A Proportional Congestion Control Solution for Real-Time Traffic

submitted by

Jinmei Yang, B.Eng., M.Eng.

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Applied Science

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Canada

April, 2005

0-494-06809-4

Abstract

For many real-time applications (e.g. VoIP), some absolute quality of service (QoS) requirements must be satisfied. In this thesis, we propose a novel dynamic proportional control framework (DPCF) for the scalable support of absolute end-to-end QoS in differentiated service (DiffServ) networks. Different from all other dynamic congestion control approaches, our DPCF builds on the fact that each internal node of a network can support proportional relative DiffServ quite easily. Our first contribution is to extend the proportional relative DiffServ of each node to an end-to-end relative DiffServ. This allows us to achieve the end-to-end absolute DiffServ by simply measuring the QoS of a base class at egress node and regulating the base class traffic at ingress node with the feedback from the egress node. One of the key benefits of our DPCF solution is its tolerance to dynamic path changes as long as all internal nodes maintain the same relative QoS among different classes. No per-flow state needs to be maintained at each internal node. In order to support the framework better, we propose a unified scheduler, the Dynamic Deficit Round Robin (DDRR) scheduler, which combines Generalized Processor Sharing (GPS) with relative DiffServ in the internal nodes of a network. The single-node simulations have shown that the DDRR scheduler is able to maintain the desired DVP ratios under different link utilizations and load distributions. The multi-node simulations have shown that the specified absolute end-to-end DVP requirements can be achieved effectively under the DPCF. The relative end-to-end DVP results are consistent with our analytical bound.

The undersigned recommend to the Faculty of Graduate Studies and Research

Acceptance of the thesis

A Proportional Congestion Control Solution for Real-Time Traffic

submitted by

Jinmei Yang, B.Eng., M.Eng.

in partial fulfillment of the requirements for

the degree of Master of Applied Science

Thesis Supervisor, Dr. Changcheng Huang

Chair, Department of Systems and Computer Engineering

Carleton University

April, 2005

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Changcheng Huang, for his guidance, encouragement and patience.

I must also thank those persons at Department of System and Computer Engineering for their support and help; and I must especially acknowledge and thank Prof. Richard Dansereau and Blazenka Power.

I gratefully acknowledge the funding support provided to my research by Communications and Information Technology Ontario (CITO) and Natural Sciences and Engineering Research Council of Canada (NSERC).

I want also to thank Jenny Zhu, Ximing Zeng, Yan Robichaud, Hongqing Zeng, Benjamin Feng, Houjin Li, Baohua Zhang, and Hao Wang for their kindly and generous help during my graduate study.

Finally, I am grateful to my family. Thanks to my parents, for their encouragement and support all through my life. Special thanks to my husband Xiangyang and my daughter Cynthia, for their love and supports all the way long.

Table of Contents

List of Figures

List of Tables

List of Notations

pk_D_k	End-to-end delay experienced by the kth packet
$pk\_D_{\max}$	End-to-end delay upper bound
β	Lower bound of the probability of DVP
f_i^n	General Differentiation parameter of class i at node n
d_i^n	Average delay of class i at node n
q_i^n	Performance metric of class i at node n
δ_i^n	DVP differentiation parameter of class i at node n
$q_i^n(t)$	Priority of the packet at head of class i at node n
$a_i^n(t)$	Waiting time of the head packet of class i at node n at time t in WTP
γ_i^n	Delay bound of class i at node n in EDD
$t_{k,i}^n$	Arrival time of the kth packet of class i at node n
t	Current system time
$E_i^n(t)$	Current estimated DVP of class i at node n at time t
w_i^n	Weight of class i at node n in GPS
$R_i^n(\tau,t)$	Amount of class i at node n traffic served in the time interval $[\tau,t]$ in GPS
C	Link capacity
$F_i^n(k)$	Virtual finish time of the kth packet of class i at node n in GPS
$v_i^n(k)$	Virtual arrival time of the kth packet of class i at node n in GPS
$P_i^n(k)$	*Kth* Packet size of class i at node n in GPS
g_i^n	Service rate of class i at node n in GPS
$B_i^n(t)$	Backlog of queue i at node n at time t in BPR

$\hat{r}_{j,i}$	Target DVP ratio of class i to class j
$\tilde{p}_i^n$	Normalized DVP of class i at node n
$DI[i]$	Absolute DVP of class i in a measurement interval
$DTB[i]$	Absolute DVP of class i before current measurement interval
$DT[i]$	Exponential averaging DVP of class i
$BI[i]$	Counter of packets departure with DVP from class i in measurement interval
$AI[i]$	Counter of packets departure from class i in measurement interval
$weight[i]$	Weight of class i in DDRR
ρ	Weight of the current interval DVP in exponential DVP calculation
α	Weight adjustment ratio
R	Weight adjustment interval
D_i	Hop (node) deadline at node i
D^l	End-to-end deadline on path l
$p_i^{l,n}$	Absolute DVP of class i at node n on path l
$r_{j,k}^i$	Relative DVP ratio of class k to class j at node i
$P_j^{l,n}$	End-to-end absolute DVP of class j traffic on path l with n nodes
$\mathrm{R}_{j,k}^{l,n}$	End-to-end relative DVP ratio of class k to class j on path l with n nodes
Δ	Absolute error from the lower bound
S_1	Number of users in source *1*
S_2	Number of users in source 2

List of Acronyms

ATM	Transfer Mode
ABR	Available Bit Rate
BA	Behavior Aggregate
BB	Bandwidth Broker
BPR	Backlog Proportional Rate
BRM	Backward Resource Management
CAC	Call Admission Control
DCS	Dynamic class selection
DDP	DVP Differentiation Parameter
DDRR	Dynamic Deficit Round Robin
DiffServ	Differentiated Services
DPCF	Dynamic Proportional Control Framework
DRR	Deficit Round Robin
DSCP	DiffServ Code Point
DVP	Deadline Violation Probability
ECN	Explicit Congestion Control
EDD	Earliest Due Date
EDF	Earliest Deadline First
EFCI	Explicit Forwarding Congestion Indication
FRM	Forward Resource Management
GPS	Generalized Processor Sharing
IETF	Internet Engineering Task Force

IntServ	Integrated Services
MANET	Mobile ad hoc networks
PGPS	Packet-by-Packet Generalized Processor Sharing
PHB	Per-Hop Behavior
QoS	Quality of Service
RED	Early Random Drop
RM	Resource Management
RSVP	Resource Reservation Setup Protocol
SACK	Selective Acknowledgments
SAF	Speech Activity Factor
SLA	Service Level Agreement
SLS	Service Level Specification
TCP	Transmission Control Protocol
TDP	Time Dependent Priority
WEDD	Weighted Earliest Due Date
WFQ	Weighted Fair Queueing
WRR	Weighted Round Robin
WTP	Waiting-Time Priority

Chapter 1

Introduction

1.1 Background

The current Internet provides only best-effort service with unpredictable performance. Traffic is processed as quickly as possible. However, the commercialization of the Internet has given rise to new real-time applications such as video streaming. These applications have traffic characteristics and performance requirements that are quite different from the existing data-oriented applications [1]. Consequently, the best-effort service is not sufficient for many of such applications.

To satisfy a diverse set of Quality of Service (QoS) requirements [2][3], IETF has proposed two architectures: Integrated Services (IntServ) [4] and Differentiated Services (DiffServ) [5]. IntServ approach tries to ensure end-to-end absolute performance guarantees on a per-flow basis [6][7]. The strength of IntServ lies in its per-flow reservation using Resource reSerVation Protocol (RSVP) [8] and its per-flow QoS guarantee. However, the per-flow QoS guarantee is not scalable due to the per-flow state management in internal routers where there are millions of flows traversing the network simultaneously. This leads to a network state explosion. Therefore, IntServ can not scale well in the Internet.

On the other hand, the goal of DiffServ is to provide the benefits of different levels of QoS while avoiding the limitations of the IntServ model. It aggregates traffic with similar QoS requirements into classes. Instead of achieving per-flow QoS, class based QoS metrics are guaranteed. A network core implements simple but highly scalable scheduling behaviors for traffic classes on a Per Hop Behavior (PHB) [9] basis. This makes the DiffServ model more feasible and scalable.

The DiffServ model has two approaches: absolute and relative differentiated services. The absolute DiffServ aims at providing IntServ type of end-to-end absolute performance at a granularity of class level [10] [11] [12]. It achieves this goal through a combination of conservative traffic engineering, aggressive traffic shaping at edges, and class-based traffic treatment at each internal node. On the other hand, the relative DiffServ aims at providing relative services [13] [14] in the network internal nodes at a per-hop and per-class granularity. The relative DiffServ model assures that the performance of the higher class will be better than the performance of the lower classes. In this model, the performance of a higher priority class is better than those of a lower priority class. Let the local performance metric of class i be denoted by q_i, the relative differentiation between class i and class j can be expressed as a ratio $q_i/q_j = r$.

Many real-time applications such as Voice over IP (VoIP) require that a packet is delivered within a predefined end-to-end deadline. Packets delivered beyond these end-to-end deadlines are considered useless. In order to have an end-to-end delay guarantee, real-time applications rely on absolute differentiated services. The challenge for DiffServ to provide absolute end-to-end service is to develop solutions that can deliver suitable network control mechanisms with scalable and efficient network state management.

In this thesis, we propose a dynamic proportional control framework (DPCF) in order to provide absolute end-to-end differentiated services. The framework is the combination of relative DiffServ mechanism in the network internal nodes and traffic regulation at edges. With dynamic measurement-based feedback control and traffic shaping at edges, the proposed DPCF implements quantitative per-class deadline violation probability (DVP) guarantee. The relative DiffServ in the internal nodes of a network makes our approach different from other work, in which we can provide QoS performance guarantee for all classes of traffic by maintaining only one or several classes' state information. This reduces overhead at edges and leaves the network internal nodes as simple as possible. Our approach is simple but effective in implementation. In order to support the DPCF efficiently, we also introduce a unified scheduler, the Dynamic Deficit Round Robin (DDRR) scheduler, which combines Generalized Processor Sharing (GPS) with relative differentiated services. The DDRR scheduler achieves relative DVP among different classes at each internal node for satisfying real-time application requirements. Based on the single node result, a bound for the end-to-end proportional DVP ratio is developed.

The relative differentiation model can only offer the relative service among different classes on a per-node base. Most works so far end with studying the single node case [14][15][16]. This is clearly not enough from the users' point of view. The extension from a single node to an end-to-end path is definitely necessary but certainly more challengeable. In the following chapters, we show that some kind of end-to-end relative QoS can be guaranteed if each node of a network supports the relative QoS on a per-node base. However, for some specific applications, especially for those value-added business customers, the end-to-end relative QoS is still not good enough. An end-to-end absolute QoS is required in order to meet the stringent delay and loss requirements [17] [18]. In [19] and [20], users have to adjust dynamically the class of a flow in order to match its end-to-end absolute QoS requirements. Nevertheless, under heavy load condition of any flow, the absolute performance of all classes will become worse simultaneously even they are still proportional. When users with the smallest delay requirements are unsatisfied in the highest class, both approaches could not provide a satisfactory solution. Furthermore, changing the class of a traffic flow can introduce out of order packets.

To satisfy end-to-end absolute QoS requirements for real time applications, we need traffic engineering and traffic shaping. Traffic engineering is a difficult task due to the dynamic nature of Internet traffic [21]. Some examples are the range-based model [22] and the hose model [23]. From these models we can see that, to make the problem tractable, intentional over-provisioning is typical in real practice to guarantee a certain level of QoS. In [24], in order to support absolute delay bounds, traffic aggregates of each class are dimensioned at edges by a distributed control mechanism that employ statistics of the entire traffic present in the network. In practice, full knowledge of the traffic traversing a network is generally not available and at the same time control derived from unrealistic model M/M/1/K can lead to poor performance. The approach presented in [25] maintains the control states of internal routers by a complex bandwidth broker algorithm and perform admission control for the whole network. It can support strict QoS guarantee but also lead to more complexity in the control plane.

Our approach, with measurement-based traffic control at edges and relative DiffServ algorithm inside the network, bears similarity to the work in [24] but differs in that we provide traffic regulation only for one class rather than all classes discussed in [24]. The QoS requirements for the other classes can be automatically achieved by the end-to-end relative differentiation maintained in our DPCF without complex centralized core algorithms as in [24]. This allows high utilization can be achieved without the cost of traffic engineering.

Some schedulers have been proposed for realizing relative differentiation of delay, loss and DVP in literature. Link sharing schedulers dynamically change the service rates allocated to classes, such as BPR [14] for relative delay and JoBS [16] for relative loss. Alternatively, priority-based schedulers schedule packets according to their priority, such as WTP [14] and MDP [19] for relative delay, PLR [26] for relative loss, and WEDD [15] for relative DVP.

An important advantage of the link sharing over the priority-based schedulers is to guarantee some minimum bandwidth for each class so that they are not completely distorted by some badly behaved classes. Suppose one real-time traffic class is extremely overloaded under no traffic engineering condition. All other real-time classes under the priority-based scheduler such as WEDD maybe affected so much that they receive too little bandwidth to be useful while trying to maintain certain relative QoS. In the extreme case, some kind of starvation may happen. In this thesis, we propose DDRR, a link sharing scheduler for the network internal nodes. It can provide real time traffic with the proportional DVP differentiation and guarantee a minimum bandwidth for each class. Although BPR and JoBS can satisfy a minimum bandwidth requirement, they are trying to achieve relative ratios of mean delay, not our relative DVP differentiation. We will describe our algorithm in more detail in chapter III.

1.2 Contributions

The contributions of this research are briefly summarized in the following:

- To propose a Dynamic Proportional Control Framework (DPCF) to meet absolute end-to-end differentiated services requirements. With dynamic measurement-based feedback control and traffic shaping at edges and the relative DiffServ in the network internal nodes, the proposed DPCF can satisfy quantitative per-class Deadline Violation Probability (DVP) guarantee. By maintaining only one or several classes' state information, the approach reduces overhead at edges and leaves the network internal nodes as simple as possible.

- In order to support the DPCF efficiently, a novel Dynamic Deficit Round Robin (DDRR) scheduling algorithm is developed to achieve proportional DVP differentiation among different traffic classes at a single node. The GPS-based scheduler DDRR provides the isolation among the different classes of DiffServ and guarantees the minimum bandwidth for each class, which is not the case for other DVP schedulers such as WEDD.

- Based on the relative DVP of a single node, a bound for the end-to-end proportional DVP ratio is developed, which can provide a general idea to configure the relative DVP of a single node for network administrators. We simulate the end-to-end proportional DVP ratio for the real time traffic streams. We compare analytical results with simulation results and show the agreements.

1.3 Organization

The remaining chapters of the thesis are organized as follows.

Chapter 2: Reviews some related topics on DiffServ network including QoS definitions, the proportional differentiation model, the related scheduling algorithms and traffic engineering methods. Chapter 3: Introduces the proposed scheduler, Dynamic Deficit Round Robin (DDRR). Develops a bound for the end-to-end relative DVP based on the relative DVP of a single node. At last, proposes the DPCF for converting relative DVP to absolute DVP. Chapter 4: Presents the simulation results under different topologies: the single node topology and the multi-node topologies. Chapter 5: Presents conclusions and recommendations for future research.

Chapter 2

Differentiated Services Network

DiffServ mechanisms allow network providers to allocate different levels of service to different users. In the DiffServ architecture, the complex traffic engineering is treated at the edges and class-based traffic processing is done at each internal node. More specifically, the edges in a DiffServ model are responsible for marking packets by setting the value of DiffServ Code Point (DSCP) [27] based on the Service Level Agreement (SLA) [28]. In DiffServ, the first six bits of the 8-bit Type of Service (ToS, or DS) fields in the IPv4 header contains the DSCP. By using DSCP, packets within the network internal nodes can be subjected to specific per-hop behavior (PHB), such as queueing or scheduling behavior. Each Behavior Aggregate (BA) is identified by a single DSCP. Service differentiation is provided by implementing different per-hop behaviors.

The Internet can support a wide variety of applications and thus needs to satisfy diverse QoS requirements. In this chapter, we first introduce some basic QoS definitions related to IP networks and the QoS metrics of interest. The proportional differentiation model is then introduced. We also review some existing scheduling algorithms providing relative services guarantees in the network internal nodes at a per-hop and per-class basis and some traffic engineering methods that can be applied to DiffServ networks at the edges.

2.1 QoS Definitions

Quality of Service (QoS) refers to the capability of a network to provide better services to selected network traffic over various technologies. The objective of network QoS is to quantify the treatment that a particular packet can expect as it transits a network.

In this thesis, QoS is described by a set of measurable parameters [29], such as delay, jitter, and loss rate etc., which have been widely used in network research and operations. The delay of a packet experiences at a node has four major components: queuing delay, handling delay, transmission delay, and propagation delay. It can be considered either in an end-to-end manner or with regard to a particular node element. The delay requirement can be specified as the average delay or worst-case delay. In general, the most complicated and interesting component of the node delay is the queuing delay. Unlike the other delays, the queuing delay can vary from packet to packet (Transmission delay can also vary because of varying packet length, but less variance). The delay variation must be bounded or minimized if adequate real-time service is to be achieved.

The traffic characteristics and performance requirements of real-time applications are quite different from data-oriented applications [1]. In a real-time application, the total delay for each packet across the network is bounded by the specific maximum delay (i.e. delay bound), resulting in a deadline being associated with each packet. In [30], ITU specifies that for good voice quality, no more than 150 ms of one-way, end-to-end delay should be tolerated. End-to-end delays between 150 and 400 milliseconds can be acceptable but are not ideal; and delays exceeding 400 milliseconds can seriously hinder the interactivity in voice conversations [18][31]. If a packet arrives at the destination after its deadline has expired, which is called deadline violation, it may become useless.

The delay bound guarantees could be either deterministic or statistical. Deterministic delay guarantees promise an absolute end-to-end delay bound for each packet. On the other hand, statistical guarantees aim in providing the probability of deadline violation to the end-to-end delay bound [18].

- Deterministic delay guarantee

$$pk\_D_k < pk\_D_{\max} \quad \textit{for all packets} \qquad (2.1)$$

where pk_D_k is the end-to-end delay experienced by the *k-th* packet between its injection into the network and its arrival at its destination, $pk\_D_{\max}$ is the end-to-end delay bound specified by the network administrator.

- Statistical delay guarantee

$$\text{Prob}(pk\_D_k \leq pk\_D_{\max}) \geq \beta \qquad (2.2)$$

where pk_D_k and $pk\_D_{\max}$ are defined as the same in deterministic delay guarantee and β is the lower bound of the deadline violation probability (DVP), which is specified by the network administrator.

Deterministic services provide a simple model to the applications; but they are excessively conservative in the use of the network resources. They account for the worst-case scenarios, even though such worst cases may occur with very low probability. As the real time applications are resilient to infrequent deadline violation (e.g. the tolerable loss rates for high-quality audio is five percent for speech and ten percent for music [32]), statistical services greatly increase the efficiency of the network by statistical multiplexing.

2.2 Proportional Differentiation Model

For relative differentiated services, the traffic is grouped into N classes of service. The service provided to class i will be better than class $(i-1)$ for $1 \le i \le N$, in terms of local forwarding metrics. An application selects an appropriate class based on its QoS requirements such as end-to-end delay or throughput.

In relative differentiation networks, a better service will be provided to the higher classes. There are several ways to provide the relative differentiation. We briefly discuss four existing approaches [33],

- Strict Prioritization

In this service model, the highest backlogged class is always serviced first. If higher classes have heavy load, lower classes can experience service starvation. Moreover, the network operator has no means to control the QoS differentiation between classes.

- Capacity Differentiation

In this approach, the network operator allocates more resources per unit load to higher classes traffic. For example, if the load of class i is l_i and the weight is w_i at node n, then the allocated bandwidth satisfies $w_i^n / l_i^n > w_j^n / l_j^n \quad for\ i > j$.

- Additive Differentiation

The basic idea in this model is to differentiate the class service level by using additive property, e.g. the delay of class i at node n is larger than that of class $(i+1)$ by a certain constant, $d_i^n = d_{i+1}^n + A$, where d_i^n is the delay of class i at node n and A is the constant.

- Proportional Differentiation

This model provides that the performance metric of each class is proportional to certain differentiation parameters. For example, the delays of class i and j satisfy the relationship $d_i^n / d_j^n = f_i^n / f_j^n$, where d_i^n is the delay of class i at node n and f_i^n is the differentiation parameter at node n set by the network operator.

