Quality of Service provisioning for IP/MPLS networks and EPON networks

Yin, Yong Ning

2009

Yin, Y. N. (2009). Quality of Service provisioning for IP/MPLS networks and EPON networks.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/15167>

<https://doi.org/10.32657/10356/15167>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

**QUALITY OF SERVICE PROVISIONING FOR IP/MPLS
NETWORKS AND EPON NETWORKS**

**YIN YONGNING
SCHOOL OF ELECTRICAL AND ELECTRONIC
ENGINEERING
2009**

**QUALITY OF SERVICE PROVISIONING FOR IP/MPLS
NETWORKS AND EPON NETWORKS**

YIN YONGNING

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2009

Acknowledgements

First and foremost I would like to express my deep and sincere gratitude to my supervisor, Associate Professor Poo Gee Swee. His wide knowledge and scientific way of thinking have inspired me to find solutions to problems. His strong support and patient guidance is invaluable for me to complete this challenging journey towards a PhD degree.

I wish to thank all my research group colleagues, Dr. Wang Haibo, Mr. Chen Sheng, Mr. He Xuzhou, Dr. Zhou Yinzhu, Mr. Kelvin Yong, Dr. Chen Bin, and Ms. Shao Xuguang. They helped me immensely by giving me friendship, encouragement, and support that enable me pass through my tough research years. I would also like to thank Dr. Zhou Yinzhu for their insightful assistance during my thesis writing.

I owe my loving thanks to my wife Ou Hanyan. Without her encouragement, understanding, and support, it would be impossible for me to finish this hard work. I would also like to thank my parents and sisters for their encouragement when I am facing challenges and in despair during life.

I am very grateful to the Nanyang Technological University for the financial support. I would also like to thank all professors and colleagues from Radar Lab and Network Technology Research Centre (NTRC) for providing a friendly and effective research environment.

Yin Yongning

20-Feb-2008

Contents

Acknowledgements	iii
List of Figures	vii
List of Tables.....	ix
Acronyms.....	x
Summary	xii
Chapter 1 Introduction	1
1.1 Background	1
1.1.1 Network Service Models and MPLS Architecture	2
1.1.2 Traffic Aggregation and Aggregate Scheduling	3
1.1.3 EPON Network and Dynamic Bandwidth Allocation (DBA).....	5
1.2 Motivations	6
1.3 Objectives.....	9
1.4 Contributions.....	11
1.5 Thesis Organization	13
Chapter 2 Literature Review	15
2.1 Flow-based Scheduling Algorithms and Network Calculus	15
2.1.1 Scheduling Algorithms and Performance Results.....	16
2.1.2 Scheduling Classes	20
2.1.3 Network Calculus	23
2.2 Aggregate Scheduling	25
2.2.1 Route Interference (<i>RI</i>) based Aggregate Scheduling	26
2.2.2 Aggregate Scheduling Featuring Network Utilization.....	28
2.2.3 Aggregate Scheduling with Fair Aggregator.....	30
2.3 Dynamic Bandwidth Allocation (DBA) for EPON	33
2.3.1 Algorithms with Statistical Multiplexing and Bandwidth Guarantee	35
2.3.2 Service Differentiation Support for QoS Assurance	38
2.3.3 User-oriented Hierarchical Scheduling	43
2.4 Summary	45

Chapter 3 Deterministic End-to-End QoS: A Latency-Rate Max-Min Service Curve Approach	47
3.1 Introduction	47
3.1.1 Motivations	48
3.2 Network Model and Latency-Rate Max-Min Service Curve	49
3.3 Performance Analysis on End-to-end Queuing Delay and Buffer Size	52
3.3.1 Queuing Delay through a Super-chain.....	56
3.3.2 Property of Super-chain with respect to source rate condition.....	59
3.3.3 Bounds on end-to-end queuing delay and buffer-size	61
3.4 Evaluation.....	66
3.5 Conclusion.....	75
Chapter 4 Deterministic End-to-end QoS: Aggregate Scheduling with Fair Aggregation.....	77
4.1 Introduction	77
4.2 Aggregate Scheduling based on GRFA.....	78
4.2.1 GRFA aggregator.....	78
4.2.2 Aggregate scheduling of GR with GRFA.....	81
4.2.3 End-to-end Delay Bound	83
4.2.4 Performance Comparison	85
4.2.5 Summary	88
4.3 Aggregate Scheduling of FA-GR and P-shaper	88
4.3.1 FR-GR Scheduling.....	88
4.3.2 P-shaper	90
4.3.3 Node Model	91
4.3.4 Performance Analysis on a Network of FA-GR and P-shaper	92
4.3.5 Performance Comparison	101
4.3.6 Summary.....	110
Chapter 5 Hybrid Cycle Scheduling: A Fair Bandwidth Allocation Algorithm with QoS Support in EPON	112
5.1 Introduction	112
5.2 Hybrid Cycle Bandwidth Allocation Scheme (HCS).....	113
5.2.1 EF Cycle and AF Cycle	114

5.2.2 Performance Analysis for EF Traffic	119
5.2.3 Dynamic Bandwidth Allocation Algorithm for AF and BE Traffic	121
5.3 Performance Evaluation	126
5.3.1 Delay and Delay Variation	127
5.3.2 Throughput	132
5.3.3 Loss Ratio	133
5.3.4 Delay Bound Comparison and Throughput for ONUs	134
5.4 Summary	137
Chapter 6 User-oriented Hierarchical Scheduling for EPON	138
6.1 Introduction	138
6.2 User-oriented Hierarchical Bandwidth Scheduling Algorithms (UHSAs)	140
6.2.1 Two-part Frame Partition Strategy	140
6.2.2 Credit-based Common Queue and Transmission Priority Scheme	144
6.2.3 Report and Grant Mechanism	151
6.3 Performance Evaluation	154
6.3.1 Delay and Delay Variation	155
6.3.2 Throughput and Fairness	160
6.3.3 Scheduling Efficiency and Queuing Number	163
6.4 Summary	165
Chapter 7 Conclusion and Future Work	167
7.1 Summary	167
7.2 Future Research	170
Author's Publications	174
Bibliography	175

List of Figures

Figure 1.1: Different Scheduling Architectures.....	4
Figure 1.2: Illustration of EPON network topology and scheduling architecture.....	5
Figure 2.1: An illustration of a heterogeneous scheduling network.	20
Figure 2.2: An example of per-node delay mapping.....	26
Figure 2.3: Aggregate Scheduling Architecture in DiffServ.	28
Figure 2.4: An Illustration of Aggregator Architecture.	31
Figure 2.5: Taxonomy of dynamic bandwidth allocation (DBA) algorithms.....	34
Figure 2.6: An Illustration of the Interleaved Polling Mechanism.	35
Figure 2.7: An illustration of the interleaved polling with stop mechanism.....	39
Figure 3.1: Bounds on backlog and per-node delay.....	63
Figure 3.2: Network Topology.....	67
Figure 3.3: Network topology: example 2.	71
Figure 3.4: Comparison of end-to-end delay for flows of f_1 and f_2 in our model and $f_1(f_2)$ in Ref. [OB03].	72
Figure 3.5: Comparison of end-to-end delay with varying number of flows in f_1	73
Figure 4.1: GRFA and the architecture of the aggregate scheduling.....	80
Figure 4.2: Path configuration and flow aggregation.....	84
Figure 4.3: The architecture of FA-GR scheduling	89
Figure 4.4: Flow separation occurs at the network.	99
Figure 4.5: An example network of configuration.....	102
Figure 4.6: End-to-end delay comparison.	104
Figure 4.7: End-to-end delay bound with varying aggregated flows.....	106
Figure 5.1: Illustration of EF cycle and AF cycle.....	115
Figure 5.2: Illustration of queuing delay for ONU_i	120
Figure 5.3: Example of AF distribution.	122
Figure 5.4: Simulation results of AD for EF, AF, and BE Class.....	128
Figure 5.5: Simulation results of DV for EF and AF Class.	130

Figure 5.6: Throughput.....	132
Figure 5.7: Loss Ratio.....	133
Figure 6.1: Illustration of EF cycle and AF cycle (a) UHSA-S. (b) UHSA-M.	140
Figure 6.2: Architecture of the User-oriented Hierarchical Scheduling.....	145
Figure 6.3: Pseudo code for UHSA.	147
Figure 6.4: Simulation results of AD for EF, AF, and BE Class.....	156
Figure 6.5: Simulation results of DV for EF and AF Class.....	159
Figure 6.6: Throughput of different classes.....	160
Figure 6.7: Throughput of test users.....	162
Figure 6.8: Percent of packets served from CCQs.	164
Figure 6.9: Average number of queues (AQ).	165

List of Tables

Table 2-1: E value, end-to-end delay bound, and complexity of GPS schedulers.....	18
Table 2-2: Single packet delay bound and fairness of RR schedulers	19
Table 3-1: Parameters used for analytical calculation and simulation.....	67
Table 3-2: Analytical and simulation results for variable packet size scenarios	68
Table 3-3: Analytical and simulation results for fixed packet size scenarios	68
Table 3-4: Parameters used for the second example	71
Table 4-1: Parameter used for performance analysis.....	79
Table 4-2: End-to-end delay (ms)	86
Table 5-1: Parameter used for performance analysis.....	119
Table 5-2 Maximum queuing delay under HCS	134
Table 5-3: Throughput guarantee validation.....	136
Table 6-1: Parameters used for UHSAs formulization	143

Acronyms

AF	assured forwarding
ATM	asynchronous transfer mode
BE	best effort
CCQ	credit-based common queue
DBA	dynamic bandwidth allocation
DiffServ	differentiated services
EF	expedited forwarding
EPON	Ethernet passive optical network
FIFO	first-in-first-out
GPS	generalized processor sharing
GR	guaranteed rate
GRC	guaranteed rate clock
GRFA	guaranteed rate fair aggregator
HCS	hybrid cycle scheme
HDTV	high-definition TV
HGP	hybrid granting protocol
HP	high-priority
IETF	Internet engineering task force
IP	Internet protocol
IPS	Interleaved polling with stop
IPACT	interleaved polling with adaptive cycle time
IntServ	integrated services

LP	low priority
LR	latency-rate
LRMMSC	latency-rate max-min service curve
LSP	label switched path
MAC	medium access control
MPCP	multi-point control protocol
MPLS	multi-protocol label switching
OLT	optical line terminal
ONU	optical network unit
PSRG	packet scale rate guarantee
QoS	quality of service
RI	route interference
RR	round robin
RTT	round-trip time
SLA	service level agreement
UHSA	user-oriented hierarchical bandwidth scheduling algorithm
WDM	wavelength-division multiplexing

Summary

Scheduling disciplines are used in network routers/switches to provide a wide range of Quality of Service (QoS) assurances. With the development of the Internet technologies, two new branches of scheduling family arise in order to provide QoS in the context of emerging networking environments. In particular, aggregate scheduling is developed to provide scalable QoS in aggregation-based high-speed networks, while dynamic bandwidth allocation (DBA) is developed to offer efficient QoS support in Ethernet passive optical network (EPON)-based access networks. This thesis contributes to the design, analysis, and evaluation of various aggregate scheduling disciplines and DBA algorithms.

The first part of this work studies aggregate scheduling using the concept of route interference. A general service scheme is proposed which features wide-range implementations. We consider a general network model and define a source rate condition in terms of the route interference. We show that there exist deterministic bounds on end-to-end delay and buffer size at all nodes provided that each flow obeys its source rate condition when entering the network. Experiments show good agreement between analytical and simulation results. To exploit the advantage of fair aggregation in aggregate scheduling, we then propose a guaranteed rate fair aggregator (GRFA) for flow aggregation and show analytically the advantage of aggregate scheduling with GRFA in delivering deterministic end-to-end delay bound. Subsequently, we devise a flow aggregation capable of guaranteed rate (FA-GR) scheduling and a priority-shaper for aggregate scheduling. This approach obtains the

same level of end-to-end delay bound as GRFA-based approach while achieving lower complexity and implementation cost.

We then address dynamic bandwidth allocation problem in EPON. A new hybrid cycle scheme (HCS) is proposed which adopts two-part frame partition approach. High priority traffic is transmitted in a fixed cycle to ensure guaranteed service while low priority traffic is delivered in an adaptive dynamic cycle to adapt to its bursty nature in order to improve network throughput. In addition, a novel feature of multiple transmission opportunities per-cycle for high priority traffic is developed by the proposal of a suitable local buffer management scheme. This is significant in improving average delay by a range of 20.0% to 95.6% and delay variation by a range of 12.8% to 85.4% as compared with traditional approaches through simulation experiments. We further propose a collection of credit-based hierarchical scheduling algorithms in order to support DiffServ and fairness to end-users. Two-part frame partition approach is adopted for inter-ONU scheduling, while a credit-based scheduling is proposed to guarantee fairness among end users for intra-ONU scheduling. On the other hand, a credit-based common queue (CCQ) for each class is proposed to improve scheduling efficiency and lower queuing management complexity. A transmission priority scheme is proposed for different queue groups, which together with CCQ substantially improve delay and jitter performance of high priority traffic, while guaranteeing throughput for medium priority traffic. Simulation experiments are conducted to demonstrate the effectiveness and efficiency of the algorithms.

Chapter 1

Introduction

1.1 Background

The startling expansion of Internet and the emergence of content-rich multimedia applications have catalyzed stupendous evolution in communication networks. The carrying capacity of core networks has been increased drastically with the advances of wavelength division multiplexing (WDM) technologies. Multi-protocol label switching (MPLS) technology together with differentiated services (DiffServ) is to be developed to provide quality of service (QoS) for the next generation high speed networks.

QoS is a crucial issue in broadband networks. Besides data transactions such as e-mail and web surfing, the Internet has been experiencing fast growth in value-added real-time audio and video applications, like video conferencing, high-definition TV (HDTV), and IP-voice communication. These applications have stringent QoS requirements in term of throughput, delay, jitter, and loss rate. The Internet was originally designed to provide one simple service: the best-effort service. Therefore, there is an urgent need to introduce quality services in the Internet with performance

guarantees. *Scheduling discipline*, in particular, plays a crucial role in implementing the QoS control.

Scheduling discipline is the algorithm deployed at network switches (nodes, routers) to provide link bandwidth sharing among a number of competing traffic flows. A flow represents a stream of data packets ejected from a single or an aggregate of applications (connections). The scheduling discipline determines the transmission order of queuing packets to fulfill differentiated QoS to different flows. A switch employs a scheduling algorithm at each input or output port for link sharing. Our study focuses exclusively on output-buffered switches, where a packet is immediately available for transmission upon arriving at the switch.

1.1.1 Network Service Models and MPLS Architecture

To provide Internet QoS, two primary service architectures have been defined. The Integrated Services (IntServ) [BCS94] architecture, proposed by the Internet Engineering Task Force (IETF), was the first attempt to support QoS on the Internet. IntServ defines two enhanced service models: guaranteed service and controlled load service, in addition to best-effort service, and is characterized by per-flow (individual or aggregate) resource reservation. IntServ provides powerful service that is capable to support a variety of applications with diverse QoS requirements.

Differentiated Services (DiffServ) [BBC98] architecture, developed by IETF as an alternative QoS model, provides QoS in a significantly simple and scalable way. At edge nodes all traffic is classified into a small set of behavior aggregates (BAs), while at core nodes each BA receives a particular forwarding treatment, termed Per-Hop-Behavior, or PHB. This suggests, in DiffServ, while edge routers perform traffic

operation at finer granularity, e.g., per-flow, per-user, etc., core routers distinguish only a small set of aggregate flows, i.e., providing services in the granularity of traffic class. Thus, DiffServ achieves a scalable core. However, the drawback is that this architecture does not address end-to-end QoS delivery to individual flows, although it offers per-hop (nodal) QoS guarantees for traffic aggregates (classes).

IntServ and DiffServ define two primary service architectures. Multi-protocol label switching (MPLS) [RVC01] architecture, on the other hand, introduces a simple but advanced traffic engineering solution to IP networks where logical connections, referred as label switched paths (LSPs), are established across the network using constraint-based QoS routing for optimizing network performance. By incorporating the concepts and features of IntServ and DiffServ [FWD01], LSPs can be established edge-to-edge subject to some constraints such as bandwidth and QoS requirements, and each LSP may support one or more DiffServ traffic classes. In order to reduce the number of LSPs in core nodes for improving scalability, LSPs sharing the whole or part of the path can be merged together into a single LSP. This is another form of traffic aggregation, distinct from that of traffic classification in DiffServ. MPLS coupling with DiffServ is becoming the best candidate for next generation high speed network in providing efficient traffic engineering and scalable QoS.

1.1.2 Traffic Aggregation and Aggregate Scheduling

IntServ architecture is not applicable for a large-scale network due to its poor scalability. In particular, to implement per-flow reservation, each network node needs to maintain per-flow QoS state and perform per-flow classification, scheduling, and buffer management, which consume large memory cost and processing complexity

when the number of flows is large. *Traffic aggregation* is the only way to overcome this problem by substantially reducing the number of flows in the network core as discussed in DiffServ and MPLS technologies. The development of aggregation-based networks naturally raises the demand of a new class of scheduling, namely *aggregate scheduling*. An aggregate scheduling is comprised of a number of aggregators each for aggregating different flows into an aggregated flow using some mechanism and a flow-based scheduler for scheduling aggregated flows. Figure 1.1 (b) illustrates the typical architecture of aggregate scheduling, while Figure 1.1 (a) shows the architecture of traditional flow-based scheduling for comparison. In an aggregation-based network, edge nodes implement traffic classification and / or flow aggregation. At core routers, only a small number of aggregate flows are aware and flow aggregation may also be implemented in order to further reduce the number of flows.

Figure 1.1: Different Scheduling Architectures.

Figure 1.2: Illustration of EPON network topology and scheduling architecture.

