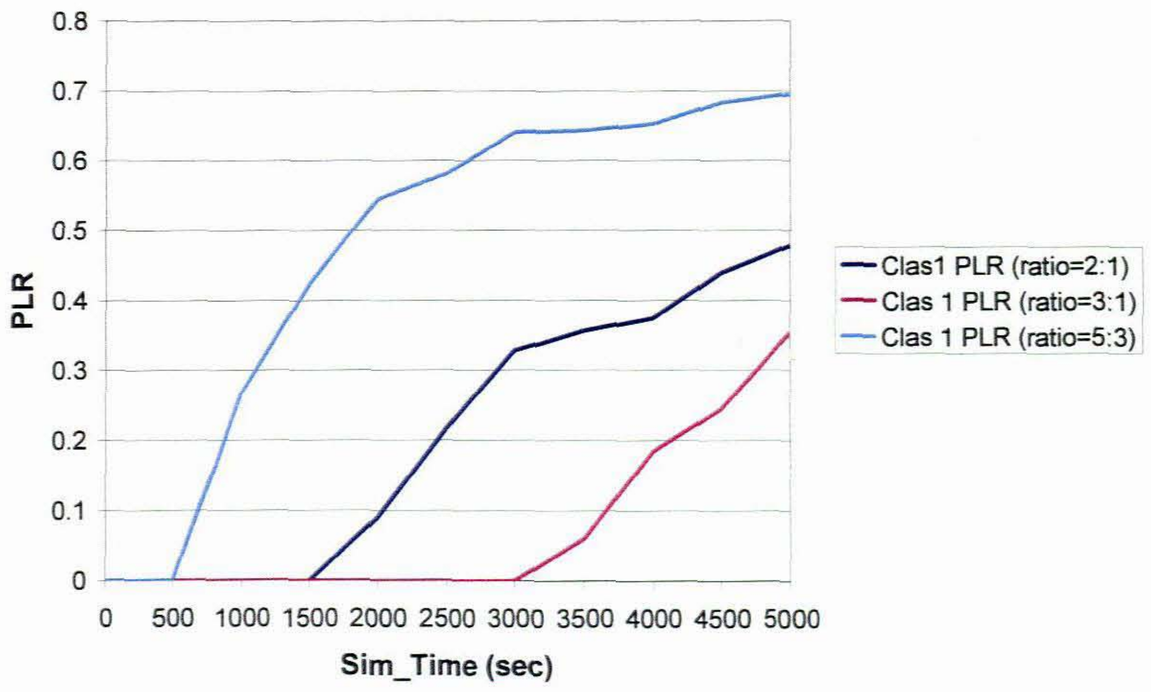


Figure 5.13: PLR when varying ratio of Class basic volume for DTBAS

➢ Analysis

From the graph of Figure 5.13, it can be observed that, as the ratio of class basic volume increases, Class 1 PLR decreases. This is because when the class basic volume increases, comparatively more buffer space is made available to Class 1 packets. Hence few Class 1 packets are dropped. In this experiment the maximum ratio was set as 3:1 and it can be seen that in this scenario, the least amount of packets is lost.

b) Variation of the upper threshold of Class 1 queues.

➢ Test

This experiment was conducted to observe how packet loss rate in the DTBAS varies with the upper threshold of Class 1 queues. For this experiment the initial thresholds for Class 1 queue was set at: 7 (7/30=23%), 12 (12/30=40%) and 17 (17/30=56.7%). The simulation ran for 5000 sec.

➢ Result

The data generated during simulation is presented in a tabular form, in Table 5.12. Figure 5.14 shows a graphical representation of the tabulated data.

Table 5.12: PLR when varying the upper threshold of Class 1 queue

	Sim_time (sec)										
PLR of Class 1	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
th = 17	0	0	0	0	0.43	0.5	0.57	0.64	0.68	0.71	0.75
th = 12	0	0	0.27	0.42	0.57	0.65	0.70	0.73	0.74	0.77	0.79
th = 7	0	0.33	0.69	0.76	0.82	0.85	0.88	0.89	0.89	0.90	0.91

