Proportional Deadline Violation Probability Differentiation Scheduling for Real-Time Traffic

by
Jie Zhu

A thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Masters of Applied Science
at the
Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
April 7, 2004
©Jie Zhu, 2004

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitons et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-93954-5

Our file Notre référence

ISBN: 0-612-93954-5

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this dissertation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de ce manuscrit.

While these forms may be included in the document page count, their removal does not represent any loss of content from the dissertation.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

Deadline Violation Probability (DVP) is the probability that the end-to-end delays of packets exceed a certain value and it is an important performance of real-time applications. One of the important issues in providing differentiated services is how to provide relative differentiated DVP service between different classes. In this thesis, we examine a proportional DVP differentiation model which provides controllable and predictable relative differentiated services for real-time traffic based on the DVP differentiation parameters. To realize such proportional DVP differentiation model, we present a new packet scheduling algorithm, called Adaptive Weighted Earliest Due Date (AWEDD), which improves the Weighted Earliest Due Date (WEDD) scheduling scheme such that the achieved DVP ratios between different classes approximate the target DVP ratios, independent of the aggregate traffic load and class load distributions in both long and short timescales. The simulation results show that AWEDD is capable of providing controllable and predictable services for classes with different load distributions under various load conditions. In particular, the AWEDD scheduler exceeds the WEDD scheduler in terms of adaptiveness and it is comparable to the WEDD scheduler in terms of convergence. The results also show that the AWEDD scheduler can efficiently support Connection Admission Control (CAC) to provide absolute QoS for real-time applications.

Acknowledgements

This thesis could not have been carried out without the help from a number of people, to whom I owe a great debt of gratitude and would like to thank for their contribution.

First and foremost, I would like to give my deepest thanks to my supervisor, Professor Jerome Talim, for his unconditional support, scientific guidance and confidence. The completion of this thesis project would have not been possible without his help. I would also like to thank Professor Changcheng Huang for granting me an enjoyable workplace.

I would like to thank Yan Robichaud and Mark Fransisco for their help. I am also grateful to my friends, Roy Zhang, Michael He, Wei Yi and Wenhong Ma, for their friendship, advice and encouragement. Also I would like to thank all the people I know at Carleton University, for making my stay more memorable.

Last but not least, I would like to thank my family for their emotional and financial support.

Contents

List of Figures	vii
List of Tables	viii
List of Symbols	ix
List of Acronyms	xiii
1 Introduction	1
1.1 Quality of service	1
1.2 Diffserv	6
1.3 Thesis objectives and contributions	9
1.4 Thesis organization	11
2 Problem Formulation	13
3 Review of the Packet Scheduling Algorithms for Proportional Differentiation	
Model	17
3.1 Static priority queueing	17
3.2 Weighted fair queueing	18

3.3	Mean delay proportional schedulers	20
3.4	Earliest due date scheduler	22
4	Adaptive Weighted Earliest Due Date Scheduling Scheme	24
4.1	Concept	24
4.1.1	Proportional DVP differentiation model	24
4.1.2	Weighted earliest due date scheduling	27
4.2	Algorithm	29
4.2.1	General description	30
4.2.2	Congestion indicator algorithm	32
4.2.3	DVP estimator algorithm	35
4.2.4	Packet selector algorithm	38
5	Simulation Evaluation	43
5.1	Voice traffic model	44
5.2	Simulation topology and parameters	46
5.3	Convergence of the AWEDD scheduler	48
5.3.1	Case A: convergence of the AWEDD scheduler under different system utilizations and effect of the aggregate load on the measurement window length L_{min}	49
5.3.2	Case B: convergence of the AWEDD scheduler under different load ratios and effect of the load ratio between classes on the measurement window length L_{min}	52

5.3.3	Case C: convergence of the AWEDD scheduler under different delay bounds and effect of the delay bounds of classes on the measurement window length L_{min}	55
5.3.4	Case D: convergence of the AWEDD scheduler under different target DVP ratios and effect of the target DVP ratios on the measurement window length L_{min}	57
5.4	Adaptiveness of the AWEDD scheduler	60
5.4.1	Short-term deviations under stationary traffic load distribution	61
5.4.2	Short-term deviations under non-stationary traffic load distribution . .	64
5.5	Examples of infeasible proportional DVP differentiation	68
5.6	From relative QoS to hard QoS	70
6	Conclusion	75
6.1	Conclusion	75
6.2	Future work	77

List of Figures

1.1	Nodal (per-hop) delay at router A	3
3.1	Priority functions with slopes $1/b_i$ and $1/b_j$ in the WTP scheduler	22
4.1	Queueing model	25
4.2	Proposed system architecture.	32
4.3	Deadline for packets	33
4.4	Structure of DVP History Table	37
5.1	Slow speech activity model	45
5.2	Simulation topology	47
5.3	Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different system utilizations, $\hat{r}_{1,2} = 4$	51
5.4	Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different load ratios, $\hat{r}_{1,2} = 4$	54
5.5	Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different delay bounds, $\hat{r}_{1,2} = 4$	56
5.6	Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different target DVP ratios, $\hat{r}_{1,2} = 4$ or 8	58

5.7	Summary of parameters and conclusions in Case A, B, C, and D	59
5.8	A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under stationary load distributions ($\tau = 50,000$ p-units), $\hat{r}_{1,2} = 2$	63
5.9	A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under stationary load distributions ($\tau = 100,000$ p-units), $\hat{r}_{1,2} = 2$	63
5.10	A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under nonstationary load distributions ($\tau = 50,000$ p-units), $\hat{r}_{1,2} = 2$	66
5.11	A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under nonstationary load distributions ($\tau = 100,000$ p-units), $\hat{r}_{1,2} = 2$	67
5.12	The behavior of AWEDD and WEDD when the proportional DVP differ- entiation model is at the onset of infeasibility	69
5.13	Simulation topology used in Experiment G	71
5.14	Achieved DVP experienced by packets in class-1 and class-2 vs. source-2 average throughput under different source-1 average throughput, $\hat{r}_{1,2} = 2$.	72

List of Tables

5.1	Parameters for slow speech activity model	46
5.2	Simulation parameters	47
5.3	Parameters used in Experiment A.1	50
5.4	Parameters used in Experiment A.2	50
5.5	Parameters used in Experiment B.2	53
5.6	Parameters used in Experiment B.3	53
5.7	Parameters used in Experiment C.2	55
5.8	Parameters used in Experiment D.2	57
5.9	Parameters used in Experiment E	62
5.10	Parameters used in Experiment F	65
5.11	Simulation parameter used in Experiment G	71

List of Symbols

d_{nodal}	Nodal delay
d_{proc}	Processing delay
d_{prop}	Propagation delay
d_{queue}	Queueing delay
d_{trans}	Transmission delay
$d_{end-to-end}$	End-to-end delay
C	Link capacity (total bandwidth)
c_i	Service rate for class i
D_{max}	End-to-end delay bound
D_{max}^i	Nodal delay bound at the i -th node
DVP_{max}	Deadline violation probability bound
DVP_{max}^i	Nodal deadline violation probability bound at the i -th node
Q_i	Local performance metric of class i

a_i	Generic quality differentiation parameter of class i
D_i	Nodal (local) delay bound assigned to the packets of class i
τ	Observation period
$P_i(t, t + \tau)$	Probability that D_i will be exceeded by packets at class i during the time interval $(t, t + \tau)$
$Z_{vio}^i(t, t + \tau)$	Number of total packets departed from class i with deadline violation during the time interval $(t, t + \tau)$
$Z_{total}^i(t, t + \tau)$	Number of total packets departed from class i during the time interval $(t, t + \tau)$
w_i	DVP differentiation parameter
$S_i(t_1, t_2)$	Amount of class i traffic served by the server during the time interval $(t_1, t_2]$ in the GPS scheduler
ϕ_i	Weight of class i in the GPS scheduler
g_i	Guaranteed minimum service rate of class i in the GPS scheduler
$\bar{d}_i$	Average queueing delay of the class i packets
b_i	Delay differentiation parameter
$q_i(t)$	Backlog of queue i at time t in the BPR scheduler
$y_i(t)$	Priority of the head-of-line packet of class i in the WTP scheduler
u	System utilization

t_A	Arrival epoch of the packet
λ_i	Average arrival rate of class i
x_i	Average packet size of class i
λ	Arrival rate of the aggregate traffic
$\hat{r}_{i,j}$	Target DVP ratio between class i and class j
$r_{i,j}(t, t + \tau)$	Achieved DVP ratio between class i and class i during the time interval $(t, t + \tau)$
t_s	Current system time
δ_i	Safety margin of class i in the AWEDD scheduler
$A(t_s)$	Set of backlogged classes at time t_s
$B(t_s)$	Set of backlogged classes in which the first packet meets the congestion condition in the AWEDD scheduler
$E_i(t_s)$	Current estimated DVP of class i at t_s
$\tilde{E}_i(t_s)$	Current normalized DVP of class i at t_s
$V_i(t_s)$	Number of total packets departed with deadline violation from class i during the time interval $(0, t_s)$
$X_i(t_s)$	Number of total packets departed from class i during the time interval $(0, t_s)$
L	Length of the measurement window in the AWEDD scheduler

$V_i(L)$	Number of total packets departed with deadline violation from class i in the measurement window of last L packets
$X_i(L)$	Number of total packets departed from class i in the measurement window of last L packets
$pk\ size_{max}$	Maximum packet size
$congestion$	Congestion indicator in the AWEDD scheduler
$tail$	Tail of the VHT in the AWEDD scheduler
$flag$	“Cold-start” phase indicator in the AWEDD scheduler

List of Acronyms

AWEDD	Adaptive Weighted Earliest Due Date
BPR	Backlog-Proportional Rate
CAC	Connection Admission Control
CBR	Constant Bit Rate
DDP	Delay Differentiation Parameter
DHT	DVP History Table
Diffserv	Differentiated Services
DSCP	Diffserv Code Point
DVP	Deadline Violation Probability
EDD	Earliest Due Date
EDF	Earliest Deadline First
FCFS	First-Come-First-Served
GPS	Generalized Processor Sharing

IETF	Internet Engineering Task Force
Intserv	Integrated Services
LHB	Loss History Buffer
PCM	Pulse-Code Modulation
PDD	Proportional Delay Differentiation
PHB	Per-Hop Behavior
PGPS	Packet-by-Packet Generalized Processor Sharing
QoS	Quality of Service
RSVP	Resource Reservation Setup Protocol
SAF	Speech Activity Factor
SLA	Service Level Agreement
SLS	Service Level Specification
TASI	Time Assignment Interpolation
UBAC	Utilization-Based Admission Control
WEDD	Weighted Earliest Due Date
WTP	Waiting-Time Priority
WFQ	Weighted Fair Queueing

Chapter 1

Introduction

1.1 Quality of service

Today's Internet provides the so called best-effort service which provides no guarantees for packet loss or delay bounds. For traditional non-real-time applications such as web browsing, e-mail or file transfer, the best-effort service is acceptable. However, as we move further into the age of multimedia communications, many delay sensitive applications, such as video conferencing and IP telephony, are being developed to the point where the best-effort service model cannot be adequate [2, 3]. To satisfy the real-time requirements of such applications, the Internet is changing from a best-effort service model to one that can provide differentiated Quality of Service (QoS) levels.

QoS requirements are application-specific. For example, real-time applications, such as IP telephony, are sensitive to end-to-end delays. When the delay of the packets exceeds certain levels (typically 150 to 300ms [4, 5, 24]), these packets become useless for the users. In contrast, a traditional reliable data transfer is more sensitive to the average transmission rate or the total amount of time for transferring the data. The following common parameters

have been widely used to describe QoS requirements [8]:

- Bandwidth

In a given time period, the capacity of the network to transmit data is limited by the physical media of communication links. Such limitation is defined as bandwidth and measured in terms of the number of bits transmitted in a time unit, such as bits/second.

- Jitter

In computer networks, the delay that each packet experiences is various due to the corresponding random end-to-end delay. The largest variation among all packets' end-to-end delays is defined as jitter.

- Loss Rate

Loss rate is the ratio of lost packets to the total packets transmitted. Packet losses are often caused by congestion.

- Delay

Delay is the total time from when a packet is generated at the source host until it is received at the destination host. As a packet travels from the source host to the destination host, the packet experiences several types of delays at each node. These delays are the nodal processing delay, queueing delay, transmission delay, and propagation delay. Then, end-to-end delay is the sum of processing, queueing and transmission delays in nodes and propagation delays in the link between nodes. Let us examine these different types of delays in the context of Figure 1.1.

Figure 1.1: Nodal (per-hop) delay at router A

For example, when the packet arrives at router A, it is stored in to the router A's buffer which is in front of the outbound link, router A tries to examine the packet's header to decide where to direct the packet, and then directs the packet to the appropriate link. In this scenario, router A sends the packet to the outbound link that is connected to router B.

- Processing delay

When the packet arrives at router A, router A tries to examine the packet's header in order to determine the appropriate outbound link for the packet. Meanwhile, router A also checks for the bit-level errors in the packet that occurred in transmission. This elapsed time is called the processing delay.

- Queueing delay

At the queue, the packet experiences a queueing delay, waiting for its transmission. The queue length decides the queueing delay of a specific packet. The queueing delay of a given packet can vary significantly from packet to packet.

- Transmission delay

The transmission delay is the size of a packet divided by the link rate. This is the amount of time required to transmit all of the packet's bits onto the link.

- Propagation delay

The propagation delay is the time required to propagate a packet in links. The physical medium of the link decides the propagation speed of the link. The propagation delay is the distance between routers divided by the propagation speed.

If we let d_{proc} , d_{queue} , d_{trans} , d_{prop} denote the processing, queueing, transmission, and propagation delays, then the total nodal delay is given by

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} \quad (1.1)$$

As we know, the processing delay d_{proc} can be the negligible term in d_{nodal} and it does not vary from packet to packet. If the packet's size is fixed, the transmission delay d_{trans} is also fixed. However, even though the packet size is fixed, the queueing delay d_{queue} can vary significantly from packet to packet. Therefore, d_{queue} is the most complicated term of nodal delays d_{nodal} .