As a further refinement of the relative differentiation services, the proportional differentiation model has generated much interest in the research community. In this model [13] [14] [26], QoS parameters discussed above such as delay, loss rate or jitter are proportional to certain differentiation parameters set by the network administrator. In DiffServ networks, the traffic is grouped into N classes of service. *For each class i, the service provided to class i will be better (or at least no worse) than the service provided to class (i-1), where* $1 < i \le N$, *in terms of queuing delay and packet losses* [13].

Considering N classes of traffic and q_i as a performance measure for class i at node n, the proportional differentiation model imposes constraints of the following form between classes,

$$\frac{q_i^n}{q_j^n} = \frac{a_i^n}{a_j^n} \quad 1 \le i, j \le N \tag{2.3}$$

where $a_1^n < a_2^n < \cdots < a_N^n$ are the quality differentiation parameters at node n. When the class loads vary in the actual network, the QoS performance ratio between classes remains fixed and controlled by the network administrator. In real time applications, the performance depends much on the probability that the end-to-end delay exceeds the delay bound, not on the average delay. In this thesis, we focus on deadline violation probability (DVP) differentiation only and apply it to the proportional differentiation model. Specifically, if p_i^n is the long-term average DVP of the class i traffic at node n, then the proportional DVP differentiation model states that

$$\frac{p_i^n}{p_j^n} = \frac{\delta_i^n}{\delta_j^n} \quad 1 \le i, j \le N \tag{2.4}$$

The parameter δ_i^n is referred to as DVP differentiation parameter (DDP) at node n. The higher classes with larger i have better quality, so $\delta_1^n > \delta_2^n > \cdots > \delta_N^n$.

With the proportional differentiation model, the network administrator could adjust the QoS performance ratio between classes, independent of the class loads; this cannot be achieved by the other relative differentiation models. The (relative) proportional differentiation model can provide,

- Predictability: the differentiation between different classes should be consistent with differentiation parameters, independent of the class loads distribution.

- Controllability: the differentiation parameters set by the network administrator can be adjusted between classes based on pricing and policy requirements.

2.3 Traffic engineering

Traffic engineering is concerned with the performance optimization of existing network resources and includes the application of principles to the measurement, characterization, and control of traffic. It addresses the problem of efficiently allocating resources in the network so that user constraints are met and operator benefit is maximized. In this thesis, we briefly review some major traffic engineering approaches and do not address issues not directly related to our topic, for instance, the issues of routing and the interaction of domains.

Services with different traffic characteristics may have different QoS requirements, for example, deterministic QoS is appropriate for intolerant and rigid applications while statistical QoS is suitable for some real-time applications. To satisfy the diversified QoS, there are numerous traffic engineering mechanisms proposed in the literature. They can be classified into two major types in the controlled QoS context [34].

2.3.1 Guaranteed Service

The first class is *guaranteed* service. This approach can provide guaranteed QoS performance in the form of packet delay, jitter or loss rate experienced at the receiving end of the packet transportation. When a specified share of bandwidth for each source is assigned appropriately and the traffic sources conform to certain stochastic properties, the guaranteed service is met by using a proper scheduling method. Some typical algorithms are WFQ [35] PGPS [36][37], Delay-EDD[38] and Jitter-EDD[39]. These scheduling algorithms have been designed mainly to provide isolation between flows with the specified bandwidth. The specified bandwidth is derived from calculating the worst-cast behavior using *a priori* characterizations of sources. The worst-case guaranteed bounds delivered by these algorithms are appropriated for intolerant and rigid applications, since they need deterministic QoS assurances about the service they receive.

However, the actual traffic may neither be stationary nor have an accurate stochastic description model. Hence, such static bandwidth allocation for each flow in guaranteed service may be inefficient due to bandwidth over-provisioning resulting in low network utilization. Therefore, instead of worst-case guarantee, statistic QoS provisioning is preferred both to maintain high network efficiency and to ensure the QoS requirement for real-time applications

2.3.2 Effective Bandwidth

To overcome the problem of inefficiency in guaranteed service, the second class, Effective Bandwidth, is proposed to guarantee a bound on a specified QoS parameter such as loss or late packets based on statistical characterization of the traffic [40][41][42]. The theory of effective bandwidth is designed to accurately estimate resources that a traffic flow requires to satisfy given QoS requirements. In this approach, each flow is allotted an effective bandwidth that is larger than its average rate but less than its peak rate. The effective bandwidth is computed for a steady-state of the network based on statistical characterization of traffic in most cases. By independent and additive property of effective bandwidth, we can compare the sum of the effective bandwidth of individual flows in a collection to the link bandwidth to determine whether the QoS requirements can be satisfied.

However, the estimation of the effective bandwidth through *a priori* characterization of flows is difficult to provide accurate and tight statistical models for each individual flow. In DiffServ network, it is desirable to estimate the bandwidth requirements of an internal link based on edge nodes computations. The current effective bandwidth methods may turn out to provide conservative estimation for the class-based DiffServ traffic in the network. Therefore, effective bandwidth solutions for DiffServ will miss significant potential statistical multiplexing gain and result in low network utilization.

The above two classes of static traffic engineering approaches are inefficient when traffic is bursty, resulting in dissatisfied QoS requirements. Higher network utilization can be achieved by dynamic bandwidth provisioning. In dynamic bandwidth provisioning, the network can take into account the recent traffic load change when estimating the kind of service it can deliver reliably. The bandwidth assigned to a flow is adaptively adjusted based on the flow's current condition and its QoS requirement.

We can further group dynamic traffic engineering mechanisms into two sets: proactive control and reactive control based on how to react to congestion.

2.3.3 Proactive Congestion Control

In the proactive approach [22][23][43]-[47], the traffic engineering process continuously optimizes network parameters by taking preventive action, such as Call Admission Control (CAC) [43] [44] [45]. This approach can protect the network from overload situations due to applications misbehavior and system malfunctions.

CAC can be further classified into two categories: (a) admission control using *a priori* traffic descriptors [43], (b) Measurement based admission control (MBAC) [44][45]. For the first approach, admission control is based on the assumption that it has perfect knowledge of each type of traffic source and current number of connected services. Hence, it will only accept a new service request if the minimum amount of bandwidth required by the total number of connected services, including the new one, is less than the available service capacity. However, since no traffic measurements are taken into consideration, if the provided traffic descriptors do not depict the actual behavior of sources or the appropriate traffic descriptors are not known *a priori*, the performance of such admission control schemes would be very low. Instead of explicitly specifying traffic descriptors, MBAC approaches use real time measurements to characterize those existing flows. When traffic from different flows is multiplexed, the QoS experienced depends often on their aggregate behavior. At the same time, MBAC relies on measurements, and source behavior is generally not static. Hence, MBAC can deliver significant gain in utilization only when there is a high degree of statistical multiplexing.

CAC is an open loop control system. Although effective, using open-loop control to achieve a given target QoS for DiffServ is difficult due to the lack of explicit relationship between the predicted traffic rate and the target QoS. The unrealistic assumption about the stochastic distribution of the traffic load in many situations also makes the actual QoS deviated from the desired level.

In the last few years, a Bandwidth Broker (BB) architecture has been proposed for admission control and resource provisioning in each network domain [25][46][47]. The BB architecture decouples QoS control from internal routers. Internal routers do not maintain any reservation state; all reservation states are stored in, and managed by, bandwidth brokers. Due to the requirement of end-to-end QoS through different networks, the concept of Bandwidth Broker has gained in popularity. However, while the internal routers in the BB architecture are freed from performing admission control decisions, the Bandwidth Broker needs to maintain the overall network domain and to store information about all elements, flows and paths, which introduce scalability concerns. Moreover, all edge routers sending requests to the broker would lead to high overhead on the links leading to the broker node.

In addition to CAC and BB, hose model [23] and range-based model [22] are alternative TE approaches in efficient resource allocation and load balancing. These models, which is applicable to both IP based networks and Virtual Private Network services (VPN), fall into the category of the proactive approach.

In the hose model, instead of a traffic matrix containing point-to-point demands for each node pair, the characterization of user traffic is expressed as per-node aggregate ingress and egress traffic volumes. There is no need for a complete traffic matrix, but only the total amount of traffic which a node injects into the network and the total amount of traffic which it receives from the network shall be specified. Comparing with the traditional pipe model, the hose model improves the traditional pipe model over-provisioning of the resources in the network internal nodes. Similarly, in the range-based model, the characterization of user traffic is expressed as a range of quantitative values for the services required. Range-based model deploys automated provisioning systems that are able to logically partition the capacity at the edge and the internal routers to various classes, where a class here refers to a guaranteed bound specified as a bandwidth bound. Range-based model provides the customer with a choice among multiple groups of bandwidth guarantees. With the hose model and the range-based model, resource provision is easier to specify and users have the flexibility to allocate bandwidth. Therefore, both models take advantage of the statistical multiplexing gains from the aggregation of flows in the network internal nodes.

However, the hose model and the range-based model increase the complexity of the already difficult problem of resource management to support QoS. Various suitable policies and algorithms are needed to dynamically provision and allocate resources at the edges. We also need to configure the interior nodes of a transit network to meet the assurances offered at the boundaries of the network, but dynamic and frequent reconfigurations of an interior device are not desired. This leads to scalability problem and also defeats the purpose of the DiffServ architecture that suggests moving all complexities towards the edges. In addition, resource provisioning in both approaches lack explicit relationship with the QoS metric of interest. Therefore, both approaches as open-loop control mechanisms are difficult to directly apply to DiffServ network.

2.3.4 Reactive Congestion Control

In the reactive approach, or closed-loop control [24][48]-[60], the control system responds adaptively to events that have already occurred in the network, in which the QoS performance parameters are regularly monitored to provide the feedback to adjust the allocated bandwidth. This approach regulates traffic flows in order to avoid network overload and to obtain a high network throughput. It can further be classified as host-based and router-based mechanisms. In host-based scheme, the congestion controls are usually implemented in the end-hosts, such as the sources and destinations, which also called end-to-end flow control. On the other hand, router-based scheme relies on the routers participating in congestion control and resources management.

Most of the control approaches to closed-loop congestion control have appeared in the ATM community, motivated by the Available Bit Rate (ABR) [48][49]. ABR service supports applications with vague requirements for throughput and delay which can be expressed as ranges of acceptable value. The goal of ABR is, given the limited bandwidth available to set the sending rate of sources so that the bandwidth is fully utilized and without congestion happening causing severe queueing delay and packet loss. The source is connected with the destination via a number of switches (in Figure

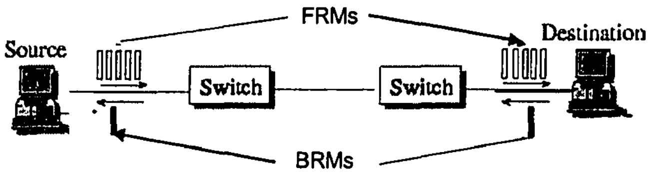


Figure 2.1 Basic configuration for rate-based congestion control in ABR connections

2.1 only two switches are shown). The switches monitor traffic load and compute the available bandwidth, dividing it fairly among active flows. The feedbacks from the switches to the sources are indicated in resource management (RM) cells which are generated by the source forward RM (FRM) and turned around by the destination backward RM (BRM) in Figure 2.1. The sources adjust their allowed cell rates based on the feedbacks received in BRMs.

The different schemes of feedback, either sending Explicit Forwarding Congestion Indication (EFCI) or sending an explicit rate directly in RM, have been proposed in ATM history. The rate based scheme can allow the rate of a source to adapt more rapidly and to oscillate less widely than EFCI. The congestion control approach in ABR is mainly about bandwidth control. The object being controlled is the sending rate of the sources.

Although the ABR service is potentially useful for a wide variety of applications, the main motivation for its development has been the support of data traffic. It does not support different QoS requirements for each connection except for a guaranteed minimum bandwidth. In ATM service, the congestion control provides qualitative QoS only for ABR traffic. While in DiffServ network, the diversified QoS requirements for different classes such as delay, loss, and jitter not only on local nodes but also on the end-to-end basis need to be satisfied for the aggregated class-based traffic. DiffServ can support bother types of services, services with absolute guarantees and services with relative differentiation.

The end-to-end Transmission Control Protocol (TCP) has been widely used in the current Internet to offer congestion control. The basic idea is that the sources should decrease their sending rates by reducing their sending window size upon detection of congestion. The TCP sender maintains its congestion window and can inject new packets into the network up to the congestion window without receipt of acknowledgements. As a congestion control mechanism, the TCP sender dynamically increases or decreases the window size according to the degree of the network congestion. The congestion level is estimated via packet losses. TCP relies on almost no help from individual routers. The essential operations of TCP include the following congestion control algorithms: slow start, congestion avoidance, fast retransmit and fast recovery. The most popular end-to-end flow control schemes in the Internet are various versions of TCP protocols, such as Tahoe TCP [50], Reno TCP [51], New-Reno TCP [52], and Selective Acknowledgments (SACK) TCP [53].

The advantage of TCP congestion control is that it is a simple end-to-end mechanism and it requires very little participation from routers. This allows it to be used for heterogeneous networks. Although TCP mechanisms work fairly well in practice, they still have some weaknesses. First, because there is no bandwidth guarantee in TCP congestion control, it is unable to provide good QoS control, which is necessary for real-time applications such as video conferencing. This also causes TCP not able to enforce fairness very well, allowing one connection to use a large portion of the bandwidth and causing jitter for all other connections. Further, the requirement for accurate timeout in order to maintain the congestion control rate is a burden when implementing TCP on some platforms. These disadvantages make TCP congestion control unsuitable for real-time applications and class-based DiffServ networks.

In reactive congestion control, the control systems rely heavily on feedback to detect congestion. There are two kinds of feedbacks: implicit/indirect feedback and explicit/direct feedback. In implicit/indirect feedback, the sources do not receive the needed feedback information from the network directly; instead they rely on a certain mapping relationship between the measurement and the interested QoS. By monitoring the most commonly used implicit signals such as delay, loss etc, the state of the network (congestion or not) can be inferred. On the other hand, in direct/explicit feedback, the network sends explicit feedback information to the sources, in the form of either congestion notification or rate indication. Explicit Congestion Control (ECN) scheme is proposed for TCP/IP network, in which the source is responsible for adjusting the sending window upon receiving the congestion acknowledgment packet. EFCI scheme in ATM performs the similar function as ECN, but in EFCI the sources adjust their sending rates directly. Explicit rate notification has also been used in ATM networks, in which the feedback information is the sending rate that is calculated by the switches in the path.

As the Internet expands rapidly, it is no longer possible to exclusively rely on end hosts to perform end-to-end congestion control. Therefore, the router-based mechanisms for congestion control have become unavoidable. Generally, the router-based mechanisms can be divided into two categories: scheduling and queue management.

The role of queue management is to control the length of queue by selecting which packet to drop and determining when such operation is appropriate. The prominent and widely studied active queue management schemes are: Early Random Drop (RED) [54], BLUE [55], FRED [56] etc. However, in this thesis we focus only on scheduling schemes to perform the required QoS; and we will review the scheduling schemes in more details in the next separate section.

Recently, several real-time QoS assurance solutions such as INSIGNIA [57], SWAN [58], and NPDD [59] have been proposed for wireless networks.

INSIGNIA [57] is a resource signaling protocol to support end-to-end adaptive services in mobile ad hoc networks (MANET). It depends on in-band signaling and soft-state bandwidth reservation to set up certain bandwidth guarantees for traffic flows along the path. In INSIGNIA, each IP packet carries the maximum and minimum bandwidth requests of a flow; routers along the path mark the packet header to indicate whether the requests can be guaranteed, and reserve the corresponding bandwidth with soft-state.

The advantages of INSIGNIA are that it can provide fast, per-flow bandwidth reservation, and it reacts quickly to route changes by in-band signaling. Hence, it is quickly responsive to bandwidth variation and network topology changes in MANET. The difficulty with INSIGNIA is its flow-based resource reservation in the intermediate nodes. As the available bandwidth for each node varies with time, it is hard for an intermediate node to perform admission control, resource reservation and flow-based soft state maintenance. When millions of flows traverse the network, INSIGNIA will have the scalability problem as the RSVP protocol.

SWAN [58] is a distributed and stateless network model using feedback-based control mechanisms to support service differentiation for best-effort and real-time traffic in MANET. In SWAN, rate control (shaper) is used for best-effort traffic and sender-based admission control for real-time traffic. Before sending a real-time flow, the sender must probe the path to see how much bandwidth is left at the intermediate routers, after that the sender uses the probed rate as the flow's sending rate. When a real-time flow can no longer be supported, it re-establishes its rate by probing the path again. Each node in MANET independently regulates best-effort traffic based on feedback of the packet delay; hence, real-time traffic is guaranteed having minimum delay at each node.

One limitation of SWAN is that it only supports the differentiated service for two classes of traffic. But usually, there are more than two classes of traffic traversing in DiffServ networks. In addition, in order to alleviate congestion, SWAN recommends randomly marking a subset of the real-time flows to stop sending traffic; as a result, no fairness can be guaranteed among flows. Furthermore, although SWAN works well by locally degrading a certain portion of the traffic at each node; there still exist fairness issues among nodes to solve congestion.

NPDD [59] provides QoS assurance through class selection and proportional differentiation in wireless networks. In the NPDD model, while the core network assures consistent proportional delay differentiation at all times for several service classes using waiting time proportional scheduler, the absolute class delays vary according to the traffic class distribution. To satisfy the absolute end-to-end delay requirement, applications monitor QoS periodically and dynamically adjust their service classes using

Dynamic Class Selection (DCS). If all applications follow the DCS rules and if the network can satisfy all the QoS requirements, the existence and feasibility of a QoS satisfying solution can be guaranteed.

One serious drawback in NPDD is that there is no traffic regulation scheme in the model. Under heavy load conditions, applications try to achieve their desired QoS and adjust the service classes of their flows to higher ones. It is possible that even after the applications set their flows to the highest class, the achieved QoS are still unsatisfactory. In addition, changing the class of a traffic flow can introduce out of order packets, which are highly undesirable for many real-time applications.

The services proposed in [24] [60] further explore the design space of class-based DiffServ architecture under the reactive control mechanism. The Dynamic Core Provisioning service [24] supports absolute delay bound and qualitative loss, but not proportional differentiation. The dynamic capacity provisioning architecture illustrated in Figure 2.2 comprises dynamic core provisioning module and dynamic node provisioning module. Dynamic core provisioning sets appropriate ingress traffic conditioners located at access routers by utilizing a *core traffic load matrix* to apply rate-adjusting. The mechanism used in dynamic node provisioning provides service guarantees by dynamically adjusting scheduler service weights and packet dropping thresholds in core routers. Persistent service level violations are reported to the core provisioning algorithm, which dimensions traffic aggregates at the network ingress edge.

Unfortunately, the core provisioning algorithm uses knowledge of the entire traffic present in the network. In practice, full knowledge of the traffic traversing a

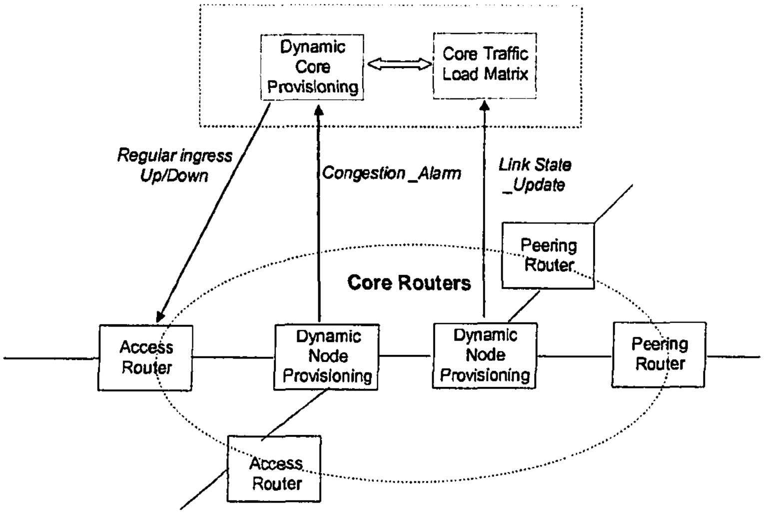


Figure 2.2 Dynamic capacity provisioning model

network is generally not available; the algorithm has to approximate the traffic when it is deployed in a large network. Furthermore, the measurement accuracy of traffic distributions of the ingress aggregate traffic over all links is also questionable. The computation complexity will increase greatly with the number of nodes in the network.