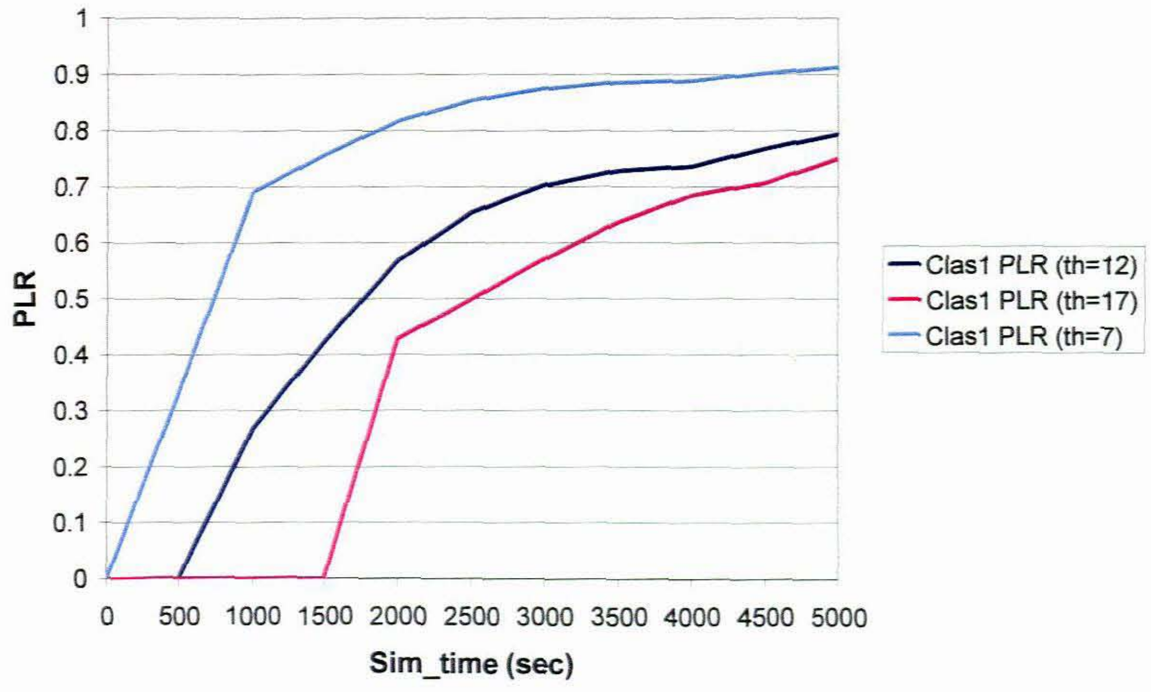


Figure 5.14: PLR vs. Sim_time while varying Class 1 upper threshold

➢ Analysis

From the graph of Figure 5.14, it can be observed that, when the upper threshold for Class 1 queue increases, the PLR for Class 1 decreases. This is due to the fact that as the threshold increases, much buffer space is made available to Class 1 packets.

c) ***Variation of the total buffer size.***

➢ Test

This experiment was conducted to observe how the packet loss rate in the DTBAS varies with the total buffer size. For this experiment the total buffer volume used were 50 (100%), 40 (80%) and 30 (60%). The simulation ran for 5000 sec.

➢ Result

The data generated during simulation is presented in a tabular form, in Table 5.13. Figure 5.15 shows a graphical representation of the tabulated data.

Table 5.13: PLR when varying the total buffer size

PLR of Class1	Sim_time (sec)										
	0	500	1000	1500	2000	2500	3000	3500	4000	4500	5000
tbsize = 50	0	0	0	0	0	0	0	0	0	0	0.15
tbsize = 40	0	0	0	0	0	0.04	0.17	0.21	0.24	0.32	0.37
tbsize = 30	0	0	0	0.12	0.32	0.4	0.48	0.5	0.51	0.56	0.59