Assume that there are $N - 1$ routers between the source host and the destination host. Let d_{proc} be the processing delay at each router and at the source host, d_{trans} be the transmission delay at each router and at the source host, and d_{prop} be the propagation delay. Therefore, the nodal delays and the propagation delays d_{prop} accumulate and give an end-

to-end delay,

$$d_{end-to-end} = N * (d_{proc} + d_{trans} + d_{prop}) + \sum_{i=1}^{N-1} d_{queue}^i \quad (1.2)$$

where d_{queue}^i is the queueing delay at the i -th router or source host.

Real-time applications require the total delay experienced by a packet in a network to be bounded on an end-to-end basis. In this context, the packet is required to be transmitted from its source host to the destination host within a predefined end-to-end delay bound. If the packet arrives at the destination host after its end-to-end delay bound, it may be useless or have significantly lost its value and the packet is called as deadline violation. Some of the forms may characterize the delay bounds requirement as follows [7, 22, 23]:

- *deterministic delay bound (deterministic services):*

$$d_{end-to-end}(k) \leq D_{max} \quad \text{for all } k \quad (1.3)$$

where $d_{end-to-end}(k)$ is the delay with respect to which the k -th packet sent by the user is delivered to the destination, and D_{max} is the end-to-end delay bound specified by the user.

- *statistical delay bound (statistical services):*

$$Prob(d_{end-to-end}(k) > D_{max}) \leq DVP_{max} \quad \text{for all } k \quad (1.4)$$

where $d_{end-to-end}(k)$ and D_{max} are defined as above, and DVP_{max} is the upper bound of the end-to-end deadline violation probability (DVP). D_{max} and DVP_{max} are specified by the user.

The Internet Engineering Task Force (IETF) has proposed many QoS architectures to meet the demand of QoS. Notably among them are the Integrated Services (Intserv) architecture [11] and the Differentiated Services (Diffserv) architecture [13, 14, 15].

Intserv is a flow-based resource reservation service, which is used to provide deterministic or statistical assurances related to performance measures. For real-time applications, before packets are transmitted, the applications must first set up paths and reserve resources using a signaling protocol, called Resource Reservation Setup Protocol (RSVP) [9, 10].

In order for the Intserv framework to provide QoS guarantee, each flow must be able to request and reserve the necessary resources in the network. Specifically, if a new flow requires QoS guarantee, in order to setup the path across the network for this flow, each router along the path must examine whether or not it has enough resources to provide the QoS requested by the new flow. The criterion for this examination is to provide the QoS for the new flow without sacrificing the already committed QoS of the existing flows. Meanwhile, the new flow must be able to provide each router along its path with its QoS requirement and the characteristic of the traffic it will be carrying.

However, in a large and complicated network such as Internet, this approach causes large amount of overhead. This per-flow resource reservation approach forces each backbone router to maintain the state information for each flow that passes through it. With the larger number of flows sharing the back-bone routers, the overhead will be significant.

1.2 Diffserv

In order to cover the problems of Intserv, Diffserv has been proposed, which only provides a limited number of service classes. It aggregates flows of the same service requirements, and serves the packets according to the predefined traffic agreement. Instead of achieving

per-flow QoS, Diffserv guarantees the class-based service qualities and makes itself more scalable and feasible.

The Diffserv architecture consists of two sets of functional components:

- Edge functions:

The network edge host must be a Diffserv-capable host that generates traffic or a Diffserv-capable router that forwards the traffic. Each arriving packet must be marked at the edge of the network to reflect its class so it can receive different treatment in the core network. The Diffserv-capable edge node marks the packet by setting the Diffserv Code Point (DSCP) of the packet according to the Service Level Agreement (SLA) [18] between the service provider and the user.

- Core functions:

When a Diffserv-marked packet arrives at a Diffserv-capable router, its marking reflects the class it belongs to and this packet is forwarded onto its next hop according to the Per-Hop Behavior (PHB) associated with that packet's class. When forwarding the packets, the routers' resources, such as its buffer and bandwidth, are divided among the competing classes according to classes' PHB.

Two types of QoS differentiation services have been defined as follows:

- Absolute Differentiation Service

Absolute Differentiation Service relies on admission control and resource reservation mechanism to give guarantees and statistical guarantees related to certain QoS measures, such as average throughput and end-to-end delay.

- Relative Differentiation Service

“In this approach, the network traffic is grouped into N classes of service which are ordered, such that class i is better (or at least no worse) than class $(i-1)$ for $1 < i \leq N$, in terms of local (per-hop) forwarding metrics” [20]. Although there is no general agreement on the most appropriate performance metrics for packet forwarding, it is been commonly accepted that a class performance metric is defined as “better” if it guarantees smaller delays, smaller probability of the packet losses, or smaller DVPs. In this relative Diffserv approach, classes do not get an absolute QoS guarantee since there is no admission control and resource reservations.

The relative service differentiation model can be classified into a few sub models [19].

- Capacity Differentiation

This sub model allocates more resources per unit load to class i than class $i - 1$ for $1 < i \leq N$. If the load of class is ρ_i and the weight is a_i , then the allocated bandwidth c_i satisfies $(c_i * a_i) / \rho_i > (c_{i-1} * a_{i-1}) / \rho_{i-1}$.

- (QoS) Additive Differentiation

For example, the delay of class i is less than that of class $(i - 1)$ by a certain constant. That is, $d_i = d_{i-1} + \theta_i$, where d_i is the delay of class i and θ_i is the constant.

- (QoS) Proportional Differentiation

In this sub model, the packet forwarding performance of classes is proportional to certain differentiation parameters. Thus, for example, the delays of class i and j satisfy the relationship $d_i/d_j = a_i/a_j$, where d_i is the delay of class i and a_i is the weight of class i .

1.3 Thesis objectives and contributions

In real-time applications, the maximum allowed end-to-end delay is bounded by an application-specified value, called the end-to-end delay bound. If a packet arrives after its delay bound, the received information is discarded and considered lost. For example, for IP telephony, the end-to-end delay should be approximately 150ms. While a lower delay cannot really be appreciated, the delay above this threshold will be noticed by the users and will become a hindrance to interactivity. The performance of real-time applications does not depend on the average delay much, but on the probability that the end-to-end delay exceeds the delay bound.

Under the Diffserv model, traffic flows are aggregated and identified as classes. Rather than providing end-to-end performance guarantees for individual flows like Intserv, the Diffserv model objective is to differentiate among classes of traffic using *per-hop* packet forwarding behaviors. Suppose that there are $N - 1$ routers between the source host and the destination host. As noted above, the nodal delays accumulate to give an end-to-end delay $d_{end-to-end}$. To satisfy the end-to-end delay bound D_{max} , it is sufficient (though not necessary) to satisfy the bound D_{max}^i (or D_{max}^i, DVP_{max}^i) in each node along the source-to-destination path, where D_{max}^i is the nodal delay bound at the i -th node, and DVP_{max}^i

is the upper bound of the nodal DVP at the i -th node. For simplicity, we assume that to satisfy this sufficient condition is the goal to be met by each node along the path. That is,

$$d_{nodal}(k) \leq D_{max}^i \quad \text{for all } k, 1 \leq i \leq N \quad (1.5)$$

or

$$Prob(d_{nodal}(k) > D_{max}^i) \leq DVP_{max}^i \quad \text{for all } k, 1 \leq i \leq N \quad (1.6)$$

where $d_{nodal}(k)$ is the nodal delay with respect to the k -th packet at the i -th node, D_{max}^i is the nodal delay bound at the i -th node, and DVP_{max}^i is upper bound of the nodal DVP at the i -th node.

As an approach to a Diffserv network, the relative Diffserv network only offers local (per-hop) relative QoS differentiation between a number of classes. In this context, applications and users do not get an absolute service level assurance, such as deterministic delay bound in Equation 1.5 or a statistical delay bound in Equation 1.6, since there is no admission control and resource reservations. Instead, the relative Diffserv network assures that higher classes will offer better QoS (i.e., lower probability of delay violations) than the lower classes, and so the users can select the class that best meets their requirements and costs. The proportional differentiation model, which is a further refinement of the relative Diffserv approach, states that the QoS obtained by classes are ratioed proportionally to certain differentiation parameters set by the network administrator [20]. The objective of this thesis is to achieve proportional DVP guarantees in the context of the relative differentiation model.

As we know, one of the most important tools for the provision of proportional differentiation model is scheduling. The packet scheduler decides the order in which packets are

sent to the output link, and in turn determines the queueing delay experienced by a packet at each router along the path. The basic purpose of scheduling is to allow sharing of common resources in a controlled way. There exist some scheduling schemes in order to achieve proportional bandwidth (Weighted Fair Queueing [28, 29] and others) and proportional delay (Waiting Time Priority [25, 33]). Meanwhile, close attention has been devoted to meet the requirement of proportional differentiation service model in terms of DVP. In this thesis, we attempt to guarantee proportional DVP for the real-time traffic using an adaptive scheduling scheme.

Starting from the Weighted Earliest Due Date (WEDD) algorithm, proposed by Stefan Bodamer in [1], we develop an improved packet scheduling algorithm called Adaptive Weighted Earliest Due Date (AWEDD), which enhances the WEDD service discipline such that the ratio of deadline violation probabilities between classes can be fixed to certain values, independent of the aggregate load or the class load distribution in short as well as in long timescales. This mechanism is easy to implement and efficiently supports Connection Admission Control (CAC) to provide absolute QoS for real-time applications.

1.4 Thesis organization

The rest of the thesis concentrates on describing a scheduling mechanism for the real-time applications that provides proportional DVP differentiation in an appropriate manner. In chapter 2, we present the proportional differentiation model in terms of DVP and describe the packet scheduling problem in details. In chapter 3, we describe several previous scheduling mechanisms, explaining why they are not suitable for proportional DVP differentiation services. In chapter 4, we start to analyze a related scheduling algorithm WEDD and then propose our scheduling scheme AWEDD. Chapter 5 evaluates and compares those

two schedulers, WEDD and AWEDD. Finally, we conclude with a discussion of future work in chapter 6.

Chapter 2

Problem Formulation

The *proportional differentiation model* [25], as a further refinement of the relative differentiated services approach, has generated much interest in the research community. The goal of the proportional differentiation model is to appropriately “*space*” local (per-hop) class performance metrics according to the differentiation parameters chosen by the network operator [15]. Consider N classes and let the local performance metric of a class i be denoted by Q_i . The proportional differentiation model imposes a constraint for class i such that:

$$\frac{Q_i}{Q_j} = \frac{a_i}{a_j} \quad 1 \leq i, j \leq N \quad (2.1)$$

where a_i is defined as the generic quality differentiation parameter. Through these quality differentiation parameters, the network administrator can control the relative quality spacing (ratio) of the offered classes based on pricing or policy requirements.

Recently, real-time multimedia applications such as IP telephony and Video on demand over Internet are becoming more and more popular. In these real-time applications, the performance does not depend on the average delay much, but on the probability that the

end-to-end delay exceeds the delay bound. If we consider the DVP as the QoS performance metric, the proportional differentiation model can be represented as the *proportional DVP differentiation model*. The proportional DVP differentiation model controls the ratios of DVPs between classes based on the specified DVP differentiation parameters.

Suppose that a network provides N classes and D_i is the delay bound assigned to the class i ($1 \leq i \leq N$). Let $Z_{total}^i(t, t + \tau)$ be the total number of packets departed from the class i during the time interval $(t, t + \tau)$ and $Z_{vio}^i(t, t + \tau)$ be the number of packets departed from class i with deadline violation during time interval $(t, t + \tau)$. Then the probability that D_i will be exceeded for packets in class i during the time interval $(t, t + \tau)$, say $P_i(t, t + \tau)$, is defined to be the ratio of $Z_{vio}^i(t, t + \tau)$ to $Z_{total}^i(t, t + \tau)$ [1], that is:

$$P_i(t, t + \tau) = \frac{Z_{vio}^i(t, t + \tau)}{Z_{total}^i(t, t + \tau)} \quad 1 \leq i \leq N \quad (2.2)$$

The idea of the proportional DVP differentiation model can be stated as:

$$\frac{P_i(t, t + \tau)}{P_j(t, t + \tau)} = \frac{w_i}{w_j} \quad 1 \leq i, j \leq N \quad (2.3)$$

where $P_i(t, t + \tau)$ is the DVP obtained by class i during the time interval $(t, t + \tau)$, and w_i is the *DVP differentiation parameter* of class i and it is defined in the traffic contract. τ is a configurable parameter which denotes an observation period.

The proportional DVP differentiation model provides an appropriate per-hop behavior in terms of DVP due to the following two reasons [25, 26]:

- First, the proportional DVP differentiation model is *controllable*, since the proportional DVP model provides the DVP differentiation parameters which allow the network administrator to provision the performance differentiation between classes.

Since each class is differentiated by their pricing and policy objective, such tuning capability is indispensable to network administrators since the administrators must be able to adjust the individual treatment corresponding to each class.

- Secondly, the proportional DVP differentiation model is *predictable*, since the relative DVP differentiation service is consistent in both long and short timescales with respect to its DVP differentiation parameter and it is not affected by the traffic load conditions. Specifically, even with strongly nonstationary traffic load distribution, such as Internet traffic which is dynamic and unpredictable, the proportional DVP differentiation still exhibits predictability. The provision of DVP differentiation service is more important in short timescales than in long timescales since a majority of real-time applications on the Internet is small in size [5, 28].

As we know, the *packet scheduler* determines which classes should be served next, so as to achieve the necessary DVP differentiation between classes. Therefore, the aim of our work is to find out how to design a scheduler which can provide controllable and predictable service over DVP ratios between classes of traffic in various load conditions and class load distributions. We can examine the proportional DVP differentiation scheduler from two aspects:

- We grade the *controllability* of a scheduler by studying whether or not the network administrator can adjust the DVP ratios between classes based on class differentiation parameters.
- We grade the *predictability* of a scheduler by studying if the performance between different classes should be consistent with their DVP differentiation parameters in both long and short time scales, independent of the aggregate load or the class load

distribution. We consider a proportional DVP differentiation scheduler *adaptive* if the achieved DVP ratios have negligible deviations from the target ratios of the DVP differentiation parameters in short timescales. In contrast, we consider a proportional DVP differentiation scheduler *convergent* if the achieved DVP ratios closely approximate the target ratios of DVP differentiation parameters in long timescales.