In [60] the authors propose an intra-domain monitoring system for the DiffServ network. The monitoring system includes (a) Node monitor: responsible for node-related measurements and there exists one node monitor per router. It is able to perform active measurements at the path or hop level. (b) Network monitor: responsible for network-wide post-processing of measurement data using a library of statistical functions. Although the Scalable Monitoring System provides the measurement accuracy and scalability, it has the drawback that node monitors need to collect information on all classes of traffic at the PHB level and routers in DiffServ network.

In this thesis, we propose a new dynamic proportional control framework (DPCF) to implement the absolute end-to-end QoS in DiffServ network. Since the above proactive control algorithms do not provide fast mechanisms that are capable of reacting to sudden traffic pattern changes for satisfied QoS, we choose the reactive control as congestion control method in our approach. Each internal node dynamically adjusts the service rate for each class to achieve the proportional differentiation among all classes of traffic. The volume of traffic at the network ingress is controlled by sending measurement-based feedback from the destinations to traffic sources. We can choose to send feedback information only from one class because the relative service differentiation scheme maintains the QoS performance for other classes in the network internal nodes. Our dynamic proportional control is capable of quickly restoring service differentiation under severely congestion and device failures. In contrast to the other reactive control algorithms presented above, our DPCF differs from ABR by providing quantitative QoS guarantees for multi-class traffic except only for ABR class in ATM network. In addition, our approach supports different quantitative QoS requirements contrary to qualitative service in ABR. By monitoring feedback information for one class not all ABR flows, this results in reducing the processing and bandwidth overheads in the network. Our DPCF differs from INSIGNIA in providing scalable class-based QoS guarantee instead of flow-based resource reservation. In SWAN, the service differentiation supports only for two traffic classes; while our DPCF can support multi-class traffic differentiation. At the same time, instead of guaranteeing the mean delay as in SWAN and NPDD, DPCF supports the absolute DVP QoS metric, which is more critical for real-time applications. The advantage of DPCF over NPDD is that DPCF provide absolute QoS guarantee with traffic regulation at the edge; in contrast, under heavy loaded networks, NPDD can not satisfy QoS requirements, because it has no traffic regulation support. Our approach also bears similarity to the work in [24] but differs in that we provide traffic control only for one class rather than all classes discussed in [24]. The QoS requirements for the other classes can be implemented by relative differentiation in the network internal nodes without complex centralized core algorithms in [24]. Without using the full knowledge of the entire traffic in the network [24] and information on all classes of traffic in [60], our approach is more scalable to large systems. We will describe our proposed DPCF in more detail in chapter 3.

2.4 Scheduling algorithms

The scheduling of resources is a key to provide performance guarantees to applications that require QoS support from the network. A scheduler (server) determines the order of the classes in which it serves packets. The service order has an impact on the delay suffered by packets waiting in the queue. The server can allocate bandwidth to packets from a class by serving a certain number of packets within a time interval.

In the last several years, Generalized Processor Sharing (GPS) schedulers and the priority-based schedulers have emerged as among the most popular packet scheduling schemes to satisfy the QoS guarantees for real-time communication services. The packetized version of the GPS scheduler (PGPS) can guarantee the minimum throughput and delay bound with flow isolation [36] [37]. The priority-based scheduler has the ability to minimize the maximum lateness or deadline of packets depending on the related control variables [61].

2.4.1 Priority-based schedulers

Based on Time Dependent Priority (TDP) algorithm [62], priority-based schedulers are developed by controlling the corresponding QoS metric. We will first describe the TDP scheduling.

In general, TDP is a non-preemptive packet scheduling algorithm that services N classes of traffic. If a server is ready to transmit a packet at time t, it only needs to consider the priority of the packet at the head of the queue for each class, since for the same queue the earlier arrival packets always have a higher priority than the later arrival packets. Let $q_i^n(t)$ be defined as the priority of the packet at the head of class i at node n, the scheduler chooses a packet from class i^*

$$i^*(t) = \arg \max_{i=1,2\cdots N} \{q_i^n(t)\} \tag{2.5}$$

Ties for the highest priority can be broken by serving the packet with the longest waiting time in the system. The different methods to define the priority of the packets and rules to break the ties result in the following different schedulers.

2.4.1.1 Priority queueing

Priority queueing scheduler is a special case of the priority-based schedulers. A simple way to provide differential treatment to classes is to assign different classes with associated priorities. If there are packets backlogged in both higher and lower priority queues, the scheduler will serve packets from the lower priority queue until it finishes serving the higher priority queue. The highest priority queue has the least delay, highest throughput and lowest loss.

Priority queueing is simple from an implementation point of view, as it needs to maintain only a few states per queue. However, this scheme has the potential starvation problem of lower classes under heavy load conditions. Moreover, a priority queueing is not controllable in function of QoS requirements; the relative differentiation between classes is very dependent on the load and the load distribution.

2.4.1.2 Waiting-Time Priority scheduler (WTP)

In WTP scheduler [13], the priority of a packet increases proportionally with its waiting-time in the queue. By controlling the ratios of individual packet's delay, the WTP scheduler can achieve the desired long-term average delays between different classes of traffic. Specifically, the priority of a packet in class i at node n at time t is,

$$q_i^n(t) = a_i^n(t) * f_i^n \qquad (2.6)$$

where $a_i^n(t)$ is the waiting time of the class i packet at node n at time t, f_i^n is the differentiation parameter, which is constant for class i denoted by the network administrator. The load of a class in the recent past is reflected on the waiting-time of the packet at the head of that queue. When the server is ready to send packet, the scheduler calculates the priority of each class according to equation 2.6 and chooses the packet with the highest priority.

Based on the priority-based scheduler, WTP cannot provide the minimum bandwidth guarantee. The overloading of one class will result in the starvation of the other classes. Also, it is designed to provide mean delay differentiation, which is not our desired DVP metric used for real time applications.

2.4.1.3 Earliest Due Date (EDD) scheduler

The EDD scheduler [63] , also known as Earliest Deadline First (EDF), is a deadline-based priority scheduler, which requires a packet to be transmitted before its deadline. It is a mechanism to provide absolute delay differentiation. Each class i is associated with a delay bound γ_i^n at node n. When the kth packet of class i arrives at time $t_{k,i}^n$, it receives a timestamp $t_{k,i}^n + \gamma_i^n$ representing its deadline. The scheduler serves the packets of all classes of traffic in increasing order of their deadline.

A characteristic of the EDD scheduler is that the probabilities of deadline violations due to congestion are equal in all classes. This property holds independent of the total load and load ratio of different classes. At the same time, it is uncontrollable when the network administrator demands performance differentiation between classes.

2.4.1.4 Weighted Earliest Due Date (WEDD) scheduler

The WEDD [15] scheduler is a modified version of the EDD. It enhances EDD by providing not only different deadlines but also different deadline violation probabilities. The deadline violation probabilities p_i^n of class i at node n are constrained by

$$\frac{p_i^n}{p_j^n} = \frac{\delta_i^n}{\delta_j^n} \tag{2.7}$$

where δ_i^n is the parameter set by the network administrator to control the performances of different classes. The basic operation of WEDD is the same as EDD, i.e. setting packets' deadlines $t_{k,i}^n + \gamma_i^n$ on arrival at $t_{k,i}^n$ and scheduling packets in increasing order of their deadlines. When there is more than one traffic class being backlogged with the head packet's deadline $t_{k,i}^n + \gamma_i^n < t$ (t defined as the current system time), the system is in "congestion mode". In congestion mode, we timestamp each backlogged class with g_i^n

$$g_i^n = \frac{\delta_i^n}{E_i^n(t)} \tag{2.8}$$

where $E_i^n(t)$ is the current measured deadline violation probability of class i. The scheduler serves the packet with the lowest congestion stamp g_i^n.

Though WEDD is able to provide the relative DVP probability for real time traffic, the priority-based scheduler cannot provide certain isolation among different traffic classes. Some badly behaved classes may impact negatively the performance of other classes when no other traffic engineering mechanisms are used.

2.4.2 GPS schedulers

Generalized Processor Sharing (GPS) has gained much popularity in recent years as a simple and effective scheduling mechanism for the provisioning of QoS. GPS is an idealized fluid work-conserving scheme. It assumes that each class (each flow) is kept in a separate logical queue and serves an infinitesimal amount of data from each nonempty queue in a finite time interval. Each queue can have an associated weight and can be served in proportion to its weight.

Let us assume that there are K classes to be served in a GPS scheduler and class i at node n has a weight of w_i^n. The service rate for the *i-th* class in interval $[\tau,t]$ is represented as $R_i^n(\tau,t)$. For any backlogged class i and class j in interval $[\tau,t]$, the following equation holds,

$$\frac{R_i^n(\tau,t)}{R_j^n(\tau,t)} \geq \frac{w_i^n}{w_j^n} \qquad (2.9)$$

Since serving an infinitesimal amount of data cannot be implemented in practice, we discuss next some variations of the GPS.

2.4.2.1 Deficit Round Robin

A simple implementation of GPS is the round robin (RR) scheduling where the servicing is done packet by packet. The round robin scheduler maintains one queue for each class. The queues are served in a round robin fashion, taking one packet from each nonempty queue in turn. Empty queues are skipped over. Instead of serving a single packet from a queue per turn, weighted round robin (WRR) [64] serves n_i packets, which is calculated from the weight w_i and the link capacity C, i.e. $n_i = w_i * C / pksize$. The WRR works well with fixed size packets.

Deficit round robin (DRR) [65] improves WRR by being able to serve variable-length packets without knowing the average packet size. The algorithm works as follows: initially a variable quantum is initialized to represent the number of bits to be served from each queue. The scheduler starts serving each nonempty queue. If the packet size is less than or equal to the quantum, the packet is served. However, if the packet is bigger than the quantum size, the packet has to wait for another round. In this case, a deficit counter is initialized for this queue. If a packet can't be served in a round, its deficit counter is incremented by the size of the quantum. The disadvantage of DRR is that the delay of a packet can be long and it depends on the number of queues. Suppose a packet just misses its scheduling turn, it has to wait until its next round. However, since class-based DiffServ usually only has several classes at the internal nodes, this problem is not a big issue in DiffServ networks. DRR scheme achieves fairness in terms of throughput for each queue, and is simple enough to implement in hardware.

2.4.2.2 Weighted Fair Queueing

Weighted Fair Queueing (WFQ) [36][35][66], which is also known as Packet Generalized Processor Sharing (PGPS), is another well-known algorithm based on the GPS scheduler. By introducing the concept of virtual time, which measures the work progress in the ideal GPS system, WFQ determines the packet service order according to the packet virtual finish time. The following equation is used to calculate the virtual finish time,

$$F_i^n(k) = \max\{F_i^n(k-1), V_i^n(k)\} + \frac{P_i^n(k)}{w_i^n} \tag{2.10}$$

where $F_i^n(k)$ is the virtual finish time of the *kth* packet in class *i* at node *n*. $V_i^n(k)$ is the virtual time of the *k-th* packet arrival. w_i^n is the weight of class *i* and $P_i^n(k)$ is the packet size. $Max\{F_i^n(k-1), V_i^n(k)\}$ can be considered as the virtual start time of the *kth* packet. As shown in equation 2.10, if w_i^n is large, $F_i^n(k)$ is small and the packet will be served earlier. Let *C* be the link capacity and *N* be the total number of classes being served. When all classes of the traffic are greedy sources, the guaranteed minimum service rate can be achieved for each class. The guaranteed service rate of class *i* at node *n* can be expressed as g_i^n,

$$g_i^n = \frac{w_i^n}{\sum_{j=1}^{N} w_j^n} * C \qquad (2.11)$$

In comparison with DRR, WFQ is more complex. When a class joins or leaves the competition, the virtual time has to be updated. This makes the implementation more complex and more processing intensive.

Using DRR or WFQ, we can guarantee the better performance of a class with respect to DVP by assigning a larger weight to that class. Due to the dynamic nature of the Internet traffic [21], it is a complex task to find the appropriate values of the weights according to the proportional DVP differentiation requirement. When the load distribution is non-stationary, or the load distribution varies quickly due to traffic burstiness, the resulting DVP differentiation can be dramatically different. A solution is to dynamically adjust the weights based on the current class loads. This idea results in the following scheduler.

2.4.2.3 Backlog Proportional Rate (BPR)

Based on GPS, the basic idea in BPR scheduler [13] is to adjust the class service rates dynamically so that they are divided proportionally to the corresponding ratios of measured class loads. The relation between the class loads is reflected on the relation of the class backlogs. Specifically, let R_i^n be the service rate that assigned to queue i at node n at time t. If queue i is empty at time t, then $R_i^n(t) = 0$. For two backlogged queues i and j, the service rate allocation in BPR satisfies the proportional constraint.

$$\frac{R_i^n(t)}{R_j^n(t)} = \frac{a_i^n}{a_j^n} \frac{B_i^n(t)}{B_j^n(t)} \tag{2.12}$$

where $B_i^n(t)$ is the backlog of queue i at node n at time t, and a_i^n is a set of control variables. The actual service rate of each class during a busy period can be calculated from the work-conservation constraint.

$$\sum_{i=1}^{N} R_i^n(t) = C \tag{2.13}$$

where C is the link capacity.

Although the BPR scheduler can guarantee the minimum bandwidth requirement, it is designed to achieve only the proportional delay differentiation and is only applicable to delay-sensitive applications. However, the performance of real-time applications such as IP telephony and video-conferencing will not so much depend on mean delays but on the probability that the end-to-end delay exceeds a certain threshold. Hence, we are more interested in the DVP QoS metric and choose it as our object of study. In this thesis, we provide the service differentiation in terms of DVP in DiffServ networks.

Chapter 3

From Relative Differentiation to Absolute QoS

In this chapter, we first describe the proposed dynamic proportional control framework DPCF to guarantee the end-to-end absolute DVP, and then develop a bound for the end-to-end relative DVP based on per-node relative DVP, and at last present deficit round robin (DDRR) scheduler that provides the relative DVP among all classes of traffic at single node.

3.1 The proposed framework: DPCF

In this section, we describe the architecture and basic operation of the DPCF.

3.1.1 Framework architecture

To give different packets different treatment, the network infrastructure must be capable of distinguishing between packets through means of classification, enqueueing packets separately as a result of the classification, scheduling packet queues to implement differential treatments, as well as providing means for measuring, monitoring and conditioning packet streams to meet requirements of different QoS levels.

If there is no traffic engineering and traffic shaping, a network with relative differentiation cannot provide absolute QoS guarantees. Figure 3.1 illustrates the proposed DPCF structure.

In the DPCF, the main objective is to provide the end-to-end absolute deadline violation probability (DVP) guarantee for each class in DiffServ networks. We implement this goal by combining relative DiffServ at the network internal nodes and traffic policing/shaping at edges. The DPCF is consistent with DiffServ architecture in terms of keeping complexity at edges and leaving the network internal nodes as simple as possible. The edge and internal nodes provide the following functions respectively:

Edge nodes: we implement the measurement of the end-to-end absolute DVP for

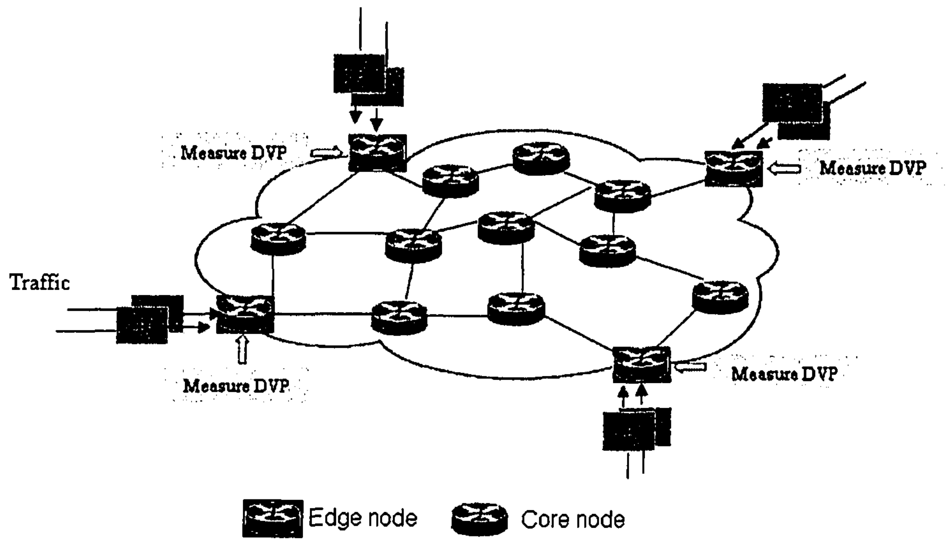


Figure 3.1 The proposed dynamic proportional control framework DPCF structure

base class at egress nodes. Traffic conditioners perform traffic shaping at ingress nodes. In the DPCF, we use leaky buckets at ingress nodes for traffic shaping.

Internal nodes: we implement the relative differentiation by the relative DiffServ schedulers at internal nodes. We choose the proposed DDRR scheduler to provide the relative DVP differentiation in the DPCF.

Assume that the end-to-end absolute DVP requirements for class 1 and class 2 are A and B respectively. We can assign a relative DVP ratio $r = B / A$ to each internal node. According to the theorem 2 in Chapter 3.2, if we configure each node with equal relative DVP ratio, the end-to-end relative DVP will be the same as each internal node. Hence, by maintaining the end-to-end absolute DVP requirement A for class 1, the requirement B for class 2 can be guaranteed automated by relative differentiation between class 1 and class 2.

One key issue for any DVP based algorithms is the setting of the delay deadline for each class at each internal node. If a path consists of only a single hop, it is obvious how to choose the delay deadline. However, for a multi-hop real network, it is not so trivial. The issue of how to split an end-to-end delay deadline into delay deadlines at each hop is an interesting and complex problem. Here we assume a simple and conservative approach.

We assume that a packet can satisfy its end-to-end delay bound only if it satisfies its per-hop delay bound. This assumption is more conservative than necessary. A packet could exceed its delay bound at one hop and be serviced earlier at another hop and still meet its end-to-end delay bound. However modeling such general scenario is not simple.

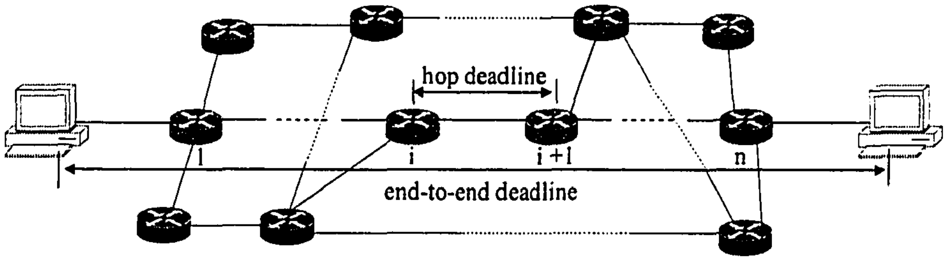


Figure 3.2 End-to-End deadline vs. hop-deadline

Hence, we make the conservative assumption that a packet missed its hop deadline will violate its end-to-end deadline. Violated packets at each node are marked and continue to be transmitted to egress nodes. Figure 3.2 illustrates our conservative approach which actually assumes an additive property. The sum of all deadlines at individual hops is equal to the deadline of the end-to-end path.

3.1.2 Basic Operation

The basic operation performed in our proposed DPCF is as the following steps:

1. **Divide delay budget at the internal nodes of a network.** We split the end-to-end delay deadline into hop deadline with equal splitting in this thesis. For a node with multi-path, we can choose the stringent deadline as the hop deadline.

2. **Configure the relative DVP ratio at the internal nodes of a network.** According to the QoS requirements of different classes, we allot the relative DVP ratios among classes at each internal node. We configure the same class *i* and class *j* with the same ratio, and then the end-to-end relative DVP between class *i* and class *j* can be achieved with a very similar value based on the theorem 2 in Chapter 3.2.

3. **Implement relative differentiation at the internal nodes of a network.** The proposed DDRR scheduler implements the relative DVP differentiation among classes at each internal node. The scheduler only needs to maintain local measurement for each class and local state management.