1.1.3 EPON Network and Dynamic Bandwidth Allocation (DBA)

The techniques of QoS support have been widely adopted in high-speed core networks and local area networks (LANs). Recently, these techniques, in particular DiffServ, are being developed in *access network*, or the “last mile” network. Access network is a critical component in providing end-to-end QoS delivery; however, little work has been done in the last decade. To replenish the growing gulf between high-capacity backbone networks and LANs, Ethernet passive optical network (EPON) is being developed as one of the most promising solutions for the next generation broadband access networks. An EPON as shown in Figure 1.2 (a), is a point-to-multipoint optical network. It consists of a centralized optical line terminal (OLT) and multiple distributed optical network units (ONUs) [KMP01]. In the upstream direction, all ONUs transmit data to the OLT through a shared uplink channel. Thus, a scheduler is needed at the OLT to arbitrate data transmission of ONUs to avoid collision. Furthermore, as pointed out in [KBS04], a remote scheduling system like EPON, where the queues and the scheduler are located at a large distance from one

another, experiences *significant queue switch-over overhead and large control-plane propagation*. This excludes the applicability of packet-based scheduling algorithms proposed before and leads to the development of a new category of scheduling algorithms, namely *dynamic bandwidth allocation (DBA)*, which features hierarchical and round-robin fashion scheduling architecture as shown in Figure 1.2 (b).

To ensure end-to-end QoS delivery to end users, it is essential to investigate QoS provisioning over three indispensable components of the Internet: core network, access network, and LAN. DiffServ and IP/MPLS have been developed to provide scalable QoS in next generation core network and LAN, while EPON is being developed as one of the best solutions for access network. In this thesis, we investigate QoS provisioning in both IP/MPLS high speed networks and EPON access networks. Our research focus is to address traffic scheduling issues which pose one of the biggest challenges in implementing QoS in these networks. In particular, we propose aggregate scheduling algorithms for end-to-end QoS guarantees over heterogeneous high speed networks, and dynamic bandwidth allocation algorithms for EPON networks.

The rest of the chapter is organized as follows. Section 1.2 presents motivations of the thesis. Section 1.3 defines issues and objectives. Section 1.4 and section 1.5 cover contributions and organization of the thesis, respectively.

1.2 Motivations

The evolvement in network technology has brought new challenges in QoS provisioning and implementation, such as admission control, resource provisioning,

buffer management, QoS routing, traffic scheduling, etc. In this thesis, we focus on the design and analysis of traffic scheduling algorithms. Fundamentally, the scheduling discipline architectures are distinct for different network service models and architectures. IntServ features per-flow operation and accordingly necessitates flow-based scheduling algorithm as shown in Figure 1.1 (a). Extensive studies have been done on scheduling protocols to support IntServ QoS, resulting in numerous scheduling protocols (surveyed in [Zha95], [JG02]) and a framework of network calculus for scheduling modeling and performance analysis [BT03,Cha00]. However, IntServ and flow-based scheduling architectures have serious scalability problem [SZ99]. It is difficult to implement and deploy IntServ to obtain end-to-end QoS in a large-scale network at reasonable cost.

DiffServ and MPLS, widely different, have a nature of traffic aggregation. Naturally in these networks, *aggregate scheduling* is required. Unfortunately, the rich results obtained from flow-based scheduling are not directly available for aggregation-based scheduling. Therefore, there is an urgent need to design new aggregate scheduling protocols and investigate end-to-end QoS delivery in aggregation-based networks. Aggregate scheduling with class-based FIFO aggregation has been studied in [CB00,Jia02,BBC02,ZDH01] where end-to-end delay bounds are derived, but paying big price of very low network utilization (e.g., 10%). This is due to large increase of burstiness brought in by FIFO aggregation [ZDH01]. FIFO aggregation is suitable for connectionless DiffServ networks. Nonetheless, for connection-oriented MPLS networks, traffic paths (LSPs) can be established with QoS constraints such as bandwidth requirement and different LSPs can carry traffic of the same class. In this scenario, simple FIFO aggregation and priority scheduling are

not applicable. Therefore, more advanced aggregate scheduling algorithms need to be developed with regard to their potentially complex scheduling architecture. The work has just started.

On the other hand, some works based on the concept of *route interference (RI)* have been done in [CFZ98] to explore the deterministic end-to-end queuing delay and backlog. One salient feature of RI approach is that the end-to-end queuing delay of a flow (with the highest priority) is only related to the number of joining flows along its path and the maximum backlog size at an output port is bounded by the number of flows departing from it. These performance properties are especially valuable for MPLS networks since the number of flows can be a small value in the network core with traffic aggregation. However, these studies tend to confine to a simple priority scheduling and study the traffic in a single-class environment or the highest priority traffic in a multi-class environment. This is rather restrictive. Therefore, it is needed to extend RI approach into a general network model by proposing a general scheduling scheme that has wide implementations.

DBA algorithms have been studied extensively recently in the context of EPON. The early work mainly focused on upstream link utilization improvement through statistical multiplexing techniques, such as in [KM02, SRC04]. Then, the research focus is shifted to QoS support, where DiffServ model is generally adopted with three identified classes, namely expedited forwarding (EF), assured forwarding (AF) and best effort (BE). All of the proposed algorithms are aware of providing prior services to high priority traffic. However, the fundamental weakness of these algorithms is that transmission opportunities (a piece of timeslot) distributed to ONUs falls into either

single transmission opportunity (S-opportunity) per-ONU-per-cycle fashion or S-opportunity per-class-per-cycle fashion. Thus, there is at most one transmission opportunity in each OLT scheduling cycle for an ONU or a traffic class of an ONU. This limitation is inadequate. Since EF traffic has rigorous delay requirement and should be scheduled more frequently and be delivered as soon as possible, while AF traffic is not sensitive to delay and jitter but requiring statistical throughput guarantees and BE traffic has no QoS demanding. Therefore, it is desired to develop new DBA protocols to support QoS in a more efficient way.

On the other hand, in practical EPON networks, an ONU normally hosts multiple end users and each user generates heterogeneous QoS-demanding traffic. Hence, a scheduling system in EPON is responsible for QoS delivery not only to aggregate traffic (i.e., traffic classes) of an ONU, but also to individual traffic of end users. Up to now, only a few research efforts have been done on QoS delivery at the level of end users in [KBS04,CCS06,ABH03]. Unfortunately, all of these algorithms adopt S-opportunity per-ONU-per-cycle strategy for inter-ONU scheduling that limits the performance of high priority traffic and suffers round trip time (RTT) idling problem, thus, resulting in lower link utilization. Therefore, designing an efficient hierarchical scheduling system and offering QoS support at the level of end users is another important topic.

1.3 Objectives

Internet QoS is one of the most comprehensively studied topics among the network community, which poses a primary challenge for successful evolution of the Internet

into a commercial infrastructure. QoS refers to the capability of a network to provide preferential delivery service to certain applications and is typically measured in terms of end-to-end delay, jitter (delay variation), throughput, and packet loss rate. In general, end-to-end delay is considered to be the single most important performance parameter [Zha95]. In this thesis, we focus mainly on delay performance. We also study throughput issues. In particular, we investigate *deterministic (worst-case)* end-to-end delay as well as queuing size for MPLS-compliant networks. End-to-end deterministic performance is crucial in the context of guaranteed service over core networks for real-time and performance-critical applications. While for EPON, we target both deterministic and statistical performance evaluation. This is because the communication in EPON is characterized by point-to-multipoint style between an OLT and a collection of ONUs, where both deterministic and statistical performances are very meaningful. The main objectives of this thesis are summarized as follows:

- *To conduct a comprehensive review on scheduling algorithms.* This covers traditional flow-based scheduling algorithms as well as emerging aggregate scheduling algorithms and dynamic bandwidth allocation (DBA) algorithms. Scheduling algorithms are classified and existing problems are analyzed.
- *To extend RI approach in studying deterministic end-to-end QoS into a general network model.* General scheduling scheme and network model need to be introduced to enable derived results of RI approach applicable to more complex networks, like MPLS networks.
- *To investigate aggregate scheduling with fair aggregation to obtain end-to-end QoS.* Efficient and fair aggregator should be devised to construct an aggregate

scheduling in order to achieve end-to-end QoS. Also, the choice of scheduling algorithm for the design of aggregator and aggregate scheduling should have wide implementation in order to utilize the rich results of flow-based scheduling algorithms.

- *To study dynamic bandwidth allocation among multiple ONUs with QoS support in EPON networks.* Efficient DBA algorithm need to be developed in order to provide guaranteed service to delay and delay variation sensitive traffic as well as guaranteeing throughput for bandwidth required traffic and maximizing total network throughput.
- *To investigate user-oriented hierarchical scheduling with QoS support in EPON networks.* Efficient user-oriented hierarchical DBA algorithm need to be designed in order to provide QoS delivery not only to aggregate traffic of an ONU, but also to individual traffic of different users hosted by an ONU.

1.4 Contributions

This thesis contributes to the design, analysis, and implementation of two suits of aggregate scheduling algorithms for MPLS-compliant high speed networks and two DBA protocols for EPON access networks. The performances of these algorithms have been analyzed by both analytical and simulation approaches and compared with related algorithms. The specific contributions are as follows:

1. A general service scheme represented by a latency-rate max-min service curve (LRMMSC) has been proposed, which aims to extend the application of route interference (RI) in the study of aggregate scheduling. Our approach presents two

salient features: 1) the scheme has wide-range implementations owing to the general nature of the concept of service curve, 2) a general network model is considered without the constraints on the manner of packet queuing that makes it applicable to many complex networks, e.g., MPLS network. For a network of LRMMSC, we prove the existence of tight bounds on end-to-end queuing delay and buffer size needed for loss-free packet delivery, provided that all flows obey a given source rate condition in the form of their route interference when entering the network. We conduct simulation experiments to verify the tightness of analytical results and demonstrate the advantage of LRMMSC in providing maximum end-to-end delay.

2. On the other hand, we have proposed a guaranteed rate fair aggregator (GRFA) for traffic aggregation and investigated end-to-end delay bound of a network of aggregate scheduling that incorporates a guaranteed rate (GR) scheduler and a GRFA. Since a GRFA is constructed from a GR scheduler, it inherits the advantage of wide-range implementations of GR that make it suitable for a heterogeneous network environment. We then move forward to develop a fair aggregation capable guaranteed rate (FA-GR) scheduling and a novel priority shaper (P-shaper) with respect to aggregate scheduling. This algorithm has the same level of end-to-end delay bound as the former one and other two-level fair aggregation approaches in [Cob02], while achieving relative lower implementation cost.
3. A new DBA protocol, named hybrid cycle scheme (HCS), has been proposed for dynamic bandwidth allocation with DiffServ support in EPON network. This

- algorithm exhibits a novel feature that potentially provides multiple transmission opportunities (M-opportunities) per-cycle for high priority traffic, which is significant in improving delay and delay variation performance. Specially, it reduces average delay by a range of 20.0% to 95.6% and delay variation by a range of 12.8% to 85.4% at different network loads as compared with previous approaches.
4. We have further developed a number of user-oriented hierarchical bandwidth scheduling algorithms (UHSAs) that support DiffServ and guaranteed fairness to end-users. For inter-ONU scheduling, the hybrid cycle scheme has been modified for the purpose. For intra-ONU scheduling, a new credit-based scheduling approach has been developed. In addition, to improve scheduling efficiency and lower queue management complexity, we design a novel credit-based common queue (CCQ) for each traffic class to enhance scheduling architecture and minimize average number of queues in an ONU. On the other hand, we propose a transmission priority scheme for different queue groups, which together with CCQ mechanism serves the objective of improving delay and jitter performance of high priority traffic, while guaranteeing throughput for bandwidth sensitive medium priority traffic. Detailed simulation experiments have been conducted to study the performance and validated the effectiveness of the proposed protocols.

1.5 Thesis Organization

This thesis is organized as follows:

Chapter 2 conducts a literature review of scheduling algorithms, including per-flow scheduling algorithms, network calculus, aggregate scheduling, and DBA algorithms.

Chapter 3 presents a general service model, namely latency-rate max-min service curve (LRMMSC), to extend route interference approach in the study of aggregate scheduling for a general network model.

Chapter 4 develops a guaranteed rate fair aggregator (GRFA) for aggregate scheduling to study end-to-end delay bound. In addition, another fair aggregation capable guaranteed rate (FA-GR) scheduling is also introduced, which together with a new priority shaper (P-shaper) are used to investigate end-to-end QoS performance.

Chapter 5 proposes a new DBA protocol, called hybrid cycle scheme (HCS), which provides potentially M -opportunities per-cycle for high priority traffic that notably improves delay and delay variation performances.

Chapter 6 proposes a number of user-oriented hierarchical scheduling algorithms (UHSAs) for providing fairness to end-users. Also, a novel credit-based common queue (CCQ) mechanism is devised to enhance scheduling efficiency and minimize average queue number in an ONU.

Chapter 7 provides a summary of the thesis, and discusses directions for future research.

Chapter 2

Literature Review

Scheduling algorithms have been studied extensively since early 1990s in the context of providing performance guaranteed services in packet switched networks. This chapter presents a comprehensive literature review on related works that serve as the background for the research. In particular, the review covers per-flow based scheduling disciplines developed in the context of integrated services (IntServ) architecture [BCS94], network calculus giving a set of mathematical results for performance bounds analysis, aggregate scheduling algorithms proposed in the context of traffic aggregation based on differentiated services (DiffServ) architecture [BBC98] and multi-protocol label switching (MPLS) architecture [FWD01], and dynamic bandwidth allocation (DBA) schemes for Ethernet passive optical networks (EPONs) [KMP01].

2.1 Flow-based Scheduling Algorithms and Network Calculus

We refer to a scheduling algorithm as a flow-based scheduling algorithm if 1) it operates flows in a per-flow basis, i.e., maintaining per-flow QoS state and per-flow queue, and implementing a specific scheduling treatment for each flow, and 2) it does not perform traffic aggregation, i.e., receiving a set of flows as input and outputting

the same set of flows as output. This is in contrast to aggregate scheduling which is a combination of a number of traffic aggregators and a flow-based scheduler, where 1) the aggregator performs traffic aggregation, i.e., receiving a set of flows as input and producing an aggregate as output, and 2) the flow-based scheduler schedules aggregate flows accordingly in the same way as a normal flow-based scheduler. Therefore for a flow-based scheduling network, an individual flow is aware at each node along its path through the network, while for an aggregate scheduling network, an individual flow is normally only aware at edge node and at core nodes only a small number of aggregate flows are aware. The basic architecture of flow-based scheduling algorithm is shown in Figure 1.1 (a). Many known flow-based scheduling disciplines have been proposed in literature and some survey works can be found in [Zha95,JG02]. Although flow-based scheduling disciplines are developed in the context of IntServ architecture to provide deterministic (worst-case) end-to-end performance guarantees, they also serve as the foundation of aggregate scheduling algorithms and DBA protocols. In essence, the latter two originate from the former one to solve scheduling issues in emerging aggregation-based networks and remote scheduling systems. Thus, some fundamental results are applicable for all scheduling systems. Below, we have conducted surveys on flow-based scheduling disciplines, scheduling classes for end-to-end QoS study in heterogeneous scheduling environments, and network calculus for performance bounds analysis.

2.1.1 Scheduling Algorithms and Performance Results

Generalized processor sharing (GPS) [PG93,PG94] is an ideal fluid scheduling policy that provides an exact max-min fair share allocation. More precisely, GPS allocates

the whole outgoing capacity to all backlogged connections in proportion to their minimum rate (bandwidth) requirements. Let $B(t)$ denote the set of backlogged connections at time t , the service rate $g_i(t)$ allocated to connection i at t is given as:

$$g_i(t) = \frac{r_i}{\sum_{j \in B(t)} r_j} \times C$$

where r_i denotes the minimum allocated rate for connection i , C denotes the outgoing link capacity and satisfies $\sum_{i \in B(t)} r_i \leq C$. GPS provides the best fairness, as well as achieving minimum bandwidth guarantee to each flow regardless of the behavior of other flows and deterministic end-to-end delay bounds to flows whose traffic is envelope constraint. Nonetheless, GPS makes assumption of infinitesimal packet size which is unimplementable. Therefore, GPS generally works as a reference to measure the effectiveness of an entire class of GPS-based packet scheduling algorithms, which includes many well known disciplines like weighted fair queuing (WFQ) [DKS90], self-clocked fair queuing (SCFQ) [Gol94], virtual clock (VC) [XL95], start-time fair queuing (SFQ) [GVC96], worst-case fair weighted fair queuing (WF²Q) [BZ96], WF²Q+ [BZ97], frame-based fair queuing (FFQ) [SV96b], etc. These algorithms are also classified as sorted priority scheduling where a timestamp (virtual time) is computed for each packet and packets are sorted based on their timestamps and are transmitted in the order.

In order to compare the performance of GPS and its approximations, an error term E is introduced. Let $d(i)$ and $G(i)$ be the time of the i th departure under scheduler S and GPS with rate R , respectively. The accuracy of S with respect to GPS is determined by E , such that $d(i) \leq G(i) + E$. E determines the maximum per-hop

delay bound. Table 2-1 summarizes the error term E for several known algorithms. On the other hand, another two critical properties in terms of end-to-end delay bound and scheduling complexity are also presented.

Table 2-1: E value, end-to-end delay bound, and complexity of GPS schedulers.

Scheduler	E	End-to-end delay bound*	Complexity
GPS	0	σ_j / r_j	---
WFQ	$\frac{L_{\max}}{C}$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H \frac{L_{\max}}{C_i}$	$O(N)$
WF ² Q	$\frac{L_{\max}}{C}$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H \frac{L_{\max}}{C_i}$	$O(N)$
WF ² Q+	$\frac{L_{\max}}{C}$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H \frac{L_{\max}}{C_i}$	$O(\log N)$
SCFQ	$\frac{L_{\max}}{C} (N-1)$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H K_i \frac{L_{\max}}{C_i}$	$O(\log N)$
SFQ	$\frac{L_{\max}}{C} (N-1)$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H K_i \frac{L_{\max}}{C_i}$	$O(\log N)$
VC	$\frac{L_{\max}}{C}$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H \frac{L_{\max}}{C_i}$	$O(\log N)$
FFQ	$\frac{L_{\max}}{C}$	$\frac{\sigma_j + HL_{\max}}{r_j} + \sum_{i=1}^H \frac{L_{\max}}{C_i}$	$O(\log N)$

* Leaky bucket model in terms of (σ_f, ρ_f) is used as the traffic constraint function in deriving end-to-end delay bound for a flow. Refer to Definition 2.5.