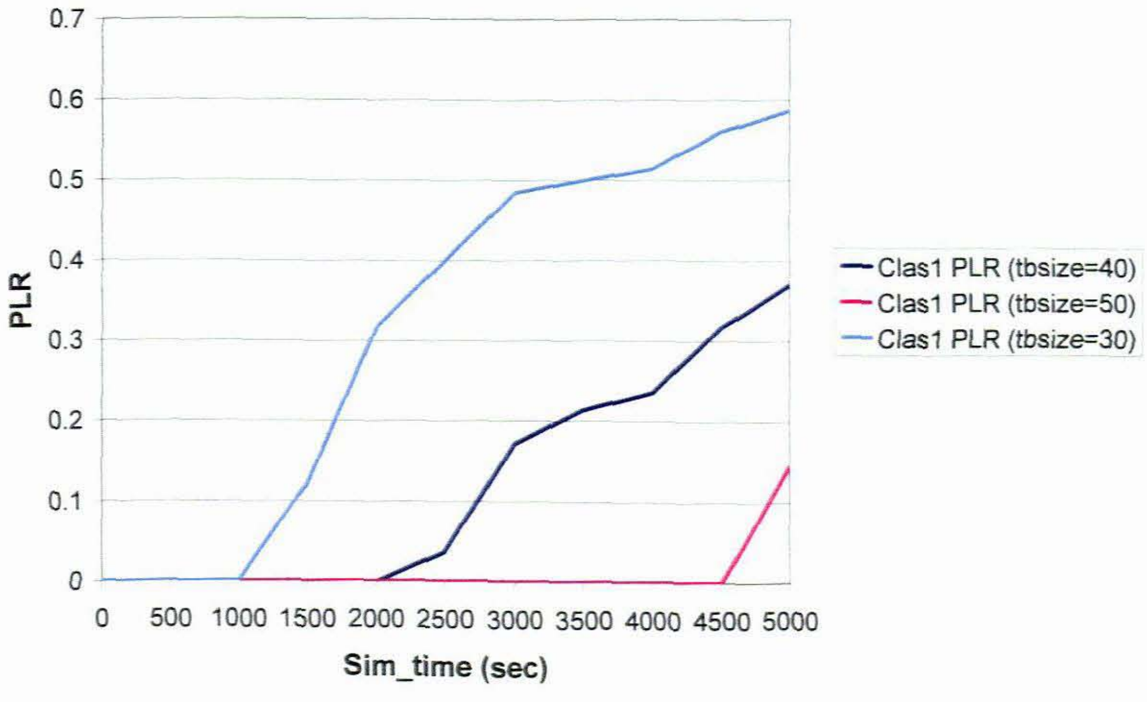


Figure 5.15: PLR when varying the total buffer size

➢ Analysis

From the graph of Figure 5.15, it can be observed that, when the total buffer size increases, the PLR decreases. This is because when the total buffer size increases, more buffer space is available for packets to be stored. Hence fewer packets are dropped.

In summary, from the detailed experiments conducted, the following can be observed:

- Class 1 PLR in the DTBAS is lower than Class 2 PLR.
- Class 1 PLR of the DTBAS is very low when compared with Class 1 PLR of the CS or the CP schemes. This is independent of the traffic used.
- The buffer utilisation of the DTBAS is very good and is comparable to that of the CS scheme.
- Increasing the ratio of Class 1 basic volume to Class 2 basic volume reduces the PLR in the DTBAS scheme.
- Setting the upper threshold of Class 1 queue to high, decreases the PLR.
- The PLR of the DTBAS decreases with an increase in total buffer volume.

Chapter 6

CONCLUSIONS

6.1 Introduction

Congestion on the network results in long queues forcing packets to be dropped. This especially affects the quality of real-time multimedia applications. This is the case with the current Internet as it strives to accommodate these delay and loss-sensitive applications. Many mechanisms to alleviate congestion have been proposed but very few have dwelt on the use of buffer allocation strategies to reduce the effect of congestion. In this study, a Dynamic Threshold Buffer Allocation scheme (DTBAS) has been proposed, that will efficiently allocate buffers to real-time multimedia applications on the Internet while also providing excellent utilisation of the buffers.

This chapter reviews the DTBAS strategy. It also reviews the achievement of the research work. Reasonable critiques of the work and some suggestions on how it can be extended in future have also been included. Limitations of this work and some suggestions on how it can be extended in future have also been included.

6.2 Summary

The relevant design goals and principles of the DTBAS are to:

1) *Minimise packet loss rate*: - this was achieved by using separate buffer space for different classes of applications that entered the router. Real-time applications were classified as Class 1 while non-real-time applications were classified as Class 2. Pre-emption was used during periods of congestion in order to further minimise the PLR of Class 1 applications.

2) *Accommodate and manage different types of applications*: - by dividing the buffer into two logical buffer spaces, it was easy to handle the congestion and packet loss for both real-time and non-real-time applications. The buffer was virtually partitioned into two spaces and the buffer was then completely shared among two different types of applications. Real-time applications consist of IN-profile and OUT-profile packets, where out-profile packets are not sensitive to loss. Thus we manage the efficiency of transmission of real-time applications by dropping some OUT-profile packets. Drop tail packet discarding strategy was used to pre-empt and drop packets within the buffer. Weighted Round Robin scheduling strategy was also used to manage the scheduling of the packets out of the buffer.

3) *Provide efficient buffer utilisation*: - by using complete sharing with virtual partitioning; we efficiently utilise the buffer; as different types of applications share the buffer space with one another.

As packets arrive at the output buffer of a router, these packets are classified and allocated buffer space according to the information extracted from the header of the packet. A hierarchical buffer allocation structure is used, where the buffer in an output port is virtually partitioned into two classes, according to the type of applications; Class 1 is for real-time applications while Class 2 is for non-real-time applications. Real-time applications consist of IN-profile and OUT-profiles packet. IN-profile packets are sensitive to packet loss and they affect the quality of real-time applications when these packets are dropped. The OUT-profile packets are not sensitive to loss and thus dropping these packets does not affect the quality of the application. Each application class is allocated a buffer volume. Real-time applications are given higher priority than non-real-time applications. In case an arriving real-time packet finds a full buffer space, excess buffer space is borrowed from the buffer space of non-real-time applications. If there is buffer space available, then the application will access this excess buffer space. If Class 1 buffer volume is full and Class 2 buffer volume is also full, an arriving Class 1 packet will pre-empt a Class 2 packet. The Class 2 buffer space is accessed by Class 1 packets, until the upper threshold of Class 2 reaches its minimum threshold (min_{th}). During this pre-emption period, Class 1 OUT-profile packets are randomly dropped so as to reduce the amount of load into the Class 1 queue. Arriving Class 2 packets are all dropped.