4. **Measure the end-to-end absolute DVP at egress nodes.** We measure the number of marked packets at egress nodes and calculate the end-to-end absolute DVP by a ratio of marked packets to the total packets. We do not need to measure QoS performance for all classes of traffic. Instead, we choose a class as a base class and only measure the end-to-end absolute DVP for the base class. It should be noted that the path that carries the feedback information does not need to be the same as the forwarding path because each internal does not need the feedback information.

5. **Feed back the results to ingress nodes.** The measured end-to-end absolute DVP results of the base class traffic at egress nodes are fed back to the ingress nodes. At each ingress node, we add the destination-based leaky bucket shaper for the base class.

6. **Shape the traffic at ingress nodes**. We adjust the token rate of the leaky bucket for the base class traffic based on the feedback of the achieved end-to-end absolute DVP of the base class traffic, until the end-to-end absolute DVP of the base class meets its target DVP. Because a relative end-to-end QoS can be achieved among different classes under our configuration , if the base class can meet its target absolute DVP, all other classes can meet their targets as well, as long as the ratios are designed based on the ratios of their absolute QoS.

It is important to note that typically we do not need shaping all traffic classes for meeting QoS requirements under our proposed DPCF. We can choose adjusting token rates of either one (typically the base class) or several traffic classes (typically with lower priorities) depending on whether the target absolute DVP for the base class is easy to achieve or not. Shaping lower traffic class is a good choice because it has the minimum impact on revenue. Usually there is a large mount of lower class traffic in the network. Achieving absolute QoS by shaping only the lower priority traffic is one of the benefits of our proposed framework.

Choosing the base class and measuring its absolute end-to-end DVP is the critical part of our proposed framework. The traffic of the base class should travel the same path as other classes in order to maintain the relative QoS as indicated in the last section. If the traffic of various classes at an ingress node travels to different destinations (or egress nodes), a base class flow for each destination is necessary. In the worst case, a mesh of all ingress nodes to all egress nodes may be necessary. This seems to raise a scalability issue. However we argue that this is not a big problem for the following reasons:

1. We periodically measure the absolute end-to-end DVP at egress nodes and perform traffic regulation at ingress nodes in a DiffServ network. At each ingress node, destination-based shaper is provided only for the base class. This edge-to-edge distributed solution greatly decreases the complexity from $O(n^2)$ inside the network to $O(n\ )$ at edges. The intelligent measurement and shaping at edges allow it to scale well to large networks and hence achieve the scalability.
2. Our measurements are only conducted for the base class traffic. We can choose the

base class to be the lowest traffic class to minimize the impact on the QoS seen by end users. If no such low priority traffic flow exists for a particular path, we can inject a small amount of low traffic into the path that we are interested in simply for the measurement purpose. This certainly can cause a little bit overhead in terms of utilization. But in comparison with the over-provisioning of the traditional traffic engineering approach, this overhead is a very small cost to pay simply because the overhead is a low priority class.

Whether we use low priority user traffic or injected low priority test traffic as the base class, the measurement results at the egress nodes need to be sent back to the corresponding ingress nodes. The ingress nodes then make decisions on how to shape ingress traffic to maintain absolute end-to-end QoS.

3.2 End-to-end relative DVP bound analysis

Assume that two flows from class *j* and *k* enter a network and they traverse the same path *l*. Let $p_j^{l,i}$ denote the absolute DVP of class *j* traffic at node *i* on path *l*, $r_{j,k}^i$ denote the relative DVP ratio of class *k* to class *j* at node *i*, so $r_{j,k}^i = p_k^{l,i} / p_j^{l,i}$; $P_j^{l,n}$ and $R_{j,k}^{l,n}$ are the corresponding end-to-end values on path *l* with *n* nodes and $R_{j,k}^{l,n} = P_k^{l,n} / P_j^{l,n}$. Assuming $p_k^{l,i} < p_j^{l,i}$ *and* $0 < r_{j,k}^i < 1$, we say that class *k* has higher priority than class *j*. The relative DVP ratios at different nodes may be different. In the following paragraphs, we address the issue that, given the relative DVP ratio $r_{j,k}^i$ at node *i*, where $i = 1, \ldots, n$, how to calculate the end-to-end relative DVP ratio $R_{j,k}^{l,n}$.

Theorem 1: Assume the events of delay deadline violation are independent across a network. Given two flows from class j and k traversing the same path l, the end-to-end relative DVP ratio $R_{j,k}^{l,n}$ of the specific path l with n nodes is bounded by

$$\mathrm{Min}\,(r_{j,k}^{1},\ r_{j,k}^{2}, \ldots\ r_{j,k}^{n}) \le R_{j,k}^{l,n}$$
$$\le \sum_{i=1}^{n} r_{j,k}^{i} - \sum_{\substack{i,d=1 \\ i>d}}^{n} r_{j,k}^{i}\, r_{j,k}^{d} + \sum_{\substack{i,d,m=1 \\ i>d>m}}^{n} r_{j,k}^{i}\, r_{j,k}^{d}\, r_{j,k}^{m} - \cdots + (-1)^{n-1}\, r_{j,k}^{1} r_{j,k}^{2} \cdots r_{j,k}^{n} \qquad (3.1)$$

Proof: Given the relative DVP $r_{j,k}^{i}$ at each node, Theorem 1 gives the upper and lower bound of the end-to-end relative DVP for an end-to-end path. To prove these results, we first try to get the expression of the end-to-end absolute DVP of each class; then deduce the end-to-end relative DVP from the end-to-end absolute DVP.

Given that all deadline violation events are independent across a network, since the end-to-end absolute DVP $P_j^{l,n}$ can be expressed as the complement of the end-to-end probability without deadline violation, and since the end-to-end probability without deadline violation 1- $P_j^{l,n}$ can be calculated by the multiplicative of each node's probability without deadline violation, i.e. $\prod_i (1 - p_j^{l,i})$, the end-to-end absolute DVP of class j traffic $P_j^{l,n}$ with n nodes on path l can be expressed as,

$$P_j^{l,n} = 1 - \prod_i (1 - p_j^{l,i}) \qquad (3.2)$$

Equation 3.2 makes no assumption about topology or traffic pattern of the given network. Therefore it is quite general.

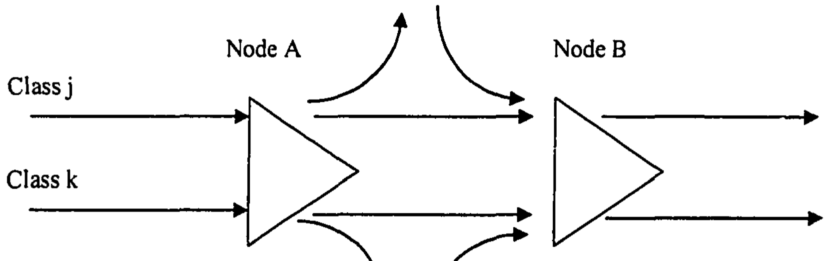
Without loss of generality, here we consider only two classes of traffic: class j and class k traversing a network following the same path l. Figure 3.3 shows an example path with two nodes. Assume class k has higher priority than class j, i.e. $p_k^{l,i} < p_j^{l,i}$ *and* $0 < r_{j,k}^i < 1$.

In the following we will use mathematical induction approach.

For a path with one node, it is obvious that the node relative DVP is equal to the end-to-end relative DVP ratio. Therefore Theorem 1 is true.

Suppose Theorem 1 is true with n-1 nodes, let $R_{j,k}^{l,n-1}$ be the end-to-end relative DVP ratio for n-1 nodes, we have

$$\begin{aligned} &\mathrm{Min}\,(r_{j,k}^1,\ r_{j,k}^2, \ldots\ r_{j,k}^{n-1}) \le R_{j,k}^{l,n-1} \\ &\le \sum_{i=1}^{n-1} r_{j,k}^i - \sum_{\substack{i,d=1 \\ i>d}}^{n-1} r_{j,k}^i\, r_{j,k}^d + \sum_{\substack{i,d,m=1 \\ i>d>m}}^{n-1} r_{j,k}^i\, r_{j,k}^d\, r_{j,k}^m - \cdots + (-1)^{n-2}\, r_{j,k}^1 r_{j,k}^2 \cdots r_{j,k}^{n-1} \end{aligned} \tag{3.3}$$



$$\frac{p_k^{l,A}}{p_j^{l,A}} = r_{j,k}^A \qquad \frac{p_k^{l,B}}{p_j^{l,B}} = r_{j,k}^B$$

Figure 3.3 The model for the end-to-end relative DVP analysis

Then for n nodes, we can treat the path as a two-node path, i.e. the first $n-1$ nodes as node A with $R_{j,k}^{l,n-1} = r_{j,k}^{A}$ and the n-th node as node B $r_{j,k}^{n} = r_{j,k}^{B}$. By Equation 3.2, the end-to-end relative DVP ratio on path l for the two-node network is,

$$R_{j,k}^{l,2} = \frac{P_k^{l,2}}{P_j^{l,2}} = \frac{p_k^{l,A} + p_k^{l,B} - p_k^{l,A} p_k^{l,B}}{p_j^{l,A} + p_j^{l,B} - p_j^{l,A} p_j^{l,B}}$$

$$= \frac{r_{j,k}^{A} p_j^{l,A} + r_{j,k}^{B} p_j^{l,B} - r_{j,k}^{A} r_{j,k}^{B} p_j^{l,A} p_j^{l,B}}{p_j^{l,A} + p_j^{l,B} - p_j^{l,A} p_j^{l,B}}$$

$$= \frac{r_{j,k}^{A} / p_j^{l,B} + r_{j,k}^{B} / p_j^{l,A} - r_{j,k}^{A} r_{j,k}^{B}}{1/p_j^{l,B} + 1/p_j^{l,A} - 1}$$

Let $a = 1/p_j^{l,B}, b = 1/p_j^{l,A}$, then $1 < a, b < \infty$.

$$R_{j,k}^{l,2} = \frac{a r_{j,k}^{A} + b r_{j,k}^{B} - r_{j,k}^{A} r_{j,k}^{B}}{a + b - 1} \tag{3.4}$$

$$= r_{j,k}^{A} + \frac{b(r_{j,k}^{B} - r_{j,k}^{A}) + r_{j,k}^{A}(1 - r_{j,k}^{B})}{a + b - 1} \tag{3.5}$$

$$= r_{j,k}^{B} + \frac{a(r_{j,k}^{A} - r_{j,k}^{B}) + r_{j,k}^{B}(1 - r_{j,k}^{A})}{a + b - 1} \tag{3.6}$$

Considering $r_{j,k}^{B} > r_{j,k}^{A}$, $r_{j,k}^{B} = r_{j,k}^{A}$, and $r_{j,k}^{B} < r_{j,k}^{A}$ three different cases, and the condition $0 < r_{j,k}^{A}, r_{j,k}^{B} < 1$,

1) If $r^B_{j,k} > r^A_{j,k}$, from equation (3.5), the second term is larger than 0. When $a = \infty$ *and* $b \neq \infty$, we can get $\text{Min}\,(R^{l,2}_{j,k}) = r^A_{j,k}$. When $a = 1$ *and* $b = 1$, we can get $\text{Max}\,(R^{l,2}_{j,k}) = r^A_{j,k} + r^B_{j,k} - r^A_{j,k} r^B_{j,k}$.

2) If $r^B_{j,k} = r^A_{j,k}$, from equation (3.5), the second term is larger than 0. When $a = \infty$ *or* $b = \infty$, we can get $\text{Min}\,(R^{l,2}_{j,k}) = r^A_{j,k}$. When $a = 1$ *and* $b = 1$, we can get $\text{Max}\,(R^{l,2}_{j,k}) = r^A_{j,k} + r^B_{j,k} - r^A_{j,k} r^B_{j,k}$.

3) If $r^B_{j,k} < r^A_{j,k}$, from equation (3.6), the second term is larger than 0. When $b = \infty$ *and* $a \neq \infty$, we can get $\text{Min}\,(R^{l,2}_{j,k}) = r^B_{j,k}$. When $a = 1$ *and* $b = 1$, we can get $\text{Max}\,(R^{l,2}_{j,k}) = r^A_{j,k} + r^B_{j,k} - r^A_{j,k} r^B_{j,k}$.

From above three conditions, we can get

$$\text{Min}\,(\mathrm{r}^{\mathrm{A}}_{\mathrm{j,k}}, r^B_{j,k}) \leq R^{l,2}_{j,k} \leq \mathrm{r}^{\mathrm{A}}_{\mathrm{j,k}} + r^B_{j,k} - \mathrm{r}^{\mathrm{A}}_{\mathrm{j,k}} r^B_{j,k} \qquad (3.7)$$

Equation (3.7) can be rewritten as the following,

$$\text{Min}\,(R^{l,n-1}_{\mathrm{j,k}}, r^n_{j,k}) \leq R^{l,n}_{j,k} \leq R^{l,n-1}_{\mathrm{j,k}} + r^n_{j,k} - R^{l,n-1}_{\mathrm{j,k}} r^n_{j,k} \qquad (3.8)$$

From (3.3) and (3.8), the end-to-end relative DVP $R^{l,n}_{j,k}$ is larger than the minimum relative DVP $r^i_{j,k}$ among all the nodes across the path, which is the left side of equation (3.1). The right side of (3.8) can be further rewritten as,

$$R^{l,n}_{j,k} < R^{l,n-1}_{j,k} * (1 - r^n_{j,k}) + r^n_{j,k} \qquad (3.9)$$

Because $r^n_{j,k}$ is a configuration value and $0 < r^n_{j,k} < 1$, and $R^{l,n-1}_{j,k}$ satisfies Equation (3.3), when $R^{l,n-1}_{j,k}$ is the upper bound of Equation (3.3), $R^{l,n}_{j,k}$ can be the upper bound too. Substituting the upper bound of $R^{l,n-1}_{j,k}$ into the right side of Equation (3.9), we can get the right side of the equation (3.1). Hence we prove theorem 1. ■

The assumption in Theorem 1 is reasonable. It is pointed out in [67] that *the single hop delays incurred by a packet are independent random variables, particularly when link utilizations are high, with ON/OFF traffic sources traversing 5 hops in length.* Similarly, the independence assumption on the link delays in an end-to-end path has been referred in [68]. In general, with the occurrence of events of deadline violation, the network will have relatively heavy traffic. Hence, the above conclusion can be applied to end-to-end relative DVP.

From Theorem 1, we have the following observation: The end-to-end relative DVP ratio $R^{l,n}_{j,k}$ for an end-to-end path is dependent on the relative DVP ratios $r^i_{j,k}$ of all nodes on the path. After configuring each single node DVP ratio $r^i_{j,k}$, we can derive the bound on the end-to-end DVP ratio $R^{l,n}_{j,k}$. If traffic traverses the network through different paths, the bounds on the end-to-end relative DVP are different based on Theorem 1. Hence, Theorem 1 depends on path in general.

Figure 3.4 shows the case that the relative DVP ratio of node 1 is set to 0.5 and the relative DVP ratio of node 2 changes from 0 to 1. The difference between upper and lower bounds is the smallest when the two DVP ratios are equal. This conclusion is not difficult to see intuitively.

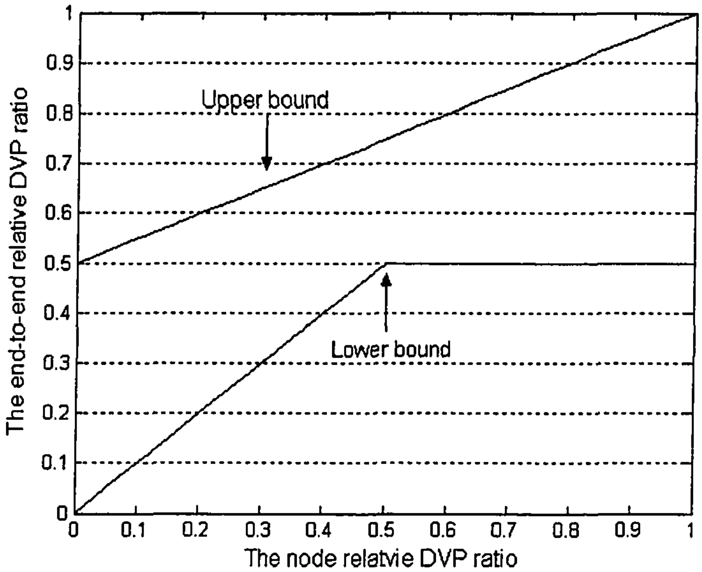


Figure 3.4 The end-to-end relative DVP vs. the node relative DVP: node 2 relative DVP ratio is set to vary with $r_{j,k}^{2} \in (0,1)$ and node 1 relative DVP ratio is set to $r_{j,k}^{1} = 0.5$

The results derived from Theorem 1 are hardly impressive especially when the number of nodes is large. Different paths also result in different end-to-end bounds. Hence, it is hard to apply Theorem 1 directly to the real network. We study some special cases of Theorem 1 and try to find much tighter bound on the end-to-end DVP analysis.

Figure 3.5 shows the end-to-end relative DVP bounds for paths with 2, 3 and 4 nodes respectively where all nodes are configured with the same relative DVP ratios. As is increased n, the bounds seem to become looser.

An interesting and very useful special case is that, if we configure all $r_{j,k}^{i}$ for the same class j and k to be the same across all nodes in the network, the actual values are typically very close to the lower bound, i.e. $R_{j,k}^{l,n} \approx r_{j,k}^{i}$.

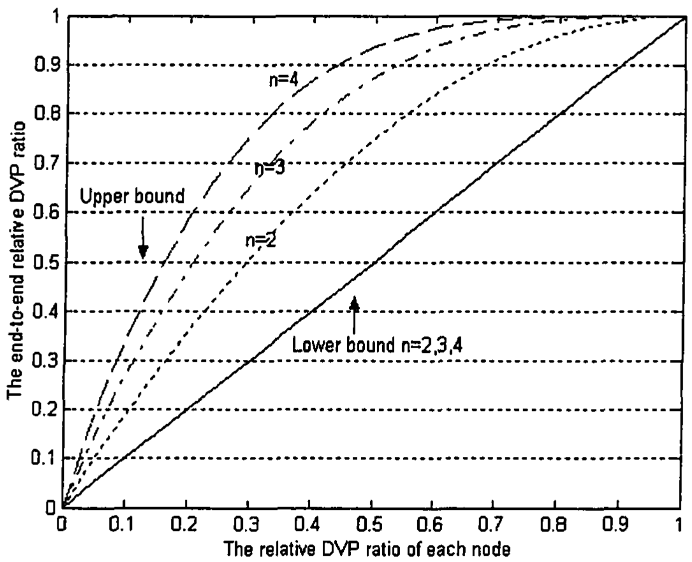


Figure 3.5 The end-to-end relative DVP bounds for paths with 2. 3 and 4 nodes respectively under the equal node relative DVP ratios $r_{j,k}^{i} = r$ $and\ 1 \le i \le 4$

Theorem 2: Assume $a\ and\ b >> 1$ in equation (3.4), if all $r_{j,k}^{i}$ for the same class j and k across all nodes on path l are the same, i.e. $r_{j,k}^{i} = r$, then the end-to-end relative DVP ratio $R_{j,k}^{l,n}$ of the specific path l is $R_{j,k}^{l,n} \approx r_{j,k}^{i} = r$.

Proof: Consider an example path with two nodes (more general case can be proved using induction), from equation (3.4), we have

$$R_{j,k}^{l,2} = \frac{ar + br - r * r}{a + b - 1} = r + \frac{r(1-r)}{a+b-1} \qquad (3.10)$$

The absolute error from the lower bound will be,

$$\Delta = R_{j,k}^{l,2} - r = \frac{r(1-r)}{a+b-1} \qquad (3.11)$$

We define variables a and b as the reciprocals of the absolute DVP. Typically, the absolute DVP is much smaller than 1, so we have $a \text{ and } b >> 1$, also $0 < r < 1$, then the absolute error $\Delta << r$. Therefore, we have $R^{l,2}_{j,k} \approx r^{i}_{j,k} = r$. ■

For example, assume that the absolute DVP $p^{l,i}_{j}$ of each node is 10% and $a = b = 10$ then $r_{\varepsilon} \in (0\,, 0.013)$ as shown in Figure 3.6.

Theorem 2 with the configuration of equal relative ratio for all nodes is important and complementary to Theorem 1. The results derived from Theorem 2 are much tighter compared with equation 1 in Theorem 1 even when the number of hops is large.