Round robin (RR) scheduler is the simplest approximation of GPS. It polls each queue in a cyclic order and serves a packet from any nonempty queue. Thus, RR has $O(1)$ complexity that is significantly simpler than timestamp-based GPS approximations. Many RR-based scheduling protocols have been designed in literature. Weighed round robin (WRR) [KSC91] is a natural extension of RR to serve connections with different weights. Deficit round robin (DRR) [SV96a] is a further modification of WRR in improving scheduling efficiency in variable packet size scenario. While smoothed round robin (SRR) [Guo01], stratified round robin [RP03],

and fair round robin [YD05] serve the objective to improve delay, burstiness, and fairness performance of DRR. RR-based algorithms are also categorized as frame-based scheduling where transmission timeline of an outgoing link is divided into static or variable frames and one or more packets of each flow are allowed to transmit during a frame period. Table 2-2 presents the properties of single packet delay bound, i.e., maximum delay experienced by a packet at the head of a queue, and fairness for the popular RR schedulers. Specially, fairness is measured by a relative fairness bound (RFB), i.e., the maximum difference in the normalized service $S_i(t_1, t_2)$ received by any two flows during backlogged period of (t_1, t_2) . The normalization factor can be the share f_i , weight w_i , or rate r_i .

Table 2-2: Single packet delay bound and fairness of RR schedulers

Scheduler	Single packet delay bound	Fairness
DRR	$\frac{3\sum_m Q_m - 2Q_i}{C} - \frac{L_{\min}}{r_i}$	$\left \frac{S_i(t_1, t_2)}{f_i} - \frac{S_j(t_1, t_2)}{f_j} \right \leq 2L_{\max} + Q$
SRR	$\frac{2L_{\max}}{w_f} + (N-1)\frac{2L_{\max}}{C}$	$\left \frac{S_i(t_1, t_2)}{w_i} - \frac{S_j(t_1, t_2)}{w_j} \right < \frac{(k+2)L_{\max}}{2\min(w_i, w_j)}$
Stratified RR	$\frac{12L_{\max}}{r_i}$	$\left \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right \leq 5L_{\max} \left(\frac{1}{r_i} - \frac{1}{r_j} \right)$
FRR	$\frac{c_1 L_{\max}}{r_i}$	$\left \frac{S_i(t_1, t_2)}{r_i} - \frac{S_j(t_1, t_2)}{r_j} \right \leq L_{\max} \left(\frac{c_1}{r_i} - \frac{c_2}{r_j} \right)$

*1: Q_m is quantum worth of bits given to a flow m , $Q = \min_m(Q_m)$. C is the link capacity.

*2. N is the number of flows in the system. k is the order of weight spread sequence (WSS) used by SRR.

*3. c_1, c_2 are constants. $L_{\max}, L_{\min}$ are the maximum and minimum packet length in the network, respectively.

GPS approximations discussed above are also classified as *work-conserving* disciplines, i.e., a server is never idle when there is a packet waiting service. Inversely, a server of non-work conserving discipline assigns each packet an eligibility time and never sends a packet before its eligibility time, even it is idle. The theory behind non-

work conserving discipline is that by re-shaping (regulating) traffic at each node traffic arriving at the next hop is more predictable, resulting in improved end-to-end delay and jitter performance and reduced buffer size requirement. However, non-work conserving schedulers yield higher average delay and lower network throughput compared with work-conserving schedulers. The known non-work conserving disciplines include jitter earliest-due-date (jitter-EDD) [VZF91], stop-and-go [Gol91], hierarchical round robin [KKK90], and rate-controlled static priority (RCSP) [ZF94]. [Zha95] conducted a comprehensive survey on non-work conserving disciplines and presented a unified framework called rate-controlled service discipline to generalize them.

Figure 2.1: An illustration of a heterogeneous scheduling network.

2.1.2 Scheduling Classes

In a commercial wide area networking environment, each switch may employ a different scheduling algorithm as illustrated in Figure 2.1. In order to analyze end-to-end QoS for such heterogeneous scheduling environments, several general scheduling classes are proposed, each of which covers a number of scheduling algorithms. For a network of switches each of which deploys a scheduling algorithm belonging to the same scheduling class, deterministic end-to-end QoS can be derived. Thus, a well defined scheduling class plays a critical role in studying end-to-end QoS in a

heterogeneous network environment. Several known scheduling classes have been defined. Below, we mainly focus on two scheduling classes, namely *guaranteed rate (GR)* and *packet scale rate guarantee (PSRG)*, which are closely related to our work, and discuss their relationships.

GR scheduling class was first defined in [GLV95] and later generalized in [GV97]. It covers many known work-conserving disciplines such as VC, WFQ, SCFQ, SFQ, SRR, D-EDD, etc, and a class of non-work conserving disciplines, i.e., rate controlled service disciplines. The idea of GR is to guarantee a delay deadline for each packet with regard to a calculated virtual finish time, termed guaranteed rate clock (GRC). Formally, the definition of GR is given as follows:

Definition 2.1 [GR]: A scheduling algorithm at node j is said to be GR for a flow f , if for every packet p_f^i of f , the departure time $d_{f,i}^j$ is less than $GRC_{f,i}^j + \beta_j \cdot \beta_j$ is the latency term of the scheduling algorithm and $GRC_{f,i}^j$ is defined as:

$$\begin{cases} GRC_{f,i}^0 = 0 \\ GRC_{f,i}^j = \max\{a_{f,i}^j, GRC_{f,i-1}^j\} + l_f^i / R_f \end{cases}$$

where $a_{f,i}^j$ is the arrival time of the packet at node j , l_f^i is the packet length of p_f^i and R_f is the rate reserved for flow f at node j . A collection of β values of some known scheduling algorithms can be found in [GV97,Jia03]. The E values listed in Table 2-1 are also β values of the corresponding GR schedulers.

PSRG is a stronger definition of service model than GR [BBC02,BC03], which is introduced as a new definition of service for expedited forwarding (EF) traffic [DCB02] in the context of DiffServ network. The definition of PSGR is as follows:

Definition 2.2 [PSRG]: A scheduler j is said to offer an EF aggregate f a “PSRG with rate R_f and latency E_j ” if the departure time $d_{f,i}^j$ of the i th packet p_f^i satisfies the following condition for all $i \geq 0$:

$$d_{f,i}^j \leq F_{f,i}^j + E_j$$

where $F_{f,i}^j$ is defined iteratively by:

$$\begin{cases} F_{f,0}^j = 0, & d_{f,0}^j = 0 \\ F_{f,i}^j = \max[a_{f,i}^j, \min(d_{f,i-1}^j, F_{f,i-1}^j)] + \frac{l_f^i}{R_f} \end{cases}$$

It shows from the definition that PSRG implies GR, or PSRG is a special subclass of GR. Thus, all properties known for GR are also applicable to PSRG. In particular, PSRG server captures not only how much later but also how much earlier a server transmits a packet than GPS, i.e., it has finite error term E_1 and E_2 for $G_{f,i}^j - E_1 \leq d_{f,i}^j \leq G_{f,i}^j + E_2$, where $G_{f,i}^j$ is the i th departure time under scheduler GPS. Thus, PSRG has an important property of delay-from-backlog, while GR has not [BBC02].

Another important scheduling class is *latency-rate (LR)* that was introduced in [SV98] to describe the worst-case behavior of scheduling algorithms. It is based on a concept of *busy period*, a period of time during which the average arrival rate of a connection remains at or above its reserved rate. A LR server offers service rate to a connection f at least equal to its reserved rate R_f with a latency Θ during its busy period, i.e., $W_f^i(\tau, t) \geq R_f(t - \tau - \Theta)^+$, where $(x)^+ \equiv \max\{x, 0\}$, $W_f^i(\tau, t)$ is the service

received by f during $[\tau, t]$, and τ is the beginning of a busy period. LR is equivalent to GR in essence. An in-depth discussion on GR and LR is conducted in [Jia03].

2.1.3 Network Calculus

Network calculus is a theory of deterministic queuing systems developed in computer networks that can be used for computing tight bounds on delay, backlog, and arrival envelope in a lossless environment. Fundamental works have been done in [Cru91a, Cru91b, Cru95, Kur92, Sar96, BCO92, Cha00, AR96, Bou98, BT03]. Here, we present the basic network calculus concepts of arrival curve, service curve, and shaper, and the fundamental network calculus results.

Consider a data flow, described by its cumulative function $R(t)$, which is the number of bits seen in time interval $[0, t)$. The concept of arrival curve is defined as follows.

Definition 2.3 (arrival curve) [AR96, Cha00]: Given a wide sense increasing function α defined for $t \geq 0$, we say $R(t)$ has α as an arrival curve (envelope) or $R(t)$ is α -smooth (or α -constraint) if for all $0 \leq s \leq t$,

$$R(t) - R(s) \leq \alpha(t - s), \text{ or equivalently, } R \leq R \otimes \alpha.$$

where $\otimes$ is the min-plus convolution operation that is defined as

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} \{f(t - s) + g(s)\}.$$

Let $R(t)$ and $R^*(t)$ be the cumulative input function and departure function, respectively, at a server, a service curve for a flow is defined as follows:

Definition 2.4 (service curve) [Cha00, CO96]: A server is said to provide a service curve β to a flow, if and only if for any $t \geq 0$, there exists some t_0 , such that:

$$R^*(t) - R(t_0) \geq \beta(t - t_0), \text{ or equivalently, } R^* \geq R \otimes \beta.$$

Three fundamental network calculus results are summarized as follows:

Theorem 2.1 (backlog) [CO96]: Assume a flow, having α -constraint and receiving a service curve of β at a server. The backlog $R(t) - R^*(t)$ at the server satisfies:

$$R(t) - R^*(t) \leq \sup_{s \geq 0} \{ \alpha(s) - \beta(s) \} = (\alpha \oslash \beta)(0)$$

where $\oslash$ is the min-plus de-convolution operation that is defined as

$$(f \oslash g)(t) = \sup_{s \geq 0} \{ f(t + s) - g(s) \}.$$

Theorem 2.2 (delay bound) [CO96]: Assume a flow, having α -constraint and receiving a service curve of β . The virtual delay $d(t)$ for all $t \geq 0$ satisfies:

$$d(t) \leq h(\alpha, \beta) = \sup_{s \geq 0} \{ T : T \geq 0 \text{ and } \alpha(s) \leq \beta(s + T) \}$$

where $h(\alpha, \beta)$ is the horizontal deviation between the two functions.

Theorem 2.3 (output flow) [CO96]: Assume a flow, having α -constraint and receiving a service curve of β . The output flow is constrained by $\alpha^* \leq \alpha \oslash \beta$.

Next, we introduce another important concept called *shaper*. A shaper, with shaping curve α , is a bit-processing device that forces its output to have α as an arrival curve. A greedy shaper is a shaper that delays the input bits in a buffer, whenever sending a bit would violate the constraint α , but outputs them as soon as possible.

Theorem 2.4 (input-output characterization of greedy shapers) [AR96, Cha00, Sar96]: Consider a greedy shaper with shaping curve α . Assume that the shaper buffer is empty at time 0, and that it is large enough so that there is no data loss. For an input flow R , the output R^* is given by: $R^* = R \otimes \bar{\alpha}$.

$\bar{\alpha}$ is the sub-additive closure of α which is defined as $\bar{\alpha} = \delta_0 \wedge \alpha \wedge (\alpha \otimes \alpha) \wedge (\alpha \otimes \alpha \otimes \alpha) \wedge \dots$. $\delta_0(t)$ is the burst-delay function that is defined as $\delta_T(t) = 0$ for $t \leq T$ and ∞ for $t > T$. $\wedge$ is define as $f \wedge g = \min(f, g)$.

It has been shown [AR96, Cha00, Sar96] that a shaper with a sub-additive α such that $\alpha(0) = 0$ offers α as a service curve. Specially, a *leaky bucket* is a shaper defined as follows:

Definition 2.5 (Leaky bucket): a buffered leaky bucket is a shaper with α of the form $\alpha(t) = \sigma + \rho t$, where σ is the burst size and ρ is the average rate. It forces a flow to be constraint by the arrival curve α , i.e., the output is (σ, ρ) -constraint.

2.2 Aggregate Scheduling

Instead of scheduling traffic at the granularity of per-flow at each node, aggregate scheduling attempts to aggregating flows into a limited number of aggregate flows, or aggregates for short, using some mechanisms at each node and scheduling traffic at the scale of aggregate flows. Aggregate scheduling greatly reduces scheduling complexity and improves network scalability by reducing the number of flows in the network, while still providing end-to-end QoS guarantees to individual flows in spite of traffic aggregation. Generally, an aggregate scheduler is composed of an aggregator for traffic aggregation and a per-flow scheduler for aggregate flows

scheduling, as illustrated in Figure 1.1 (b). Aggregate scheduling can be categorized into two basic classes according to their flow aggregation methods: FIFO aggregation based solution and fair aggregation based solution. The first one is further divided into two subclasses according to their different approaches to achieve end-to-end delay bound. One is characterized by route interference approach and the other features network utilization constraint in order to provide guaranteed end-to-end QoS.

Figure 2.2: An example of per-node delay mapping.

2.2.1 Route Interference (RI) based Aggregate Scheduling

The concept of RI was first introduced in [CFZ98] to explore deterministic end-to-end delay bound and backlog for a connection-oriented global network (e.g., ATM network) of arbitrary topology, homogeneous link speed, fixed packet size, single-class traffic, and discrete time model. The idea of RI is inspired by examining the delay of a packet experienced at a single node. Specially, for a packet arriving from a link I_j at a node, the queuing delay of the packet is bounded by the service time of packets arriving on incoming links except I_j during the buffer busy period and departing before that packet. An example is illustrated in Figure 2.2. Packets $p, 1, 2, 3$, and 4 are in the same busy period, and 1 has just departed while $2, 3$, and 4 are still in the queue when p arrives. Thus, p experiences three time units queuing delay induced by $2, 3$, and 4 . Assume p and 4 come from the same incoming link I_j while

1, 2, and 3 are not. Then, the delay of p induced by 4 can be mapped to 1, i.e., the queuing delay of p can be expressed by 1, 2, and 3. This implies that the part of delay caused by packets arriving over I_j can be mapped to packets from other incoming links not directly delaying p . This observation has great significance if it can be extended into the global network and can meet the requirement that there is at most one packet from each interfering flow delaying p at any node along its path. This is because if the requirement is satisfied throughout a global network, then the end-to-end queuing delay of a packet of a flow is bounded by the service time to transmit a number of packets equal to the amount of interfering flows through the network (so-called RI of the flow). Note that a flow is called as an interfering flow of the other flow if they arrive over different incoming links and queue at the same buffer of an outgoing link at a node. Chalmtac [CFZ98] has provided the implementation in the environment of homogeneous ATM network that by strictly controlling traffic flows to obey a specific source rate condition in terms of minimum inter-packet space at the network boundary to ensure at most one packet of each flow exists during a chained buffer backlogged (busy) period (so-called superchain). In [BH00,Zha99], the result on buffer capacity is improved for the same network model. In [OB03], the authors extended the results into a more generic connection-oriented network, covering heterogeneous link speeds, variable packet sizes and multi-class traffic. For the traffic of each class, packets are FIFO queued in a single buffer and a strict priority scheduling is used to schedule packets of the several class-based buffers. In the paper, per-node queuing delay and buffer capacity required are proved to be bounded for the traffic with the highest priority.

Previous works tend to confine to a simple scheduling scheme and study traffic in a single-class environment or the highest priority traffic in a multi-class environment. This is rather restrictive. First, for future networks, it is quite likely that many other scheduling disciplines are deployed to better offer QoS, such as the scheduling family of the GPS and RR. Second, the assumption of the network model is rather restrictive and makes the result not applicable to complex networks where packets belonging to the same class may queue in different buffers. In addition, the results that have been obtained for traffic with the highest priority are not applicable to traffic with low priority. Therefore, further studies are required to extend RI approach into a general scheduling environment and for a general network model.

Figure 2.3: Aggregate Scheduling Architecture in DiffServ.

2.2.2 Aggregate Scheduling Featuring Network Utilization

DiffServ is characterized by traffic aggregation and accordingly raises research interests in aggregate scheduling. Some pioneer work has been done in [CB00, Jia02, BBC02] to provide deterministic end-to-end delay bound to EF traffic under the condition of traffic aggregation. This is achieved by limiting the amount of EF traffic entering the network in order to keep the utilization of the link speed or reserved rate

at each node below a specified ratio. The scheduling architecture is illustrated in Figure 2.3. The assumptions on the network model are made as follows:

- Traffic is classified into a limited number of classes as defined in DiffServ and all traffic belonging to the same class is treated as a single aggregate flow and is FIFO queued in a single buffer at each node. This is the so-called FIFO aggregation.
- At the network ingress, any flow i is (σ_i, ρ_i) -constraint. Note that a flow may itself consist of a number of micro-flows (connections) sharing the same ingress-egress pair.
- The EF aggregate receives a service of GR or PSGR at each node h with rate R_h and latency E_h . H denotes the maximum number of hops a flow traverses in a network, also called the diameter of the network.
- Let F_h denotes the set of all EF flows constituting the EF aggregate on link h , it requires the condition $\sum_{i \in F_h} \rho_i \leq \alpha R_h$ is satisfied for a certain ratio $\alpha < 1$. Also, the overall bound β on the burstiness satisfies $\sum_{i \in F_h} \sigma_i \leq \beta R_h$ at any node i .

With the above assumptions, a bound is derived on end-to-end delay.

Theorem 2.5 (End-to-end delay) [CB00,Jia02]: If $\alpha < 1/(H-1)$, a bound on end-to-

end delay is $D_{end-to-end} = \frac{H(\beta + E_{\max})}{1 - (H-1)\alpha}$, where $E_{\max} = \max_h \{ \frac{L_{\max}}{R_h} + E_h \}$.