Extensive simulation was performed using parameters such as packet loss rate and buffer utilisation. Two different types of traffic were employed; CBR and Poisson. The Class 1 PLR for the DTBAS scheme was compared with those of the CS and the CP buffer allocation schemes. It was found that the DTBAS scheme performed best in terms of PLR than the CS and the CP schemes. It was also found that the buffer utilisation of the DTBAS is comparable to that of the CS scheme. Additional tests show that, to add value to the DTBAS, a high Class 1: Class 2 buffer volume ratio, a high upper threshold value for Class 1 queue and a large total buffer volume is required. These results make the DTBAS one of the best schemes to be considered for the future Internet. The DTBAS uses dynamic thresholds to signal when packets should be dropped. It also uses pre-emption of low priority applications by high priority applications. During the pre-emption period, the dropping mechanism of real-time applications changes from Drop Tail to a modified form of RIO [15]. It is assumed that Class 1 packets are marked as either IN-profile or OUT-profile packets.

A limitation of this work is that link delay between nodes was not taken into account. This, though does not impact much on the current results. In addition only two sources were used (one for real-time, the other for non-real time,) to generate all the traffic. It would have been more realistic for many sources to contribute to the traffic into the router.

6.3 Future Work

The results obtained from the simulation showed suitability of the proposed scheme for real-time applications. However, the simulation is only an approximation of the reality; therefore another primary goal in the future work is to observe the behaviour of the DTBAS in real networks. A test-bed network could be constructed, with the DTBAS as the buffer allocation protocol, where both real-time and non-real-time applications are used.

This research concentrated on the buffer allocation strategy in the output buffers of a router, with the use of performance metrics such as packet loss rate and buffer utilization for real-time and non-real-time applications. The performance metrics used in this study only form a small part of the performance metrics that can be investigated in the output buffers of a router. The performance metrics that could be investigated include *delay* and *throughput* with respect to the load applied in the network. The DTBAS scheme could be further investigated using load as the performance parameter. Another important task for future work in the DTBAS may be to investigate its performance with a combined dropping and scheduling mechanism.

References

[1] Armitage, G., "Quality of Service in IP Networks- foundation of a multi-service internet", First Edition, McMillan, April 2000.

[2] Athuraliya, S., Li, V., Low, S., and Yin, Q., "REM: Active queue management," in IEEE Network, vol. 15, pp. 48-53, May/June 2001.

[3] Awan, I., Ahmad, A. and Ahmad, B., "Performance analysis of multimedia based web traffic with QoS constraints", Journal of Computer and System Sciences, Available online 24 April 2007.

[4] Aweya, J., Ouellette, M., and Montuno, D. Y., "Service differentiation, using a multi-level RED mechanism", Int. Journal. Network Mgmt, vol. 12, pp.81–98.

[5] Aweya, J., Ouellette, M., Dasylva, A. and Montuno, D. Y., "DRED-MP: Queue management with multiple levels of drop precedence", Int. Journal. Network Mgmt, vol.14, pp. 405–424, 2004.

[6] Aweya, J., Ouellette, M., Dasylva, A. and Montuno, D. Y., "DRED: a random early detection algorithm for TCP/IP networks", Computer Networks 2001.

[7] Aweya, J., Ouellette, M., Montuno, D.Y., "Multilevel Active Queue Management with Dynamic Thresholds", Computer Communication vol. 25, pp.756-771, 2002

[8] Bai, Y. and Ito, M. R., "A new technique for minimizing network loss from users' perspective", Journal of Network and Computer Applications, vol. 30, Issue 2, pp. 637-649, April 2007.

[9] Bartek, W. and Moshe, Z., "GREEN: An active queue management algorithm for a self managed Internet", In Proceedings of ICC 2002, vol. 4, pp. 2368–2372, April 2002.

[10] Blake, S., Black, D., Calson, M., Davies, E., Wang Z. and Weiss, W., "An Architecture for Differentiated Services", IETF RFC 2475, December 1998.

[11] Branden, B., Clark, D., Davie, B., Floyd, S., Minshall, G., Peterson, L., Ramakrishnan, K. K., Shenker, S., Zhang, L., "Recommendation on Queue Management and Congestion Avoidance in the Internet", IETF RFC 2309, April. 1998.