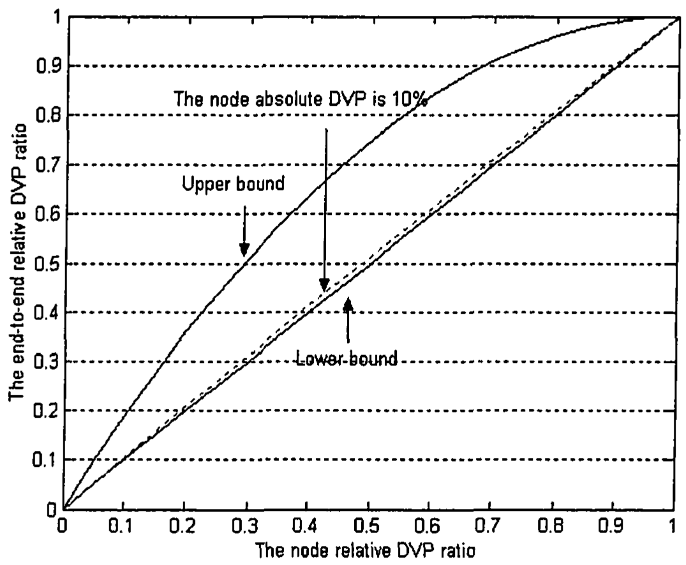


Figure 3.6 The special case of the end-to-end relative DVP for path with 2 nodes under the equal node relative DVP ratios $r^{1}_{j,k} = r^{2}_{j,k} \in (0\,, 1)$ and $p^{l,1}_{j} = p^{l,2}_{k} = 10\%$

The advantage of Theorem 2 over Theorem 1 is that Theorem 2 is not dependable on path. Based on Theorem 2, when we configure all nodes with equal node relative ratios, we can have the same end-to-end relative value across the same network. Hence, we can apply the Theorem 2 directly to the real networks. Therefore we recommend the ratios $r_{j,k}^{i}$ for the same class j and k be set to the same value across a network wherever it is feasible.

3.3 The proposed scheduler: Dynamic Deficit Round Robin (DDRR)

The main reasons for choosing DDRR in our work are its simplicity and its ease of implementation in hardware in comparison with other GPS based schedulers, such as WFQ scheduler. It also can guarantee the minimum bandwidth for each class. In GPS, the weight for each queue is static; DDRR is the extension of GPS, in which the weight can be adjusted dynamically to achieve the target. The novelty of our DDRR is that we apply it to maintain the DVP to satisfy real-time application QoS requirements.

A workable scheduler must provide means for specifying performance objectives for different classes of traffic as well as a means of delivering on those performance objectives. Generally, a scheduler cannot create additional bandwidth; thus, when some packets get better treatment, other packets will get worse treatment. The performance objective in the thesis is to ensure that the DVP p_i^n experienced by class i at node n satisfy the following constraint,

$$\frac{p_i^n}{p_j^n} = \frac{\delta_i^n}{\delta_j^n} = \hat{r}_{j,i} \quad 1 \le i, j \le N \qquad (3.12)$$

where δ_i^n is DVP differentiation parameter (DDP) of class i at node n and $\hat{r}_{j,i}$ is the target DVP ratio between class i and class j. The higher classes are of better quality i.e. lower DVPs, so $\delta_1^n > \delta_2^n > \cdots > \delta_N^n$. For the further discussion in the thesis, we can use the concept of normalized DVP $\tilde{p}_i^n = \frac{p_i^n}{\delta_i^n}$; then the equation 3.12 can be rewritten as the following,

$$\tilde{p}_i^n = \frac{p_i^n}{\delta_i^n} = \frac{p_j^n}{\delta_j^n} = \tilde{p}_j^n \quad 1 \le i, j \le N \qquad (3.13)$$

Equation 3.13 means the normalized DVPs should be equal for all classes of traffic, i.e. the DVP is proportional to its DVP differentiation parameter set by the network administrator.

3.3.1 System architecture

Our goal is to provide a proportional DVP among classes in a single node. The scheduler works in a work-conserving, loss-less, and non-preemptive mode. The work-conserving mechanism means that scheduler is never idle if there are backlogged packets in some (or all) classes. The loss-less assumption requires there are enough buffers for those backlogged packets. The non-preemptive assumption implies that the transmission of a packet cannot be interrupted. Figure 3.7 shows system architecture.

Traffic flows from different sources are classified into a set of classes by a classifier according to their contracts. In our model, we assume that the buffer of each class is infinite in a router and therefore packet dropping due to a full queue is not considered. We further assume that packets are marked due to delay deadline violation. For simplicity, we do not consider processing delay in this case. If sum of queueing delay and transmission delay of a packet is larger than the delay bound, the packet is considered to have a deadline violation.

The DVP measurement process first estimates the absolute DVP of each class based on the number of departures and the number of departures with deadline violation and then calculate the relative DVP ratios between classes. The weight adjustment process adjusts the weight of each class based on DVP measurement results. The DDRR scheduler schedules packets using deficit round robin algorithm after obtaining the

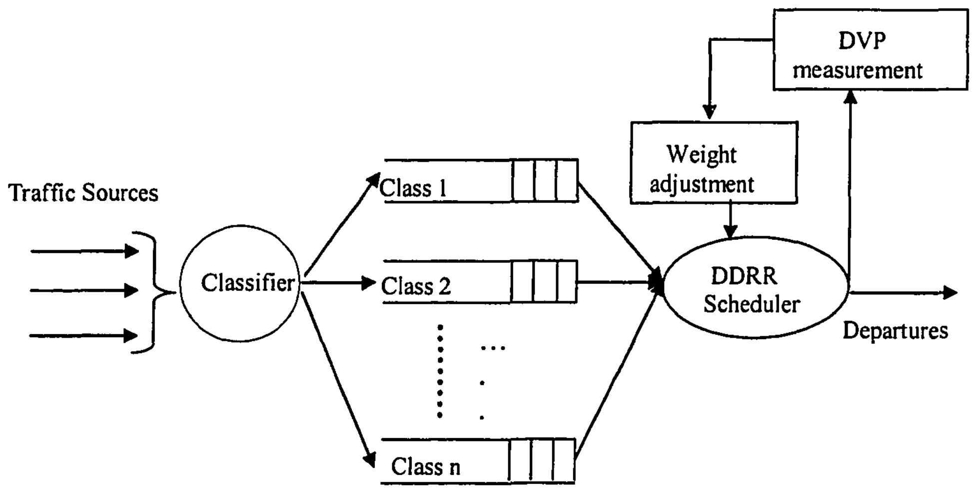


Figure 3.7 DDRR system architecture

weight of each class. We will describe the DDRR scheduler, the DVP measurement algorithm and the weight adjustment algorithm in more detail in following subsections.

3.3.2 DDRR scheduler

We start with the deficit round robin (DRR) scheduler. The goal of DRR scheduler is to choose next packet to be served. In DRR, queues are served in a round robin fashion, taking packets from each nonempty queue in turn. The number of bits of each queue to be served in one round is calculated from the weight (quantum) of this queue. If a packet size is larger than the quantum size, the packet has to wait for another round. In this case, a deficit counter is incremented by a size of the quantum and can be used by next round.

In our proposed DDRR, the weight of each class is adjusted dynamically based on measurement results. The round number R in DRR is used as the DVP measurement interval and the weight adjustment interval i.e. we measure the DVP and make an adjustment of weight every R rounds.

3.3.3 DVP measurement algorithm

There are different ways to measure the absolute DVP of each class. Here we employ an exponential averaging algorithm as shown in Figure 3.8. Assuming $DI[i]$ is the absolute DVP value of class i in a measurement interval, $DTB[i]$ is the averaged exponential averaging DVP value of class i of the history and the current measurement interval, we have,

Algorithm 1: exponential averaging algorithm

ρ : the configuration parameter.
I : the set of all classes
$BI[i]$: counter of packets with deadline violation from class i in measurement interval.
$AI[i]$: counter of packets from class i in measurement interval.
$B[i]$: counter of packets departed with deadline violation from class i on the history
$A[i]$: counter of packets departed from class i on the history.
$DI[i]$: the DVP of class i in measurement interval.
$DTB[i]$: the DVP of class i on the history before the measurement interval.
$DT[i]$: the achieved DVP of class i based on the history and the measurement interval.

At the end of each measurement interval:

```
for ( i ∈ I )
  A[i] += AI[i]
  B[i] += BI[i]

/* to calculate the DVP in measurement interval */
  DI[i] = BI[i] / AI[i]

/* exponential averaging */
  DT[i] = (1 - ρ) * DTB[i] + ρ * DI[i]

/* to calculate the history DVP value for the next time exponential averaging */
  DTB[i] = B[i] / A[i]
endLoop
```

Figure 3.8 DVP exponential averaging measurement algorithm of DDRR

$$DT[i] = (1-\rho) * DTB[i] + \rho * DI[i] \qquad (3.14)$$

where ρ is the weight of the measurement interval DVP value in DVP calculation. As ρ is increased, it will make the averaging process more adaptive to load changes and as ρ is decreased, it will give a smoother average by keeping a longer history.

In order to estimate the DVP of class *i* in the measurement interval, the scheduler maintains two counters,

- $BI[i]$ is used to count the number of packets departed with deadline violation from class *i* in the measurement interval.

- $AI[i]$ is used to count the number of packets departed from class i in the measurement interval.

The estimated DVP for the current measurement interval for class i is given by the ratio of the current values of two counters.

3.3.4 Weight adjustment algorithm

After obtaining the absolute DVP of all classes, the scheduler dynamically adjusts the weight of each class based on the difference of the achieved DVP ratio and the target DVP ratio. Figure 3.9 illustrates the weight adjustment algorithm in more detail.

Algorithm 2: Weight adjustment algorithm

α : the configuration parameter.
I : the set of all classes
$DI[i]$: the DVP of class i in measurement interval.
$DT[i]$: the achieved DVP of class i based on the history and the measurement interval.
k : the first class number which $DI[k]$!= 0 in measurement interval as base class
weight [i] : the weight of class i in DDRR at next measurement interval.

At the end of each measurement interval:

```
if ( DI[i] == 0, i ∈ I )
   /* no DVPs in measurement interval, keep the weights */
elseif
   /*adjust weight toward target ratio */
   for ( i ∈ I )
       /* the DVP ratio is larger than the target ratio, increase the weight of class i */
       if( DT[i] / DT[k] > target )
          weight [i] = weight [i] * (1 + α )
       elseif
          /* the DVP ratio is smaller than the target ratio, decrease the weight of class i */
          weight [i] = weight [i] * (1 - α )
       endif
   endLoop
endif
```

Figure 3.9 Weight adjustment algorithm of DDRR

The scheduler first chooses the lowest class with non-zero DVP in the measurement interval as a base class. Then it compares all the other classes with the base class to get achieved DVP ratios. The weight of each class is adjusted in turn. If the achieved DVP ratio of class i to the base class is larger than the target DVP ratio, the scheduler increases the weight of class i proportionally by a ratio α,

$$weight[i] = weight[i] * (1+\alpha) \qquad (3.15)$$

If the achieved DVP ratio is smaller than the target DVP ratio, it decreases the weight of class i proportionally by a ratio α,

$$weight[i] = weight[i] * (1-\alpha) \qquad (3.16)$$

where α is a configuration parameter (between 0 and 1) and controls how much the weight is adjusted each time.

The round number R in DRR is the weight adjustment interval. A larger R will keep the weight steadier and make DVP ratio less synchronized with the target ratio. Weight is updated more frequently given a smaller R, and this will increase complexity. By always keeping DVP ratios of traffic contract as weights updating goal, the scheduler would achieve proportional deadline violation probability gradually.

These parameters α, ρ and R are configuration parameters. For simplicity, we set the DVP measurement interval and the weight adjustment interval using same round number R parameter in the scheduler. The method of choosing these parameters is a design issue. We will study the effect of these parameters on performance of the scheduler and set them through a set of empirical study in next chapter.

In addition to the algorithms described in Figure 3.8 and 3.9, we can also configure each class or some classes with the minimum bandwidth guarantee. When the adjustment of the DDRR algorithm reach a point that one of the guaranteed class is receiving bandwidth less than or equal to the guaranteed minimum bandwidth, the adjustment stops.

The computational complexity of DDRR depends on the number of traffic classes. In DiffServ networks, traffic flows have been grouped to a very limited number of classes. So the complexity of DDRR is not a real issue.

Chapter 4

Performance Evaluation

In this chapter, we start with evaluating the effectiveness of the proposed DDRR scheduler in providing proportional DVP differentiation for a single node network. After applying it to a multi-node network, we illustrate the achieved end-to-end relative DVP results and compare them with analytical bounds. The efficiency of the proposed DPCF, in terms of the end-to-end absolute DVP, is then assessed. Three different topologies have been employed in our simulations for the evaluation of a single node case and a multi-node network respectively. All simulations are executed on the OPNET 8.0 platform.

4.1 DDRR scheduler at the single node

We present simulation results of DDRR, with respect to its controllability and predictability, in various load conditions. Its sensitivities to some configuration parameters, such as the weight of the interval DVP in the DVP calculation, the weight adjustment interval and the weight adjustment ratio, are examined. Also we simply discuss feasibility issues on this model and give an example of its infeasibility. At last, we compare the proposed DDRR algorithm with WEDD algorithm.

4.1.1 Simulation setup

The simulated topology adopted for a single node is shown in Figure 4.1. The traffic sources in our simulation are ON/OFF model. Each user has exponential ON and OFF periods. The average durations of ON period is 100 ms and OFF period is 158 ms respectively. The packet size distribution is a Poisson distribution with the mean packet size 1024 bits.

Two groups of users (source 1 and source 2) generate packets corresponding to class 1 and class 2 of traffic respectively. At both sources, each user adopts the ON/OFF traffic model to generate packets streams. Let S_1 be the number of users in source 1 and S_2 be the number of users in source 2. Hence, by adjusting value of S_1 and S_2 or the sending rate of traffic source during ON periods, we can obtain different traffic load conditions, such as different utilizations and different traffic load distributions. We assume that class 1 has a lower priority i.e. higher DVP, and class 2 has a higher priority.

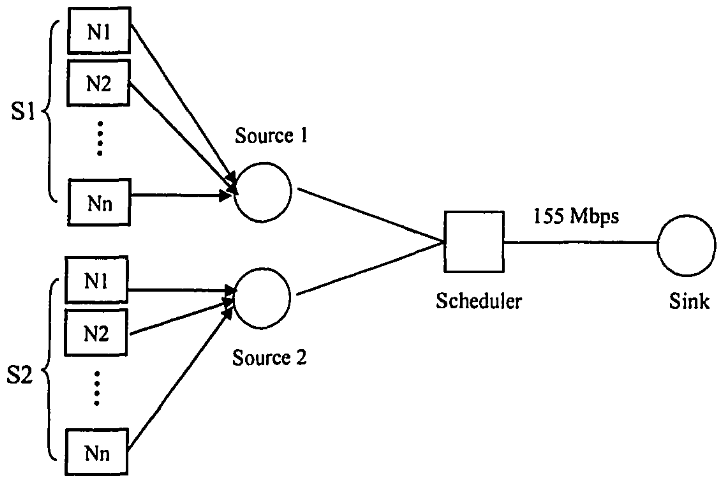


Figure 4.1 The single node network topology

The packets in each class are characterized by a delay bound based on configuration parameters. Here, we set the delay bound for both classes of traffic to 10 ms. the link speed is set to OC-3 155 Mbps.

In the following two sections, we investigate two requirements for the proportional differentiation model in DDRR implementation: predictability and controllability. Predictability means that the relative differentiated service between different classes should be consistent with their DVP differentiation parameters. Controllability means that the network administrator should be able to adjust the quality spacing between two classes. When it is feasible, such controllability should be valid regardless of traffic characteristics.

4.1.2 Effect of link utilization

We first investigate two requirements presented above under different link utilizations, with two classes of traffic. The goal of this simulation is to evaluate the consistency of achieved DVP ratios $r_{2,1}$ with the target DVP ratios $\hat{r}_{2,1}$ in long timescales and also under different target DVP ratios $\hat{r}_{2,1} = 2, 8, 16$ respectively.

We varied link utilization by adjusting the sending rate during ON periods. The utilization is set from 65% to 100%. The traffic load ratio of source 1 to source 2 is 1:1 and we set $S_1 = S_2 = 20$ users. Through an empirical study as shown in section 4.1.4, the configuration parameters are set as follows: R=1600, $\alpha = 0.1$ and ρ ranges from 0.01 to 0.00001 as utilization changes from 100% to 65%.

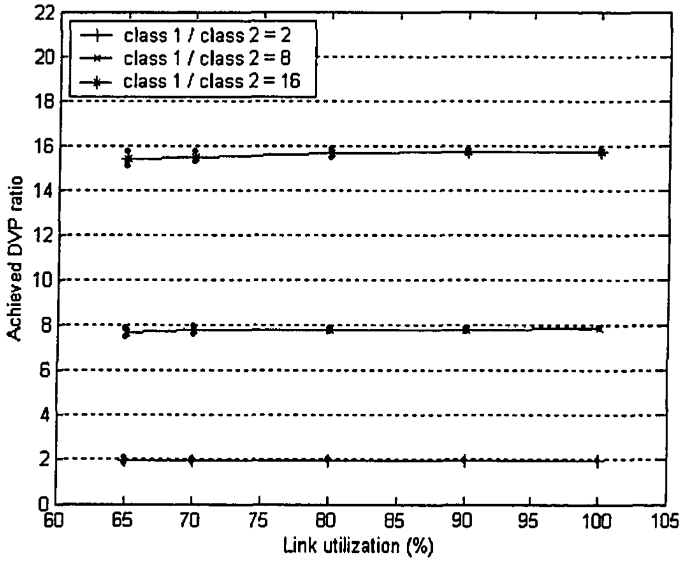


Figure 4.2 Achieved DVP ratios vs. link utilization under different DVP target ratios $\hat{r}_{2,1} = 2, 8, 16$ with 95% CI

The simulation results are summarized in Figure 4.2 in which the x-axis is the link utilization and the y-axis is the achieved DVP ratio between class 1 and class 2. Figure 4.3 shows the detailed scheduler working process at utilization 75%. From Figure 4.2 and 4.3, we make the following observations:

1. DDRR scheduler satisfies the predictability and the controllability requirements: the achieved DVP ratios are consistent with the target DVP differentiation parameters and the differentiation parameters can be adjusted to the different quality 2, 8 and 16 respectively.

2. In Figure 4.3, when the DVP ratio is higher than the target ratio 2 in time periods t_1 and t_2 , t_3 and t_4, the weight of class 1 traffic is increased, thus decreasing the achieved DVP ratio. Between t_2 and t_3, t_4 and t_5 time periods, the DVP ratio is

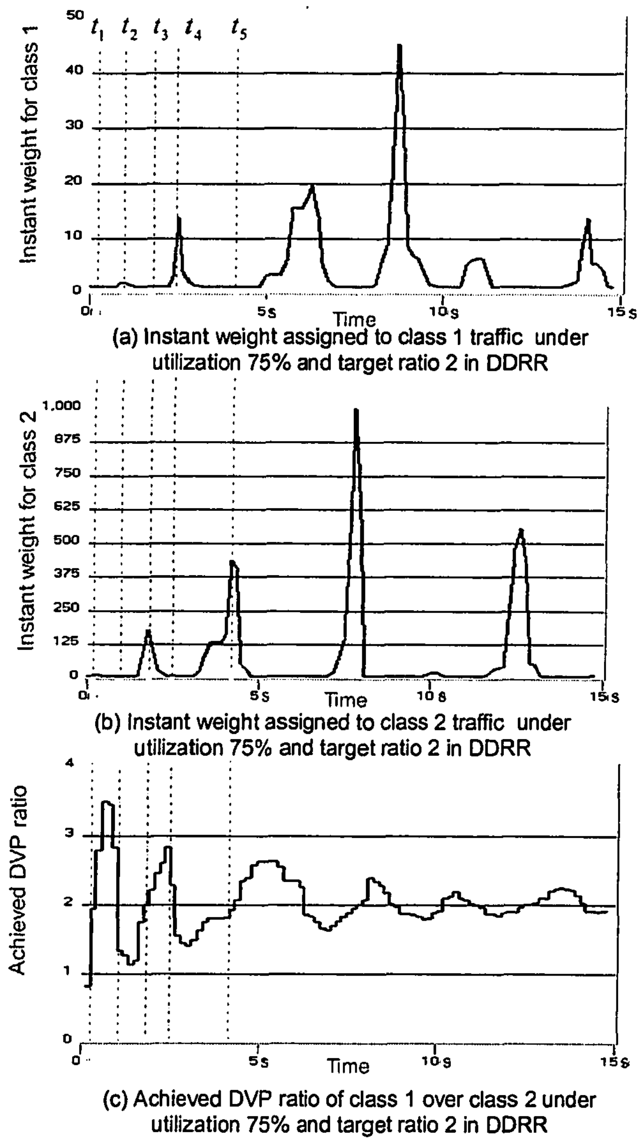


Figure 4.3 A sample path of the achieved DVP ratio at utilization 75% and the target ratio 2

lower than the target ratio, the weight of class 2 traffic is increased. DDRR converges faster at lower DVP target ratio or with heavy load condition. Because we set the load ratio between two classes of traffic is 1:1, it is easier to adjust the weight to meet the lower target ratio. When a network is underutilized, there are not enough packets to exceed the deadline. The work-conserving mechanism of schedulers is never idle if there are backlogged packets in classes. Hence, the underutilized condition will make the scheduler take a longer time to achieve the desired DVP differentiation. However, in an underutilized network, the network does not suffer congestion and the packets experience very low DVP. For example, in this scenario, the DVP value can be in the order of about 10^{-5} at utilization 65%; therefore DVP differentiation is not really necessary in this case. DiffServ is designed to classify and prioritize traffic such that when congestion occurs, higher priority traffic will be forwarded and lower priority traffic will suffer as necessary.