It shows that a closed form expression of end-to-end delay bound exists given the network utilization below a certain value. This solution of aggregate scheduling has some similarity with the one of RI-based in that they both depend on a global network parameter to perform traffic shaping or admission control at network boundary. While RI-based solution adjusts inter-packet space of a flow with regard to its RI value, this

solution conducts traffic admission control to ensure any link or reserve rate utilization below a certain ratio which is determined by the network diameter. The major weakness of the solution featuring network utilization is that the network utilization can be very low. For example, when $H = 10$, $\alpha < 11\%$. This is due to the increase of burstiness of each flow when passing each node along its path [BT03]. To overcome this weakness, fair aggregation is needed. This leads to the development of the other class of aggregate scheduling.

2.2.3 Aggregate Scheduling with Fair Aggregator

The fundamental work of aggregate scheduling with fair aggregator was pioneered by [Cob02]. The concept of aggregator in [Cob02] is fundamentally different from the FIFO aggregator as discussed in Section 2.2.2 where the constituent flows of an EF aggregate upon departing a node may take different paths and become part of other aggregates at other nodes. In [Cob02], an *aggregator* is defined as a special scheduler that receives as input a set of flows and produces as output a single aggregate flow (aggregate for short). The set of input flows are said to be the immediate constituents of the output aggregate. A flow g is said to be the root of another flow f at some point in the network if g is not a constituent of any other flow while f is a constituent of g . A scheduler is said to be an *aggregate scheduler* to an input flow if the flow is aggregated with other flows to form the output aggregate, and is said to be a *non-aggregate scheduler* if the output flow is just the input flow. Inversely, a *separator* is defined as a process that receives as input an aggregate and produces as output the set of constituents of the aggregate. A separator does not introduce packet delay. Therefore, the set of flows that are aggregated into an aggregate at a node must

destine for the same output port at the next hop and will continue sharing their path at least before they are separated by a separator. This is fundamentally different from the concept of FIFO aggregator introduced in section 2.2.2 although the fair aggregator in [Cob02] can be implemented by FIFO approach as well.

Figure 2.4: An Illustration of Aggregator Architecture.

A fair aggregator is considered to be constructed by a fictitious scheduler and an traffic assembler as illustrated in Figure 2.4. The fictitious scheduler v works as a normal scheduler that schedules traffic from a number of constituent flows with output capacity limited to R_g , with $R_g \geq \sum_i r_{f_i}$. The assembler s is used to aggregate traffic from different input flows into an aggregate such that all constituents of the aggregate are transparent to the following schedulers. Different aggregation mechanisms can be implemented. Aggregation approach in MPLS protocol can be found in [FWD01]. The assembler s forwards packets at a rate of R_g although its capacity is C_s ($C_s \geq R_g$). Meanwhile, s forwards each packet to its output channel at exactly the same time v would forward the same packet.

In [Cob02], a fair aggregator is constructed by a start-time scheduler. In particular, an aggregator is said to be fair, if and only if the start-time values of a packet at an aggregator and its next start-time scheduler (can also be a fair aggregator) have

maximum constant difference. Two types of fair aggregator are devised in [Cob02]. A basic fair aggregator forwards packet with rate always limited to R_g . A greedy fair aggregator could forward packet with rate higher than R_g when all constituents have an arrival rate higher than their reserved rates, such that $R_g < \sum_i r_{fi}$. The work of [Cob02] has two limitations: 1) both of the aggregators are non-work conserving, 2) it implies there is only one outgoing aggregate from each aggregator at an output interface. In [SS05], the authors extended the result in [Cob02] by several ways. First, a fair aggregator is defined that is applicable to both work-conserving and non-work conserving GR scheduling algorithms. Specially, an aggregator is said to be fair, if a packet's GRC values at the aggregator and its next hop have maximum constant difference. By using token-bucket traffic model and output constraint derivation theory of Theorem 2.3, end-to-end delay bounds are derived for both stand-alone aggregators and hierarchical aggregators. Second, a non-work conserving rate-controlled fair aggregator (RCFA) is introduced similar with that of [Cob02] but end-to-end delay bound is derived based on the assumption that multiple aggregators exist at a node output.

In [Jia06], a packet scale fair aggregator (PSFA) is introduced that makes a big leap in several ways. First, PSFA is constructed from PSRG [BBC02,BC03] and aims to provide per-domain PSRG in the presence of flow aggregation. PSRG is the new definition of EF service in the context of DiffServ which is a much stronger service definition than start-time scheduler and GR scheduler. Second, PSFA can be implemented by both one-level schedulers and hierarchical schedulers, while start-time aggregators and GR aggregators are implemented by only one-level schedulers.

Third, the definition of start-time and GR aggregators depends on the behavior of the next scheduler as discussed above while PSFA does not have such dependency, i.e., PSFA is based on the behavior of the server alone.

2.3 Dynamic Bandwidth Allocation (DBA) for EPON

In an EPON system, all ONUs share the common uplink channel, thus some medium access control (MAC) mechanism is needed in the OLT to arbitrate transmission to avoid data collision and at the same time efficiently and fairly distribute upstream bandwidth among all ONUs according to their service level agreement (SLA) with the network operator. The MAC mechanism divides the upstream transmission timeline into a stream of static or variable length mini-slots, termed timeslots. Each timeslot represents a transmission opportunity that can deliver a number of packets of the ONU which the timeslot is assigned to. OLT determines the size of each timeslot and controls access to these timeslots by the ONUs using a specific scheduling algorithm.

To support timeslot scheduling, the multi-point control protocol (MPCP) has been developed by the IEEE 802.3ah Task force [802]. MPCP does not mandate a particular scheduling algorithm; rather, it is a supporting protocol that facilitates the implementation of various scheduling algorithms. MPCP relies on two control messages, namely GATE and REPORT, to manage upstream bandwidth. An ONU sends a REPORT message to the OLT, containing the timestamp and queue status. The OLT calculates the round-trip time (RTT) using the reported timestamp and determines bandwidth allocation using the reported queue information. The OLT sends a GATE message to an ONU with the beginning and length of the transmission timeslot for the ONU. A REPORT message can support up to eight queue reports, and

a GATE message can grant at most four timeslots. Therefore, this mechanism is efficient in implementing differentiated services for multiple class queues in an ONU.

Many different scheduling algorithms may be implemented in an OLT. Static timeslot assignment is the simple way for bandwidth allocation [KMP01]. A fixed timeslot is allocated to an ONU in fixed cycle. This method is not adaptive to the bursty nature of network traffic, resulting in inefficient channel utilization. As a result, dynamic bandwidth allocation (DBA) is highly desired, which dynamically allocates a variable size of timeslot to an ONU based on the instantaneous network traffic condition. DBA also provides statistical multiplexing between ONUs to improve bandwidth utilization. Thus, researches on EPON scheduling are focused on DBA algorithm design and analysis. In the following, we present a literature survey on DBA algorithms and categorize them into different classes according to their major characteristics.

Figure 2.5: Taxonomy of dynamic bandwidth allocation (DBA) algorithms.

The taxonomy to be used in the survey is constructed in a hierarchical architecture as shown in Figure 2.5. We categorize DBA algorithms into three classes: Class of algorithms with statistical multiplexing and bandwidth guarantee, Class of algorithms with service differentiation support for QoS assurance, and Class of algorithms of

user (queue) oriented hierarchical scheduling. The second class includes two subclasses: algorithms featured in Single transmission opportunity (S-opportunity) per-ONU-per-cycle, and algorithms featured in S-opportunity per-class-per-cycle.

2.3.1 Algorithms with Statistical Multiplexing and Bandwidth Guarantee

This class of algorithms has two salient features: 1) providing statistical multiplexing and using interleaved polling approach to overcome the fundamental drawback of static allocation methods, 2) providing minimum bandwidth guarantee to each ONU according to their SLAs. The algorithms include interleaved polling with adaptive cycle time (IPACT) approach and its extensions.

Figure 2.6: An Illustration of the Interleaved Polling Mechanism.

IPACT [KM02] is designed to overcome the drawback of static allocation methods. To this end, an interleaved polling mechanism is proposed which allows the OLT to send a GATE message to the next ONU before the REPORT message and data from previous ONU(s) arrive. An illustration is presented in Figure 2.6. The OLT sends GATE message to ONU₃ before the transmission of ONU₂ and even ONU₁ finishes. This is workable since the OLT knows when the last bit of ONU₂ will arrive.

Therefore, knowing the RTT of ONU₃, the OLT can schedule a Grant to ONU₃ such that the first bit from ONU₃ will arrive a small guard time after the last bit from ONU₂. The interleaved polling approach can significantly improve channel utilization by reducing the accumulation of walk times (switchover times) associated with a polling system [HO87].

The OLT polls ONUs orderly in round-robin (RR) fashion. To prevent the channel monopolization of ONUs with high data volume, an ONU specific maximum transmission window (MTW) size W_i^{Max} is defined, which subsequently determines the maximum polling cycle time T_{max} . Several grant scheduling services are introduced for performance evaluation, including fixed, limited, gated, constant credit, linear credit, and elastic. The simulation results show that the limited service has the best performance. The limited service grants bandwidth by $W_i = \min(R_i, W_i^{Max})$, i.e., satisfying the bandwidth request R_i , but no more than its limit. Accordingly, the cycle time is dynamic and adaptive to the instantaneous bandwidth demands. If at least one ONU has less data, the cycle time becomes shorter than T_{max} , and therefore the available bandwidth to all other ONUs will increase proportionally to their W_i^{Max} .

IPACT has made a quantum leap from static assignment approaches. Nonetheless, there remains room for improvement. In [SRC04], the authors argued that IPACT limited service suffers severe utilization degradation under non-uniform traffic. This is due to strictly enforcing MTW on ONUs such that the guard time occupies large part of cycle time under non-uniform traffic condition. To resolve this problem, a new scheme was proposed which relaxes maximum timeslot restriction. Another algorithm

for this purpose, called sliding cycle time-based (SLICT) protocol, is proposed in [KPK05]. SLICT introduces a concept of remnant time T_r , the accumulated savings from previous ONUs are used, to calculate the next grant. Furthermore, SLICT introduces sliding cycle time constraint to guarantee the upper bound of maximum polling cycle interval of any ONU to be $T_{\max}$. By these two techniques, SLICT achieves significant improvement in utilization, delay, loss, and queue size under various traffic conditions, compared with IPACT.

In order to improve performance of delay and queuing length, several estimation-based algorithms have been proposed by predicting arrivals in granting intervals. More precisely, next grant is derived from the previous grant by deducting an adaptive factor, i.e., $G(n+1) = G(n) - D(n)$. In [BNL03], $D(n)$ is equal to $\alpha R(n)$. α is the gain factor and $R(n) = G(n) - [q(n-1) + a(n-1)]$, where $q(n)$ is the queue length at cycle n and $a(n)$ is the amount of arrival during $[n-1, n]$ cycle. Using control theoretic arguments, it shows that for piecewise constant traffic with seldom jumps, the system is asymptotically stable for $0 < \alpha < 2$. Whereas in [JB06], $D(n)$ is designed to be $k_1\{q(n-1) - q^*\} - k_2\{q(n) - q^*\}$, where k_1 and k_2 are the control gains, and q^* is the desired queue length. The asymptotical stable condition is $0 < k_1 < k_2 < 2$.

Most DBAs fall into RR-fashion scheduling that schedules ONUs orderly in each cycle. In [ZM05], two new methods are introduced. One is called the longest-queue-first (LQF) algorithm, which polls ONUs in descending order of instantaneous queue length in each cycle. The other is called the earliest-packet-first (EPF) algorithm, which polls ONUs in ascending order of the arrival time of the first packet waiting in

each ONU queue. These two algorithms improve packet delay compared with RR-based DBA algorithm under medium network load.

Summarily, IPACT-style algorithms achieve efficient bandwidth utilization through statistical multiplexing and interleaving polling, and provide minimum bandwidth guarantee through weight-based granting. The main drawback of the algorithms is the lack of QoS support and service differentiation.

2.3.2 Service Differentiation Support for QoS Assurance

Supporting QoS is a crucial issue for successful implementation of EPON. To this end, traffic classification (e.g., DiffServ) is needed to implement differentiated services to effectively support traffic with diverse QoS requirements. We classify DBA algorithms of inter-ONU scheduling and with QoS support into two groups, and discuss each of them in the following subsections.

2.3.2.1 Algorithms of S-opportunity per-ONU-per-cycle

This class of algorithms supports traffic classification and service differentiation, and features delivering single transmission opportunity to each ONU in each cycle. We mark this scheduling class as S-opportunity per-ONU-per-cycle solution.

In [KMD02], the authors studied an IPACT-extended scheme for QoS support which combines limited service [KM02] and strict priority scheduling, and found that queuing delay for lower priority classes increases when the network load decreases at light load region, a phenomenon they called light-load penalty. This is due to strict priority scheduling where new arriving high priority (*HP*) packets can preempt reported lower priority (*LP*) packets in the next timeslot, deferring the transmission of

LP traffic by one or more cycles. To cope with this problem, two approaches were proposed, namely two-stage buffers and constant-bit-rate (CBR) credit. Two-stage buffers method employs a first-come-first service (FCFS) second-stage queue to prevent preemption of reported LP traffic to eliminate light load penalty. Nonetheless, it substantially increases delay for HP traffic. CBR credit predicts HP traffic arrivals by the next grant and allocates more timeslot size accordingly. It partially eliminates light-load penalty but needs to acquire extra information on arrival process.

Figure 2.7: An illustration of the interleaved polling with stop mechanism.

In [AYD03,SBG05,Zhe06], several algorithms that make use of interleaved polling with stop (IPS) mechanism have been developed. As shown in Figure 2.7, the OLT receives reports from all ONUs before starting the next polling cycle, thus it could make more intelligent bandwidth allocation to provide enhanced QoS and fairness performance by taking into account the bandwidth demands from all ONUs and their SLAs. The DBA protocol in [AYD03] partitions ONUs into under-loaded and over-loaded groups, and distributes total excess bandwidth from under-loaded ONUs among over-loaded ONUs based on their bandwidth demands. In addition, a local priority-based queue management algorithm is proposed to alleviate the light load penalty problem by transmitting reported packets ahead of new coming packets. This

approach assumes that total excess bandwidth is always fully occupied by over-loaded ONUs, which is not necessarily true. In [SBG05], a measure is introduced to remedy this deficiency by limiting grant to an ONU not exceeding its request. The deficiency of IPS mechanism is that it brings some idle time ($RTT + \text{processing time}$) in each cycle as shown in Figure 2.7 during which the EPON channel is not utilized. This is because the grant program is invoked only after all requests have been received. An early allocation mechanism is introduced in [AYD03] to overcome this problem by scheduling a lightly loaded ONU instantaneously without waiting. In [Zhe06], an improved approach is proposed which introduces a tracker to monitor the end of the idle period and allows scheduling heavily loaded ONUs in the idle period to make sufficient use of the idle period. The early allocation mechanism improves utilization but at the expense of large delay and delay variation due to huge fluctuation of ONU positions in each cycle.

In [CH02,Cho04], two DBA algorithms are designed for multimedia services support, which use fixed polling cycle and prioritize bandwidth allocation at the frame level that HP requests are always granted prior to lower ones. Since HP traffic can monopolize upstream channel, this brings unlimited unfairness to ONUs and further to end-users. Moreover, these approaches adopt IPS mechanism that suffers RTT idling and use fixed polling cycle resulting in large delay at light load.

In [MZC03], a bandwidth guaranteed polling [BGP] approach is proposed, where ONUs are partitioned into bandwidth guaranteed [BG] group and non-bandwidth guaranteed (non-BG) group. This algorithm is able to provide guaranteed bandwidth to BG ONUs by allocating each BG ONU one or multiple evenly divided and

distributed bandwidth granules in each scheduling frame. In [ZMC05], BGP is extended to support DiffServ by adopting a two-stage queuing mechanism like that in [KMD02]. The drawback of BGP is that OLT polls ONUs in roll-call polling manner which brings severe channel utilization problem due to walk time cost. Furthermore, upstream channel division in fine granules substantially increases guard time expense. Thus, this approach is not suitable for a large network of ONUs. Note that, although BGP could distribute more than one granule to some ONUs, we classify it into S-opportunity per-ONU-per-cycle solution since only few premium ONUs could receive such service due to the inefficient roll-call polling method in the context of EPON.

In summary, DBA algorithms of S-opportunity per-ONU-per-cycle solution provide service differentiation to support enhanced QoS. However, although these proposals are aware of offering prior services to HP traffic, each ONU has only a single transmission opportunity in each cycle, thus still yielding large queuing delay for HP traffic and even worse for delay variation due to the burstiness of LP traffic. Lastly, these algorithms normally experience poor bandwidth utilization for their IPS or roll-call polling mechanism.

2.3.2.2 Algorithms of S-opportunity per-class-per-cycle solution

This class of algorithms divides a frame into multiple parts each for one or two traffic class(es). By separating transmission of different traffic classes, this solution provides improved delay and delay variation performance for HP traffic. We mark this scheduling class as S-opportunity per-class-per-cycle solution.

In [AHK03], a hybrid slot-size/rate (HSSR) protocol is proposed which separates a frame into two parts, the initial part reserved for HP traffic including N fixed

timeslots each for an ONU and the latter used for LP traffic consisting of variable timeslots. This algorithm delivers improved performance in terms of delay and jitter to HP traffic. However, the fixed bandwidth cannot always satisfy instantaneous demands. Furthermore, it provides best-effort service to LP traffic. In real network scenario, a more efficient and fair algorithm is desired, such as delivering a guaranteed minimum bandwidth to LP traffic for an ONU.

To maximize priority support while delivering fairness among different classes, a two-layer bandwidth allocation (TLBA) scheme is proposed in [XJJ04], which firstly proportionally allocates bandwidth for different classes according to their demands and weights, then distributes the bandwidth allocated to a class among all requesting ONUs. In this algorithm, the weights of different classes are set empirically. An improved approach in [JX06] dynamically calculates the weights for different classes regarding to access delay bounds and instant traffic loads to minimize delay bound violation. These algorithms achieve efficient priority support by always scheduling HP traffic prior to lower ones. The major weakness lies in poor link utilization brought by RTT idling and $M \times N$ guard intervals, where M and N are the number of traffic classes (typically 3) and active ONUs, respectively. For the latter one, an extra scheme is also required to measure instant traffic loads.