[12] Braden, R., Clark, D. and Shenker, S., "Integrated Services in the Internet Architecture: an Overview", IETF RFC 1633, June 1994.

[13] Chuang, L., and Li, Y., "Dynamic Partial Buffer Sharing scheme: Proportional Packet Loss Rate", International Conference on Communication Technology, Beijing, China, pp. 255-258, April 2003.

[14] Cisco Technical Specification, "Distributed Weighted Random Early Detection", URL:
http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/wred.pdf. (Last accessed 26 August 2007)

[15] Clark, D. and Fang, W., "Explicit Allocation of Best-Effort Packet Delivery Service", IEEE/ACM Transactions on Networking, vol. 6, Issue 4, pp. 362–373, August 1998.

[16] Duan, Q., "Quality of service provision in combined input and cross-point queued switches without output queuing match", Computer Communications, vol. 30, Issue 4, pp. 830-840, 26 February 2007.

[17] Feng, W., Kandlur, D., Saha, D. and Shin, K., "BLUE: A New Class of Active Queue Management Algorithms", Technical Report CSE-TR- Dept. EECS, Univ. MI, pp. 387-99, Apr. 1999.

[18] Floyd, S. and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, vol. 1, Issue 4, pp. 397–413, 1993.

[19] Floyd, S., Gummadi, R., and Shenker, S., "Adaptive RED: An algorithm for increasing the robustness of RED", Technical report, ICSI, August 2001.

[20] Forouzan, B., "Data Communications and Networking", Third Edition, McGraw-Hill, 2003.

[21] García-Macías, J. A., Rousseau, F., Berger-Sabbatel, G., Toumi, L., and Duda, A., "Quality of Service and Mobility for the Wireless Internet", Proceedings of the first ACM workshop on Wireless Mobile Internet", pp. 34-42, July 2001.

[22] Guerin, R. and Peris, V., "Quality-of-Service in Packet Networks: Basic Mechanisms and Directions", Computer Networks vol. 31, pp. 169–189, 1999.

[23] Ghosh, S., "A predictive dynamic output buffer reconfiguration (PDOBR) architecture for ATM networks", Computer Communications, vol. 29, Issue 17, pp. 3609-3624, 8 November 2006.

[24] Han, S. C., Zaroo, P., Yau, D. K. Y., Dong, Y., Gopalan, P., and Lui, J. C. S., "Quality of service provisioning for composable routing elements", Computer Networks, vol. 50, Issue 13, pp. 2255-2270, 15 September 2006.

[25] Kamal, A. E. and Hassanein, H. S., "Performance Evaluation of Prioritized Scheduling with Buffer Management for Differentiated Services Architectures", Computer Networks, vol. 46, Issue 2, pp. 169-180, 2004.

[26] Labrador, M. A. and Banerjee, S., "Enhancing Application Throughput by Selective Packet Dropping", Proc. of IEEE ICC, pp. 1217-1222, 1999.

[27] Labrador, M. A. and Banerjee, S., "Packet Dropping Policies for ATM and IP networks", IEEE/ACM Transactions on Networking, vol. 2, Issue 1, pp: 1-15, 1994.

[28] Lakshman, T. V., Neidhardt, A., and Ott, T., "The Drop from Front Strategy in TCP and in TCP over ATM", Proc. of IEEE INFOCOM, pp. 1242-1250, 1996.

[29] Lai, W. K. and Mu-Rung-Shiu, "Improving good-puts of IP packets under ATM UBR traffic with port-based queuing schemes", Computer Networks, vol. 34, pp.419-437 ,2000.

[30] Liebehrr, J., Christin, N., "Rate allocation and buffer management for differential services", Computer Networks, vol. 40, pp. 89-110, 2002.

[31] Lin, A. Y. M., Silvester, J. A., "Priority Queuing Strategies and Buffer Allocation Protocols for Traffic Control at an ATM Integrated Broadband Switching Systems",

IEEE Journal on Selected Areas in Communications, vol. 9, pp. 1524-1536, December 1991.

[32] Malla, A., El-Kadi, M., Todorova, P., "A Fair Resource Allocation Protocol for Multimedia Wireless Networks", Proceedings of International Conference on Parallel Processing, pp.437, 2001.

[33] Menth, M. and Martin, R., "Service differentiation with MEDF scheduling in TCP/IP networks", Computer Communications, vol. 29, Issue 7, pp. 812-819, 24 April 2006.

[34] Minami, K., Tode, H., Murakami, K., "Class-Based QoS Control Scheme by Flow Management in the Internet Router", Proceedings of the 25th Annual IEEE Conference on Local Computer Networks, pp. 58-65, 2000.