4.1.3 Effect of traffic load distribution

In this section, we continue to investigate the predictability and controllability of the DDRR scheduler under different traffic load distributions with two classes of traffic. The goal of the simulation is to achieve the target DVP ratios (set to 2, 4, and 8 respectively) as close as possible.

The total link utilization is set at 90%, with the total number of users S_1 and S_2 set to 40. The distributions of two classes are varied by changing the number of users S_1 and S_2. The other parameters are the same as the preceding scenario using different link utilizations with R=1600, $\alpha = 0.1$ and $\rho = 0.01$.

Figure 4.4 depicts the performance of DVP ratios under the condition that the traffic load distribution of class 2 is varied from 10% to 90%. The x-axis is the percentage of class 2 in the total traffic and the y-axis is the achieved DVP ratios between class 1 and class 2. From Figure 4.4, we make the following observations:

1. The achieved DVP ratios with the DDRR scheduler closely approximate different target DVP ratios between two classes under different load distributions. Both predictability and controllability can be satisfied under this scenario.

2. The achieved DVP ratios do not much depend on traffic load distributions if the proportional DVP model is feasible. From 10% to 90% of percentage of class 2 traffic, the achieved DVP ratios are very similar. The feasibility issues of proportional DVP differentiation model will be discussed in more detail in section 4.1.5.

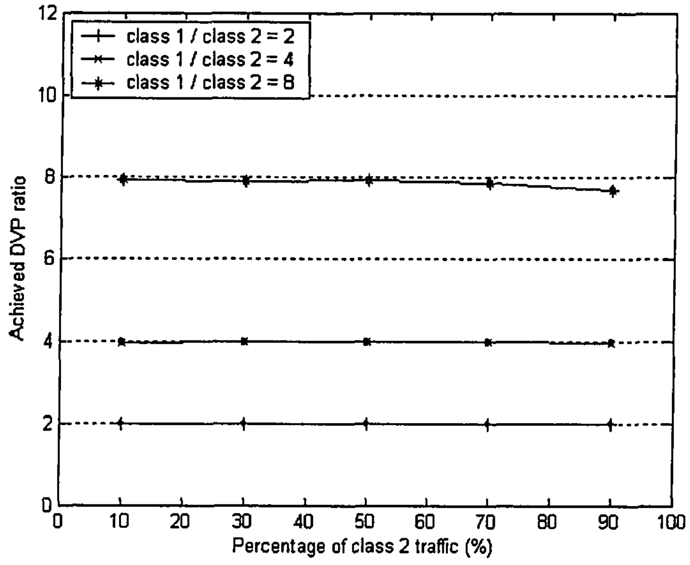


Figure 4.4 Achieved DVP ratios vs. traffic load distribution under different DVP target ratios $\hat{r}_{2,1}$ = 2, 4, 8 with 95% CI

4.1.4 Sensitivity of performance to configuration parameters

There are some configuration parameters such as the weight ρ of the interval DVP in the DVP calculation, the weight adjustment interval R, the weight adjustment ratio α in the DDRR scheduler. How to choose these parameters is an important design issue. An appropriate choice should strike a good balance between system stability and responsiveness. Therefore, these parameters can be tunable parameters for network administrators.

In this section, we study the effect of configuration parameters on performance of the DDRR scheduler. We classify simulations into three types, with the following goals,

Type A: Evaluate the sensitivity of performance to the weight ρ of the interval DVP in the DVP calculation under different link utilizations.

Type B: Evaluate the sensitivity of performance to the weight adjustment interval R under different link utilizations.

Type C: Evaluate the sensitivity of performance to the weight adjustment ratio α under different link utilizations.

The simulation parameters are the same used in the varying link utilization scenario. Here we choose two utilizations 80% and 90%, and adjust the concerned parameter while keeping the other parameters same for each experiment.

Type A: Effect of the weight ρ of the interval DVP in the DVP calculation

As mentioned in chapter 3, the larger ρ makes an averaging process more adaptive to load changes and the smaller ρ gives a smoother average by keeping a longer history. We carry out simulations with different weight settings in the section. The simulation results are summarized in Figure 4.5.

The required weight ρ to meet the target DVP ratio within a certain error tolerance decreases as the utilization decreases. In heavy load conditions (u=90%), the selection of ρ need to be set 0.01 or smaller and DVP ratio is closely approximated to the target ratio. In medium load conditions (u=80%), ρ needs to be close to 0.001 or smaller in order to arrive to the target DVP ratio. The reason is that the lower load utilization, the lower the chance of departures with deadline violation. When DVP is lower, the scheduler needs keeping a longer history to achieve the target value. Hence, ρ is sensitive to the traffic load and depends on absolute DVP. In real networks, in order to make the DDRR scheduler work very well under different traffic load, we should set ρ small to decrease the effect of parameter and do not need to measure the real traffic load. For example, in section 4.1.1, ρ is 0.01 at 90% utilization, 0.001 at 80% utilization and 0.0001 at 70 % utilization.

When utilization is constant, the absolute DVP value of the same class is similar under different traffic load distributions i.e. in scenario 4.1.2. Hence, ρ is not sensitive to traffic load distributions and we set ρ a constant in section 4.1.2. We only evaluate the sensitivity of ρ under different link utilizations in this section.

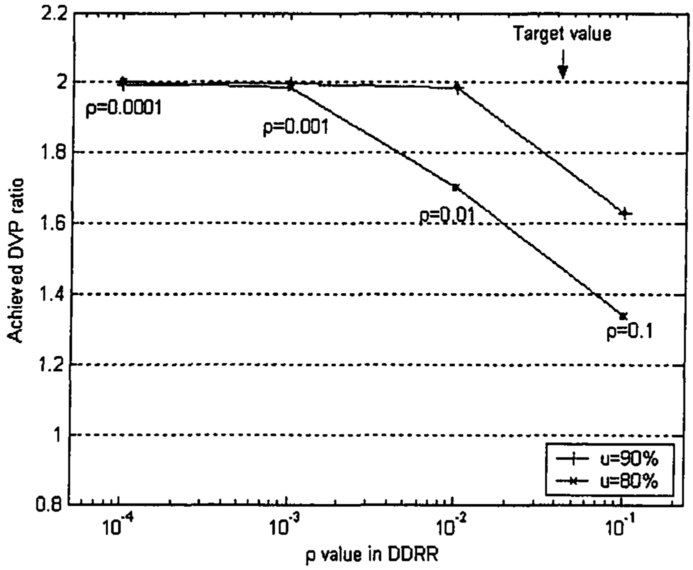


Figure 4.5 Achieved DVP ratios vs. ρ value in DDRR under different utilizations with target ratio $\hat{r}_{2,1} = 2$ and 95% CI

Type B: Effect of the weight adjustment interval R

We carry out simulations with different weight adjustment interval settings. The simulation results are summarized in Figure 4.6. The required interval R to meet the target DVP ratio within a certain error tolerance increases as the utilization decreases. A larger R keep the weight more steady and make DVP less synchronized with the target ratio. A smaller R make the weight update frequently and increase the complexity. As an optimal value R makes a good tradeoff between performance and overhead, we choose R = 1600.

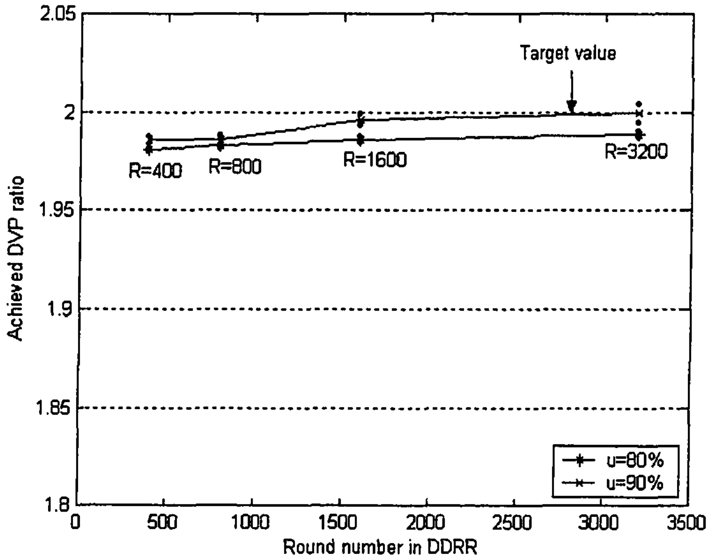


Figure 4.6 Achieved DVP ratios vs. round number in DDRR under different utilizations with target ratio $\hat{r}_{2,1} = 2$ and 95% CI

Type C: Effect of the weight adjustment ratio α

We carry out simulations with different weight adjustment ratio settings. The simulation results are summarized in Figure 4.7. From the figure, we can observe that the scheduler performance becomes insensitive to the weight adjustment ratio under both heavy load conditions and medium load conditions. A larger weight adjustment ratio can make a fast convergence to the target DVP value because we adjust the weight quickly at each time.

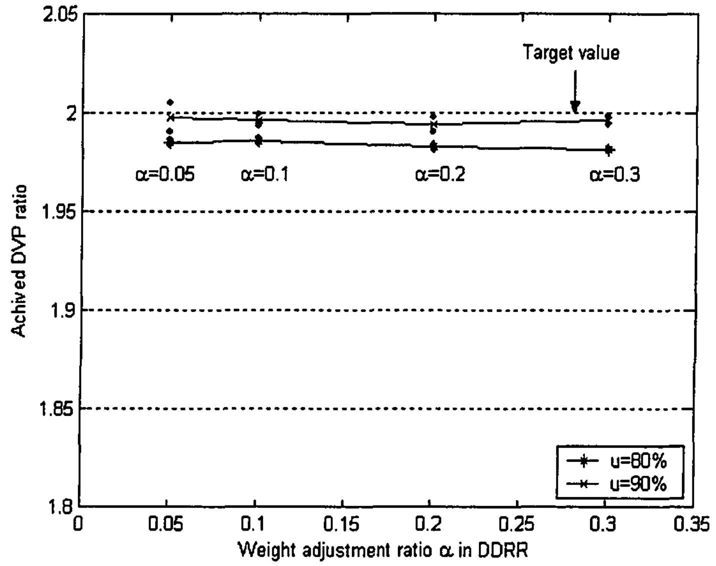


Figure 4.7 Achieved DVP ratios vs. weight adjustment ratio in DDRR under different utilizations with target ratio $\hat{r}_{2,1} = 2$ and 95% CI

4.1.5 Feasibility issues

We have assumed so far the proportional DVP differentiation model is feasible. However, this is not always possible, i.e. there may not exist a work-conserving scheduler that can achieve target DVP ratios specified by DVP differentiation parameters. The achieved DVP ratios will deviate from target DVP ratios. The infeasibility of the schedulers on proportional differentiation model is common, e.g. in [13][14] [15][16] [26] the schedulers are infeasible under some specified conditions.

These deviations in DDRR can become a general case, depending on class load distributions and target DVP ratios etc. To illustrate some of these conditions, Figure 4.8 shows the achieved DVP ratios of DDRR, an example of 'hard to achieve' DVP differentiation.

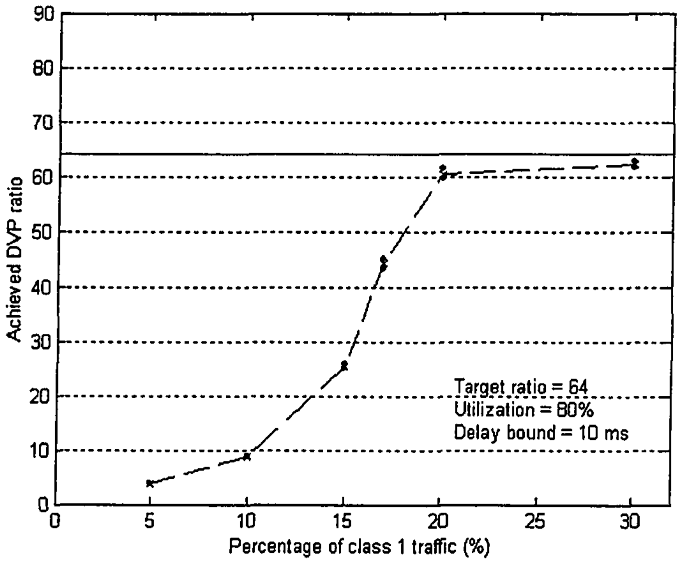


Figure 4.8 An example of DDRR when the proportional DVP differentiation is infeasible with target ratio $\hat{r}_{2,1} = 64$ and 95% CI

In Figure 4.8, class 1 does not have enough arrivals compared with class 2 when the percentage of class 1 is lower than 17%, and so it cannot maintain the required backlog for the DVP differentiation. A higher target DVP ratio ($\hat{r}_{2,1} = 64$), also makes the DVP differentiation infeasible, because there are not enough class 1 packets with deadline violation in order to make the DVP of class 1 64 times larger than the DVP in class 2. The weight adjustment method of DDRR couldn't achieve the exact target DVP.

The example of this section is intended to provide some intuition on feasibility issues. In general, due to many factors that affect the resulting DVP ratio, it is hard to provide quantitative guidelines for the model. This is a common problem for all relative QoS schemes. [13][14][15][16] [26].

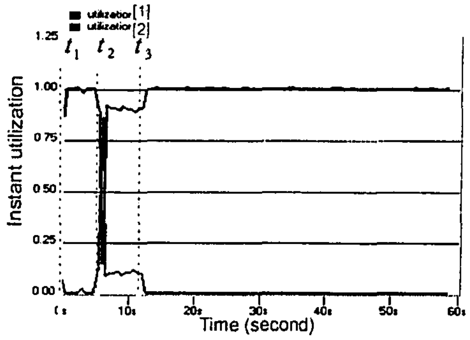
4.1.6 Starvation problem in WEDD

As we mentioned above, our DDRR can provide each class with the minimum bandwidth share and other priority-based schedulers have the starvation problem when one class is overloaded. We will illustrate this by comparing WEDD and DDRR performance in the following simulation results.

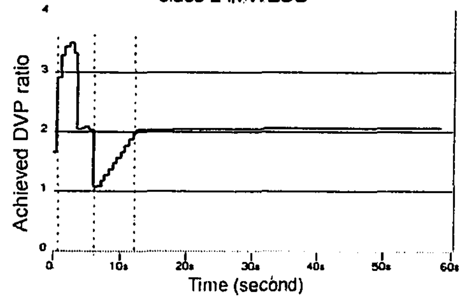
The simulation topology is shown in Section 4.1.1. The traffic source model is the same as the scenario of link utilization (Section 4.1.2). Both class 1 and class 2 have 20 users respectively with different sending rates: 18.94 packets/ms for class 1 and 1.89 packets/ms for class 2. Hence the class 1 generates 150 Mbps traffic and class 2 is 15 Mbps. The traffic load ratio is 10:1. The link capacity is 155 Mbps. The delay bound for both classes are set to 10 ms. The target DVP differentiation ratio is 2.

Although this scenario is not a steady-state scenario, the total traffic of all classes is higher than link bandwidth. This phenomenon exists for short term periods in real network. We study behaviors of two schedulers, WEDD and DDRR, under this scenario. In DDRR, when we adjust the weight distributed to each class, we set the minimum bandwidth 1/10 of total link bandwidth, i.e. we can guarantee that each class can obtain at least the minimum bandwidth.

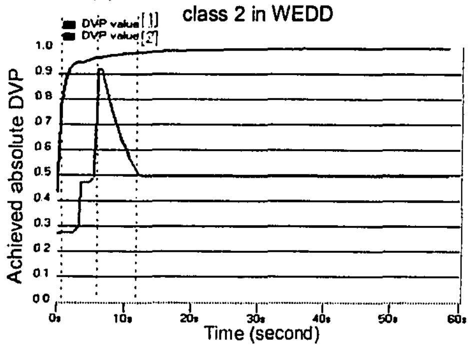
The simulation results for WEDD scheduler are summarized in Figure 4.9 in which the x-axis is the simulation time and three figures are: (a) the utilization for each class, (b) the achieved DVP ratio of class 1 to class 2 and (c) the achieved absolute DVP value for each class respectively. From Figure 4.9, we make the following observations:



(a) Instant utilization for class 1 and class 2 in WEDD



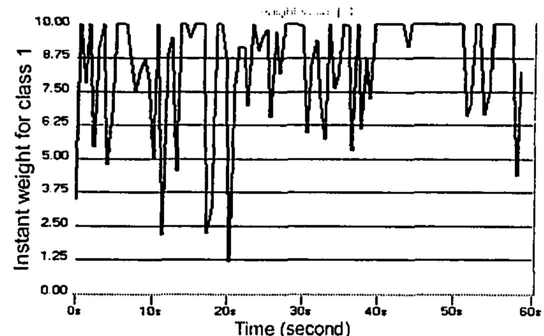
(b) Achieved DVP ratio of class 1 over class 2 in WEDD



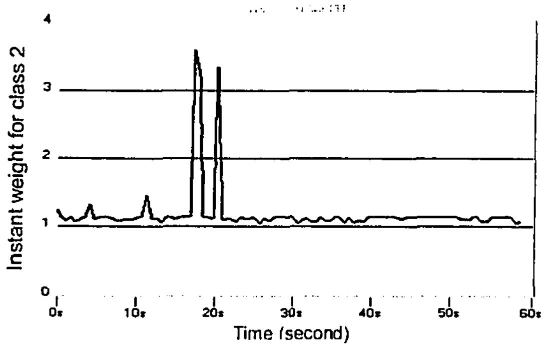
(c) Achieved absolute DVP for class 1 and class 2 in WEDD

Figure 4.9 Simulation results obtained by WEDD with utilization 106%, traffic load ratio 10:1 and target ratio $\hat{r}_{2,1} = 2$

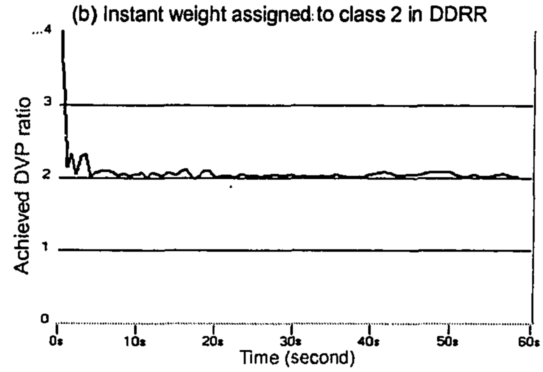
The overloading of one class results in the starvation of the other classes in WEDD scheduler. For example, when class 1 of traffic is approximately the link capacity and is 10 times of class 2 traffic, the measured utilization for class 1 is almost 100% and class 2 is starved. At the beginning, in time period t_1 and t_2, the achieved DVP ratio is higher than the target ratio, the WEDD scheduler try to schedule more class 1 packets and decrease the DVP ratio at the same time. Between t_2 and t_3 period, the achieved ratio is lower than the target ratio, more bandwidth is distributed to class 2 and thus increase the DVP ratio towards the target ratio. At time t_3, the achieved DVP ratio is higher than the target ratio again. Even the 100% bandwidth is distrusted to class 1 after time t_3, it is still hard to improve the DVP ratio to the target ratio at short time. This results in the starvation of class 2.