In [BSG05,SBA05], a hybrid granting protocol (HGP) is proposed, which features a novel grant-before-report (GBR) service for EF traffic. This is achieved by properly dividing a frame into an EF sub-cycle and an AF sub-cycle and making allocation decision at the start of the AF sub-cycle in order to accurately predict EF arrivals by the next EF transmission start time for each ONU and accordingly grant bandwidth in

advance. Nonetheless, this is fine only for constant rate EF traffic but not so for dynamic rate. Besides, it has RTT idling problem and consumes $2N$ guard time in each cycle, resulting lower link utilization.

In summary, DBA algorithms of this solution produce improved performance for HP traffic by dividing a transmission frame into multiple service class parts and scheduling HP traffic in the earliest one. The major weakness of this solution is that it usually achieves inefficient bandwidth utilization due to either fixed frame strategy or RTT idling and guard time consumption.

2.3.3 User-oriented Hierarchical Scheduling

In practical networks, an ONU normally serves multiple end users and each user generates heterogeneous QoS-demanding traffic. Hence, a scheduling system in EPON is responsible for QoS delivery not only to aggregate traffic (i.e., traffic classes), but also to handle individual traffic of end users. Particularly, we group DBA algorithms intending to provide QoS to end users as user-oriented hierarchical scheduling class.

In [KBS04], a hierarchical scheduler called fair queuing with service envelopes (FQSE) was developed which delivered a unique feature of cousin fairness service to end-users. Specifically, for any queues i and j with unfinished work, the bandwidth w is allocated in proportion to their weights ϕ , i.e., $w_i / \phi_i = w_j / \phi_j$. The root scheduler at the OLT determines a satisfiability parameter (SP) for each ONU based on their service envelopes (SEs), each of which is the sum of SEs of their leaf queues. The SE of a leaf is a function of SP and SP measures how much the bandwidth

demand is satisfied. Upon receiving the grant message, an ONU further calculates the SP for each queue which maps to allocated bandwidth through $SE-SP$ curve of the queue. This algorithm provides excellent performance in fairness to end-users.

In [CCH06], an efficient and fine hierarchical scheduling is proposed, where each REPORT carries two requests: maximal request B_i^{Mreq} and minimal request B_i^{mreq} ($\leq B_i^{\min}$, the minimum guaranteed bandwidth), and the grant mechanism is defined in

a recursive formula as $G_i = \begin{cases} B_i^{Mreq} & i \in P, \text{ such that } B_i^{Mreq} \leq B_i^{mreq} + B_i^{ex} \\ B_i^{mreq} + B_i^{ex} & i \in Q, \text{ such that } B_i^{Mreq} > B_i^{mreq} + B_i^{ex} \end{cases}$ where

$$B_i^{ex} = \left(B^{total} - \sum_{n \in P} B_n^{Mreq} - \sum_{m \in Q} B_m^{mreq} \right) \frac{\phi_i}{\sum_{m \in Q} \phi_m} . \quad \text{This algorithm guarantees minimal}$$

bandwidth to an ONU, while fairly distributing excess bandwidth among busy ONUs according to their weights. The intra-ONU scheduling combines the M-TB algorithm [CCH05] for inter-class scheduling and M-SFQ algorithm [GSA04] for intra-class scheduling. M-TB guarantees minimal bandwidth to each class meanwhile allocating the remnant bandwidth in priority order. M-SFQ guarantees fine fairness among different queues in each class. Unlike FQSE, this algorithm provides neither cousin fairness nor sliding fairness to end-user; rather, it guarantees fairness among different ONUs and different queues in each class, and provides priority and fairness to differentiated services.

A per-subscription-rate-queuing with round-robin scheduling algorithm was proposed in [ABH03], which guaranteed fairness among users of the same ONU by allocating excess bandwidth according to their subscription rate. The HP traffic of each user is firstly directed into a threshold predetermined FIFO queue before

injecting into a common HP queue. This guarantees the instantaneous bandwidth limited by subscription rate. Meanwhile, there are a number of lower priority queues, each for a subscription rate, and a round-robin scheduling is used to ensure fairness among users of the same subscription rate at an ONU.

Summarily, the user-oriented hierarchical scheduling provides efficient and fine QoS to individual users of an ONU beyond that aggregated QoS to aggregated traffic of the same traffic class. However, the algorithms in [KBS04, CCH06] use per-class-per-user queuing mechanism that brings in large scheduling overhead and queuing complexity, while the algorithm in [ABH03] does not clearly explain how round-robin scheduler works to ensure fairness among users since the LP traffic is bursty in nature. On the other hand, all of the algorithms use S-opportunity per-ONU-per-cycle method for inter-ONU scheduling that limits the performance of HP traffic, and suffers RTT idling problem.

2.4 Summary

In this chapter, we have presented a comprehensive review on scheduling algorithms. The earliest work was motivated to address end-to-end performance guarantees in the context of IntServ networks and has generated many well-known flow-based scheduling algorithms including timestamp-based GPS approximations, frame-based RR scheduling family, and non-work conserving scheduling disciplines. Then, we reviewed the concept of scheduling class which serves the objective to analyze end-to-end QoS in a heterogeneous scheduling network. In particular, we surveyed GR scheduling class and its important PSRG subclass which are closely related to our work. On the other hand, we presented the fundamental work of network calculus, a

number of mathematical results developed for analyzing deterministic performance bounds in a lossless scheduling environment.

Subsequently, we focused our review on aggregate scheduling and DBA protocols which represent the current topics of interest. We classify aggregate scheduling into two basic categories with regard to different traffic aggregation approaches. One gives FIFO aggregation based solution which has two subclasses: *RI* approach and network utilization based approach according to different techniques to achieve end-to-end QoS. On the other hand, we define three classes when reviewing the DBA protocols. The first class includes algorithms that deal with efficient and fair bandwidth allocation among ONUs through statistical multiplexing and intelligent sharing approaches. The second class covers algorithms targeting to provide differentiated services support for QoS assurance to ONUs. With respect to different traffic transmission mechanisms in a cycle, this class is further divided into two subclasses: one featuring S-opportunity per-ONU-per-cycle and the other featuring S-opportunity per-class-per-cycle. The third class comprises of algorithms that are aiming to provide finer bandwidth allocation to end-users, one or multiple of which are hosted by a single ONU. We group these algorithms as user (queue) oriented hierarchical scheduling. To the best of our knowledge, the survey is one of the few works that have reviewed aggregate scheduling and DBA protocols extensively in such breaths and depths. Another notable survey on DBA can be found in [MMR04].

Chapter 3

Deterministic End-to-End QoS: A Latency-Rate

Max-Min Service Curve Approach

3.1 Introduction

Aggregate scheduling arises naturally with the development of traffic aggregation based networks, such as DiffServ and MPLS networks. Recently, a concept of route interference (*RI*) is developed for analyzing deterministic end-to-end behavior of a connection-oriented networks using FIFO aggregation approach [CFZ98]. *RI* is a route system parameter that describes the number of interfering (joining) flows of a specific flow along its path through a network. A source rate condition is derived using the *RI* that specifies the minimum inter-packet space. By controlling traffic flows to obey their source rate condition at the network boundary, end-to-end queuing delay and all queue sizes in the network are proved to be deterministically bounded. This *RI*-based aggregate scheduling achieves scalability and simplicity by using FIFO aggregation. Moreover, it does not sacrifice network utilization in achieving bounded end-to-end delay, unlike other FIFO-based aggregate scheduling in [CB00,Jia02].

3.1.1 Motivations

As discussed in Chapter 2, previous works [CFZ98,OB03] on RI-based aggregate scheduling tend to confine to a simple FIFO scheduling or strict priority scheduling and study the traffic in a single-class environment or the highest priority traffic in a multi-class environment. This is rather restrictive. First, it is quite likely that more advanced scheduling disciplines will be deployed in future networks, such as GPS and RR scheduling family. Therefore, it is necessary to extend the results to a general scheduling environment covering many scheduling disciplines. Second, a simple network model has been assumed. There is either single-class traffic with a single buffer accommodating all packets, or multi-class traffic with packets of each class queuing in a single buffer and a priority scheduling serving the class-based queues. This restricts the results applicable to complex networks, where packets belonging to the same class queue in a number of different buffers. For example, in MPLS networks, an outgoing link could be divided into several logical Label Switched Paths (LSPs), each of which could carry one or several traffic class(es) and be allocated certain bandwidth for use. In this case, it is not suitable to use a simple priority scheduling, because more than one buffer may accommodate the same class traffic and hence have the same priority. In addition, the results that have been obtained for traffic with the highest priority are not applicable to traffic with low priority. These two main deficiencies motivate our study.

In this chapter, we propose a service scheme of latency-rate max-min service curve (LRMMSC), which could be implemented by many well-known scheduling disciplines. Then, we build a general network model with no constraints on whether packets of the same class queue in one or more buffer(s) and how they queue in the

buffer. The only requirement is that each buffer receives a service of LRMMSC. Subsequently, we analyze the deterministic performance on end-to-end queuing delay and buffer size needed for loss-free packet delivery.

The rest of the chapter is organized as follows. In Section 3.2, we introduce the network model and service scheme of LRMMSC. In Section 3.3, we present analytical analysis on end-to-end queuing delay and buffer size. In Section 3.4, we carry out simulation studies. Finally, in Section 5, we give the conclusion.

3.2 Network Model and Latency-Rate Max-Min Service Curve

In this section, we introduce the network model and present the service scheme of LRMMSC. To make the results applicable to many networks, a general approach is needed. Thus, our network model carries the following assumptions.

- Consider a connection-oriented network of arbitrary topology, heterogeneous link capacities, multi-class traffic and variable packet sizes. All packets of a flow traverse the network along a pre-established route, e.g., along a label switched path in MPLS networks. Flows of each traffic class may queue in one or several different buffer(s), each of which receives a service of LRMMSC. That is, packets of flows in the same buffer are served in FIFO manner, while packets in different buffers are competing for the shared link capacity using a number of different disciplines. Note that there is no dedicated bandwidth reservation for any flow, rather, link capacity is shared among all packets.
- Time is assumed to be continuous. Let $C_f(s, t)$ ($C'_f(s, t)$) be the cumulative function that is defined as the number of bits seen on the flow f at the input (output) of a node in the

time interval $[s, t]$. Here, we take the convention that a packet has been received (transmitted) if and only if the last bit of it has been received (transmitted).

- Each link is unidirectional. We say that flow e joins flow m at node l if the two flows arrive from different incoming links and queue at the same buffer of an outgoing link at node l . An “interference unit” is defined as a set $(l, \{e, m\})$ where l is a link, $\{e, m\}$ is a set of two distinct flows joining at node l . We call the number of interference units that contain flow e the *Route Interference (RI)* of e .

Next, we introduce the service scheme of LRMMSC. LRMMSC makes sense in guaranteeing the minimum service and constraining the maximum service for a flow (possibly a buffer, i.e., all the flows queuing in the buffer) at a server. To define LRMMSC, we first introduce two service concepts of Latency-Rate Minimum Service Curve (LRMiSC) and Latency-Rate Maximum Service Curve (LRMaSC), which represent the minimum service and maximum service, respectively. Last, we give the definition of LRMMSC by combining LRMiSC and LRMaSC.

Definition 3.1 [LRMiSC]: A server s is said to provide a service of LRMiSC with rate r_s^f and latency e_s , represented by $\beta_{r_s^f, e_s}$, to a flow f , if for any departure time $D^s(p_i)$, and for any backlogged interval $[t, D^s(p_i)]$ of the flow,

$$C_f'(t, D^s(p_i)) \geq r_s^f (D^s(p_i) - t - e_s)^+ \quad (3.1)$$

where $(.)^+ = \max\{., 0\}$, $D^s(p_i)$ is the departure time of packet p_i of flow f from s .

LRMiSC defines the minimum service that a flow can receive during any of its backlogged (busy) interval. There are many scheduling disciplines that provide such service. The scheduling class of packet scale rate guarantee (PSRG) is an important

one. PSRG is defined as the service for expedited forwarding (EF) traffic in the context of DiffServ network, it covers many well known scheduling disciplines [Jia06], such as scheduling class of GPS, Round Robin and its varieties. From the important characterization of PSRG presented in Theorem II.1 [BC03], we can easily prove that if a server provides PSRG with rate r and latency e to a flow, it also provides the flow a service of LRMISC with rate r and latency e as well.

Definition 3.2 [LRMaSC]: A server is said to provide a service of LRMaSC with rate R_s^f and latency τ_s , represented by $\gamma_{R_s^f, \tau_s}$, to a flow, if for all time t and $0 \leq t' \leq t$,

$$C_f'(t', t) \leq R_s^f (t - t' + \tau_s) \quad (3.2)$$

LRMaSC, on the other hand, defines the maximum service that a flow can receive in any time interval. LRMaSC can be implemented through a token bucket controller or generic cell rate algorithm (GCRA) as shown in Boudec [BT03].

Now, we combine the two service concepts to form the LRMMSC.

Definition 3.3 [LRMMSC]: A server s is said to provide a service of LRMMSC to a flow f , if it offers the flow a service of LRMISC $\beta_{r_s^f, e_s}$ as well as a service of LRMaSC $\gamma_{R_s^f, \tau_s}$, represented by $W(r_s^f, e_s, R_s^f, \tau_s)$.

LRMMSC defines both the minimum service and maximum service that a flow (possibly a buffer) can receive in a busy time interval. The significance of using LRMMSC lies in the wide range of implementations as indicated above. This makes the results in this chapter applicable to heterogeneous scheduling networks. From the

network providers' point of view, this allows them to flexibly deploy the network to economically meet the customers' needs.

3.3 Performance Analysis on End-to-end Queuing Delay and Buffer Size

In this section, we analyze the deterministic end-to-end queuing delay and buffer size needed for loss-free packet delivery in a network as specified in Section 3.2. We prove that if traffic flows obey a strict source rate condition when entering the network, the end-to-end queuing delay and buffer size are bounded. Furthermore, a simple yet useful corollary can be deduced from the main theorem, which provides a concise expression with little requirement of information input. More precisely, only the number of flows sharing the buffer with the considered flow and the LRMISC service of the buffer are needed to obtain bounds on buffer length and end-to-end queuing delay.

Now, we give the source rate condition for a flow f . In doing so, we denote the LRMISC service received by the buffer containing f at a node l_j as $W(r_j^{B(f)}, e_j, R_j^{B(f)}, \tau_j)$. Here, $B(f)$ denotes the buffer containing flow f .

Definition 3.4 [source rate condition]: The source rate condition is satisfied by a flow f if the inter-packet departure time is at least:

$$\frac{(RI_{source,dest} + 1) \times l_{\max}}{r_{source,dest}^{B(f)}} + l_{\max} \sum_{j=source}^{dest} S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ + \sum_{j=source}^{dest} (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}})$$

where $RI_{source,dest}$, route interference of flow f over the global network; $r_{source,dest}^{B(f)}$, the minimum rate for the buffer containing f along the path of f , i.e., $\min_j r_j^{B(f)}$; S_j ,

number of flows sharing the same buffer with f at node l_{j-1} and l_j ; $l_{\max}$, maximum packet length among all packets in the network.

The first term is related to the RI of f . The second term is related to the number of flows sharing the same buffer with f at each node. The last term gives the latency sum of the service curves.

Comment: The source rate condition controls interpacket space of a flow arriving at the network so that at most one packet of each flow exists during any chained buffer busy period, i.e., super-chain. To determine the source rate condition of a flow, the key is to get the RI of the flow. For per-flow based networks without flow aggregation, it is difficult to get the accurate number of interfering flows on a main flow because there are thousands of flows interfering with the flow along its path and many of the flows pose short-time living period. This makes the route interference of the flow vary greatly in short-time-scale. Therefore, from the engineering point of view, the RI-based approach is more applicable for connection-oriented aggregation networks, in particular MPLS networks, where it is relatively easier to get the information of interfering flows as the number of flows is significantly reduced in the network through flow aggregation. In addition, the aggregate flows pose long-time living period. Even though the constituent flows of an aggregate flow vary in short-time-scale, the aggregate flow itself will exist in a long-time period. Thus, the RI of an aggregate flow will remain stable in a long-time scale. To alleviate the cost of control message reporting RI change triggered by flow creation and cessation, an adaptive mechanism needs to be designed to report the changes.

We give an example to illustrate the implementation of RI approach. Let N be the number of edge nodes of a network. Flows belonging to the same class M and heading to the same egress node are aggregated into one single aggregate flow at the ingress node. Then, there are at most $N - 1$ flows of class M starting from an ingress node and total $N \times (N - 1)$ flows of class M in the network. For instance, if there are 30 edge nodes, then there are at most 870 flows of each class in the network which is a small value. A core router maintains information of flow interfering and flow delay requirements of the network. Each ingress node only needs to know the source rate condition of flows starting from it. When an end-to-end flow path needs to be established, the ingress node sends the path-establishment requirement with certain delay bound to the core router. The core router selects the path and determines the queuing buffer for the flow at each node according to some principle, e.g., joining the newly added flow into an exiting buffer as possible provided that it is not hurting the end-to-end delay bound requirement of existing flows. Meanwhile, the core router updates the source rate condition of each affected flow. On the other hand, when a flow path is deleted, the core router will inform the related ingress node to change source rate condition of its flow(s) such that the flow can increase its rate. In addition, admission control can be implemented to determine whether a new flow can join an aggregate flow by checking its source rate condition at the ingress node. The end-to-end queuing delay bound of a flow is dependent on the number of interfering flows and has no relationship with the rate of the interfering flows. Thus, a flow only makes ensure to follow its own source rate condition and the changing of its rate has no affect on other flows. Thus, the admission control can be implemented at the ingress node. The detailed implementation of admission control and route interfering

information gathering is beyond the scope of this thesis. Here, we focus on the derivation of end-to-end delay bound and related buffer size under the route interference.