On the other hand, the proportional DVP differentiation scheduler should have low complexity. The time complexity of choosing the next packet to transmit should be small, and the scheduling algorithm should be amenable to an efficient hardware implementation.

Chapter 3

Review of the Packet Scheduling Algorithms for Proportional Differentiation Model

In this chapter, we review several existing packet scheduling algorithms for differentiation model, and examine them from the perspective of relative DVP differentiated services.

3.1 Static priority queueing

A number of existing packet scheduling schemes for Diffserv are represented in recent literatures. Among these schemes, the static priority queueing[27] is a straightforward and simple implementation. In a priority queueing scheme, the arriving packets are queued in one of N queues, before being sent to the link. Each queue is assigned with a priority. In some other schemes, the priorities can be dynamically changed with the cost of complexity. However, in the static priority queueing scheme, such priorities are fixed once they are

assigned. In this scheme, packets in queue with higher priority are transmitted prior to their counterparts in queues with lower priorities, i.e., packets in some a queue are not transmitted until all queues with higher priorities are empty.

Unfortunately, the benefits of simplicity is not obtained without costs. The disadvantages of static priority queueing scheme are as follows:

- The packets in queues with lower priorities may have no chance to be transmitted if there is a heavy load of high-priority packets. Consequently, a congestion of lower-priority packets may occur.
- The performance of differentiation between classes is seriously affected by the queue lengths, which is determined by the load distribution and ratio. This influence may result in performance variation due to the load fluctuation.
- Under this scheme, the network administrator cannot control the class differentiation service, such as DVP ratios defined in chapter 2.

3.2 Weighted fair queueing

Several packet schedulers aim at providing each class with a minimum bandwidth share of the link capacity. The most popular one is the Weighted Fair Queueing (WFQ) [28, 29, 30, 31, 32] queueing discipline, also known as Packet-by-Packet Generalized Processor Sharing (PGPS), which is packet-based approximation of Generalized Processor Sharing (GPS) fluid-flow service model[27].

A GPS scheduler is a work-conserving scheduler that serves the incoming packets in each class with a fixed rate. Each class i has a weight ϕ_i . Let $S_i(t_1, t_2)$ be the amount of

traffic served in class i during the time interval $(t_1, t_2]$. Then, a GPS scheduler is characterized by:

$$\frac{S_i(t_1, t_2)}{S_j(t_1, t_2)} \geq \frac{\phi_i}{\phi_j} \quad (3.1)$$

for any pair of class i and j that are continuously backlogged during the interval $(t_1, t_2]$. Because the service rate of each class i during a busy period also satisfies the work-conservation constraint, if $A(t)$ is the set of backlogged class at time t and the total bandwidth is C , then each class $i \in A(t)$ is served at the service rate of

$$c_i(t) = \frac{\phi_i}{\sum_{j \in A(t)} \phi_j} C \quad (3.2)$$

Therefore, a class i is guaranteed to be served at a minimum rate of:

$$g_i = \frac{\phi_i}{\sum_{j=1}^N \phi_j} C \quad (3.3)$$

where N is the maximum number of classes that are being serviced by the GPS scheduler.

Since the GPS scheduler treats the wordload in a fluid manner, the wordload of different classes can be served in parallel. However, in a realistic packet scheduler, only one class can be served at a time. Moreover, the service of a packet cannot be interrupted unless it is finished. Many approximation techniques are proposed to emulate the idealized GPS scheduler. The most popular one is the WFQ, also known as PGPS, in which when the scheduler is ready to serve the next packet, it selects the first packet that would be served completely under the corresponding GPS scheduler.

In WFQ scheduler, if the weight of a traffic class is fixed, that class can reserve a fixed portion of link bandwidth for it. Increasing the weight of a class generally results in better performance with respect to DVP. However, it is a complex task to find appropriate values

for the weights even in an ideal GPS scheduler in order to achieve DVP differentiation model. This is due to the fact that the distributions of load on different classes are unknown in a Diffserv network. On the other hand, since the traffic on the Internet is fluctuant in nature and a short term burst will increase DVP even for a class with a high and fixed weight, WFQ may not suitable to implement proportional DVP differentiation service model in terms of predictability, as defined in chapter 2.

A solution could be an adjustment of the weights based on on-line traffic measurements. A low frequent adjustment, however, may result in lower performance of high priority classes in short timescales. On the other hand, if the frequency of the adjustment is very high, the algorithm no longer has much in common with the fair queueing [1].

3.3 Mean delay proportional schedulers

Recently, several schedulers [20, 25, 33] have been proposed that implement the Proportional Delay Differentiation (PDD). The PDD model requires that the ratios of the mean delays for N classes are fixed to the ratio of their Delay Differentiation Parameters (DDPs) called b_i . Specifically, let $\bar{d}_i$ be the mean delay of the class i packets. The objective of the PDD model is to control the ratio of mean delays between any pair of class i and j according to the ratio of the corresponding DDPs:

$$\frac{\bar{d}_i}{\bar{d}_j} = \frac{b_i}{b_j} \quad 1 \leq i, j \leq N \quad (3.4)$$

Assume that higher classes get better services, i.e., smaller delays, than lower classes, the DDPs satisfy the relation $b_1 > b_2 > \dots > b_N > 0$.

In [20, 25], the authors proposed two scheduling algorithms to approximate the PDD

model under heavy-load conditions in terms of delay differentiation. The first one is called the Backlog-Proportional Rate (BPR) scheduler. Let $c_i(t)$ be the service rate of class i at time t . If class i is empty at time t , we set $c_i(t) = 0$. On the other hand, for two backlogged classes i and j , the allocation of service rate c_i in BPR scheduler has the constraint as follows:

$$\frac{c_i(t)}{c_j(t)} = \frac{b_i q_i(t)}{b_j q_j(t)} \quad (3.5)$$

where $q_i(t)$ is the backlog of class i at time t and b_i is the the DDP of class i . Moreover, the service rate of each class i during a busy period also satisfies the work-conservation constraint:

$$\sum_{i=1}^N c_i(t) = C \quad (3.6)$$

where C is the total capacity.

The second algorithm, called the Waiting-Time Priority (WTP) [25, 34] scheduler, is proved to give a relatively exact proportional differentiation in terms of the mean delay under heavy-load conditions in both short and long timescales. The basic idea of the WTP scheduler is to control the ratios of individual packets' delays in order to achieve the target ratios of mean delays in long timescales between different classes. The WTP scheduler assigns a DDP b_i to each class i , which represents the increasing rate of the packet's priority. For example, the priority of the heading packet in class i arrived at time t_0 , increases with the rate $1/b_i$ (see Figure 3.1):

$$y_i(t) = (t - t_0)/b_i \quad (3.7)$$

Let $A(t_s)$ be the set of backlogged classes at time t_s and assume that a packet has to be selected to be transmitted at time t_s . The packet selector chooses the backlogged class j in

Figure 3.1: Priority functions with slopes $1/b_i$ and $1/b_j$ in the WTP scheduler

which the heading packet has the maximum priority,

$$j = \arg \max_{i \in A(t_s)} y_i(t) \quad (3.8)$$

where t_A is the arrival epoch of the heading packet in class i and D_i is the delay bound of class i .

The disadvantage of these schedulers is that they are designed to provide differentiation in terms of mean delays. The performance of real-time applications such as IP telephony do not depend so much on the mean delays but on the DVP [22].

3.4 Earliest due date scheduler

In general, all deadline-based scheduling policy requires a packet to be served before its deadline, which is assigned at its arrival. Various deadline-based scheduling policies such as *Earliest Due Date (EDD)* [31, 35, 37, 48] and *Delay-EDD* [23] have been proposed. They differ in how deadlines are calculated and assigned for each arrived packet, and which

packet is selected to be served.

The EDD schedule, also known as Earliest Deadline First (EDF) is a scheduling scheme used to differentiate services in terms of the absolute delay. Each traffic class i is assigned a pre-defined QoS parameter called a delay bound D_i . Each packet in class i is assigned a deadline $t_A + D_i$, where t_A is the arrival epoch of the packet and D_i is its delay bound. Let $A(t_s)$ be the set of backlogged classes at time t_s and assume that a packet has to be selected to be transmitted at time t_s . The packet selector chooses the backlogged class j in which the heading packet has the minimal deadlines,

$$j = \arg \min_{i \in A(t_s)} (t_A + D_i) \quad (3.9)$$

where t_A is the arrival epoch of the heading packet in class i and D_i is the delay bound of class i .

In a single-node scenario, the EDD scheduler is proved to be the optimal scheduling policy, in terms of the schedulable region combined with the scheduling policy [37]. More exactly, EDD is worst-case optimal because if any scheduling scheme can meet a set for deadlines with certainty, then the EDD scheduler can meet these deadlines with certainty [36, 43]. This property is independent of the total traffic load and the load ratio of different classes.

The disadvantage of these schedulers is that the EDD scheduler is uncontrollable as it lacks any tuning capability which could help the network administrator adjust the performance spacing between classes.

Chapter 4

Adaptive Weighted Earliest Due Date Scheduling Scheme

4.1 Concept

In this section, we define proportional DVP differentiation model and then review the Weighted Earliest Due Date (WEDD) algorithm [1].

4.1.1 Proportional DVP differentiation model

Before discussing the proportional DVP differentiation scheduling algorithm, let us define the notations for the proportional DVP differentiation model. In the proportional DVP differentiation model, we consider a general queueing system $G/G/1$ [21], in which packet arrival distribution and service times (packet sizes) are arbitrary. The packets are classified into N classes, each of which has a dedicated queue i , $1 \leq i \leq N$. Let λ_i denote the average arrival rate of class i and x_i denote the average packet size of class i . The arrival

Figure 4.1: Queueing model

rate of the aggregate traffic is $\lambda = \sum_{i=1}^N \lambda_i$. Let C be the capacity of the link and u be the system utilization: $u = \sum_{i=1}^N \lambda_i x_i / C$.

The proportional DVP differentiation model consists of a lossless, work-conserving, and non-preemptive link with capacity C and N First-Come-First-Served (FCFS) queues. Depending on the arrival packet's class, the packet is sent to the corresponding queue of its class before it is transmitted. The packet scheduler S determines the order of which queue is served. In order for the lossless assumption to be valid, the average packet arrival rate must be less than the link capacity C and each queue is large enough to handle its backlog packets at any time. The work-conserving assumption implies that as long as there is at least one packet left in any of the queues, the scheduler S will not be in the idle state. The non-preemptive assumption means that once the transmission of the any packets begins, the transmission cannot be interrupted until it is done. The proportional DVP differentiation model is depicted in 4.1.

For simplicity, we do not consider processing delay d_{proc} in this thesis. Thus, Equation 1.1 becomes to:

$$d_{nodal} = d_{queue} + d_{trans} \quad (4.1)$$

where d_{queue} is the queueing delay and d_{trans} is the transmission delay. If the nodal delay of the packet is larger than the nodal delay-bound D_i assigned to class i , that is $d_{nodal} > D_i$, the packet is considered as deadline violation.

The scheduler S aims at ensuring that the DVP experienced by packets in each class is proportional to its DVP differentiation parameter. Suppose that a network provides N service classes, the following equation should be satisfied for all pairs of class i and j :

$$\frac{P_i(t, t + \tau)}{P_j(t, t + \tau)} = \frac{w_i}{w_j} \quad 1 \leq i, j \leq N \quad (4.2)$$

where $P_i(t, t + \tau)$ is the DVP experienced by packets in class i during the time interval $(t, t + \tau)$, and w_i is the DVP differentiation parameter of class i and it is defined in the traffic contract. τ is a configurable parameter which denotes an observation period.

Assume that higher class gets better service, i.e., lower DVPs, and so we set $w_1 > w_2 > \dots > w_N > 0$. The goal of the proportional DVP differentiation scheduler S is that the normalized DVP, defined as $\tilde{P}_i(t, t + \tau) = P_i(t, t + \tau)/w_i$, must be equal for all classes, that is,

$$\tilde{P}_i(t, t + \tau) = \frac{P_i(t, t + \tau)}{w_i} = \frac{P_j(t, t + \tau)}{w_j} = \tilde{P}_j(t, t + \tau) \quad i = 2, \dots, N \quad (4.3)$$

independent of the aggregate traffic load and the class load distribution in short as well as in long time scales.

Another way to interpret the proportional DVP differentiation model of Equation 4.3 is: let $\hat{r}_{i,j}$ be the target DVP ratio between class i and j :

$$\hat{r}_{i,j} = \frac{w_i}{w_j} \quad 1 \leq i, j \leq N \quad (4.4)$$

and $r_{i,j}(t, t + \tau)$ be the achieved DVP ratio between class i and j during the time interval $(t, t + \tau)$:

$$r_{i,j}(t, t + \tau) = \frac{P_i(t, t + \tau)}{P_j(t, t + \tau)} \quad 1 \leq i, j \leq N \quad (4.5)$$

In other words, the goal of proportional DVP differentiation scheduler S is to make the achieved DVP ratio $r_{i,j}(t, t + \tau)$ equal to the target DVP ratio $\hat{r}_{i,j}$, i.e.,

$$r_{i,j}(t, t + \tau) = \hat{r}_{i,j} \quad 1 \leq i, j \leq N \quad (4.6)$$

where τ is the observation period. Adjusting the value of τ , we can observe the behavior of scheduler S in terms of long timescales and short timescales.