[35] Mobasseri, M. and Leung, V. M., "A New Buffer Management Scheme for Multimedia Terminals in Broadband Satellite Networks", Proc of 35th Hawaii International Conference on System Science, Hawaii, 2002.

[36] Ojong, G., and Takawira, F., "Rainbow Services: A New Architecture for QoS Provisioning in the Future Internet", Proceedings of SATNAC '03, September 2003.

[37] Olifer, N. and Olifer, V., "Computer Networks; principles, Technologies and protocols for network design", John Wiley and Sons. Ltd, England, Copyright 2006.

[38] Ott, T. J., Lakshman, T. V., and Wong L. H., "SRED: Stabilized RED", Proc. IEEE INFOCOM '99, pp. 1346–55, Mar. 1999.

[39] Pletka, R., Waldvogel, M., Mannal, S., "PURPLE: Predictive Active Queue Management Utilizing Congestion Information", Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks, 2003.

[40] Semeria, C., "Supporting Differentiated Service Classes: Queue Scheduling Disciplines", Juniper Network, 2001.

[41] Sidahmed, E. and Ali, H. H., "A QoS based model for supporting Multimedia Applications for Cellular IP", In Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05), 2005

[42] Wang, C., Li, B., Hou, Y., Sohraby, K., and Lin, Y., "LRED: a robust active queue management scheme based on packet loss ratio", In Proceedings of the INFOCOM, Hong Kong, March 2004.

[43] Welzl, M. and Mühlhäuser, M., "Scalability and Quality of Service: A trade-Off", IEEE Communications Magazine, June 2003.

[44] Wu, G. L. and Mark, J. W., "A Buffer allocation Scheme for ATM Networks: Complete Sharing Based on Virtual Partition", IEEE/ACM Transactions on Networking, vol. 3, Issue 6, December 1995.

[45] Lotker, Z. and Patt-Shamir, B., "Nearly Optimal FIFO buffer management for two packet classes", Computer Networks, vol.42, pp.481-492, 27 January 2003.

[46] Xiao, X. and Ni, L. M., "Internet QoS: A Big Picture", IEEE Network, pp.8–18, Mar/Apr. 1999.

[47] Yuzukirmizi, M. and Smith, J. M., "A customer threshold property for closed finite queuing networks", Performance Evaluation, vol.63, Issues 4-5, pp.463-492, May 2006.

[48] Zhou, Y. and Sethu, H., "On achieving fairness in the joint allocation of buffer and bandwidth resources: Principles and algorithms", Computer Networks, vol.50, Issue13, pp.2239-2254, 15 September 2006.

[49] Zhu, C., Yang, O. W. W., Aweya, J., Ouellette, M., "A Comparison of Active Queue Management Algorithms Using the OPNET Modeler", IEEE Communications Magazine, June 2002.

[50] Yaprak, E., Chronopoulos, A.T., Psarris, K., Xiao, Y., "Dynamic buffer allocation in an ATM switch", Computer Networks, vol.31, pp. 1927-1933,1999.

[51] Li, J. and Liu, C., "Improve TCP performance with a novel adaptive queue management scheme in IEEE 802.17 RPR optical networks", ISNN: 1389-1286, vol. 50, pp. 2762-2786, 2006.

[52] Lai, Y-C, "Two Schedulers to Provide Delay Proportion and Reduce Queuing Delay Simultaneously", Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 51, Issue11, pp. 3220-3231, August 2007.

[53] "Weighted round robin", URL: http://en.wikipedia.org/wiki/Weighted_round_robin , Wikipedia, the free encyclopaedia, (Last accessed 26 August 2006).

[54] Awan, I., Ahmad, S., and Ahmad, B., "Performance analysis of multimedia based web traffic with QoS constraint", Journal of Computer ar.d System Sciences, vol. 74, pp. 232–242, 2008.

Appendix

A-1. Glossary of Abbreviations used

QoS: Quality of Service

AQM: Active queue Management

CS: Complete Sharing scheme

CP: Complete Partitioning

PS: Partial sharing

DTBAS: Dynamic Threshold Buffer Allocation scheme

CBR: Constant Bit Rate traffic

PLR: Packet Loss Rate

DT: Drop Tail mechanism

sec: simulation seconds

A-2. Simulator User Interface

Figure A-1 shows the user interface of the simulator in a non-running mode. The "Configure Simulator" button on Interface 1 leads to the simulation Configuration centre (shown in Figure A-2), where simulation settings are done, with respect to experiment being conducted.