Next, we introduce the definition of super-chain which is crucial for deriving the results in this chapter. We first give the following relation.

Definition 3.5 [delay-chain]: For two packets p_m and p_n at node l , we say $p_m \prec_l p_n$, if they are in the same buffer busy period and p_m leaves l before p_n .

Note that there is no constraint that p_m and p_n must belong to the same flow. The super-chain is a sequence of packets, such that any pair of consecutive packets constitutes a delay-chain at some link along a flow path. The exact definition is as follows.

Definition 3.6 [super-chain]: Consider a sequence of packets $\underline{p} = (p_0, p_1, \dots, p_k)$ and a sequence of nodes $\underline{l} = (l_1, \dots, l_k)$. We say that $(\underline{p}, \underline{l})$ is a super-chain if

- $p_i \prec_{l_{i+1}} p_{i+1}$ for all $i = 0$ to $k-1$.
- $l_1, \dots, l_k$ are all on the path P of packet p_0 (but not necessarily consecutive).
- The path of packet p_i from l_i to l_{i+1} is a sub-path of P .

We say that the sub-path of p_0 that spans from node l_1 to node l_k is the path of the super-chain.

Super-chain expresses the chained buffer busy period at successive nodes along a flow path. The following definition describes other packets interfering with the

packets of the super-chain. This is crucial for analyzing queuing delay of the super-chain, which in turn facilitates the analysis of end-to-end queuing delay.

Definition 3.7 [segment interfering with super-chain]. For a given super-chain, we call ‘segment’ a couple (p', P') where P' is the sub-path of both the super-chain and the path of the packet p' , and P' is maximal. If there exists a node l_i on P' , such that $p' \prec_{l_i} p_i$, we say this segment (p', P') is interfering with the super-chain.

In the following, we investigate the queuing delay through a super-chain, and develop an important property of super-chain with regard to the source rate condition as shown above. In the end, we present the main results of end-to-end queuing delay and buffer size needed for loss-free packet delivery.

3.3.1 Queuing Delay through a Super-chain

Some symbols and notations are defined for use throughout this chapter. $A^j(p_i)$ denotes the arrival time of packet p_i at node l_j and, respectively, $D^j(p_i)$ denotes the corresponding departure time. Assume X and Y are a set of packets. $X \setminus Y$ denotes the set of packets in X , but not in Y , $X \cup Y$ denotes the set of packets in X and / or Y . $L(X)$ denotes the number of bits of packets in X . In addition, we assume the considered flow is f and $W(r_j^{B(f)}, e_j, R_j^{B(f)}, \tau_j)$ is the LRMMSC service for the buffer containing f at node l_j . Unless otherwise stated, when referring to “flows”, “packets”, “interfering segments”, “traffic” we implicitly refer to the traffic of the class of flow f . Also, “super-chain” and “buffer” indicate the super-chain of f and the buffer containing f , i.e., $B(f)$, respectively.

We consider a network as described in Section 3.2. Let $(\underline{p}, \underline{l})$ be a super-chain. $S_{l,k}$ is the number of segments interfering with $(\underline{p}, \underline{l})$ and $T_{l,k}$ the total transmission delay and propagation time on the path of the super-chain. S_j is the number of flows sharing the same buffer with flow f at both node l_{j-1} and l_j . p_0 is the packet of f .

Consider some node l_j on the super-chain. Let v_j be the beginning of the buffer busy period that packet p_j is in. Define U_j as the set of interference segments (p', P') where p' is a packet arriving at the node no earlier than time v_j over a link other than the input link on the super-chain path and departing no later than packet p_j , and P' is the maximal common sub-path for p' and the super-chain that l_j is in. Correspondingly, define U_j^0 as a set of interference segments similar with U_j , however, with packets arriving at the node no earlier than v_j over a link on the super-chain path. Further, define V_j^0 as the subset of U_j^0 that contains packets departing no earlier than packet p_{j-1} . Note $p_j \notin U_j \cup U_j^0$ and $p_{j-1} \in V_j^0$ (the incoming link of p_{j-1} is necessarily on the super-chain path by the definition of the super-chain).

Since at each node, the buffer containing f receives an LRMaSC service of $\gamma_{R_j^{B(f)}, \tau_j}$, we have:

$$L(U_j^0 \setminus V_j^0) \leq R_{j-1}^{B(f)}(A^j(p_{j-1}) - v_j + \tau_{j-1})$$

In addition, at each node, the buffer containing f receives a LRMiSC service of $\beta_{r_j^{B(f)}, e_j}$, we have:

$$L(U_j \cup U_j^0) + L(p_j) \geq r_j^{B(f)}(D^j(p_j) - v_j - e_j)$$

where $L(p_j)$ is the length of packet p_j .

Combining the above two inequalities, we get:

$$\begin{aligned} D^j(p_j) - A^j(p_{j-1}) &\leq \frac{L(U_j \cup U_j^0) + L(p_j)}{r_j^{B(f)}} - \frac{L(U_j^0 \setminus V_j^0)}{R_{j-1}^{B(f)}} + e_j + \tau_{j-1} \\ &\leq \frac{L(U_j \cup V_j^0)}{r_j^{B(f)}} + L(U_j^0 \setminus V_j^0) \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right) + (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) \end{aligned}$$

By varying j from 1 to k , we get:

$$\begin{aligned} D^k(p_k) - A^1(p_0) &\leq \sum_{j=1}^k \frac{L(U_j \cup V_j^0)}{r_j^{B(f)}} + \sum_{j=1}^k L(U_j^0 \setminus V_j^0) \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right) \\ &\quad + \sum_{j=1}^k (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) + T_{1,k} \end{aligned}$$

From Lemma 6.4.2 of [BT03] we know that the elements in collection (U_j, V_j^0) are two-by-two disjoint, so we get $\sum_{j=1}^k L(U_j \cup V_j^0) \leq S_{1,k} \times l_{\max}$. Furthermore, as V_j^0 contains at least one packet p_{j-1} , so $L(U_j^0 \setminus V_j^0) \leq S_j \times l_{\max}$. In addition, we define

$r_{1,k}^{B(f)} = \min_{1 \leq j \leq k} r_j^{B(f)}$, then the above inequality becomes:

$$D^k(p_k) - A^1(p_0) \leq \frac{S_{1,k} \times l_{\max}}{r_{1,k}^{B(f)}} + l_{\max} \sum_{j=1}^k S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ + \sum_{j=1}^k (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) + T_{1,k} \quad (3.3)$$

This is the queuing delay along a super-chain path of $(\underline{p}, \underline{l})$. The first term is related to the number of segments interfering with the super-chain. The second term is related to the number of flows arriving at the incoming link on the path of the super-chain and sharing the buffer with the flow f at each node. The last term gives the latency sum of LRMMSC service along the path.

3.3.2 Property of Super-chain with respect to source rate condition

Now, we present an important property of super-chain with respect to source rate condition.

Proposition 3.1. If the source rate condition is satisfied by all flows, let $(\underline{p}, \underline{l})$ be a super-chain.

- For every interference unit of packet p_0 , there is at most one packet interfering with the super-chain.
- p_k does not belong to the same flow as p_0 .

Proof. We sequentially number the events of all packets receptions and transmissions by time order as $0, 1, \dots, m, \dots$ over the network, 0 means the network is in idle state. This means each packet reception or transmission has a sole time number. Define the time of a super-chain as the exit time of the last packet p_k on the last link l_k . We use a recursion on the time t of the super-chain.

If $t = 1$, the proposition is true, because any flow has at most one packet on a link up to time t . Assume now that the proposition is true for any super-chain, with time $\leq t$ and consider a super-chain with time $t' \leq t + 1$.

Without loss of generality, we assume there are two interference segments (p', P') and (p'', P'') associated with the same interference unit $(l_e, \{f', f\})$, where “associated” means that packet p' (p'') belonging to flow f' and sub-path P' (P'') containing node l_e . f is the flow of packet p_0 . Assume packet p' is transmitted before p'' . Let l_i be a node on P' (clearly on the super-chain too) such that $p' \prec_{l_i} p_i$.

If p'' leaves node l_i before p_i , then $p' \prec_{l_i} p''$, thus $\{(p', p''), (l_i)\}$ is a super-chain.

Since p'' is an interfering packet, it must leave node l_i before time t as the departure time of packet p_i is no later than t . So the proposition is true for super-chain $\{(p', p''), (l_i)\}$, which contradicts item 2, thus p'' should leave node l_i after p_i . As p'' is an interfering packet, there must be other node $l_j (j \leq k)$ such that $p'' \prec_{l_j} p_j$. p'' leaves node l_i after p_i , and leaves node l_j before p_j , then there must be some node $n (i < n \leq j)$, such that p'' leaves node l_n after packet p_{n-1} before packet p_n by the definition of super-chain. Then $\{(p', p_i, \dots, p_{n-1}, p''), (l_i, \dots, l_n)\}$ is a super-chain with time $\leq t$ which would again contradict item 2. Thus in all cases we have a contradiction. So item 1 is true for super-chain.

We then derive the end-to-end queuing delay for packet p_0 . Call s_0 the emission time of p_0 from the source. P_0 is the sub-path of p_0 from its source to, but excluding node l_1 , and T the transmission and propagation time for the flow of p_0 over the network. The transmission and propagation time along P_0 is thus $T - T_{1,k}$. By the same reasoning as in the above paragraphs, there is at most one packet p per interference unit of p_0 such that $p \prec_{l_e} p_0$ at node l_e along path P_0 . Define $RI_{0,1}$ as the number of interference units of the flow of p_0 on P_0 , then we can get similarly as Eq. (3.3):

$$\begin{aligned}
 A^1(p_0) \leq & s_0 + \frac{RI_{0,1} \times l_{\max}}{r_{0,1}^{B(f)}} + l_{\max} \sum_{j=0}^{\text{Prev}(l_1)} S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ \\
 & + \sum_{j=0}^{\text{Prev}(l_1)} \left(e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}} \right) + T - T_{1,k}
 \end{aligned} \tag{3.4}$$

Along the super-chain path, with Eq. (3.3) and $S_{1,k} \leq RI_{1,k}$, this is obviously true for there is only one packet per-flow interfering with the super-chain, so the number of interfering segments is less than the interference units. We get:

$$D^k(p_k) - A^1(p_0) \leq \frac{RI_{1,k} \times l_{\max}}{r_{1,k}^{B(f)}} + l_{\max} \sum_{j=1}^k S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ + \sum_{j=1}^k (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) + T_{1,k} \quad (3.5)$$

Combining Eq. (3.4) and (3.5), we get:

$$D^k(p_k) \leq s_0 + \frac{RI_{0,k} \times l_{\max}}{r_{0,k}^{B(f)}} + l_{\max} \sum_{j=0}^k S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ + \sum_{j=0}^k (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) + T \quad (3.6)$$

Assuming the packet p_k and p_0 belong to the same flow, by the source rate condition, its emission time s_k from the source must satisfy:

$$s_k \geq s_0 + \frac{(RI_{0,k} + 1) \times l_{\max}}{r_{0,k}^{B(f)}} + l_{\max} \sum_{j=0}^k S_j \left(\frac{1}{r_j^{B(f)}} - \frac{1}{R_{j-1}^{B(f)}} \right)^+ + \sum_{j=0}^k (e_j + \tau_{j-1} + \frac{l_{\max}}{r_j^{B(f)}}) + T$$

which contradicts inequality (3.6) since $D^k(p_k) \geq s_k + T$. This shows that the item 2 must hold for the super-chain.

3.3.3 Bounds on end-to-end queuing delay and buffer-size

In this subsection, we conduct performance analysis on end-to-end queuing and buffer size. The main theorem is given as follows.

Theorem 3.1. For a network as described in Section 3.2, if all traffic flows obey the source rate condition, then

- The buffer size required for loss-free packet delivery at any node and the end-to-end queuing delay are bounded.

- There is at most one packet per-flow present during any buffer busy period.

Proof. Item 2 follows from Proposition 3.1. Otherwise if there are two packets p' and p'' of the same flow in the same buffer busy period at some node l , then $\{(p', p''), (l)\}$ would be a super-chain, which contradicts item 2 of Proposition 3.1.

Now, we show item 1. Consider some node j with I incoming links. Let $E_{j,i}(t)$ be the arrival curve of flows arriving over incoming link i and sharing the same buffer with the considered flow f . As $\beta_{r_j^{B(f)}, e_j}$ is the minimum service curve for the buffer containing f at node j , the backlog $Q_j^{B(f)}$ at the buffer is bounded by:

$$Q_j^{B(f)} \leq \sup_{t \geq 0} \left(\sum_{i=1}^I E_{j,i}(t) - C_{B(f)}'(s, t+s) \right) \leq \sup_{t \geq 0} \left(\sum_{i=1}^I E_{j,i}(t) - r_j^{B(f)}(t - e_j) \right)$$

where s is the beginning of the buffer busy period, $C_{B(f)}'$ is the cumulative bits of packets departed during the period, and $\sup$ indicates the upper limit of the value. The arrival curve of the flows from incoming link i is constrained by the maximum service curve of the flows at the corresponding upstream node. We use $\gamma_{R_{i'}, \tau_{i'}}$ to denote the maximum service curve, of flows arriving over the incoming link i , at the corresponding upstream node, i.e., i' is the upstream node of the incoming link i . Now using item 2 and the maximum service curve property, we have:

$$E_{j,i}(t) \leq E_{j,i}'(t) := \min[N_{j,i} \times l_{\max}, R_{i'}(t + \tau_{i'})]$$

where $N_{j,i}$ is the number of flows arriving at the node j from the incoming link i and sharing the same buffer with flow f . Thus,

$$Q_j^{B(f)} \leq \sup_{t \geq 0} \left(\sum_{i=1}^I E_{j,i}'(t) - r_j^{B(f)} \times t \right) + r_j^{B(f)} \times e_j$$

Let $t_i = \frac{N_{j,i} \times l_{\max}}{R_{i'}} - \tau_{i'}$. We renumber the t_i such that $t_{(1)} \leq t_{(2)} \leq \dots \leq t_{(I)}$. The

function $\sum_{i=1}^I E'_{j,i}(t)$ is continuous, with the linear segment in $t' \in [t_{(i-1)}, t_{(i)}]$. It has a derivative $\underline{R}_{(i)'} = R_{(i)'} + R_{(i+1)'} + \dots + R_{(I)}'$, for all $i = 1$ to I , $R_{(i)'}$ is the corresponding

value to get $t_{(i)}$, i.e., $t_{(i)} = \frac{N_{j,(i)} \times l_{\max}}{R_{(i)'}} - \tau_{(i)'}$. The ordinate at the discontinuity point $t_{(i)}$

is $\underline{R}_{(i+1)'} t_{(i)} + \sum_{m=i+1}^I R_{(m)'} \tau_{(m)'} + l_{\max} \sum_{m=1}^i N_{j,(m)}$, with a maximum value of $l_{\max} \times N_j$ at $t_{(I)}$, where $N_j = \sum_i N_{j,i}$. Note that $\underline{R}_{(I+1)'} = 0$, i.e., the derivative is zero after $t_{(I)}$.

This is illustrated in Figure 3.1.

Figure 3.1: Bounds on backlog and per-node delay.

As the node offers a minimum service curve $\beta_{r_j^{B(f)}, e_j}$ to the buffer containing flow f , the bounds on backlog and delay can be derived by the maximum vertical and horizontal distance between $\sum_{i=1}^I E'_{j,i}(t)$ and $\beta_{r_j^{B(f)}, e_j}$, respectively (refer to [Cru95] or [BT03]). To get a closed form of bounds on backlog and queuing delay, we assume that the node offers a minimum service $\beta_{r_j^{B(f)}, e_j}$ to the buffer $B(f)$, with $r_j^{B(f)} = \min_i (r_j^{B(f)}, R_{i'})$. In this case, the maximum backlog and queuing delay occurs

at $t_{(I)}$, with the value of $N_j l_{\max} - r_{j*}^{B(f)} (t_{(I)} - e_j)^+$ and $\frac{N_j l_{\max}}{r_{j*}^{B(f)}} - t_{(I)} + e_j$, respectively.

From the definition of $t_{(I)}$, it is clear $t_{(I)} \geq \frac{N_j^{j-1} \times l_{\max}}{R_{j-1}^{B(f)}} - \tau_{j-1}$, where N_j^{j-1} is the number

of flows sharing buffer $B(f)$ at both node j and $j-1$. We then finally get the least size of the buffer needed for loss-free delivery as:

$$N_j l_{\max} - r_{j*}^{B(f)} \left(\frac{N_j^{j-1} \times l_{\max}}{R_{j-1}^{B(f)}} - e_j - \tau_{j-1} \right)^+ \quad (3.7)$$

where N_j is the number of flows sharing the buffer $B(f)$ at node j , and the per-node queuing delay as:

$$l_{\max} \left(\frac{N_j}{r_{j*}^{B(f)}} - \frac{N_j^{j-1}}{R_{j-1}^{B(f)}} \right) + e_j + \tau_{j-1} \quad (3.8)$$

The end-to-end queuing delay is the sum of the maximum delay of the flow at each transit node:

$$l_{\max} \sum_{j=source}^{dest} \left(\frac{N_j}{r_{j*}^{B(f)}} - \frac{N_j^{j-1}}{R_{j-1}^{B(f)}} \right) + \sum_{j=source}^{dest} (e_j + \tau_j) \quad (3.9)$$

Theorem 3.1 shows that it is possible to obtain tight bounds on end-to-end queuing delay and buffer size needed for loss-free packet delivery, provided that the source rate condition is satisfied by all flows when entering the network at the ingress node. In addition, if we assume the maximum rate of R in all LRMaSC services is larger than the minimum rate r in all LRMiSC services in a network, the term $r_{j*}^{B(f)}$ then can be replaced by $r_j^{B(f)}$ in Eqs. (3.7)-(3.9). In this case, the calculation for buffer size, per-node delay and end-to-end queuing delay can be much easier as there is no need

to know the information of the flows arriving over each incoming link the LRMaSC service at the corresponding upstream node.