4.1.2 Weighted earliest due date scheduling

The WEDD algorithm was proposed by S. Bodamer in 2000 [1]. In this scheme, the scheduler is responsible for scheduling, i.e. deciding which of the N queues to be served next, and it can schedule only one packet at a time. Each queue corresponds to a different class, and each traffic class i is associated with a predefined QoS parameter called the delay bound D_i . Each packet in class i is assigned a deadline $t_A + D_i$, where t_A is the arrival epoch of the packet and D_i is the delay bound of class i . The packets are queued in a non-decreasing order based on their deadlines. If the packet arrived at time t_A and departed from the class i after the deadline $t_A + D_i$, the packet violates its deadline.

However, if the first packet in class i has a deadline $t_A + D_i$ and it is not transmitted before time $t_A + D_i - \delta_i$, that is $t_s > t_A + D_i - \delta_i$ where t_s denotes the current system time and δ_i is a safety margin of class i . The condition $t_s > t_A + D_i - \delta_i$ is defined as congestion condition. If there is more than one backlogged class satisfying the congestion condition,

the system is called to be in “congestion mode”.

In congestion mode, let $B(t_s)$ be the set of backlogged classes in which the first packet meets the congestion condition. Suppose that class i is backlogged at time t_s , and that $E_i(t_s)$ is the current estimated DVP of class i at t_s . We define the estimated *normalized DVP* $\tilde{E}_i(t_s)$ at t_s for class i in $B(t_s)$ as:

$$\tilde{E}_i(t_s) = \frac{E_i(t_s)}{w_i} \quad (4.7)$$

Every time a packet is to be transmitted, the WEDD scheduler selects the backlogged class j with the *maximum normalized DVP* from $B(t_s)$:

$$j = \arg \max_{i \in B(t_s)} \tilde{E}_i(t_s) \quad (4.8)$$

How to estimate DVP of each class is a key design issue in the WEDD scheduler. In order to estimate the DVP of class i , the WEDD scheduler maintains two counters for class i :

- $X_i(t_s)$ is used to count the number of total packets departed from class i during the time interval $(0, t_s)$;
- $V_i(t_s)$ is used to count the number of total packets departed with deadline violation from class i during the time interval $(0, t_s)$.

The estimated DVP of class i is then given by the ratio of the current values of V_i and X_i . Therefore, the estimated DVP of class i at time t_s is :

$$E_i(t_s) = \frac{V_i(t_s)}{X_i(t_s)} \quad (4.9)$$

The WEDD scheduler aims at ensuring that the achieved DVP ratio $r_{i,j}(t, t + \tau)$ can

closely approximate the target DVP ratio $\hat{r}_{i,j}$ in long timescales and short timescales. As a proportional DVP differentiation scheduler, WEDD should be examined in terms of controllability and predictability, which are described in chapter 2. Furthermore, the predictability of the WEDD scheduler is graded from two aspects: convergence and adaptiveness. We consider the WEDD scheduler *convergent* if long-term DVP ratios $r_{i,j}(t, t + \tau)$ can closely approximate the target DVP ratios $\hat{r}_{i,j}$ under different traffic conditions and *adaptive* if short-term target DVP ratios $r_{i,j}(t, t + \tau)$ have negligible variations, especially under non-stationary traffic load distributions.

In this thesis, we will show that the WEDD scheduler is controllable and closely approximates the target DVP ratios $\hat{r}_{i,j}$ between different classes in long timescales under different traffic load conditions. Unfortunately, changing traffic loads and load distributions (non-stationary traffic load distributions) between different classes can cause the achieved DVP ratios $r_{i,j}(t, t + \tau)$ to deviate from the target DVP ratios $\hat{r}_{i,j}$ in short timescales because the WEDD estimates the DVP of class i based on an entire history of previous departures. In general, *the WEDD scheduler gives controllable and convergent but no adaptive proportional DVP differentiation service.*

Another important disadvantage about the WEDD scheduler is the length of the counters D_i and V_i . Since the counters are required to record the entire history of previous departures, they will overflow after a certain number of departures.

4.2 Algorithm

In this section, we first introduce our new Adaptive Weighted Earliest Due Date (AWEDD) algorithm that provides tunable DVP differentiated services in an appropriate manner for real-time applications and then describe our scheduling algorithm in details. We also dis-

cuss some design issues of the algorithm.

4.2.1 General description

Two important disadvantages arise in WEDD. First, the counters will overflow after a certain number of departures. Second, changing traffic loads between different classes can cause achieved DVP ratios $r_{i,j}(t, t + \tau)$ to deviate from the target DVP ratios $\hat{r}_{i,j}$ in short timescales. Both disadvantages in WEDD are caused by the fact that WEDD estimates the DVP of class i based on an entire history of previous departures. Thus, the aim of our work is to find out how to estimate DVP of each class in order to provide controllable and predictable DVP differentiation and avoid the disadvantages arised in WEDD as well.

The basic idea in the AWEDD scheduler is to estimate the DVP of class i as the fraction of departures with deadline violation from class i in the last L departures. To do this, the AWEDD scheduler maintains a measurement window to keep last L departures in consideration. When a new packet departures from class i , the DVP estimator “slides” the measurement window to keep last L departures. Meanwhile, the AWEDD scheduler maintains two counters for traffic class i :

- $V_i(L)$ is the total number of packets departed with deadline violation from class i in the measurement window of last L departures;
- $X_i(L)$ is the total number of packets departed from class i in the measurement window of last L departures.

Therefore, the estimated DVP $E_i(t_s)$ of class i at time t_s is:

$$E_i(t_s) = \frac{V_i(L)}{X_i(L)} \quad (4.10)$$

where L is the measurement window length (L is a positive integer) and t_s is current system time. When L is infinity, AWEDD becomes equivalent to WEDD. Note that AWEDD scheduler is different from the WEDD scheduler since the AWEDD scheduler attempts to achieve the proportional DVP differentiation in successive measurement window of L departures and the WEDD scheduler attempts to achieve the proportional DVP differentiation for all departures.

The method of choosing the measurement window length L is an important design issue. An appropriate window length L should strike a good balance between adaptiveness and convergence.

- On one hand, the measurement window length L should be large enough so that at least one packet with deadline violation of each class is always one of the last L departed packets; Otherwise, the estimated DVP will be zero for classes at which no packet departs with deadline violation in last L departures. This leads to inaccurate DVP estimation which in turn causes the achieved DVP ratios to deviate from the target DVP ratios. In other words, if L is not large enough, the AWEDD scheduler cannot remember enough previous departures with deadline violation in order to schedule packets to meet DVP differentiation parameters.
- On the other hand, a large L makes the AWEDD scheduler less adaptive to traffic load fluctuation. If the measurement window length L is sufficiently large, the DVP of the packets in the window would hide the effect of bursty traffic. A small L enables a fast adaptation to the traffic fluctuation.

In chapter 5, we will study the effect of the measurement window length L on the performance of AWEDD.

Figure 4.2: Proposed system architecture.

The AWEDD scheduler can be divided into three modules as depicted in Figure 4.2. First, the *congestion indicator* checks if the system is in congestion mode. Second, the *DVP estimator* estimates the DVP of class i based on the number of departures and the number of departures with deadline violation from class i in the measurement window of last L departures. Third, the *packet selector* decides the class from which the heading packet should be sent to the link when the link becomes idle.

In the next three subsections, we describe the congestion indicator, the DVP estimator and the packet selector, discuss their properties and compare these with properties of the WEDD scheduler.

4.2.2 Congestion indicator algorithm

In the congestion indicator, the definition of class congested and the method for indicating the system congestion mode are both adopted from [1]. Each traffic class i is assigned a pre-defined QoS parameter called a delay bound D_i and a safety margin δ_i . Each packet in

Figure 4.3: Deadline for packets

class i is assigned a deadline $t_A + D_i$, where t_A is the arrival epoch of the packet and D_i is its delay bound. If the packet arrived at t_A and departed from class i after the deadline $t_A + D_i$, the packet is called deadline violation as depicted in Figure 4.3.

In the congestion indicator, we set δ_i equal to the transmission time of the packet with the maximum size, that is

$$\delta_i = \frac{pk\ size_{max}}{C} \quad (4.11)$$

where $pk\ size_{max}$ is the maximum packet size and C is the capacity of the link. If the first packet in class i has a deadline $t_A + D_i$ and it is not transmitted before time $t_A + D_i - \delta_i$, that is $t_s > t_A + D_i - \delta_i$ where t_s denotes the current system time, this means that the first packet in class i is about to violate its deadline and the class i is congested. The condition $t_s > t_A + D_i - \delta_i$ is defined as *congestion condition*. If there is more than one backlogged class meeting the congestion condition, the system is called to be in “congestion mode”. The congestion indicator uses a variable called *congestion* to show that if the system is in “congestion mode”. If *congestion* equals to 1, that means the system is in congestion mode, in which $B(t_s)$ is the set of backlogged classes meeting the congestion condition; Otherwise, *congestion* equals to 0. The complete algorithm is shown in Algorithm 1.

Algorithm 1 The algorithm of the congestion indicator in the AWEDD scheduler

D_i : the delay bound of class i ;

t_s : the current system time;

t_A : the arrival time of the first packet in class i ;

δ_i : the safety margin of class i ;

congestion: the congestion indicator;

$A(t_s)$: the set of backlogged classes;

$B(t_s)$: the set of backlogged classes meeting the congestion condition ($t_s > t_A + D_i - \delta_i$);

The link is idle:

congestion = 0;

for ($i \in A(t_s)$) {

 if ($t_s > (t_A + D_i - \delta_i)$) {

 /* the first packet in class i is about to violate its deadline and class i is congested

 put i into $B(t_s)$;

 }

}

if (the number of classes in $B(t_s) \geq 2$) {

 /* the system is in congestion mode

congestion = 1;

}

4.2.3 DVP estimator algorithm

In the AWEDD scheduler, a DVP estimator is proposed, which estimates the DVP by counting the number of departures with deadline violation in each class in the measurement window of last L departures. The DVP estimator involves two basic functions:

- measuring the number of departures and the number of departures with deadline violation from class i in the measurement window of last L departures;
- estimating the DVP of traffic class i based on the number of departures and the number of departures with deadline violation from class i in the measurement window of last L departures.

The DVP estimator maintains two counters for traffic class i at time t_s : $V_i(L)$ and $X_i(L)$.

The estimated DVP $E_i(t_s)$ of class i at t_s is:

$$E_i(t_s) = \frac{V_i(L)}{X_i(L)} \quad (4.12)$$

In the context of the AWEDD scheduler, in order to maintain the counters $X_i(L)$ and $V_i(L)$, the DVP estimator is required to record the deadline violation status and the class index of the packets in the measurement window of last L departures. Since the estimated DVP $E_i(t_s)$ of traffic class i is determined only by last L departures in the measurement window, the DVP estimator “slides” the measurement window to keep last L departures when a new packet departs from class i . The operation of sliding measurement window consists of two steps:

- deleting the packet which has become last $(L + 1)$ th departure from the measurement window;

- inserting the new packet which just departed from class i into the measurement window.

Consequently, the deadline violation status and the class index of the corresponding packets have to be updated in response to the sliding operation of the measurement window.

A critical issue in the DVP estimator is how to update efficiently the deadline violation status and class index of the packets in the measurement window of last L departures so as to calculate estimated DVP $E_i(t_s)$ of class i without causing too much overhead. The simplest approach is to use a cyclic queue called *DVP History Table (VHT)* in Figure 4.4 to record the deadline violation status and the class index of the packets in the measurement of last L departures. The DVP estimator maintains a VHT with L entries, each corresponding to one packet in the measurement window of last L departures. In each entry j ($j \in \{1, \dots, L\}$), there are two elements:

- $VHT[j].class$ ($VHT[j].class \in \{1, \dots, N\}$) is the class index of the packet;
- $VHT[j].violation$ ($VHT[j].violation \in \{0, 1\}$) records the deadline violation status of the packet. If the packet departs after its deadline, that is, $t_A + D_i < t_s$, we set the corresponding $VHT[j].violation$ to 1; Otherwise, we set the corresponding $VHT[j].violation$ to 0.

Based on the information maintained by the VHT, the DVP estimator knows the total number of departures $X_i(L)$ and the number of departures with deadline violation $V_i(L)$ from class i in the measurement window of last L departures and then calculates the estimated DVP $E_i(t_s)$ of traffic class i .

In the AWEDD scheme, the VHT is updated when a new packet departs from class i . The DVP estimator maintains two control variables:

Figure 4.4: Structure of DVP History Table

- $tail$ ($tail \in \{1, \dots, L\}$) is used to record the tail of the VHT so that the corresponding VHT entry is updated when a new packet departs from class i ;
- $flag$ ($flag \in \{0, 1\}$) is used to indicate if the VHT is full. At the beginning of the measurement window, no packet departed and the VHT is empty. In contrast, the VHT becomes full after L departures. If the total number of departures is not more than L , the system is considered to be in “cold-start” phase.

At the beginning of the measurement window, the control variables are initialized, that is, $tail = 0$ and $flag = 0$, and the system is in “cold-start” phase. In this phase, if a new packet departs from class i , the DVP estimator simply records the deadline violation status and the class index of the new departure in the VHT entry pointed by $tail$ and modifies the value of two counters $X_i(L)$ and $V_i(L)$. The “cold-start” phase is completed after L departures. From now on, if a new packet departs from class i , before the DVP estimator records the deadline violation status and the class index of packets and modifies the value of two counters $X_i(L)$ and $V_i(L)$, the DVP estimator has to modify the value of two counters

$X_k(L)$ and $V_k(L)$, where k is the class index of the packet corresponding to the VHT entry pointed by $tail$. The complete algorithm of the DVP estimator is shown in Algorithm 2.

The VHT and the control variables, $tail$ and $flag$, are the main additional implementation complexity of AWEDD as compared to WEDD. Update of the VHT and two control variables incurs non-negligible overhead. We argue that this overhead is small because such update mostly involves add operations. However, the calculation of $tail$ involves divide operations. But if the measurement window length is chosen in terms of base 2, that is, $L = 2^x$ where x is a positive integer, it means that the divide operations with L can be replaced with shift operations.