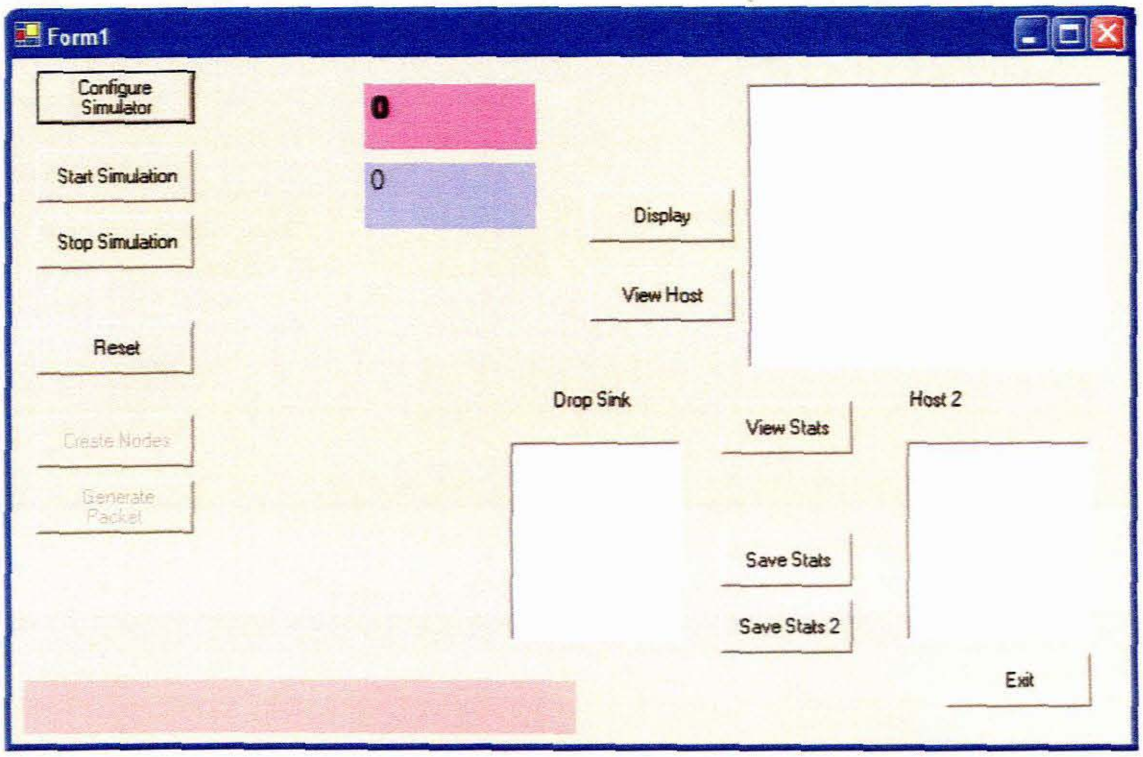
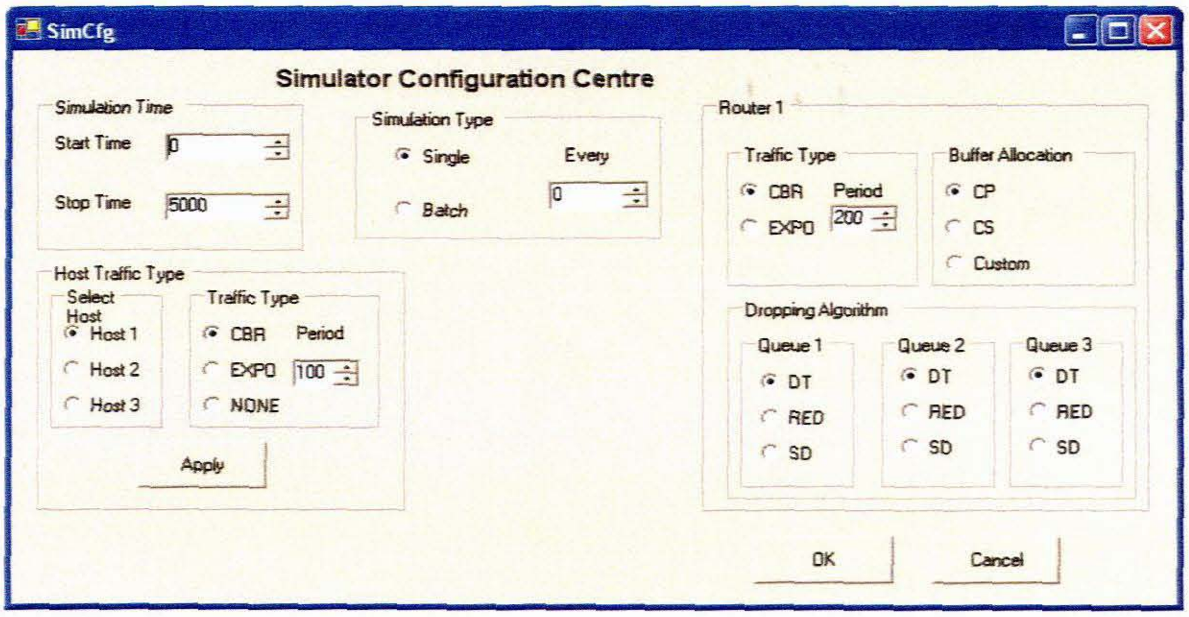