A useful corollary is obtained below:

Corollary 3.1: For a network as specified in Section 3.2, if the service offered to the buffer containing a flow f is governed by LRMiSC of $\beta_{r_j^{B(f)}, e_j}$ alone at each node j along its path, then the buffer size for loss-free packet delivery is bounded by $N_j l_{\max}$

and the end-to-end queuing delay is bounded by $\sum_{j=source}^{dest} \frac{N_j l_{\max}}{r_j^{B(f)}} + \sum_{j=source}^{dest} e_j$ where

N_j is the number of flows sharing the buffer $B(f)$ at link j .

Proof. From item 2 of Theorem 3.1, we know that there is at most one packet per-flow present during any buffer busy period. Obviously, the backlog of the buffer at any node j is bounded by $N_j l_{\max}$. The arrival curve of all flows sharing the buffer $B(f)$ during any buffer busy period is also constrained by $N_j l_{\max}$. Thus, the delay for any packet is bounded by $\sum_{j=source}^{dest} \frac{N_j l_{\max}}{r_j^{B(f)}} + \sum_{j=source}^{dest} e_j$. Note that in this case, the source rate condition is changed accordingly as:

$$\frac{(R_{source,dest} + 1) \times l_{\max}}{r_{source,dest}^{B(f)}} + l_{\max} \sum_{j=source}^{dest} S_j \frac{1}{r_j^{B(f)}} + \sum_{j=source}^{dest} (e_j + \frac{l_{\max}}{r_j^{B(f)}})$$

by setting $R_j^{B(f)}$ to be $+\infty$ in Definition 3.4.

This corollary is an interesting one and is useful in practice. First, the maximum queue length is determined by the number of flows traversing a node, precisely, there is at most one packet of a flow queuing in a buffer at any time. This bound provides

the minimum buffer capacity requirement for loss-free packet delivery in comparison with previous approaches where the buffer size is generally very large due to the burstiness increase of flows in the network such that there are a number of packets existing in a buffer at the moment. On the other hand, to prevent memory wastage by statically locking up much buffer memory for loss-free packet delivery at a node with large number of flows, dynamic buffer tuning mechanisms with soft computing techniques [LWD06] can be used to obtain efficient memory utilization. Second, the end-to-end queuing delay bound depends only on the service offered to the buffers containing the considered flow. In comparison, in Theorem 3.1, we need to know the service offered to the sets of flows arriving over each incoming link and sharing the buffer with the considered flow at the corresponding upstream node. And in [OB03], one needs to know the capacity of each incoming link of each node along the flow path. The simplicity of Corollary 3.1 implies its usefulness as minimum information is needed to make use of the results.

3.4 Evaluation

To examine the tightness of the analytical results on both end-to-end delay and buffer length, we first conduct a simple simulation using the network topology as shown in Figure 3.2 and with parameters given in Table 3-1. The “tagged” flows (i.e., 1 and 2) are the main flows of interest, which traverse the network from ingress node n_1 to egress node n_4 . Each node has two other subsidiary incoming links. Each of the subsidiary links has 1 interfering flow that shares the path with the tagged flows for only one hop. It exits the network at the next hop. One buffer accommodates all flows in FIFO manner at each node. In addition, for simulation purpose, the packets of each

flow are generated so that the first packet starts at a random value of one maximum size packet service time (i.e., $l_{\max}/C$). To simulate different traffic loads, we take three scenarios of inter-packet space for each flow. Scenario 1 (S1) has three independent and equal probability options: source rate condition plus 0, 1, or 2 maximum size packet service time $l_{\max}/C$. That is, the interval of any two successive packets of a flow is source rate condition, source rate condition + $l_{\max}/C$, or source rate condition + $2l_{\max}/C$ by entering the network. Similarly, S2 has 6 options, i.e., source rate condition plus 0, 1, ..., 5 $l_{\max}/C$, while S3 has 12 options. These were chosen to provide the flows with fairly high loads (corresponding to 92.86%, 83.87%, and 70.27% of allowed load of the tagged flows, separately) and with random fluctuations, while the source rate condition is satisfied. Moreover, in this example, $e_j = 0$ and $\tau_j = l_{\max}/C$. The buffer length is the packet bytes in queue.

Figure 3.2: Network Topology.

Table 3-1: Parameters used for analytical calculation and simulation

Packet size (L)	1) Uniformly distributed 64 ~ 800 bytes, 2) fixed 800 bytes
Link Speed (C)	200 Mbit/s
Tagged flows	1 and 2
Interfacing flows	2 flows at each node and each from one joining link
Hop count	4 for the tagged flows, 2 for the interfering flows
Inter-packet space	Three scenarios: S1, S2, and S3.

Table 3-2: Analytical and simulation results for variable packet size scenarios

Tagged flow	Analytical end-to-end delay bound	Simulation maximum end-to-end delay with variable packet size		
(A) Analytical and simulation results on maximum end-to-end delay under different scenarios				
	S1, S2, and S3	S1	S2	S3
1	416 μs	302.1 μs	311.3 μs	286.8 μs
2	416 μs	303.1 μs	296.6 μs	301.2 μs
Node	Analytical buffer length bounds(bytes)	Simulation maximum buffer length with variable packet size (bytes)		
(B) Analytical and simulation results on buffer length under different scenarios				
	S1, S2, and S3	S1	S2	S3
n1	3200	2969	2863	2924
n2	2400	2391	2399	2385
n3	2400	2395	2399	2395
n4	2400	2391	2395	2378

Table 3-3: Analytical and simulation results for fixed packet size scenarios

Tagged flow	Analytical end-to-end delay bound	Simulation maximum end-to-end delay with fixed packet size		
(A) Analytical and simulation results on maximum end-to-end delay				
	s1, s2, and s3	s1	s2	s3
1	416 μs	416 μs	416 μs	416 μs
2	416 μs	416 μs	416 μs	384 μs
Node	Analytical buffer length bounds(bytes)	Simulation maximum buffer length with fixed packet size (bytes)		
(B) Analytical and simulation results on buffer length				
	s1, s2, and s3	s1	s2	s3
n1	3200	3200	3200	3200
n2	2400	2400	2400	2400
n3	2400	2400	2400	2400
n4	2400	2400	2400	2400

The analytical and simulation results on maximum end-to-end delay and buffer length are shown in Table 3-2 and Table 3-3 for variable and fixed packet size

scenarios, respectively. For experiment with variable packet size, all packets of flow 1 and flow 2 experience end-to-end delay less than their analytical upper bounds. More precisely, the simulation maximum end-to-end delay is much lower than the analytical bound, which is due to variable packet size adopted in the simulation such that the probability that a packet experiences maximum end-to-end delay is rather low. For example, the analytical maximum end-to-end delay can only occur in the following situation: when a maximum size packet of flow 1 (or 2) arrives at each node, a maximum size packet of all other joining flows arrives at the same node at the same time, and the packet of flow 1 (or 2) is scheduled the last. Therefore, the probability of getting large end-to-end delay under variable packet size scenario is very low. Table 3-2 (B) exhibits the analytical and simulation results on maximum buffer length. The simulation maximum buffer length at each node is lower than analytical buffer length. The closeness between simulation and analytical results on buffer length is attributed to small number of flows at each node.

For experiment with fixed packet size, all packets of flow 1 and flow 2 experience end-to-end delay no more than their analytical upper bounds. Flow 1 experiences maximum end-to-end delay just the same as its upper bound at all loads while flow 2 experiences maximum end-to-end delay the same as its upper bound at load scenarios S1 and S2, and only one packet service time less than its upper bound at S3. This shows that the analytical results provide reliable upper bounds. Likewise, the simulation maximum buffer length at each node is just the same as the analytical buffer length. The closeness of the two results justifies the analytical approach to provide the upper bound estimates of the problem.

Furthermore, the analytical delay and buffer size are not related to the flow load parameter. With reasonable high load situation, the simulation results are similar under the three different load scenarios for both variable and fixed packet size experiments, as shown in Table 3-2 and Table 3-3.

Next, we demonstrate through another more complex network the advantage of the LRMMSC scheme over previous approaches in providing maximum end-to-end delay. The network topology is shown in Figure 3.3. The parameters used for analytical calculation and simulation are given in Table 3-4. The “tagged” flows are grouped into f_1 and f_2 . All flows in f_1 and f_2 traverse the network from ingress node n_1 to egress node n_8 , i.e., spanning 8 nodes through the network. Each node has four other subsidiary incoming links besides the main link carrying the tagged flows. Each of the subsidiary links has 200 interfering flows that share the path with the tagged flows for only one hop before exiting the network at the next hop. We use two-buffer method for the LRMMSC scheme, i.e., flows of groups f_1 and f_2 are queuing in two different buffers F_1 and F_2 , respectively at each node. Among the 200 interfering flows from each incoming link at each node, 20 of them are sharing the same buffer F_1 with flows of f_1 and the other 180 are queuing in F_2 together with flows of f_2 . All links have the same capacity C (Mbit/s). Start-time fair queuing (SFQ) [GVC96] scheduler is used to serve buffer F_1 and F_2 at each node, with F_1 receiving a LRMMSC service of $W(0.2C, 2l_{\max}/C, C, l_{\max}/C)$ and F_2 receiving a LRMMSC service of $W(0.8C, 2l_{\max}/C, C, l_{\max}/C)$ at each node. For previous approach using the technique as in [OB03], one-buffer method is assumed. In this

case, all flows are queuing together in one buffer at each node. Correspondingly, the analytical end-to-end delay bound is the same for all flows of f_1 and f_2 . Thus, we only need to calculate once for flows of f_1 or f_2 . For simulation purpose, the packets of each flow are generated so that the first packet starts at random time between 0~4.096ms.

Figure 3.3: Network topology: example 2.

Table 3-4: Parameters used for the second example

Packet size	800 Bytes
Tagged flows	200 flows, divided into two groups: f_1 (1~20) and f_2 (21~200)
Incoming link number	5 incoming links each node
Interfering flows	200 flows from each joining link
Hop count	8 for the tagged flows, 3 for the interfering flows
Inter-packet space	Three options: source rate condition plus 0, 1, or 2 packet service time
Number of buffers	2 for the LRMMSC scheme, 1 for the scheme in [OB03]

The analytical end-to-end delay bound is shown in Figure 3.4(a). From the Figure, we see that the delay of $f_1(f_2)$ in [OB03] lies in between f_1 and f_2 of our case with f_1 significantly lower than the delay in [OB03]. This indicates that using our network model, we are able to provide a range of delay bounds with some values much lower than the delay in [OB03] and some other values marginally higher than the delay in [OB03], by deliberately configuring the number of flows at different buffers and the

LRMMSC service for each buffer. The simulation results are shown in Figure 3.4(b). It shows the mean value of maximum end-to-end delay of flows in f_1 and f_2 for our model as well as that for [OB03]. The comparison of simulation results in Figure 3.4(b) with analytical results in Figure 3.4(a) shows a striking resemblance in behavior. The simulation results also provide a range of maximum end-to-end delay with some values much lower than that in [OB03] and some other values marginally higher than that in [OB03]. This confirms the analytical observation.

Figure 3.4: Comparison of end-to-end delay for flows of f_1 and f_2 in our model and $f_1(f_2)$ in Ref. [OB03].

Through the above evaluation, we have shown that LRMMSC approach can provide a range of maximum end-to-end delay by queuing flows in different buffers and allocating different LRMMSC services accordingly. Next, we further study how the performance varies by changing the ratio of the number of flows in f_1 over that in f_2 . The similar network scenario as above is used for evaluation and the link speed is fixed at 200 Mbit/s. Among the 200 tagged flows, the number of flows in f_1 over f_2 varies. Meanwhile, among the 200 flows from each joining link, the same number of flows will join f_1 to share the same buffer F_1 . Figure 3.5 (a) and (b) present the

analytical and simulation results, respectively. Besides the striking resemblance in behavior between the analytical delay bound and simulation mean maximum delay, the results also demonstrate that by deliberately configuring flow buffering and bandwidth allocation the LRMMSC approach can provide significantly lower end-to-end delay to some specific flows while not increasing much the delay performance of other majority flows as compared with single buffer approach of [OB03]. It is noted that the analytical bounds of both f_1 and f_2 of LRMMSC are larger than that of [OB03] when f_1 flow number is 40, i.e., the ratio of flow number in F_1 and F_2 is the same as their ratio of minimum service rate. This is due to smaller minimum service rate for each buffer and thus results in relatively larger delay bound as shown from Eq. (3.9) for multiple buffers approach in this specific case. Nonetheless, the simulation results at this point are nearly the same, which is expected since each flow can be treated to equally share the total bandwidth in both LRMMSC and [OB03] in this scenario although flows are queued in two buffers in LRMMSC.

Figure 3.5: Comparison of end-to-end delay with varying number of flows in f_1 .

Nonetheless, the mean maximum end-to-end delay obtained from the simulation is much lower than that from analytical model. This is because in a complex network,

the probability of reaching the extreme analytical upper bound is quite low for RI-based approaches. Consider a packet P of a flow in f_1 , P obtains the maximum delay at node n_1 only when 1) all other flows of f_1 generate a packet at the same time t as P , 2) all interfering flows from joining links have a packet to generate around the same time and the last packet from each joining link reaches n_1 just at time t , 3) P is the last one to leave n_1 by random selection method since many packets arrive at n_1 at time t and thus transmission order is random, 4) flows of f_2 and their interfering flows should also have packets generating around t such that they will compete for their bandwidth. Combining the above conditions, packet P is nearly impossible to experience the worst delay at n_1 . Considering the global network, it is not difficult to understand that P will achieve much lower end-to-end delay than the analytical bound, although the bound can be reached in strict scenario. Furthermore, as presented in proposition 3.1, in a super-chain, no two packets are from the same flow, thus if treat the buffers of f_1 at each node as a global chained buffer, then there is at most one packet in any buffer busy period. This increases the difficulty for a packet to experience large delay. Therefore, in a RI-based complex network, the practical maximum end-to-end delay will be significantly lower than the analytical bound.

In conclusion, the resemblance of simulation and analytical results verifies the validity of LRMMSC model. Comparing with previous models, LRMMSC model is much more flexible. This allows us to provide better QoS, in particular shorter end-to-end delay, to some specific flows of interest if needed. This feature is not supported in

[OB03] as well as in all other previous approaches where all flows belonging to the same class will receive the same QoS at each node. In addition, the result in [OB03] is not applicable to a network where non-priority scheduling is used. On the contrary, our model is applicable to a variety of heterogeneous networks with various scheduling schemes.

3.5 Conclusion

In this chapter, we have analyzed the end-to-end queuing delay and buffer size needed for loss-free packet delivery using the concept of Route Interference. Our contributions are in two areas. First, we propose a new service scheme called Latency-Rate Max-Min Service Curve (LRMMSC) to provide service. The general nature of our service scheme enables a wide range of implementations covering many known scheduling disciplines and thus makes our results applicable to heterogeneous scheduling networks. Second, we introduce a general network model with no constraints on whether packets of the same class queue in one or more buffer(s) and how they queue in the buffer, except that each buffer receives a service of LRMMSC. These new features make our results applicable to practical and complex networks, such as MPLS networks. This is not possible for previous approaches as they have adopted a simple network model. In addition, we have derived in Corollary 3.1 a concise expression that depends only on the service offered to the buffer containing the considered flow at each node. The simplicity of Corollary 3.1 implies its usefulness as minimum information is needed to make use of the results. This is highly attractive. Simulation experiments are conducted to verify the LRMMSC model. Good general agreements are obtained between analytical and simulation

results. The advantage of LRMMSC scheme in providing maximum end-to-end delay is also demonstrated.

Chapter 4

Deterministic End-to-end QoS: Aggregate

Scheduling with Fair Aggregation

4.1 Introduction

In this chapter, aggregate scheduling with fair traffic aggregation is investigated to provide deterministic end-to-end QoS. As discussed in Chapter 2, aggregate scheduling of FIFO aggregation [CB00,Jia02,BBC02] caused poor network utilization with increased network diameter due to the increase of traffic burstiness. On the other hand, aggregate scheduling of fair aggregation [Cob02,SS05,Jia06,etc] generally could avoid this deficiency. Thus, it is crucial to design of an efficient and fair traffic aggregator in the aggregate scheduling. An aggregator is normally constructed by a flow-based scheduler but is enhanced with functionality of aggregating multiple input flows into a single aggregate flow as shown in Figure 2.4 (the so-called traffic aggregation). Therefore, the design problem of an aggregate scheduling can be divided into three steps: design of an aggregator, design of an aggregate scheduling with the aggregator, and end-to-end QoS analysis of a network of the aggregate scheduling.

In Section 4.2, we develop a guaranteed rate fair aggregator (GRFA) for traffic aggregation and design an aggregate scheduling service model combining guaranteed rate (GR) scheduler and GRFA. We call it (GR with GRFA). Then, we analyze end-to-end QoS for a network of (GR with GRFA) and present the advantage of (GR with GRFA) through analytical comparison with previous approaches.

On the other hand, aggregate scheduling of GRFA normally features a two-level scheduling architecture like most current approaches [Cob02,Jia06,etc]. This increases complexity and implementation cost. In order to alleviate this situation, in Section 4.3, we further develop a flow-aggregation capable guaranteed rate (FA-GR) scheduling and a priority shaper (P-shaper) to provide end-to-end QoS in a much scalable way.

4.2 Aggregate Scheduling based on GRFA

In this section, we design an aggregate scheduling that combines a GR scheduler and a number of GRFAs and analyze the end-to-end QoS performance. The parameters used in the performance analysis in this chapter are listed in Table 4-1.