4.2.4 Packet selector algorithm

When the link is idle, the packet selector determines the class from which the heading packet should be sent to the link. We recall here that the packet selector can schedule only one packet at a time.

There are two distinct operations in the packet selector corresponding to the mode of the system indicated by the variable *congestion* which is maintained by the congestion indicator. If *congestion* equals to 0, it means that the system is not in “congestion mode” so the operation of the packet selector is the same as EDD. Let $A(t_s)$ be the set of backlogged classes at time t_s and assume that a packet has to be selected to be transmitted at time t_s . The packet selector chooses the backlogged class j in which the heading packet has the minimal deadlines,

$$j = \arg \min_{i \in A(t_s)} (t_A + D_i) \quad (4.13)$$

where t_A is the arrival epoch of the heading packet in class i and D_i is the delay bound of class i .

Algorithm 2 The algorithm of the DVP estimator in the AWEDD scheduler

VHT: Delay Violation History Table (cyclical queue with L entries);

t_A : the packet arrival time;

D_i : the delay bound of class i ;

t_s : the current system time;

tail: tail index of VHT;

flag: the “cold-start” indicator;

$X_i(L)$: Counter of packets from class i in last L departures;

$V_i(L)$: Counter for packets with deadline violation from class i in last L departures;

$E_i(t_s)$: the estimated DVP of class i ;

Packet departed in class i :

```
if (flag == 1) {
     $k = VHT[tail].class$ ;
     $X_k(L) --$ ;
     $V_k(L) - = VHT[tail].violation$ ;
}
 $X_i(L) ++$ ;
 $VHT[tail].class = i$ ;
if ( $t_A + D_i > t_s$ ) {
    /* the packet departed from class  $i$  without deadline violation
     $VHT[tail].violation = 0$ ;
} else {
    /* the packet departed from class  $i$  with deadline violation
     $VHT[tail].violation = 1$ ;
     $V_i(L) ++$ ;
}
 $E_i(t_s) = \frac{V_i(L)}{X_i(L)}$ ;
if (flag == 1) {
     $tail = (tail + 1) \bmod L$ ;
} else {
     $tail = tail + 1$ ;
    if ( $tail > L$ ) {
         $tail = tail - L$ ;
        flag = 1;
    }
}
```

If *congestion* equals to 1, the system is in “congestion mode” and $B(t_s)$ is the set of backlogged classes meeting the congestion condition as defined in the congestion indicator. Let $E_i(t_s)$ be the estimated DVP of class i at time t_s and w_i be the DVP differentiation parameter of class i that is defined in the traffic contract. Therefore, the estimated normalized DVP $\tilde{E}_i(t_s)$ of class i in $B(t_s)$ at time t_s is :

$$\tilde{E}_i(t_s) = \frac{E_i(t_s)}{w_i} \quad (4.14)$$

It has to be noted that the estimated normalized DVP $\tilde{E}_i(t_s)$ of class i in AWEDD is different from that in WEDD because only the last L departures are considered in AWEDD whereas the entire history of departures is considered in WEDD. Assume that a packet has to be selected to be transmitted at time t_s . The packet selector chooses the backlogged class j with the maximum normalized DVP at t_s from $B(t_s)$:

$$j = \arg \max_{i \in B(t_s)} \tilde{E}_i(t_s) \quad (4.15)$$

The packet at the head of class j is transmitted. The complete algorithm is shown in Algorithm 3.

The basic idea in the packet selector in “congestion mode” is that serving a packet from class j with the maximum normalized DVP tends to reduce the difference of $\tilde{E}_i(t_s)$ from the normalized DVP of the other classes. In the long-run, if the scheduler always minimizes the differences between the normalized DVP in this manner, we expect that the normalized DVP will be about the same.

Note that there are two distinct versions of AWEDD that we call *drop-AWEDD* and *non-drop-AWEDD*.

Algorithm 3 The algorithm of the packet selector in the AWEDD scheduler

t_s : the current system time;
 δ_i : the safety margin of class i ;
 t_A : the packet arrival time;
 D_i : the delay bound of class i ;
 $A(t_s)$: the set of backlogged classes;
 $B(t_s)$: the set of backlogged classes meeting the condition $(t_A + D_i < t_s + \delta_i)$;
congestion: the congestion indicator;
 w_i : the DVP differentiation parameter of class i ;
 $E_i(t_s)$: the estimated DVP of class i ;
 $\tilde{E}_i(t_s)$: the estimated normalized DVP of class i ;
 j : the class in which the heading packet will be transmitted next;

The link is idle:

```
if (congestion == 0) {  
    /* the system is not in "congestion mode"  
     $j = \arg \min_{i \in A(t_s)} (t_A + D_i)$ ;  
}  
if (congestion == 1) {  
    /* the system is in "congestion mode"  
     $\tilde{E}_i(t_s) = \frac{E_i(t_s)}{w_i}$ ;  
     $j = \arg \max_{i \in B(t_s)} \tilde{E}_i(t_s)$ ;  
}
```

- Drop-AWEDD drops a packet as soon as it passes a deadline. Specifically, if the deadline of the first packet in the class is smaller than the current time, that is $t_A + D_i < t_s$, this packet is dropped and the next packet in this class is checked in the same way.
- Non-drop-AWEDD does not drop packets although this deadline has already passed, that is, $t_A + D_i < t_s$.

As we know, one important issue related with network integration is end-to-end delay. The QoS perceived by a real-time application user, however, depends on loss and delay occurring on the end-to-end path. Probably, in most cases congestion will occur only in single node on the end-to-end path. Therefore, non-drop-AWEDD is adopted in chapter 5 to show that the performance of AWEDD is comparable to WEDD in long timescales and more adaptive than WEDD to traffic load fluctuation in short timescales.

Chapter 5

Simulation Evaluation

As we described in chapter 4, there are two requirements for the proportional DVP differentiation schedulers: *controllability* and *predictability*. The AWEDD scheduler and the WEDD scheduler can provide *controllable* proportional DVP differentiation, as the network administrator can adjust the target DVP ratio $\hat{r}_{i,j}$ ($\hat{r}_{i,j} = w_i/w_j$) based on the DVP differentiation parameters w_i . Therefore, a critical issue in the thesis is to evaluate the predictability of the AWEDD scheduler and the WEDD scheduler, i.e, to study if

$$r_{i,j}(t, t + \tau) = \hat{r}_{i,j} \quad (5.1)$$

can be achieved in both long timescales and short timescales, where $r_{i,j}(t, t + \tau)$ is the achieved DVP ratio between class i and class j traffic during a time interval $(t, t + \tau)$.

We can grade predictability of the schedulers from two aspects: *convergence* and *adaptiveness*. Convergence means that $r_{i,j}(t, t + \tau)$ can closely approximate $\hat{r}_{i,j}$ in long timescales regardless of traffic load conditions. Adaptiveness means that $r_{i,j}(t, t + \tau)$ in short timescales has negligible deviations from $\hat{r}_{i,j}$ under stationary load distributions and non-stationary

load distributions. In the context of Internet services, provisioning of proportional DVP differentiation service in short timescales is more important than in long timescales, since a number of real-time applications on the Internet is small in size [5, 6].

In this chapter, we start with an introduction of the traffic model and simulation topology used in the thesis. Also, we examine the AWEDD scheduler behavior in terms of convergence and adaptiveness, and compare it with the WEDD scheduler using simulation results. The metric of the achieved DVP ratios $r_{i,j}(t, t + \tau)$ in long timescales under various traffic load conditions provides a basis for the evaluating the convergence of the schedulers. The metric of the achieved DVP ratios $r_{i,j}(t, t + \tau)$ in short timescales, especially under non-stationary traffic distributions, provides the basis for the evaluating the adaptiveness of the schedulers. From the results which will be presented in the following, it is shown that the AWEDD scheduler exceeds the WEDD scheduler in terms of adaptiveness and is comparable to the WEDD scheduler in terms of convergence.

5.1 Voice traffic model

Voice traffic is bursty since with a conversation usually consists of talk-spurts and gaps [39, 41]. The normal conversation pattern of talk, pause, and listen creates principle-spurts and principle-gaps in voice traffic; Similarly, the short pauses in a continuous speech (to punctuate the speech) create mini-spurts *and* mini-gaps. As a result, two models have been proposed to model the process of speech generation as follows:

- The “slow” speech activity model corresponding to the principle talk-spurts and gaps [38].
- The more sensitive “fast” speech activity model corresponding to the mini-spurts and

Figure 5.1: Slow speech activity model

mini-gaps [43].

Only the slow speech activity model is used in this thesis. This model, illustrated in Figure 5.1, consists of a two-state discrete Markov process with exponential distribution time. If the mean duration of a silent gap is t_2 seconds, the probability of a transition from the silent state to the talking state during a τ seconds time slot is [42]:

$$\alpha = 1 - \exp(-\tau/t_2) \quad (5.2)$$

Correspondingly, if the mean duration of the talk-spurt is t_1 seconds, the probability of the transition from the talking state to the silent state during a τ seconds time slot is [42]:

$$\beta = 1 - \exp(-\tau/t_1) \quad (5.3)$$

Mean values of talk-spurt and silent gap durations, as reported in [42], are given in Table 5.1. The corresponding Mean Speech Activity factor(SAF), which is the percentage of time

Conditions	Values(s)
Principle Talk-spurt, t_1	1.004
Principle Gap, t_2	1.587

Table 5.1: Parameters for slow speech activity model

that a single speech source is active, is given by

$$SAF = \frac{t_1}{t_1 + t_2} = 0.387 \quad (5.4)$$

During the talk-spurt period, constant bit rate (CBR) [42] data is generated, and no data is transmitted during the silence period. In this thesis, within each talk-spurt intervals, 64Kbps PCM (Pulse-Code Modulation)-coded [40] digit voice scheme is assumed in which the packetization period is chosen as $\theta = 16ms$. Therefore, the packet size is 1024bits.

5.2 Simulation topology and parameters

An event-driven simulator is developed to evaluate and compare the performance of two schedulers, WEDD and AWEDD. Figure 5.2 shows the simulation topology adopted in the following experiments in which two classes are considered. The packets in this experiment are created by two groups of packet generators (group-1 and group-2) dedicated to class-1 and class-2, respectively. We assume that class-1 has a lower priority, i.e, higher DVP, and class-2 has a higher priority. In both groups, the packet generators use the voice traffic model to generate packets. Let N_1 be the number of packet generators in group-1 and N_2 be the number of packet generators in group-2. Hence, the load ratio between class-1 and class-2 is: N_1/N_2 . By adjust the value of N_1 and N_2 , we can get different traffic load conditions, such as different system utilizations and different traffic loads. The packets in

Figure 5.2: Simulation topology

packet size	link speed	packet service time d_{trans}	buffer size
1024bits	2000Kbps	0.5ms	∞

Table 5.2: Simulation parameters

each class are characterized by a delay bound based on the input parameter to the simulator.

We use the following simulation parameters depicted in Table 5.2. The link speed is set to 2000Kbits/s and all packets have a fixed size of 1024bits. The packet service time (transmission delay d_{trans}) on the link is therefore 0.5ms. The nodal delay consists of the packet service time and the queueing time, i.e. the time that the packet spends waiting in a buffer. Buffers are infinitely large and no packets are dropped in each experiment.

The duration of the simulations is 10800s (3 hours) in each case, meaning that at the link of 2000Kbps the scheduler can serve approximately 21 million packets in each experiment.

In the following, we examine different aspects of the AWEDD scheduler behavior and compare it with WEDD using simulation results.

5.3 Convergence of the AWEDD scheduler

Convergence means that the achieved DVP ratios $r_{i,j}(t, t + \tau)$ can closely approximate the target DVP ratios $\hat{r}_{i,j}$ regardless of traffic load conditions in long timescales. As we know, the convergence of the AWEDD scheduler depends on the measurement window length L . In order to make the AWEDD scheduler convergent, the measurement window length L is set large enough in order for the AWEDD scheduler to estimate the DVP of each class accurately. Intuitively, the larger the measurement window length L , the higher the chance of achieving the target DVP ratios $\hat{r}_{i,j}$ in long timescales. Two important questions arise as follows:

- How large is the measurement window length L to make AWEDD convergent? The lower bound of required measurement window length is called L_{min} by which the AWEDD scheduler can meet the target DVP ratios $\hat{r}_{i,j}$ within a certain error tolerance. Is it *necessary* that the measurement window length is at least L_{min} in order for the AWEDD scheduler to meet the proportional DVP differentiation constraints?
- How is L_{min} in the AWEDD scheduler, affected by traffic load conditions and service requirements?

In this section, we evaluate the convergence of the AWEDD scheduler and also investigate the measurement window length L_{min} under different traffic conditions and service requirements. We classify the experiments into four cases, A, B, C, and D, for which the goals are:

Case A: Evaluate the convergence of the AWEDD scheduler under different system utilizations and illustrate the effect of the system utilization on the measurement window

length L_{min} . In order to do it, we vary the system utilization u and compare the long term achieved DVP ratios $r_{i,j}(t, t + \tau)$ under different measurement window length L .

Case B: Evaluate the convergence of the AWEDD scheduler under different load ratios and illustrate the effect of the load ratio between classes on the measurement window length L_{min} . In order to do it, we vary the load ratio between classes and compare the long term achieved DVP ratios $r_{i,j}(t, t + \tau)$ under different measurement window length L .

Case C: Evaluate the convergence of the AWEDD scheduler under different delay bounds and illustrate the effect the delay bounds of classes on the measurement window length L_{min} . In order to do it, we vary the delay bounds and compare the long term achieved DVP ratios $r_{i,j}(t, t + \tau)$ under different measurement window length L .

Case D: Evaluate the convergence of the AWEDD scheduler under different target DVP ratios and illustrate the effect of the target DVP ratio $\hat{r}_{i,j}$ on the measurement window length L_{min} . In order to do it, we vary the target DVP ratio $\hat{r}_{i,j}$ and compare the long term achieved DVP ratios $r_{i,j}(t, t + \tau)$ under different measurement window length L .