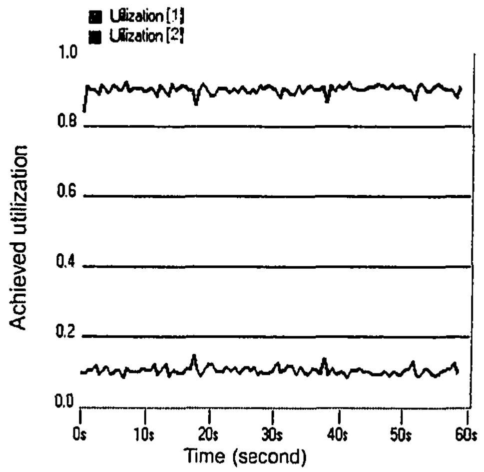
(a) Instant weight assigned to class 1 in DDRR



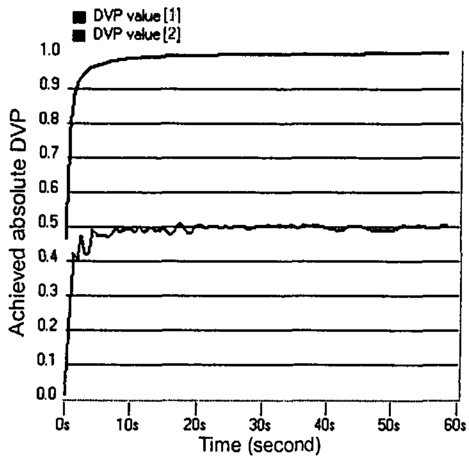
(b) Instant weight assigned to class 2 in DDRR



(c) Achieved DVP ratio of class 1 over class 2 in DDRR



(d) Utilization for class 1 and class2 in DDRR



(e) Achieved DVP value for class 1 and class 2 in DDRR

Figure 4.10 Simulation results obtained by DDRR with utilization 106%, traffic load ratio 10:1, target ratio $\hat{r}_{2,1} = 2$ and minimum bandwidth for each class is set to 1/10 of total link capacity

The simulation results are summarized in Figure 4.10 for DDRR scheduler in which the x-axis is the simulation time and five figures are: (a) the instant weight assigned to class 1, (b) the instant weight assigned to class 2, (c) the achieved DVP ratio of class 1 over class 2, (d) utilization for class 1 and class 2 and (e) the achieved DVP value for each class respectively. From Figure 4.10, we make the following observations:

There is no starvation problem in DDRR scheduler when one of the classes is overloading. From the Figure 4.9 (a) and (b), the minimum weight for each class can be guaranteed to at least 1/10 of total weight. Figure 4.9 (c) shows that the achieved DVP ratio is approximate to the target ratio. Figure 4.9 (d) illustrates that we can guarantee the minimum bandwidth for each class is 1/10 of total bandwidth by using DDRR. The utilization for class 1 is around 90% and for class 2 is around 10%. Figure 4.9 (e) shows achieved absolute DVP value for class 1 and class 2 is around

100% and 50% respectively. These results are consistent to the prediction when utilization is larger than 100%.

4.2 End-to-end relative DVP ratio

The simulation studies of previous sections focus on a single link and on the node relative DVP differentiation between different classes. Although such a study is interesting to network administrators, the network users would be more concerned about the end-to-end performance of their packet flows. Suppose that two flows from class *i* and *j* enter a network at the same time and they traverse the same path. The network attempts to provide locally in each link proportional DVP differentiation in the granularity of class traffic. Can a local relative differentiation lead to a consistent end-to- end relative differentiation? We attempt to investigate the above issue using simulations in this section and compare with our analytical results in chapter 3.

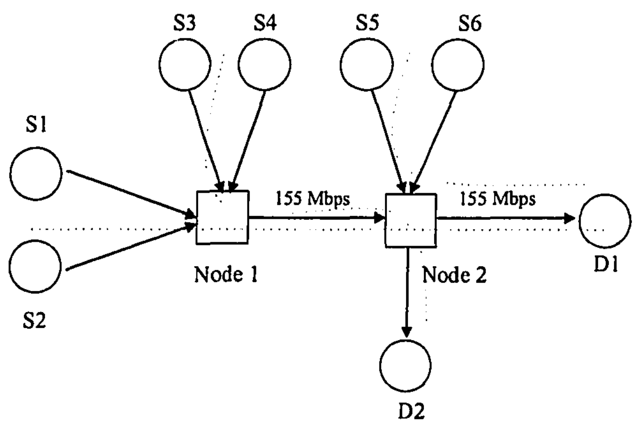


Figure 4.11 The multi-node network topology for relative DVP

In order to evaluate the end-to-end relative DVP, we consider the scenario as shown in Figure 4.11. Two classes of traffic from source 1 and 2 traverse a path that consists of two tandem nodes. Sources 3, 4, 5 and 6 are background traffic. All traffic sources are the same as the single node traffic model. We set sources 1 and 2 as having 20 users with 7.95 packets/ms sending rate in ON periods, and sources 3, 4, 5, and 6 as having 10 users with 0.88 packets/ms sending rate. Sources 1, 3 and 5 have class 1 of traffic and sources 2, 4 and 6 have class 2 traffic. Sources 1 and 2 traffic traverse node 1 and node 2 to destination 1, sources 3 and 4 to destination 2, sources 5 and 6 to destination 1. The traffic source parameters are summarized in table 4.1. The link capacity is 155 Mbps. Hence the link utilization of node 1 to node 2 and node 2 to destination 1 under this configuration are 85% and 90% respectively. We ignore propagation delays that are common to all packets and focus on queueing and transmission delays at this scenario. The equal delay bound at each node is set to 10ms. The other parameters are the same as the single node case.

The goal of the simulation is to evaluate the performance that a local relative differentiation translates to a consistent end-to-end relative differentiation and compare with our analytical bounds. In order to implement this goal, we need to measure the end-to-end relative DVP ratio given each node target relative DVP ratio. We carried out five

Table 4.1 The traffic source parameters configuration

	S1	S2	S3	S4	S5	S6
Destination	D1	D1	D2	D2	D1	D1
Priority	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2

Table 4.2 The different target ratios for node 1 and 2 and analytical end-to-end bounds

	Node 1	Node 2	ETE
case 1	0.75	0.75	0.75
case 2	0.5	0. 5	0.5
case 3	0.25	0.25	0.25
case 4	0.125	0.125	0.125
case 5	0.5	0.125	0.125–0.5625

experiment cases with different target ratios for node 1 and 2 as shown in Table 4.2. The analytical end-to-end bounds based on node target ratios are listed in the same table.

The simulation results are summarized in Table 4.3. We measure the achieved relative DVP ratio at each node and the achieved end-to-end relative DVP ratio at destination respectively. The target ratio is the DVP differentiation parameter of class 2 to class 1 of traffic. The priority of Class 2 is set higher than class 1 that implies the ratios of DVP differentiation parameter should be between 0 and 1. Table 4.3 shows the achieved end-to-end DVP ratios in five cases with different target ratios for node 1 and

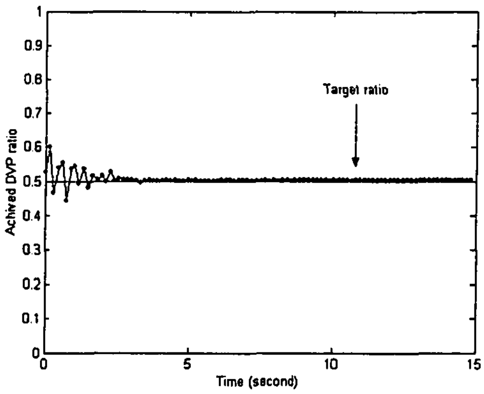
Table 4.3 The achieved end-to-end relative DVP ratios under 5 different cases

		Node 1	Node 2	ETE
Case 1	Target DVP ratio	0.75	0.75	0.75
	Achieved DVP ratio	0.753	0.7446	0.7409
Case 2	Target DVP ratio	0.5	0.5	0.5
	Achieved DVP ratio	0.5052	0.495	0.4929
Case 3	Target DVP ratio	0.25	0.25	0.25
	Achieved DVP ratio	0.2509	0.2518	0.242
Case 4	Target DVP ratio	0.125	0.125	0.125
	Achieved DVP ratio	0.1253	0.1310	0.121
Case 5	Target DVP ratio	0.5	0.125	0.125--0.5625
	Achieved DVP ratio	0.5052	0.1329	0.408

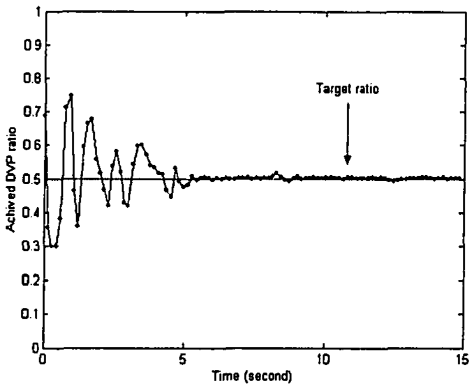
node 2. Figure 4.13 and Figure 4.14 show two sample paths under equal and unequal relative DVP ratios for each node. From the table, we make the following observations:

1. Regardless of the target ratios of node 1 and node 2, a local (node) relative DVP differentiation can translate to a consistent end-to-end relative DVP differentiation as shown in table 4.3. The difference of local target ratios configuration leads to a different end-to-end DVP ratio.

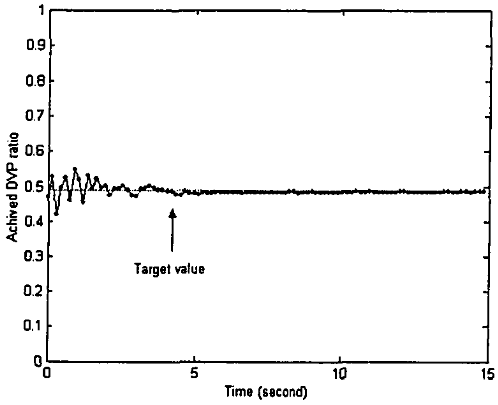
2. The achieved end-to-end relative DVP ratios are consistent to our analytical bounds in all five cases. According to Theorem 2, under the special case of equal local (node) relative DVP ratios $r_{j,k}^1 = r_{j,k}^2$ configuration (0.75, 0.5, 0.25, and 0.125 for both nodes respectively), the end-to-end relative DVP can achieve the approximate result $r_{j,k} \approx r_{j,k}^1 = r_{j,k}^2$ (0.7409, 0.4929, 0.242, and 0.121 for ETE) as shown in table 4.3. For example, an equal target relative DVP for node 1 and node 2 is set to 0.5, the achieved relative DVP ratios for node 1 and node 2 are 0.5052 and 0.4950 respectively; the achieved end-to-end relative DVP ratio is 0.4929 that is very similar to the target node ratio in Figure 4.12. Under the unequal node relative DVP settings such as 0.5 and 0.125, the achieved relative DVP ratios for node 1 and node 2 are 0.5052 and 0.132 respectively; the achieved end-to-end relative DVP ratio is 0.4080 that is between analytical bounds as shown in Figure 4.13.



(a) Achieved DVP ratio for node 1

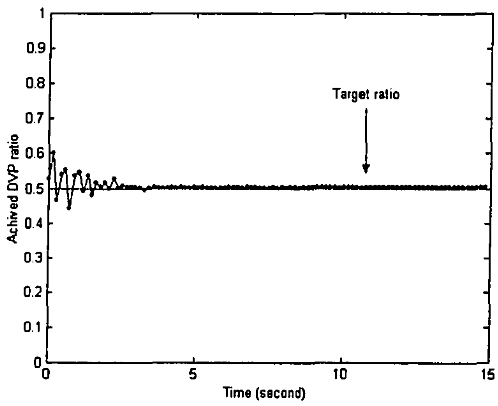


(b) Achieved DVP ratio for node 2

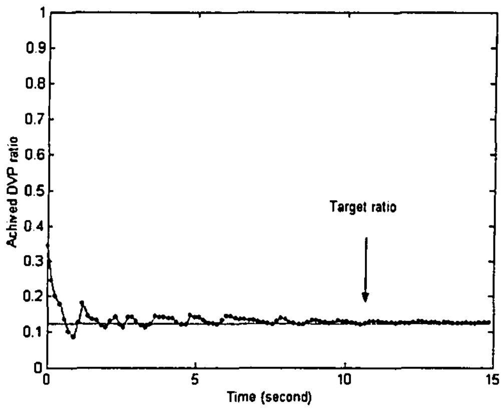


(c) Achieved DVP ratio for end-to-end

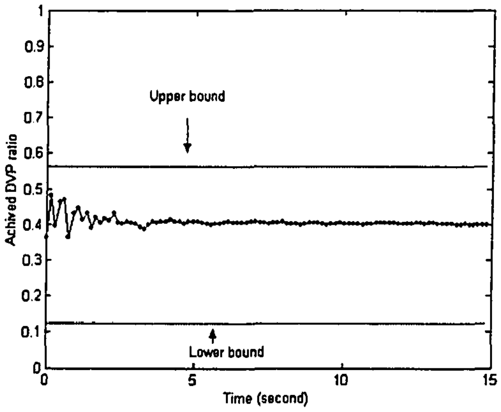
Figure 4.12 A sample path of the achieved end-to-end DVP ratio under the equal node relative DVP configuration



(a) Achieved DVP ratio for node 1



(b) Achieved DVP ratio for node 2



(c) Achieved DVP ratio for end-to-end

Figure 4.13 A sample path of the achieved end-to-end DVP ratio under the unequal node relative DVP configuration

4.3 End-to-end absolute DVP

In this section, we investigate how to provide the absolute DVP guarantee for some real time applications that require stringent DVP requirement. Two different scenarios have been studied at this section.

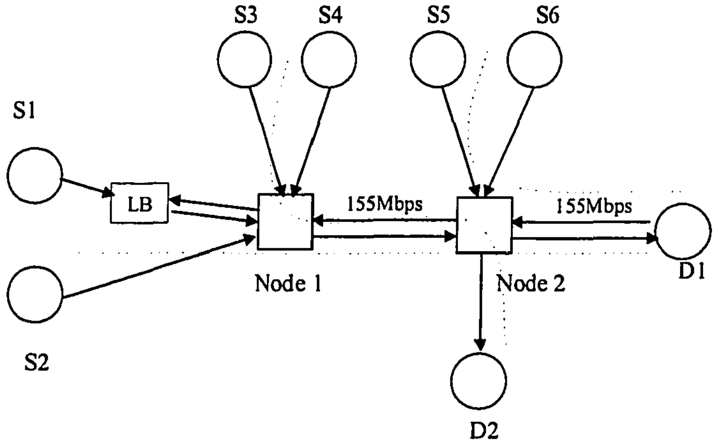


Figure 4.14 The multi-node network topology I for the end-to-end absolute DVP

4.3.1 Scenario I

To guarantee the end-to-end absolute DVP, we consider the scenario based on the end-to-end relative DVP scenario except we add a leaky bucket at source 1 as shown in Figure 4.14. The bucket depth is set to 50 packets. The token rate is adjusted dynamically. We set the class 1 as Poisson traffic with average interarrival time 0.015 ms and class 2 is still ON/OFF traffic source. The link utilizations of node 1 to node 2 and node 2 to destination 1 are 90% and 92% respectively. The other simulation setup is the same as the relative DVP scenario.

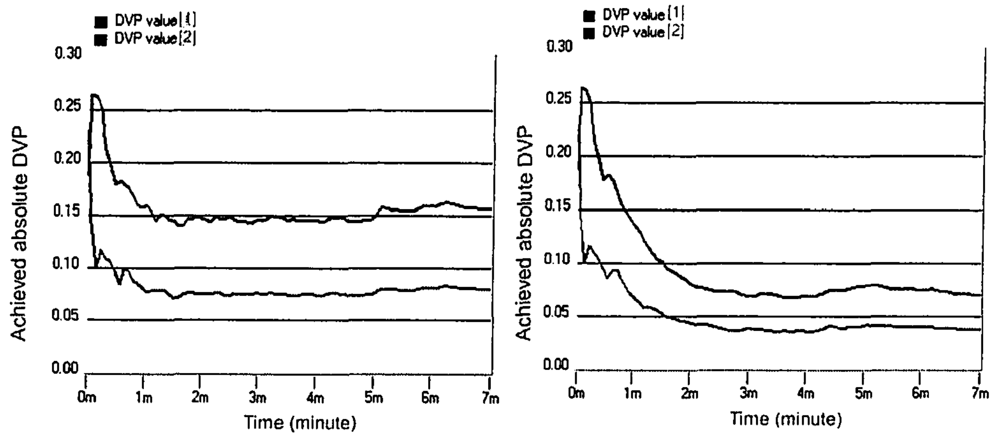
The goal of this simulation study is to find out, how to adjust the token rate in order to meet the end-to-end absolute DVP requirement on the node relative differentiation basis. We first measure the end-to-end absolute DVP of class 1 traffic at destination 1, then create a probe packet with this information and send it back to the source node. The token rate is increased or decreased dynamically in a constant rate within a time interval according to the difference between the achieved absolute DVP and the target value. Here, we set the token adjustment rate to 5000000 tokens per second and the time interval to 4 seconds in our simulation. The target end-to-end absolute DVP value of class 1 traffic is set to 7.5%. The DVP differentiation parameter of node 1 and node 2 is configured to 0.5. According to Theorem 2, the end-to-end relative DVP ratio should be approximate to 0.5.

The results obtained under both with leaky bucket and without leaky bucket conditions are summarized in table 4.4. Without the leaky bucket to control the class 1 traffic access rate, the end-to-end absolute DVP of class 1 and class 2 can achieve 15.65% and 7.94% respectively. To guarantee the absolute DVP target value, we use our

Table 4.4 The achieved end-to-end absolute DVP for scenario I

		Node 1	Node 2	ETE
Without leaky bucket	Target DVP ratio	0.5	0.5	0.5
	Achieved DVP ratio	0.5031	0.5024	0.5073
	Class 1 absolute DVP	0.0948	0.0611	0.1565
	Class 2 absolute DVP	0.0477	0.0307	0.0794
With leaky bucket	Target DVP ratio	0.5	0.5	0.5
	Achieved DVP ratio	0.5036	0.5032	0.5055
	Class 1 absolute DVP	0.0415	0.031	0.072
	Class 2 absolute DVP	0.0209	0.0156	0.0364

DPCF. After adding the control mechanism, the end-to-end absolute DVP of class 1 of traffic is 7.2%, approximately equal to our target value. Based on the node differentiation, the absolute DVP of class 2 is 3.64%. Figure 4.15 shows the sample path of the achieved end-to-end absolute DVP. At $t = 70$ second, we turn on the leaky bucket control mechanism. In addition, we set 7% as a lower limit on the absolute DVP of class 1 traffic in the simulation. If the absolute DVP of class 1 is lower than this limit, we can increase the token rate to add in more traffic to the network.



(a) ETE absolute DVP without leaky bucket (b) ETE absolute DVP with leaky bucket

Figure 4.15 Scenario I: providing the end-to-end absolute DVP

4.3.2 Scenario II

The simulation topology for scenario II is shown in Figure 4.16. Source 1 and source 2 compete the congested link between node 1 and node 2 under this scenario. The traffic source models are the same as the scenario I i.e. class 1 using Poisson traffic and class 2 using ON/OFF traffic. The link capacity is 155 Mbps. The link utilizations of node 1 to node 2 and node 2 to destination 1 are set to 90% and 92% respectively. The other simulation setup is the same as the scenario I.

The differences with the scenario I is that there are two leaky buckets corresponding to the source 1 and 2 respectively that can be adjusted. The goal of this simulation is to study the end-to-end absolute DVP requirement on per session base. The DVP differentiation parameter of node 1 and node 2 is configured to 0.5, so the end-to-end relative DVP ratio is around to 0.5 according to Theorem 2. The target end-to-end absolute DVP value of class 1 traffic is set to 7.5%.