Figure A- 1: Simulation User Interface 1

In the simulation configuration centre, the user checks the appropriate check boxes for Host traffic speed and type, the simulation speed and type, and also the router traffic speed, type and the type of allocation used for the simulation. The user can also check the type of dropping strategy used to run the simulation.

In the Simulation user interface 1 (See Figure A-3), the start simulation button begins the simulation, and the results displayed in the display list box. An example of a running simulation is shown in Figure A-3 below where the simulation time is displayed on the middle text boxes.


SimCfg

Simulator Configuration Centre

Simulation Time: Start Time 0; Stop Time 5000

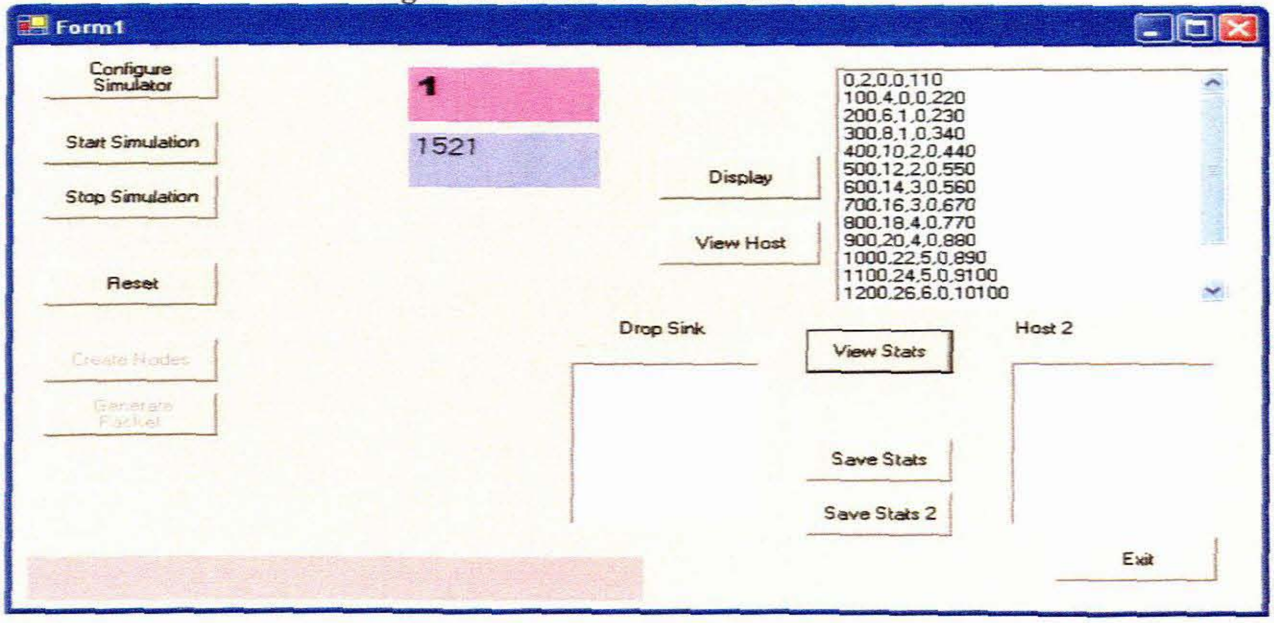
Simulation Type: [✓ Single] Batch; Every 0

Host Traffic Type: Select Host [✓ Host 1] Host 2 Host 3; Traffic Type [✓ CBR] EXPO NONE; Period 100

Router 1: Traffic Type [✓ CBR] EXPO; Period 200; Buffer Allocation [✓ CP] CS Custom

Dropping Algorithm: Queue 1 [✓ DT] RED SD; Queue 2 [✓ DT] RED SD; Queue 3 [✓ DT] RED SD

Figure A- 2: Simulation Interface 2


Form1

1

1521

```
0,2,0,0,110
100,4,0,0,220
200,6,1,0,230
300,8,1,0,340
400,10,2,0,440
500,12,2,0,550
600,14,3,0,560
700,16,3,0,670
800,18,4,0,770
900,20,4,0,880
1000,22,5,0,890
1100,24,5,0,9100
1200,26,6,0,10100
```

Figure A- 3: Simulator in a running mode