5.3.1 Case A: convergence of the AWEDD scheduler under different system utilizations and effect of the aggregate load on the measurement window length L_{min}

In this subsection, we report results from two experiments.

Experiment A.1: The parameters in Experiment A.1 are summarized in Table 5.3. The delay bounds of class-1 and class-2 are 0.3s. The DVP differentiation parameters of class-1 and class-2 are: $w_1 = 1$ and $w_2 = 0.25$, respectively. The number of users in group-1 is: $N_1 = 40$, and the number of users in group-2 is: $N_2 = 40$. This implies that the system

N_1	N_2	w_1	w_2	D_1	D_2
40	40	1	0.25	0.3s	0.3s

Table 5.3: Parameters used in Experiment A.1

N_1	N_2	w_1	w_2	D_1	D_2
38	38	1	0.25	0.3s	0.3s

Table 5.4: Parameters used in Experiment A.2

utilization is

$$u = \frac{(N_1 + N_2) * SAF * pk\ size}{C * \theta} = 0.97 \quad (5.5)$$

The load ratio between class-1 and class-2 is: $N_1/N_2 = 1$. The target DVP ratio between class-1 and class-2 is: $\hat{r}_{1,2} = w_1/w_2 = 4$. We measure the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2 in long timescales, where $\tau = 3hours$.

Experiment A.2: The experiment setting is the same as Experiment A.1, except that the number of users in group-1 is: $N_1 = 38$, and the number of users in group-2 is: $N_2 = 38$. This implies that the system utilization becomes to

$$u = \frac{(N_1 + N_2) * SAF * pk\ size}{C * \theta} = 0.92 \quad (5.6)$$

The parameters used in Experiment A.2 are summarized in Table 5.4.

In these two experiments, the length of the measurement window length L is set to $32K$, $64K$, $128K$, $256K$, and $512K$ packets. To provide a comparison between AWEDD and WEDD, the achieved DVP ratios with the WEDD scheduler are also shown as the case WEDD($L = \infty$).

The experimental results are summarized in Figure 5.3 in which the x-axis is the mea-

Figure 5.3: Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different system utilizations, $\hat{r}_{1,2} = 4$

surement window length L while the y-axis is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2. From Figure 5.3, we make two observations:

1. Regardless of the system utilization, the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ approaches the target DVP ratio $\hat{r}_{1,2}$ as we increase the measurement window length L , i.e., the larger the measurement window length L , the higher the chance of achieving the target DVP ratio $\hat{r}_{1,2}$ in long timescales. The reason is that the DVP estimation is more robust when it is based on a longer history of previous departures. As the measurement window length L increases to infinity (AWEDD becomes the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ can be exactly achieved in Experiment A.1. However, as the measurement window length L increases to infinity (AWEDD becomes to the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ cannot be achieved exactly in Experiment A.2 due to the feasibility of proportional DVP differentiation model, which will be explained in section 5.5.

2. The required measurement window length L_{min} to meet the target DVP ratio $\hat{r}_{1,2}$ within a certain error tolerance increases as the system utilization decreases. For example, if $r_{1,2}(t, t + \tau)$ is required to be larger than 90% of the target DVP ratio ($\hat{r}_{1,2} = 4$), the measurement window length L_{min} in AWEDD should be $64K$ packets in the case where the system utilization u is 97%. However, when the system utilization is 92%, the measurement window length L_{min} increases to $512K$ packets. The reason is that the lower system utilization, the lower the chance of departures with deadline violation. When the chance of departures with deadline violation is lower, the scheduler needs a longer measurement window length L to estimate the DVP of each class accurately in order to approximate the target DVP ratio $\hat{r}_{1,2}$.

5.3.2 Case B: convergence of the AWEDD scheduler under different load ratios and effect of the load ratio between classes on the measurement window length L_{min}

In this subsection, we report results from three experiments.

Experiment B.1: The experiment setting is the same as Experiment A.1. The load ratio between class-1 and class-2 is: $N_1/N_2 = 1$.

Experiment B.2: The experiment setting is the same as Experiment B.1, except that the number of users in group-1 is: $N_1 = 16$, and the number of users in group-2 is: $N_2 = 64$. This implies that the load ratio between class-1 and class-2 is: $N_1/N_2 = 0.25$. The parameters used in Experiment B.2 are summarized in Table 5.5.

Experiment B.3: The experiment setting is the same as Experiment B.1, except that the number of users in group-1 is: $N_1 = 64$, and the number of users in group-2 is: $N_2 = 16$. This implies that the load ratio between class-1 and class-2 is: $N_1/N_2 = 4$. The parameters

N_1	N_2	w_1	w_2	D_1	D_2
16	64	1	0.25	0.3s	0.3s

Table 5.5: Parameters used in Experiment B.2

N_1	N_2	w_1	w_2	D_1	D_2
64	16	1	0.25	0.3s	0.3s

Table 5.6: Parameters used in Experiment B.3

used in Experiment B.3 are summarized in Table 5.6.

In these three experiments, the length of the measurement window length L is set to $32K$, $64K$, $128K$, $256K$, and $512K$ packets. To provide a comparison between AWEDD and WEDD, the achieved DVP ratios with the WEDD scheduler are also shown as the case WEDD($L = \infty$).

The experimental results are summarized in Figure 5.4 in which the x-axis is the measurement window length L while the y-axis is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2. From Figure 5.4, we make two observations:

1. Regardless of the load ratio between classes, the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ approaches the target DVP ratio $\hat{r}_{1,2}$ as we increase the measurement window length L . As the length of the measurement window length L increases to infinity (AWEDD becomes the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ can be exactly achieved in all three experiments.
2. The required measurement window length L_{min} for meeting the target DVP ratio $\hat{r}_{1,2}$ within a certain error tolerance increases as the load ratio between classes diverges from 1. For example, if $r_{1,2}(t, t + \tau)$ is required to be larger than 90% of the target DVP ratio ($\hat{r}_{1,2} = 4$), the measurement window length L_{min} in AWEDD should be

Figure 5.4: Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different load ratios, $\hat{r}_{1,2} = 4$

64K packets in the case where the load ratio between class-1 and class-2 is: $N_1/N_2 = 1$. However, when the load ratio between classes diverges from 1, such as $N_1/N_2 = 0.25$ or 4, the measurement window length L_{min} increases to 256K packets. The reason for that behavior is as follows. For example, in the case of two classes with load ratio $N_1/N_2 = 0.25$, the DVPs are ratioed based on the last L departures, in which class-2 has four times higher rate than class-1. Consequently, only one fifth of last L packets departed from class-1 such that the DVP of class-1 is estimated inaccurately.

N_1	N_2	w_1	w_2	D_1	D_2
40	40	1	0.25	2.0s	1.0s

Table 5.7: Parameters used in Experiment C.2

5.3.3 Case C: convergence of the AWEDD scheduler under different delay bounds and effect of the delay bounds of classes on the measurement window length L_{min}

In this subsection, we report results from three experiments.

Experiment C.1: The experiment setting is the same as Experiment A.1. The delay bounds of class-1 and class-2 are 0.3s.

Experiment C.2: The experiment setting is the same as Experiment C.1, except that the delay bound of class-1 increases to 2.0s, and the delay bound of class-2 is 1.0s. The parameters used in Experiment C.2 are summarized in Table 5.7.

In these two experiments, the length of the measurement window length L is set to 32K, 64K, 128K, 256K, and 512K packets. The achieved DVP ratios with the WEDD scheduler are also shown, as the asymptotic case $L = \infty$, providing a comparison between the two schedulers.

The experimental results are summarized in Figure 5.5 in which the x-axis is the measurement window length L while the y-axis is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2. From Figure 5.5, we make two observations:

1. Regardless of the delay bounds, the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ approaches the target DVP ratio $\hat{r}_{1,2}$ as we increase the measurement window length L . As the length of the measurement window length L increases to infinity (AWEDD becomes the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ can be exactly achieved

Figure 5.5: Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different delay bounds, $\hat{r}_{1,2} = 4$

in Experiment C.1. However, as the measurement window length L increases to infinity (AWEDD becomes to the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ cannot be achieved exactly in Experiment C.2 due to the feasibility of proportional DVP differentiation model, which will be explained in section 5.5.

2. The required measurement window length L_{min} for meeting the target DVP ratio $\hat{r}_{1,2}$ within a certain error tolerance increases as the delay bounds of classes increases. For example, if $r_{1,2}(t, t + \tau)$ is required to be larger than 90% of the target DVP ratio ($\hat{r}_{1,2} = 4$), the measurement window length L_{min} in AWEDD should be 64K packets in the case where the delay bounds of class-1 and class-2 are 0.3s. However, when the delay bounds of class-1 and class-2 increase to 2.0s and 1.0s respectively, the measurement window length L_{min} increases to 256K packets. The reason behind this is that as the delay bounds increase, the chance of departures with deadline violation

N_1	N_2	w_1	w_2	D_1	D_2
40	40	1	0.125	0.3s	0.3s

Table 5.8: Parameters used in Experiment D.2

decreases. When the chance of departures with delay violation is lower, the scheduler needs a longer measurement window length L to estimate the DVP of each class accurately in order to approximate the target DVP ratio $\hat{r}_{1,2}$.

5.3.4 Case D: convergence of the AWEDD scheduler under different target DVP ratios and effect of the target DVP ratios on the measurement window length L_{min}

In this subsection, we report results from three experiments.

Experiment D.1: The experiment setting is the same as Experiment A.1. The target DVP ratio is: $\hat{r}_{1,2} = 4$.

Experiment D.2: The experiment setting is the same as Experiment D.1, except that the target DVP ratio changes to: $\hat{r}_{1,2} = 8$. The parameters used in Experiment D.2 are summarized in Table 5.8.

In these two experiments, the length of the measurement window length L is set to $32K$, $64K$, $128K$, $256K$, and $512K$ packets. The achieved DVP ratios with the WEDD scheduler are also shown, as the asymptotic case $L = \infty$, providing a comparison between the two schedulers.

The experimental results are summarized in Figure 5.6 where the x-axis is the measurement window length L while the y-axis is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2. From Figure 5.6, we make two observations:

Figure 5.6: Achieved DVP ratio $r_{1,2}$ vs. measurement window length L under different target DVP ratios, $\hat{r}_{1,2} = 4$ or 8

1. Regardless of the target DVP ratio $\hat{r}_{1,2}$, the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ approaches asymptotically the target DVP ratio $\hat{r}_{1,2}$ as we increase the measurement window length L . As the length of the measurement window length L increases to infinity (AWEDD becomes the WEDD scheduler), the target DVP ratio $\hat{r}_{1,2}$ can be exactly achieved in both experiments.
2. The required measurement window length L_{min} for meeting the target DVP ratio $\hat{r}_{1,2}$ within a certain error tolerance increases as the target DVP ratio $\hat{r}_{i,j}$ increases. For example, if $r_{1,2}(t, t + \tau)$ is required to be larger than 90% of the target DVP ratio ($\hat{r}_{1,2} = 4$ or 8), the measurement window length L_{min} in AWEDD should be $64K$ packets in the case where the target DVP ratio is: $\hat{r}_{1,2} = 4$. However, when the target DVP ratio $\hat{r}_{1,2}$ increase to 8 , the measurement window length L_{min} increases to $256K$ packets.

		Parameters					Conclusions	
		N_1	N_2	w_1	w_2	$D_1(s)$		$D_2(s)$
Case A	Experiment A.1	40	40	1	0.25	0.3	0.3	<ul style="list-style-type: none">Regardless of the system utilization, $r_{1,2}(t, t + \tau)$ approaches $\hat{r}_{1,2}$ as L increases.The required $L_{\min}$ increases as the system utilization decreases.
	Experiment A.2	38	38	1	0.25	0.3	0.3	
Case B	Experiment B.1	40	40	1	0.25	0.3	0.3	<ul style="list-style-type: none">Regardless of the load ratio between classes, $r_{1,2}(t, t + \tau)$ approaches $\hat{r}_{1,2}$ as L increases.The required $L_{\min}$ increases as the load ratio between classes diverges from 1.
	Experiment B.2	16	64	1	0.25	0.3	0.3	
	Experiment B.3	64	16	1	0.25	0.3	0.3	
Case C	Experiment C.1	40	40	1	0.25	0.3	0.3	<ul style="list-style-type: none">Regardless of the delay bounds of classes, $r_{1,2}(t, t + \tau)$ approaches $\hat{r}_{1,2}$ as L increases.The required $L_{\min}$ increases as the delay bounds increase.
	Experiment C.2	40	40	1	0.25	2.0	1.0	
Case D	Experiment D.1	40	40	1	0.25	0.3	0.3	<ul style="list-style-type: none">Regardless of the target DVP ratio $\hat{r}_{1,2}$, $r_{1,2}(t, t + \tau)$ approaches $\hat{r}_{1,2}$ as L increases.The required $L_{\min}$ increases as the target DVP ratio $\hat{r}_{1,2}$ increases.
	Experiment D.2	40	40	1	0.125	0.3	0.3	

Figure 5.7: Summary of parameters and conclusions in Case A, B, C, and D

The parameters and conclusions in Case A, B, C, and D are summarized in Figure 5.7. The conclusions from the presented results show that, the performance of the AWEDD scheduler is comparable to the WEDD scheduler in terms of convergence under different combinations of traffic load conditions, DVP differentiation parameters and delay bound requirements as long as the measurement window length L of the AWEDD scheduler is chosen large enough. In other words, the performance of the AWEDD scheduler improves in terms of convergence with the increase of the measurement window length L . This is because with the increase of the measurement window length L , the estimated DVP is more accurate since it is based on longer sequences of packet departures. It is necessary that the measurement window length L is at least L_{min} in order for the AWEDD scheduler to meet the proportional DVP differentiation constraints within a certain error tolerance. Furthermore, the required measurement window length L_{min} of the AWEDD scheduler for meeting the target DVP ratios within a certain error tolerance increases:

1. as the system utilization decreases,
2. as the load ratio between classes diverges from 1,
3. as the delay bounds of classes increase, or
4. as the ratio of the DVP differentiation parameter increases.