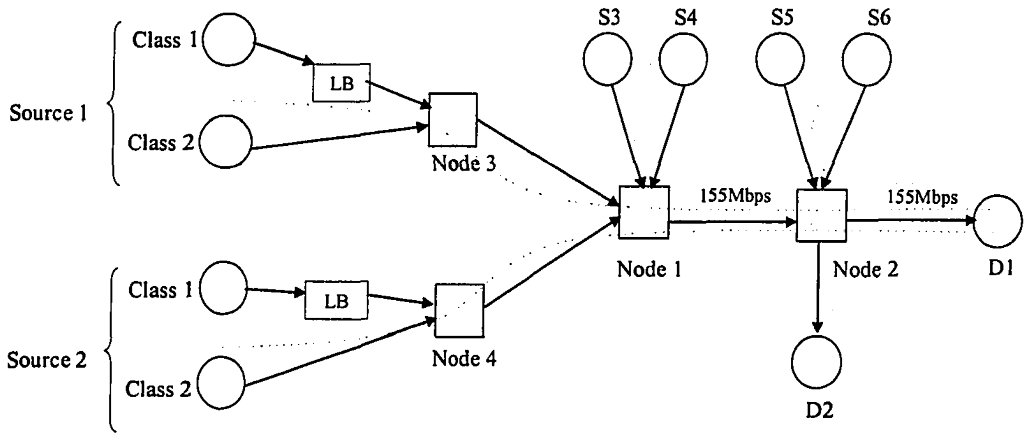


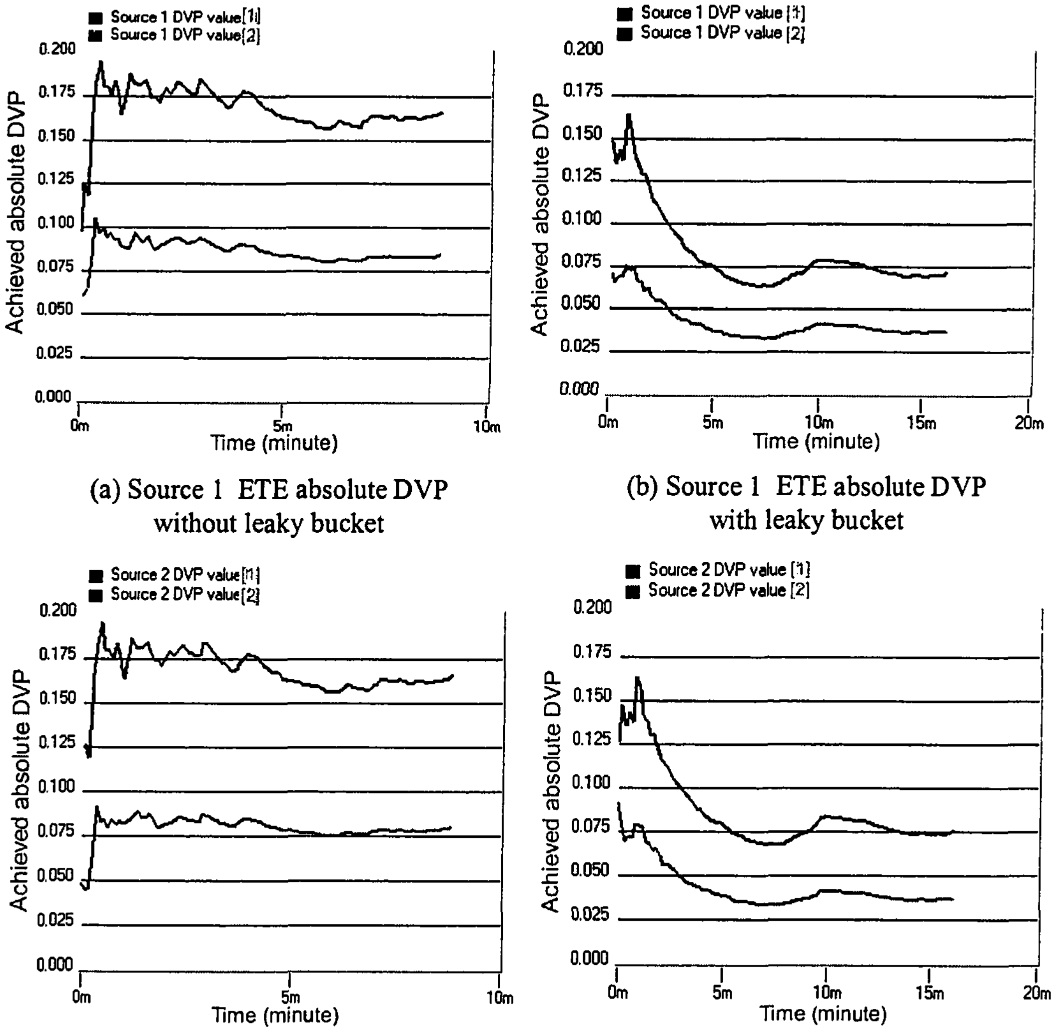
Figure 4.16 The multi-node network topology II for the end-to-end absolute DVP

Table 4.5 The achieved end-to-end absolute DVP for scenario II

		Node 1	Node 2	ETE
Without leaky bucket	Target DVP ratio	0.5	0.5	0.5
	Achieved DVP ratio	0.503	0.5034	0.5061
	Class 1 absolute DVP	0.0968	0.0578	0.1555
	Class 2 absolute DVP	0.0487	0.0291	0.0787
With leaky bucket	Target DVP ratio	0.5	0.5	0.5
	Achieved DVP ratio	0.5032	0.5019	0.5091
	Class 1 absolute DVP	0.0457	0.0257	0.0711
	Class 2 absolute DVP	0.023	0.0129	0.0362

The simulation results are summarized in Table 4.5. The measured end-to-end absolute DVP is the total DVP value including all sources traffic of the same class. Without the leaky bucket, the end-to-end absolute DVP of class 1 and class 2 can achieve 15.55% and 7.87% respectively. With leaky bucket control mechanism, the end-to-end absolute DVP of class 1 is 7.11%, close to our target value. The absolute DVP of class 2 is 3.62%.

The sample path of the achieved end-to-end absolute DVP for source 1 and source 2 are shown in Figure 4.17. We can see that the simulation results behave as we expect. That is, when we shape class 1 traffic, the absolute QoS of both classes improve at the same time while maintaining the relative QoS between the two classes unchanged.



(a) Source 1 ETE absolute DVP without leaky bucket

(b) Source 1 ETE absolute DVP with leaky bucket

(c) Source 2 ETE absolute DVP without leaky bucket

(d) Source 2 ETE absolute DVP with leaky bucket

Figure 4.17 Scenario II: providing the end-to-end absolute DVP

Chapter 5

Conclusion

5.1 Conclusion

The DiffServ architecture is a promising approach for providing scalable service differentiation in the Internet. In this thesis, we have presented a novel dynamic proportional control framework DPCF for providing absolute end-to-end differentiated service for real-time applications. The DPCF can be used to guarantee the absolute DVP on end-to-end paths. We use relative DiffServ scheduler in the network internal nodes and measurement-based feedback congestion control at edges. With the feedback measurements, the token rate of the leaky buckets at the edges can be dynamically adjusted until the end-to-end absolute DVP meet the target value. By maintaining the absolute QoS performance for the base class, the absolute QoS performance for other classes can be automatically guaranteed by relative ratios. Maintaining the state information for only one class, this approach reduces overhead at network and hence, is scalable to large systems. The simulations have shown that the specified absolute end-to-end DVP requirements can be achieved effectively under the DPCF.

The DVP is an important performance measure for all real-time applications. The proposed DDRR scheduler adjusts the weight of each class dynamically so that the DVP differentiation service can be easily specified by network administrators. The DDRR scheduler, a link sharing based scheduler, has the advantage of maintaining a minimum bandwidth amount for each class and improves on the WEDD scheduler's starvation problem. Moreover, through simulations, we evaluated the proposed DDRR scheduler from two requirements on the proportional differentiation model: predictability and controllability under various load conditions. The simulations have shown that the DDRR scheduler is able to maintain the desired ratios of DVP under different link utilizations and different load distributions.

The relative differentiation model on a single node case is then extended to an end-to-end path. A general end-to-end relative DVP ratio bound is developed given the relative DVP ratio of all nodes in an end-to-end path. According to Theorem 1 in Section 3.2, the end-to-end relative DVP ratio is dependent on the relative DVP ratios of all nodes on the path. Furthermore, from Theorem 2, a special case is presented in which we can get a much tighter bound for the end-to-end DVP ratio when we configure the relative DVP ratios of all nodes equally. The simulations have shown that a node (local) relative differentiation can lead to an end-to-end relative differentiation. The achieved end-to-end relative ratio is also consistent with the analytical bounds. Under the special case of equal relative ratios for each node, the obtained end-to-end relative ratio is approximately equal to the node relative ratio.

5.2 Future work

Our work on proportional DVP differentiation leads to several open issues that need to be further investigated.

- It is still a challenging task to provide the absolute end-to-end performance guarantee. The next stage of the research will be focused on fairness issues among traffic flows. The traffic shapers in DPCF framework are provided based on per source and per destination. When egress nodes feedback the measured DVP result to the different sources contributing to congestion, a fair solution is needed to regulate the shapers and further allocate resources to each user according to the demands. How to define and maintain such fairness could be further addressed.

- The sensitivity of system performance to the control parameters such as ρ, R and α in the DDRR scheduler can be analyzed in more detail. We tested the sensitivity of these parameters under different link utilizations and used the empirical results in this thesis. Are there any other factors affecting the setting of these parameters? How to propose a general principle to configure control parameters is left for future study.

- It has to be noted that the feasibility issue for the proportional DVP differentiation model is a complicated problem and should be analyzed in greater depth. We only presented some instances on the feasibility of the model in this thesis. It is important to provide a mathematical function as the guide of the network administrator.

Bibliography

[1] 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

[2] P. Ferguson and G. Huston, "Quality of service," *John Wiley & Sons*, 1998

[3] R. Comeford, "State of the Internet: Roundtable 4.0," *IEEE Spectrum*, Vol. 35, Issue 10, October 1998

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

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," *RFC 2475, IETF*, December 1998.

[6] S. Shenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," *RFC 2212, IETF*, September 1997.

[7] J. Wroclawski, "Specification of the controlled-load network element service," *RFC 2211, IETF*, September 1997

[8] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification," *RFC 2205, IETF*, September 1997.

[9] S. Brim, B. Carperter and F. Le Faucheur, "Per Hop Behavior Identification Codes," *RFC3140, IETF*, June 2001.

[10]R. Sivakumar, T. Kim, N. Venkitaraman and V. Bharghavan, "Achieving Per-Flow Weighted Rate Fairness in a Core Stateless Network," *IEEE Conference on Distributed Computing Systems 2000*, Taipei, March 2000.

[11] V. Jacobson, "An Expedited Forwarding PHB," *RFC 2598, IETF*, June 1999.

[12] J. Heinanen, T. Finland, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB group," *RFC 2597, IETF*, June 1999.

[13]C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," *Proceedings of ACM SIGCOMM 1999*, Cambridge, September 1999.

[14]C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: delay differentiation and packet scheduling," *IEEE/ACM Transactions on Networking,* Vol. 10, No. 1, February 2002.

[15]S. Bodamer, "A new scheduling mechanism to provide relative differentiation for real-time IP traffic," *Proceedings of IEEE GLOBECOM 2000*, San Francisco, November 2000.

[16]J. Liebeherr and N.Christin, "Rate allocation and buffer management for differentiated services" *Computer Networks*, Vol. 40 No.1, September 2002

[17]S. B. Mojn, J. Kurose, D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms," *ACM/Springer Multimedia Systems,* Vol.5,

January 1998

[18]D. Ferrari, "Client requirements for real-time communication services," *IEEE Communication Magazine*, Vol.37, Issue 11, November 1999.

[19]T. Nandagopal, N. Venkitaraman, R. Sivakumar, V. Bharghavan, "Delay differentiation and adaptation in core stateless networks," *Proceedings of IEEE INFOCOM 2000*, Tel Aviv, March 2000

[20]C. Dovrolis, P. Ramanathan, "Dynamic class selection: from relative differentiation to absolute QoS," *Proceedings of the 2001 IEEE International Conference on Network Protocol,* Riverside CA, November 2001

[21]W. E. Leland, M. S, Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," *IEEE/ACM Transaction on Networking*, Vol 2, No 1, February, 1994

[22]I. Khalil, Torsten Braun, "A range-based SLA and edge driven virtual core provisioning in DiffServ-VPNs," *The 26th Annul IEEE Conference on Local Computer Network,* Tampa, November 2001

[23]N. G. Duffield, P. Goyal, A. Greenberg, "Resource management with hoses: point-to cloud services for virtual private networks," *IEEE/ACM Transaction on Networking*, Vol 10, No 5, October, 2002

[24]R. Liao and A. Campbell, "Dynamic core provisioning for quantitative differentiated service," *IEEE/ACM Transactions on Networking,* vol. 12, No. 3, June, 2004.

[25]Z. Duan, Z. Zhang. Y. T. Hou, and L. Gao, "A core stateless bandwidth broker architecture for scalable support of guaranteed service," *IEEE Transactions on*

Parallel and Distributed Systems, vol. 15, No. 2, Feb. 2004.

[26] C. Dovrolis, and P. Ramanathan, "Proportional differentiated services, Part 2: loss rate differentiation and packet dropping," *IEEE/IFIP International Workshop on Quality of Service,* Pittsburgh, June 2000.

[27] 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*, *IETF*, December 1998.

[28] D. Goderis, " Service level specification semantics, parameters and negotiation requirements." *IETF Internet Draft*, June 2001.

[29] J. Gozdecki, A. Jajszczyk, and R. Stankiewicz, "Quality of service terminology in IP networks, " *IEEE Communication Magazine,* Vol.41, Issue 3, March 2003.

[30] The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) G.114 *recommendation*, October, 2002

[31] 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*, Alaska, April 2001.

[32] H. Nagabuchi, A. Takahashi, and N. Kitawaki, "Speech quality degraded by cell loss in ATM networks." *NTT review,* July 1992.

[33]H. Satio, C. Lukovszki, I. Moldovan, "Local optimal proportional differentiation scheduler for relative differentiated services." *Proceedings of IEEE International Conference on Computer Communications and Networks*, Las Vegas, October 2000

[34]D. D. Clark, S. Shenker and L. Zhang, "Supporting real-time applications in an integrated services packet network." In *Proc. ACM SIGOCOMM'92*, 1992

[35]H. Zhang, "Service disciplines for guaranteed performance service in Packet-switched networks," *Proceedings of the IEEE*, Vol.83, No.10, October 1995

[36]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,* Vol. 1, No. 3, June 1993

[37]A. K. Parekh, and R.G. Gallager, " A generalized processor sharing approach to flow control in integrated services networks: the multiple node case," *IEEE/ACM transactions on Networking,* Vol. 2, No. 2, April 1994

[38]S. Guo and N. D. Grorganas, "Resource and connection admission control in real-time transport protocol with deterministic QoS guarantees." In *Proc. IEEE Infocom'95,* April 1995.

[39]C. Kalmanek, H. Kanakia, and S. Keshav, "Rate controlled servers for very high-speed networks." In *Proc. IEEE GlobeCom'90,* Dallas, Dec.1990

[40]C. Courrcoubetis, V.A. Siris, and G. D. Stamoulis, "Application of the many sources asymptotic and effective bandwidths to traffic engineering," *Telecommunication Systems 12 (2-3),* 1999

[41]A. I. Elwalid and D. Mitra, "Effective bandwidth of general Markovian traffic source and admission control of high speed networks." *IEEE/ACM Transactions on*

Networking, vol. 1(3), June 1993.

[42]G. Kesidis, J. Walrand, and C.S. Chang, "Effective bandwidths for multiclass Markov fluids and other ATM sources." *IEEE/ACM Transactions on Networking,* vol. 1(4), Aug. 1993.

[43]S. Georgoulas, P. Trimintzios, and G. Pavlou, "Joint measurement and traffic descriptor-based admission control as real-time traffic aggregation points," *Proceedings of IEEE ICC*, Paris, June 2004.

[44]K. Made, and H. Kobayashi, "An efficient end-to-end measurement based admission control for VoIP networks," *Proceedings of IEEE ICC*, Paris, June 2004.

[45]J. Qiu and W. Knightly, "Measurement-based admission control with aggregate traffic envelopes," *IEEE/ACM Transactions on Networking,* vol. 9, No. 2, April, 2001.

[46]K. Nichols, V. Jacobson and L. Zhang, "A two-bit differentiated services architecture for the Internet." *RFC 2638*, July, 1999.

[47]J. Lakkakorpi, O. Strandberg and J. Salonen, "Adaptive Connection Admission Control for Differentiated Services Access Networks," *Proceedings of IEEE GlobeCom 2004*, Dallas, TX, USA, November 2004

[48]H. Saito, and K. Shiomoto, "Dynamic Call Admission Control in ATM Networks," *IEEE Journal on Selected Areas in Communications,* vol. 9, no. 7, September 1991

[49]F. Bonomi and K. W. Fendick, "The rate-based flow control framework for the available bit rate ATM service." *IEEE Transaction on Networks,* Mar. 1995.

[50]V. Jacobson, "Congestion avoidance and control," *Proceedings of the ACM SIGCOMM,* 1988

[51]V. Jacobson, "Modified TCP congestion avoidance algorithm," *Technical report,* 30, Apr. 1990, Email to the end2end-interest Mailing list, URL: ftp://ftp.ee.lbl.gov/email/vanj.90apr30.txt.

[52]J. Hoe, "Start-up dynamics of TCP's congestion control and avoidance schemes," *Master's thesis*, MIT, Jun. 1995.

[53]M. Mathis, J. Mahdavi, S. Floyd and A. Romanow, "TCP selective acknowledgement options," *RFC 2018, IETF*, Oct. 1996

[54] E. Hashem, "Analysis of random drop for gateway congestion control," *Technical Report MIT/LCS/TR-465,* MIT, Cambridge, MA, 1989.

[55] W. Feng, K. Shin, D. Kandlur, and D. Saha, "The blue active queue management algorithms," *IEEE Transaction on Networking,* Vol. 10, No. 4. Aug. 2002

[56] D. Lin, and R. Morrin, "Dynamics of random early detection," *Proceedings of the ACM SIGCOMM,* Berlin, Aug. 1997.

[57]S. Lee, G. Ahn, X. Zhang, and A. Campbell, "INSIGNIA: an IP-based quality of service framework for mobile ad hoc networks," *Journal of parallel and Distributed computing,* Vol. 60, No. 4. Apr. 2000.

[58]G. Ahn, A. Campbell, and A.veres, "SWAN: service differentiation in stateless wireless ad hoc networks." *Proceedings of. IEEE INFOCOM*, Anchorage, AK, April, 2002.

[59] K. Wang, P. Ramanathan, "QoS Assurances through class selection and proportional differentiation in wireless networks," *IEEE Journal on Selected Areas in*

Communications. Vol. 23, No. 23, March 2005.

[60]A. Asgari, R. Egan, P. Trimintzios, and G. Pavlou, "Scalable monitoring support for resource management and service assurance," *IEEE Network*, Vol. 18, No. 6, November 2004.

[61]V, Sivaraman, F. Chiussi, M. Gerla, "End-to-end statistical delay service under GPS and EDF scheduling: a comparison study." *Proceedings of IEEE INFOCOM*, Anchorage, AK, April, 2001.

[62]L. Kleinrock, "Queueing System," *Wiley interscience*, Vol.2, 1976

[63]C. Liu, J. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment." *Journal of the ACM*, Vol. 20, No. 1, January 1973

[64]M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general purpose ATM switch chip", *IEEE Journal on Selected Areas in Communications*, Vol. 9, Issue 10, October 1991.

[65]M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," *Proceedings of the ACM SIGCOMM*, Cambridge, September 1995,

[66]A. Demers, S. Keshav, and S.Shenker, "Analysis and simulation of a fair queueing algorithm." *Journal of Internetworking Research and Experience*, Vol.1, No.1, October 1990.

[67]D. Yates, J. Kurose, D. Towsley, and M. Hluchyj, "On per-session end-to-end delay and the call admission problem for real time applications with QoS requirements," *Journal of High Speed Networks*, Vol.3, Issue 4, December 1994

[68]N. G. Duffield, and F. L. Pretti, "Network tomography from measured end-to-end delay covariance" *IEEE Transactions on Networking*, vol. 12, No. 6, Dec. 2004.