5.4 Adaptiveness of the AWEDD scheduler

A good scheduler for proportional DVP differentiation model should not only achieve or closely approximate the target DVP ratio $\hat{r}_{i,j}$ in long timescales, but it also should have negligible deviations from $\hat{r}_{i,j}$ in short timescales. In this section, we focus on the adaptiveness

of the AWEDD scheduler and the WEDD scheduler. In evaluating the adaptiveness of the AWEDD scheduler and the WEDD scheduler, the key issue is to examine the short-term deviations from $\hat{r}_{i,j}$ experienced by both schedulers not only under stationary traffic load distributions, but also under non-stationary traffic load distributions. Since the Internet traffic is dynamic and unpredictable, we should pay more attention on short-term behaviors of the schedulers under non-stationary traffic load distributions.

We first give an outline of the experiments carried out. In the first experiment E.1, we study the behavior of two schedulers, AWEDD and WEDD, under stationary load distribution in short timescales. In the second experiment E.2, we study the short-term behaviors of these two schedulers under non-stationary load distributions.

5.4.1 Short-term deviations under stationary traffic load distribution

Experiment E: In this experiment, we want to find out the achieved DVP ratios between class-1 and class-2 $r_{1,2}(t, t+\tau)$ within a *fixed time interval* (or what we call the observation period τ) under stationary load distribution. In order to do it, we measure the DVP of all packets that get served within an observation period τ . The length of the observation period τ is set to 50,000 and 100,000 p-units, where a p-units is the average packet transmission time d_{trans} (or service time). In the thesis, the packet length is 1024bits and the link capacity is 2000Kbps, the observation period τ is 50,000 and 100,000 p-units, respectively.

The parameters in Experiment E are summarized in Table 5.9. The delay bounds of class-1 and class-2 are 0.3s. The DVP differentiation parameters of class-1 and class-2 are $w_1 = 1$ and $w_2 = 0.5$. The number of users in group-1 is : $N_1 = 40$, and the number of users in group-2 is: $N_2 = 40$. In this experiment, the average arrival rate in each class remains constant (stationary load distribution). The load ratio between two

N_1	N_2	w_1	w_2	D_1	D_2
40	40	1	0.5	0.3s	0.3s

Table 5.9: Parameters used in Experiment E

classes is: $N_1/N_2 = 1$. The target DVP ratio is: $\hat{r}_{1,2} = w_1/w_2 = 2$. We measure the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2 in short timescales, where $\tau = 50,000$ or $100,000$ p-units, respectively. In the AWEDD scheduler, the measurement window length L is set to $128K$ packets. The results from the WEDD scheduler are also shown for comparison.

Figure 5.8 and 5.9 show sample paths of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ that the WEDD scheduler and the AWEDD scheduler produce within different observation periods τ . In both figures, the x-axis is the time line while the y-axis is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2. A data point is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ measured over the last τ time unit. In Figure 5.8, τ is 50,000 p-units which implies that the achieved DVP ratio $r_{1,2}(t, t + \tau)$ is measured every 50,000 p-units. In Figure 5.9, τ is 100,000 p-units which implies that the achieved DVP ratio $r_{1,2}(t, t + \tau)$ is measured every 100,000 p-units.

From Figure 5.8 and Figure 5.9, we make three observations:

1. Both schedulers closely approximate the target DVP ratio $\hat{r}_{1,2}$ in most time intervals. However, the short-term achieved DVP ratios $r_{1,2}(t, t + \tau)$ are not always equal to the target DVP ratio $\hat{r}_{1,2}$, and they occasionally have significant deviation from the target DVP ratios. The first reason is that these two schedulers, WEDD and AWEDD with $L = 128K$, schedule packets based on the estimated DVP over a history of previous departures, which is larger than the number of packets departed within the

Figure 5.8: A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under stationary load distributions ($\tau = 50,000$ p-units), $\hat{r}_{1,2} = 2$

Figure 5.9: A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under stationary load distributions ($\tau = 100,000$ p-units), $\hat{r}_{1,2} = 2$

observation period τ . Consequently, the short-term achieved DVP ratios $r_{1,2}(t, t + \tau)$ can deviate from the proportional DVP differentiation model constraints. The second reason is related to the feasibility of the proportional DVP model, which will be explained in section 5.5.

2. A more careful examination of the sample paths shows that the AWEDD scheduler meets the proportionality more accurately. In other words, the achieved DVP ratios $r_{1,2}(t, t + \tau)$ obtained by AWEDD have smaller deviations from the target DVP ratios $\hat{r}_{1,2}$ than those obtained by WEDD. The reason is that the AWEDD scheduler and the WEDD scheduler differ in the time interval over which the DVPs are proportional adjusted. For example, in Experiment E, the AWEDD scheduler with the measurement window length $L = 128K$, attempts to adjust DVP ratios proportionally only for the last $128K$ departures. In contrast, the WEDD scheduler attempts to adjust DVP ratios proportionally for all departures.
3. For either scheduler, the observation period τ increased, the deviations from the target DVP ratio $\hat{r}_{1,2}$ decreased.

5.4.2 Short-term deviations under non-stationary traffic load distribution

In evaluating the adaptiveness of a scheduler, our primary focus is to examine whether the short-term achieved DVP ratios $r_{i,j}(t, t + \tau)$ have negligible deviations from $\hat{r}_{i,j}$ under stationary load distributions and nonstationary distributions. It has already been illustrated in the previous graphs that target DVP ratios $\hat{r}_{i,j}$ can be achieved in most short timescales, but only in non-stationary traffic load distributions. Are the target DVP ratios $\hat{r}_{i,j}$ between

	N_1	N_2	w_1	w_2	D_1	D_2
before time 12, 500, 000 p-units	64	16	1	0.5	0.3s	0.3s
after time 12, 500, 000 p-units	16	64	1	0.5	0.3s	0.3s

Table 5.10: Parameters used in Experiment F

classes, however, also achieved in short timescales under non-stationary traffic load distributions?

Experiment F: In this experiment, we want to find out the achieved DVP ratios $r_{1,2}(t, t + \tau)$ between class-1 and class-2 within a *fixed time interval* (or what we call the observation period τ) under non-stationary load distribution. In order to do it, we measure the DVP of all packets that get served within a observation period τ . The length of the observation period τ is varied to be 50, 000 and 100, 000 p-units.

The parameters in Experiment F are summarized in Table 5.10. The delay bounds of class-1 and class-2 are 0.3s. The DVP differentiation parameters of class-1 and class-2 are $w_1 = 1$ and $w_2 = 0.5$, respectively. The target DVP ratio is: $\hat{r}_{1,2} = w_1/w_2 = 2$. Initially, the number of users in group-1 is: $N_1 = 64$ and the number of users in group-2 is: $N_2 = 16$. At time 12, 500, 000 p-units, the number of users N_1 in group-1 decreases to 16, and the number of users N_2 in group-2 increases to 64. Therefore the load ratio between class-1 and class-2 changes from 4 to 0.25 at time 12, 500, 000 p-units. This implies that the average arrival rate in each class is nonstationary. We measure the achieved DVP ratio $r_{1,2}(t, t + \tau)$ between class-1 and class-2 in short timescales, where $\tau = 50, 000$ or $100, 000$ p-units, respectively. The results from the WEDD scheduler are also shown for comparison.

Figure 5.10 and 5.11 show sample paths of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ that the WEDD scheduler and the AWEDD scheduler produce within different observation periods τ . In both figures, the x-axis is the time line while the y-axis is the achieved DVP ratio

Figure 5.10: A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under nonstationary load distributions ($\tau = 50,000$ p-units), $\hat{r}_{1,2} = 2$

$r_{1,2}(t, t + \tau)$ between class-1 and class-2. A data point is the achieved DVP ratio $r_{1,2}(t, t + \tau)$ measured over the last τ time unit. In Figure 5.10, τ is 50,000 p-units which implies that the achieved DVP ratio $r_{1,2}(t, t + \tau)$ is measured in every 50,000 p-units. In Figure 5.11, τ is 100,000 p-units which implies that the achieved DVP ratio $r_{1,2}(t, t + \tau)$ is measured in every 100,000 p-units.

In both Figure 5.10 and Figure 5.11, the short-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ are not always equal to the target DVP ratio $\hat{r}_{1,2}$, and there are occasional “priority inversions” of significant magnitude in the achieved DVP ratios $\hat{r}_{1,2}$ obtained by the WEDD scheduler after the class load distributions change. Specifically, at time 12,500,000 p-units, the average class loads are modified as shown in the graphs. The AWEDD scheduler is unaffected by this load distribution change, since it only considers the departures from each class in the last $L = 128K$ packets. On the other hand, the WEDD scheduler is negatively influenced

Figure 5.11: A sample path of the achieved DVP ratio $r_{1,2}(t, t + \tau)$ obtained by AWEDD and WEDD under nonstationary load distributions ($\tau = 100,000$ p-units), $\hat{r}_{1,2} = 2$

by the load distribution change, and the achieved DVP ratio $r_{1,2}(t, t + \tau)$ diverges from the target DVP ratio $\hat{r}_{1,2}$. The reason for this behavior is as follows. After the load change, the class-2 has four times higher rate than class-1, the DVPs between classes are still ratioed based on the long history of previous departures, in which class-1 had four times higher rate than class-2. Such “priority inversion” can occur when the load distribution is nonstationary, or when the load distribution varies in short timescales due to traffic burstiness. In contrast, by using a measurement window, such as $L = 128K$, the AWEDD scheduler can detect any change in traffic load distributions in time and allow fast adaptations based on instantaneous detections of system behavior.

5.5 Examples of infeasible proportional DVP differentiation

Sometimes, we cannot achieve the target DVP ratios $\hat{r}_{i,j}$ specified by the DVP differentiation parameters w_i , i.e., the achieved DVP ratios $r_{i,j}(t, t + \tau)$ experienced by the proportional DVP differentiation scheduler deviate from the target DVP ratios $\hat{r}_{i,j}$ in long timescales. In this situation, we say that the DVP differentiation parameters w_i are infeasible.

To illustrate some of these conditions, Figure 5.12 shows the achieved DVP ratio for AWEDD and WEDD, in the case of two classes, for several examples of “hard to achieve” DVP differentiation.

In Figure 5.12-a, a high target DVP ratio ($\hat{r}_{1,2} = w_1/w_2 = 50$), in combination with larger delay bounds ($T_1 = 3.0s$ and $T_2 = 1.5s$), makes the DVP differentiation infeasible, because there are not enough class-1 departures with delay violation in order to make the DVP of class-1 50 times larger than the DVP of class-2. In Figure 5.12-b, class-1 does not have enough arrivals compared to class-2 when $N_1/N_2 = 2/78$, and so it cannot maintain the required backlog for the given DVP differentiation.

The general observation from our simulation results is that the proportional DVP differentiation schedulers, AWEDD and WEDD, deviate from the proportional DVP differentiation model when the proportional DVP differentiation parameters are infeasible. The reason is that the class with the the maximum normalized DVP is not backlogged at the times of packet transmissions.

(a) The target of DVP ratio $\hat{r}_{1,2}$ is too high, $\hat{r}_{1,2} = 50$

(b) Class-1 is weakly loaded, $\hat{r}_{1,2} = 4$

Figure 5.12: The behavior of AWEDD and WEDD when the proportional DVP differentiation model is at the onset of infeasibility

5.6 From relative QoS to hard QoS

Some applications may require stringent DVP requirement, dealing with an absolute bound on the value of the DVP. Proportional DVP differentiation model is a relative QoS differentiation model which cannot provide hard QoS guarantees by itself. In order to provide hard QoS guarantees, a *Connection Admission Control (CAC)* [45, 46, 47, 48] mechanism has to be used to guarantee that enough resources are available to both the new and the existing connections.

One approach of CAC scheme is Utilization-Based Admission Control (UBAC) [24]. UBAC outlines that, as long as the utilization of link is not beyond a given threshold, the performance guarantee of hard DVP requirement for the new and existing connections can be satisfied. This link utilization threshold depends on many parameters, such as aggregate traffic load, class load distribution and performance requirement of different classes. In the following experiment, we will focus on finding the utilization threshold in the UBAC scheme.

Experiment G: In this experiment, we use a different simulation topology. Figure 5.13 shows the simulation topology adopted in this experiment in which two classes are considered. The packets in this experiment are created by two random sources dedicated to class-1 and class-2, respectively. The interarrival times of packets in the same class follow Poisson distribution. We assume that class-1 has a lower priority, i.e., higher DVP, and class-2 has a higher priority. By adjusting the mean value of interarrival time of packets, we can get different average throughput (different traffic load). Although the Poisson assumption has been shown to be often invalid in packet network [44], it is a simpler traffic model than the voice traffic model used in other experiments. In this experiment, we only consider the long-term achieved DVP ratio.

Figure 5.13: Simulation topology used in Experiment G

packet size	link speed	packet service time d_{trans}	buffer size	w_1	w_2	D_1	D_2
1024bits	200Kbps	5s	∞	1	0.5	0.3s	0.3s

Table 5.11: Simulation parameter used in Experiment G

The simulation parameters used in this experiment are depicted in Table 5.11. The link speed is set to $200Kbits/s$ and all packets have a fixed size of $1024bits$. The packet service time (transmission delay d_{trans}) on the link is therefore $5ms$. Buffers are infinitely large and no packets are dropped in this experiment. The delay bounds of class-1 and class-2 are $0.3s$. The target DVP ratio between class-1 and class-2 is: $\hat{r}_{1,2} = w_1/w_2 = 2$.

The experimental results are summarized in Figure 5.14 where the x-axis is the source-2 average throughput while the y-axis is the achieved DVP experienced by packet in class-1 or class-2. From Figure 5.14, we make three observations:

1. For given source-1 (source-2) average throughput, as the source-2 (source-1) average throughput increased, the achieved DVP for class-1 and class-2 increased. For example, in the case where the source-1 average throughput is $80Kbps$, when the source-1 average throughput increases from $60Kbps$ to $80Kbps$, the DVP experienced by the packets in class-1 increases from 5.1% to 15.2% and the DVP experienced by the packets in class-2 increases from 2.5% to 7.5% as well. In addition,

(a) Achieved DVP experienced by packets in class-1

(b) Achieved DVP experienced by packets in class-2

Figure 5.14: Achieved DVP experienced by packets in class-1 and class-2 vs. source-2 average throughput under different source-1 average throughput, $\hat{r}_{1,2} = 2$

in the case where the source-2 average throughput is $100Kbps$, when the source-1 average throughput increases from $40Kbps$ to $60Kbps$, the DVP experienced by the packets in class-1 increases from 3.2% to 7.4%, and the DVP experienced by the packets in class-2 increases from 1.5% to 3.7% as well. This is because that with the increase of source-1 or source-2 average throughput, more packets would arrive at the system and the queue length becomes longer. Therefore, the packets experience larger queueing delays and the DVPs increase.

2. Regardless of the source-1 or source-2 average throughput, the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ can closely approximate the target DVP ratio $\hat{r}_{1,2}$. For example, in the case where the source-1 average throughput is $80Kbps$ and the source-2 average throughput is $60Kbps$, the DVP experienced by the packets in class-1 is 5.1% and the DVP experienced by the packets in class-2 is 2.5%, so we can say that the long-term achieved DVP ratio $r_{1,2}(t, t + \tau)$ can closely approximate the target DVP ratio $\hat{r}_{1,2}$. In this experiment, the measurement window length L is set to be $512K$ packets, meaning that the measurement window length L in this experiment is large enough in order for the AWEDD scheduler to meet the proportional DVP differentiation constraints.
3. When the total average throughput of two classes is fixed, the achieved DVP experienced by packets in class-1 and class-2 is also fixed. This is true for different combinations of source-1 and source-2, as long as the total average throughput is fixed. For example, when the total average throughput are fixed at $160Kbps$, the achieved DVP experienced by packets in class-1 is around 15% and the achieved DVP experienced by packets in class-2 is around 7.5%, no matter what the exact average throughput of each class is. This observation could be used as UBAC function. For instance, when

the hard DVP requirement is set as that hard DVP of class-1 is required less than 15% and the hard DVP of class-2 is required less than 7.5%, as long as the total average throughput of two classes is less than $160Kbps$, the hard DVP can be met and $u = 160Kbps/200Kbps = 0.8$ is called utilization threshold. However, this conclusion is only based on limited experiments in which the interarrival times of packets in the same class follow exponential distribution. Whether or not this conclusion is independent of traffic load distributions requires further studies.

Chapter 6

Conclusion

6.1 Conclusion

In this thesis, we examine a proportional DVP differentiation model for Internet differentiated service. The probability of delay violation is a very important performance measure for most of the Internet traffic, and it is especially important for all real-time applications. Under the proportional DVP differentiation model, the network administrator can control the DVP spacings between different classes of traffic. Specifically, the network administrator tries to ensure that the achieved DVP ratio between classes is kept at a constant specific ratio.

To implement such proportional DVP differentiated service, we propose an AWEDD scheduler based on the WEDD scheduler. These two schedulers, AWEDD and WEDD, differ in the time interval over which the DVPs are estimated. The WEDD scheduler estimates the DVPs based on the entire history of departures, while the AWEDD scheduler estimates the DVPs in successive measurement window of last L departures. This difference decides the accuracy in meeting the proportional DVP differentiation model.

As the proportional DVP differentiation scheduler, AWEDD and WEDD should be examined from two aspects: controllability and predictability. Obviously, both AWEDD and WEDD can provide controllable proportional DVP differentiation, since the network administrator can adjust the target DVP ratios based on DVP differentiation parameters. In order to grade the predictability of these two schedulers, we study if the achieved DVP ratios can closely approximate the target DVP ratios in long timescales and short timescales. We consider the scheduler *adaptive* if achieved DVP ratios have negligible deviations from the target DVP ratios in short timescales and *convergent* if achieved DVP ratios approximate the target DVP ratios in long timescales.

We evaluate, through simulations, the adaptiveness and convergence of the schedulers, WEDD and AWEDD. These simulations have shown that the AWEDD scheduler inherits the excellent convergence of WEDD, i.e., it provides lower DVPs to higher classes, and it approximates closely the proportional DVP differentiated model in long timescales as long as the measurement window length L in the AWEDD is set to be larger than L_{min} . The simulations have also shown that L_{min} in the AWEDD scheduler can be affected by the system utilization, load ratios, delay bounds and the DVP differentiation parameters.

About the adaptiveness of these two schedulers, the simulations have shown that using a measurement window, the AWEDD scheduler can obtain the achieved DVP ratios of classes that have smaller deviations from the target DVP ratios in short timescales under stationary traffic load and nonstationary traffic load, when compared with the results in WEDD. The simulations have also shown that as an important parameter in AWEDD, the measurement window length L provides controlled trade-off between system convergence and adaptiveness. This is because a larger L incorporates more history into the DVP estimation, giving estimates that are more robust against transient conditions. A smaller L , on

the other hand, allows faster adaptations based on more instantaneous detections of traffic load behavior.

In general, from the experiment results, it is shown that the performance of the AWEDD scheduler exceeds that of the WEDD scheduler in terms of adaptiveness and is comparable to that of the WEDD scheduler in terms of convergence. And, the AWEDD scheduler can avoid the counter overflow which occurs in the WEDD scheduler.

6.2 Future work

Future work in the context of proportional DVP differentiation can pursue a deeper mathematical treatment of the proposed scheduler AWEDD. The following issues deserve further investigation.

- First, the feasibility conditions for the proportional DVP differentiation model can be analyzed in greater depth. In the thesis, we have only shown some examples on the feasibility issues. We will study the quantitative guidelines for the selection of feasible DVP differentiation parameters.
- Second, another further research would be that how to combine the AWEDD scheduler and a CAC scheme, such as the UBAC method. The challenge of using the UBAC method with the AWEDD scheduler in the Diffserv network is how to verify the correctness of a utilization threshold. In the thesis, we have investigated the utilization threshold base on traffic with Poisson interarrival distribution. However, realistic traffic may not be Poisson distribution, and a robust utilization threshold in the UBAC method should not depend on modeling assumptions. Such a robust utilization threshold in UBAC would show more explicitly the impact of different traffic

load distributions on the hard DVP requirements.

- Third, an interesting topic is that how to dynamically adjust the measurement window length L according to the traffic load fluctuation. As we mentioned before, as long as the measurement window length L is chose larger than L_{min} , the performance of AWEDD is comparable to WEDD in terms of convergence. On the other hand, in order to achieve the better performance in terms of adaptiveness, we have to use a smaller measurement window length L to estimate DVP for each class. Therefore, the optimal measurement window length L in the AWEDD scheduler is L_{min} . However, with the change of the load conditions and class load distributions, the value of measurement window length L_{min} is changed also. By dynamically adjust the measurement window length L_{min} , the AWEDD scheduler can provide convergent and adaptive service over DVP between different classes under various load conditions and class load distributions.
- Fourth, in order to estimate DVP for each class, we use a measurement window of last L departures, in which each of departures has the same weight. Another further research would be that we use different weights for packets in measurement window.
- Finally, in our simulation, we use voice traffic model which is based on On-Off model. Another further research would be that we use video traffic model to examine the behavior of the AWEDD scheduler.

Bibliography

- [1] S. Bodamer, "A New Scheduling Mechanism to Provide Relative Differentiation for Real-time IP Traffic", *IEEE Globecom*, November 2000.
- [2] D. Ferrari, and L. Delgrossi, "Charging for QoS", *IEEE/IFIP IWQOS '98 keynote paper*, May 1998.
- [3] "QoS protocols and Architectures", *Quality of Service Forum*, 1999.
- [4] D. Ferrari, "Client Requirements for Real-Time Communication Services", *IEEE Communication Magazine*, November 1999.
- [5] I. W. Habib, and T. N. Saadawi, "Multimedia Traffic Characteristics in Broadband Networks", *IEEE Communication Magazine*, July 1992.
- [6] M. F. Arlitt, and C. L. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications", *IEEE/ACM Transactions on Networking*, October 1997
- [7] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service", *IETF Internet Draft*, July 1995.

- [8] J. Gozdecki, A. Jajszczyk, and R. Stankiewicz, "Quality of Service Terminology in IP Networks", *IEEE Communication Magazine*, March 2003.
- [9] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource ReSerVation Protocol", *IEEE Network*, September 1993.
- [10] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification", *RFC 2205*, September 1997.
- [11] P. P White, "RSVP and Integrated Services in the Internet: A Tutorial", *IEEE Communication Magazine*, May 1997.
- [12] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", *RFC 1633*, June 1994.
- [13] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", *IETF Internet Draft*, November 1997.
- [14] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", *RFC 2474*, December 1998.
- [15] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", *RFC 2475*, December 1998.
- [16] J. Heinanen, T. Finland, F. Baker, W. Weiss, and J. Wroclawski, "Assure Forwarding PHB group", *RFC 2597*, June 1999.
- [17] V. Jacobson, "An Expedited Forwarding PHB", *RFC 2598*, June 1999.
- [18] D. Goderis, et al, "Service Level Specification Semantics, Parameters and Negotiation Requirements", *IETF Internet Draft*, June 2001.

- [19] C. Dovrolis, and D. Stiliadis, "Relative Differentiation Services in the Internet: Issues and Mechanisms", *Measurement and Modeling of Computer System*, 1999.
- [20] C. Dovrolis, and P. Ramanathan, "A Case for Relative Differentiated Services and the Proportional Differentiation Model", *IEEE Network*, October 1999.
- [21] L. Kleinrock, "Queueing Systems: Volume 2", *Wiley-Interscience*, 1976.
- [22] C. M. Aras, J. Kurose, D. Reeves, and H. schulzrinne, "Real-time Communication in Packet-switched Networks", *Proceedings of the IEEE*, Vol.82, No. 1, January 1994.
- [23] D. Ferrari, and D. Verma, "A Scheme for Real-time Channel Establishment in Wide Area Networks", *IEEE Journal on Selected Areas in Communications*, April 1990.
- [24] S. Wang, D. Xuan, R. Bettai, and W. Zhao, "Providing Absolute Differentiated Services for Real-Time Applications in Static-Priority Scheduling Networks", *Proceedings of IEEE Infocom*, April 2001.
- [25] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional Differentiated Service: Delay Differentiation and Packet Scheduling", *Proceedings of ACM SIGCOMM*, September 1999.
- [26] C. Dovrolis, and P. Ramanathan, "Dynamic Class Selection: from Relative Differentiation to Absolute QoS", *Proceedings of the ICNP 2001*, November 2001.
- [27] C. Li, R. Bettati, and W. Zhao, "Static Priority Scheduling for ATM networks", *Proceeding of Real-Time Systems Symposium*, December 1997.
- [28] H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-switched Networks", *Proceedings of the IEEE*, vol.83, no.10, October 1995.

- [29] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm", *Proceedings of ACM SIGCOMM*, September 1989.
- [30] A. K. Parekh, and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-node Case", *IEEE/ACM Transactions on Networking*, June 1993.
- [31] A. Parekh, and R. Gallager, "A generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case", *IEEE/ACM Transactions on Networking*, April 1994.
- [32] D. Verma, H. Zhang, and D. Ferrari, "Guaranteeing Delay Jitter Bounds in Packet Switching Networks", *Proceedings of Tricomm*, April 1991.
- [33] A. Francini, and F. M. Chiussi, "A Weighted Fair Queueing Scheduler with Decoupled Bandwidth and Delay Gurantees for the Support of Voice Traffic", *IEEE Globecom*, November 2001.
- [34] T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "Delay Differentiaiton and Adaptation in Core Stateless Networks", *Proceedings of IEEE INFOCOM*, March 2000.
- [35] A. Netterman, and I. Adiri, "A Dynamic Prority queue with General Concave Priority Functions", *Oprations Research*, 1979.
- [36] C. L. Liu, and J. W. Leyland, "Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment", *Journal of ACM*, January 1973.

- [37] L. Georgiadis, R. Guerin, V. Peris, and K. Sivarajan, "Efficient Network QoS Provisioning Based on per Node Traffic Shaping", *Proceedings of IEEE Infocom*, August 1996.
- [38] I. Georgiadis, R. Guerin, and A. Parekh, "Optimal Multiplexing on a Single Link: Delay and Buffer Requirements", *IEEE Transactions on Information Theory*, September 1997.
- [39] P. T. Brandy, "A Model for Generating On-Off Speech Patterns in Two-Way Conversation", *Bell System Technical Journal*, September 1969.
- [40] E. N. Gilbert, "Capacity of a Burst-noise Channel", *Bell System Technology*, September 1960.
- [41] "Pulse Code Modulation (PCM) of Voice Frequencies", *ITU-G Reference*, November 1988.
- [42] P. T. Brady, "A Technique for Investigating On-Off Patterns in 16 Conversations", *Bell System Technical Journal*, January 1968.
- [43] "Artificial Conversational Speech", *ITU-T Reference*, March 1993.
- [44] R. W. Muise, T. J. Schonfeld, and G. H. Zimmerman, "Experiments in Wideband Packet Technology", *Proceedings Zurich Seminar*, 1986.
- [45] V. Paxson, and S. Floyd, "Wide Area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Transactions on Networking*, June 1995.

- [46] A. Dailianas, and A. Bovopoulos, “Real-Time Admission Control Algorithm with Delay and Loss Guarantees in ATM networks”, *Proceedings of IEEE Infocom*, June 1994.
- [47] J. Liebeherr, D. E. Wrege, and D. Ferrari, “Exact Admission Control in Networks with Bounded Delay Services”, *IEEE/ACM Transactions on Networking*, December 1996.
- [48] P. W. Tse, and M. Zukerman, “Connection Admission Control in ATM Networks”, *IEEE Globecom*, 1994.
- [49] V. Firoiu, J. Kurose, and D. Towsley, “Efficient Admission Control for EDF Schedulers”, *Proceedings of IEEE Infocom*, March 1997.