4.2.1 GRFA aggregator

As discussed in Chapter 2, GR scheduling is a very general scheduling concept that covers many known scheduling disciplines. It is also equivalent with the general service concept of service curve [BT03]. Thus, GR is an ideal candidate in the design of fair aggregator and aggregate scheduling. In this section, we propose a GRFA aggregator which is constructed by GR scheduling. Specially, GRFA is designed with the following characteristics as illustrated in Figure 4.1:

Table 4-1: Parameter used for performance analysis.

f, g	An individual or aggregate flow.
R_f	The offered rate to flow f
$L_{\max}$	The maximum packet length among all network packets
σ_f, ρ_f	The burst size and average arrival rate of flow f , respectively.
p_f^i	The i -th packet of flow f .
l_f^i	The length of packet p_f^i
$a_{f,i}^S, d_{f,i}^S$	The arrival and departure time of packet p_f^i at and from S , respectively. S represents a network block, such as a scheduler.
β^v ,	An implementation-specific error (latency) term of scheduler v .
$GRC_{f,i}^j$	The GRC value of packet p_f^i at GR scheduler j .

- 1) It forwards packets at a rate $R_g = \sum_{i \in F} r_i$, even though its capacity is C ($C \geq R_g$). F is the set of input flows f_i that will be merged into one single flow g by the aggregator.
- 2) The input flow can be a simple individual flow or an aggregate flow from the upstream nodes. The input flows of an aggregator are merged together to form a single output flow to be forwarded to subsequent nodes. The input flows are called immediate constituent flows (constituents for short) of the output flow. Along the subsequent nodes, only the aggregate flow is aware, no constituent is aware until the aggregate flow is separated into its constituents by a separator.
- 3) In particular, we give an example to show how the aggregator works in order to merge flows in MPLS network. In MPLS network, packets of flows are classified and assigned a 32-bit label at the ingress node according to some principles. At each router along the path, the label is used to forward packet to the next hop. Hence, packets with the same label are treated as from the same flow. If packets from different upstream nodes with

different labels are replaced with the same label at a node, then at the subsequent nodes, these packets are treated as from the same flow until they are assigned to different labels at some node. Hence in this case, the assembler uses only the same label to replace the labels of all packets from its input flows to merge flows into a single flow.

Figure 4.1: GRFA and the architecture of the aggregate scheduling.

The design of GRFA is illustrated in Figure 4.1. We assume that the aggregator is composed of two components: a fictitious GR-scheduler v and a simple assembler s . Let f_k represent the input flows of GRFA, and g the output aggregate of it. The assembler s forwards packets at a rate R_g . Its capacity is C_s . The assembler also performs flow aggregation, i.e., it processes packets of each constituent flow based on some special mechanism (e.g., label merging of MPLS). After that all the constituents will be treated as a single flow by the following schedulers and aggregators. The fictitious GR scheduler v has the same input flows as the assembler s . It performs fair scheduling of the constituent flows $f_1, f_2, \dots, f_n$ as a normal GR scheduler as defined by Definition 2.1, while its output channel has capacity R_g . s forwards each packet to its output channel at exactly the same time v will forward the same packet. Thus, as the output channel of v has capacity R_g , s cannot forward packets at a rate higher than R_g , even though its channel has capacity C_s , where $C_s \geq R_g$.

The combination of v and s makes up the GRFA aggregator q . As q forwards packet at the same time as a GR scheduler, q is also a GR scheduler.

4.2.2 Aggregate scheduling of GR with GRFA

Our aggregate scheduling is composed of a GR scheduler and a number of GRFAs as shown in Figure 4.1. Next, we analyze the concatenation property of two consecutive aggregate scheduling schemes in order to derive end-to-end delay bound performance. For this purpose, we first relate GRC values of a $(k-1)$ th packet at t and the arrival time of a k th packet at t in order to relate GRC values at v and t , respectively.

Lemma 4.1: Consider a node model as shown in Figure 4.1, for all $k, k > 1$, $GRC_{g,k-1}^t \leq a_{g,k}^t + l_g^{\max} / C_s - l_g^k / C_s$. l_g^k is the packet length of p_g^k and $a_{g,k}^t$ is the arrival time of p_g^k at GR scheduler t .

Proof. We prove the lemma by induction on k . First, let $k = 1$. It is obviously true, $GRC_{g,0}^t = 0 \leq a_{g,0}^t + l_g^{\max} / C_s - l_g^1 / C_s$. For the induction, we assume it is true for $k-1$, i.e., $GRC_{g,k-2}^t \leq a_{g,k-1}^t + l_g^{\max} / C_s - l_g^{k-1} / C_s$.

We consider k . Packet p_g^{k-1} will be chosen for transmission at the aggregator l_g^{k-1} / C_s seconds before arriving at t , and from the definition of a GRFA aggregator, p_g^k begins transmission at the aggregator at least l_g^{k-1} / R_g seconds after p_g^{k-1} is chosen for transmission. And it arrives at scheduler t after l_g^k / C_s seconds. We get,

$$a_{g,k}^t \geq a_{g,k-1}^t + l_g^{k-1} / R_g - l_g^{k-1} / C_s + l_g^k / C_s$$

As t is GR scheduler, we get $GRC_{g,k-1}^t = \max\{a_{g,k-1}^t, GRC_{g,k-2}^t\} + l_g^{k-1} / R_g$. There are two cases. If $a_{g,k-1}^t \leq GRC_{g,k-2}^t$, we have:

$$\begin{aligned} GRC_{g,k-1}^t &= GRC_{g,k-2}^t + l_g^{k-1} / R_g \leq a_{g,k-1}^t + l_g^{\max} / C_s - l_g^{k-1} / C_s + l_g^{k-1} / R_g \\ &\leq a_{g,k}^t + l_g^{\max} / C_s - l_g^k / C_s \end{aligned}$$

Otherwise, we have:

$$\begin{aligned} GRC_{g,k-1}^t &= a_{g,k-1}^t + l_g^{k-1} / R_g \leq a_{g,k}^t + l_g^{k-1} / C_s - l_g^k / C_s \\ &\leq a_{g,k}^t + l_g^{\max} / C_s - l_g^k / C_s \end{aligned}$$

This is true for all cases. Finally, we get result of Lemma 4.1.

Next, we obtain a relationship between the GRC values at v and t , respectively.

Consider an immediate constituent f of the output flow g and assume $p_g^j = p_f^i$.

From the Definition 2.1 of GR server, we have:

$$GRC_{g,j}^t = \max\{a_{g,j}^t, GRC_{g,j-1}^t\} + l_g^j / R_g$$

Together with Lemma 4.1, we get $GRC_{g,j}^t \leq a_{g,j}^t + l_g^{\max} / C_s + l_g^j / R_g - l_g^j / C_s$.

Since v is a GR scheduler, and the output channel rate of s is C_s , we have:

$$a_{g,j}^t = a_{f,i}^t \leq GRC_{f,i}^v + \beta^v - (l_f^i / R_g - l_f^i / C_s)$$

where β^v is the latency term of v . Since $p_g^j = p_f^i$, we get:

$$GRC_{g,j}^t \leq GRC_{f,i}^v + \beta^v + l_g^{\max} / C_s \quad (4.1)$$

We ignore the propagation delay between the aggregator q and the GR scheduler t . In fact, the aggregator and the GR scheduler are generally at the same router, so the propagation delay is zero.

Next, we derive the GRC relationship between one fictitious GR scheduler v and the next fictitious GR scheduler u . This is given in Theorem 4.1.

Theorem 4.1: Let q be a GRFA aggregator, v the fictitious GR scheduler of q , t a GR scheduler in the same node and u the fictitious GR scheduler of the next node. For each constituent f of its output flow g and for all $i, i \geq 1$,

$$GRC_{g,j}^u \leq GRC_{f,i}^v + \beta^v + \alpha^t + l_g^{\max} / R_g$$

where $\alpha^t = \beta^t + \tau^{t,u}$ and $\tau^{t,u}$ is the propagation delay between t and u ; $p_g^j = p_f^i$.

Proof. As the aggregator q is at the same node as the scheduler t , we get $GRC_{g,j}^t \leq GRC_{f,i}^v + \beta^v$ by letting $C_s \approx \text{infinity}$ in Eq. (4.1).

In addition, since $\alpha_{g,j}^u \leq GRC_{g,j}^t + \alpha^t$, we can obtain the difference of GRC values between the two consecutive schedulers t and u as:

$$GRC_{g,j}^u \leq GRC_{g,j}^t + \alpha^t + \max_{k \in [1..j]} l_g^k / R_g$$

So we obtain:

$$GRC_{g,j}^u \leq GRC_{g,j}^t + \alpha^t + \max_{k \in [1..j]} l_g^k / R_g \leq GRC_{f,i}^v + \beta^v + \alpha^t + l_g^{\max} / R_g$$

4.2.3 End-to-end Delay Bound

Consider a path configuration as shown in Figure 4.2. At each node, flows have the same egress node are aggregated together, and separation only occurs at the last egress node. Each node employs our aggregate scheduling service model. We first relate the GRC values of packet p_f^i at hop h_1 and h_N respectively, and then obtain the end-to-end delay bound.

Figure 4.2: Path configuration and flow aggregation

By induction on Theorem 4.1, we get:

$$\begin{aligned}
 GRC_{f,i}^{v_2} &\leq GRC_{f,i}^{v_1} + \beta^{v_1} + \alpha^{t_1} + \frac{L_{\max}}{R_{g1}} = GRC_{f,i}^{v_1} + \beta^{v_1} + \alpha^{t_1} + \frac{\theta_1 L_{\max}}{R_f} \\
 GRC_{f,i}^{v_3} &\leq GRC_{f,i}^{v_2} + \beta^{v_2} + \alpha^{t_2} + \frac{L_{\max}}{R_{g2}} = GRC_{f,i}^{v_2} + \beta^{v_2} + \alpha^{t_2} + \frac{\theta_2 L_{\max}}{R_{g1}} \\
 &= GRC_{f,i}^{v_2} + \beta^{v_2} + \alpha^{t_2} + \frac{\theta_1 \theta_2 L_{\max}}{R_f} \\
 &\quad \bullet \bullet \bullet \bullet \\
 GRC_{f,i}^{v_{N-1}} &\leq GRC_{f,i}^{v_{N-2}} + \beta^{v_{N-2}} + \alpha^{t_{N-2}} + \frac{L_{\max}}{R_{g(N-2)}} \\
 &= GRC_{f,i}^{v_{N-2}} + \beta^{v_{N-2}} + \alpha^{t_{N-2}} + \frac{\theta_1 \cdots \theta_{N-2} L_{\max}}{R_f} \\
 &\leq GRC_{f,i}^{v_1} + \sum_{i=1}^{N-2} (\beta^{v_i} + \alpha^{t_i}) + \sum_{i=1}^{N-2} \frac{\theta_1 \cdots \theta_i L_{\max}}{R_f}
 \end{aligned}$$

where θ_i is the bandwidth ratio of g_i that contains f among g_{i+1} . As h_N is the last hop, there is no need of aggregation. From Theorem 4.1, we get:

$$\begin{aligned}
 GRC_{f,i}^N &\leq GRC_{f,i}^{v_{N-1}} + \beta^{v_{N-1}} + \alpha^{t_{N-1}} + \frac{L_{\max}}{R_{g(N-1)}} \\
 &\leq GRC_{f,i}^{v_1} + \sum_{i=1}^{N-1} (\beta^{v_i} + \alpha^{t_i}) + \sum_{i=1}^{N-1} \frac{\theta_1 \cdots \theta_i L_{\max}}{R_f}
 \end{aligned} \tag{4.2}$$

Since the last service at node N guarantees the packet p_f^i will be transmitted by

$GRC_{f,i}^N + \beta^N$ and the packet arrives at the first node at time $a_{f,i}^1$, we get:

$$\begin{aligned}
Delay_{end-to-end} &\leq GRC_{f,i}^N - a_{f,i}^1 + \alpha^N \\
&\leq GRC_{f,i}^{v_1} - a_{f,i}^1 + \sum_{i=1}^{N-1} \beta^{v_i} + \sum_{i=1}^N \alpha^{t_i} + \sum_{i=1}^{N-1} \frac{\theta_1 \cdots \theta_i L_{\max}}{R_f}
\end{aligned} \tag{4.3}$$

If we know the traffic constraint function of each flow at the ingress router, then $GRC_{f,i}^{v_1} - a_{f,i}^1$ can be determined. For example, if traffic conforms to (σ, ρ) -constraint of Definition 2.5, we can get: $GRC_{f,i}^1 - a_{f,i}^1 \leq \sigma / \rho$ [GLV95], so:

$$Delay_{end-to-end} \leq \frac{\sigma_f}{\rho_f} + \sum_{i=1}^{N-1} \beta^{v_i} + \sum_{i=1}^N \alpha^{t_i} + \sum_{i=1}^{N-1} \frac{\theta_1 \cdots \theta_i L_{\max}}{R_f} \tag{4.4}$$

4.2.4 Performance Comparison

In order to demonstrate the advantage of (GR with GRFA), we compare the end-to-end delay bound of (GR with GRFA) with the aggregate scheduling service models of (GR with FIFO) [Jia02], (PSRG with FIFO) [BBC02], and (LBFA) [Jia04]. (PSRG with FIFO) and (GR with FIFO) have the same end-to-end delay bound as shown in Theorem 2.5 for the case of lossless system and no knowledge of input capacity. If more information is known such as total input links capacity P_h , burstiness of input flows, etc., the model of (GR with FIFO) can give improved performance. Thus, we consider four cases for comparison: (GR with FIFO with finite P_h), ((PSRG with FIFO) or (GR with FIFO) with $P_h \Rightarrow \infty$), (LBFA) and our model (GR with GRFA).

We shall consider some scenarios and compute the numerical values for comparison. For simplicity, the propagation delay is ignored. The constraint function is (σ, ρ) . Network diameter $H = 10$ and $L_{\max} = 1500$ bytes. All links have capacity of $C = 155$ Mbps. In particular, for [Jia02, Jia04], we assume each hop has 4 incoming links. $P_h = 4C$ and the reserved rate R_h for the considered traffic aggregate is $C/2$

for [Jia02]. For [Jia04] and our model, we assume the considered flow has the same rate ratio θ_i among the corresponding aggregate at each node. We compute the end-to-end delay for three utilization cases, i.e., $\alpha = 0.07$, 0.1 , and 0.12 . Meanwhile $\theta_1 = 0.02$, 0.05 , and 0.10 respectively for [LBFA] and our model to indicate the degree of flow aggregation at the ingress node. The results are shown in Table 4-2 (a)-(c).

Table 4-2: End-to-end delay (ms)

(a): $\alpha = 0.07$, $\rho_f = 32\text{Kbps}$. $\theta_1 = 0.02$, $\theta_2 = \theta_3 = \theta_4 = \theta_5 = 0.4$, $\theta_6 = \theta_7 = \theta_8 = \theta_9 = 1$

	$\sigma_f = 100\text{Bytes}$	200	400	800
GR with FIFO (P_h is finite)	41.67	76.47	146.07	285.28
(PSRG or GR with FIFO)	53.57	100.87	195.47	384.66
LBFA	55.03	80.03	130.03	230.03
GR with GRFA	52.05	77.05	127.05	227.05

(b): $\alpha = 0.1$, $\rho_f = 32\text{Kbps}$. $\theta_1 = 0.05$, $\theta_2 = \theta_3 = \theta_4 = \theta_5 = 0.4$, $\theta_6 = \theta_7 = \theta_8 = \theta_9 = 1$.

	$\sigma_f = 100\text{Bytes}$	200	400	800
GR with FIFO (P_h is finite)	124.33	233.70	452.45	889.95
(PSRG or GR with FIFO)	273.22	532.22	>1000	>1000
LBFA	94.57	119.57	169.57	269.57
GR with GRFA	91.47	116.47	166.47	266.47

(c): $\alpha = 0.12$, $\rho_f = 32\text{Kbps}$. $\theta_1 = 0.1$, $\theta_2 = \theta_3 = \theta_4 = \theta_5 = 0.4$, $\theta_6 = \theta_7 = \theta_8 = \theta_9 = 1$.

	$\sigma_f = 100\text{Bytes}$	200	400	800
GR with FIFO (P_h is finite)	730.48	>1000	>1000	>1000
(PSRG or GR with FIFO)	∞	∞	∞	∞
LBFA	160.92	185.92	235.92	335.92
GR with GRFA	157.17	182.17	232.17	332.17

All the three models are affected by σ_f in the similar manner. Table 4-2 (a) shows that the end-to-end delay bounds for all the four models are comparable for low utilization. However, as α increases from 0.07 to 0.1 as shown in Table 4-2 (b), the

(GR with FIFO) as well as (PSRG with FIFO) deteriorates rapidly and our model provides the lowest end-to-end delay bound. This shows that the models of (GR with FIFO) and (PSRG with FIFO) are rather sensitive to network utilization α and provide good results only at the low utilization. When α increases to 0.12, (PSRG+FIFO) results in infinite end-to-end delay bound indicates the utilization of (PSRG with FIFO) must be less than 0.12 to achieve finite delay bound. As for (GR with FIFO), the utilization constraint is improved owing to extra information of total incoming links capacity introduced to derive end-to-end delay bound. Nonetheless, the delay bound of (GR with FIFO) has significantly deteriorated at load 0.12. Our model is not sensitive to α and is affected mainly by the first term of σ_f / ρ_f . LBFA achieves comparable delay bounds as ours in all cases. However, their delay values are higher, caused by the shaper at the front of LBFA aggregator. In addition, if σ_f / ρ_f is very low, the ratio of bandwidth of each flow affects the end-to-end delay greatly. When θ_1 varies from 0.02 to 0.1, the delay bound increases a lot. This indicates our model can achieve good performance when there are many flows towards the same direction being aggregated together, especially at the ingress node. On the other hand, LBFA and (GR+GRFA) are non-working scheduling system in nature, thus they likely yield higher average delay as compared with (GR+FIFO) and (PSRG+FIFO) which are work-conserving scheduling system. However, in guaranteed performance service, the more important performance index is the end-to-end delay bound rather than the average delay [Zha95]. Since PSRG is a special subclass of GR, (PSRG+FIFO) and (GR+FIFO) will deliver the same actual delay given the same scheduler is used. The difference in Table 4-2 is because

(PSRG+FIFO) in [BBC02] only presents the end-to-end delay bound for a more general network scenario while (GR+FIFO) in [Jia02] improves the delay bound by introducing information of incoming links capacity.

4.2.5 Summary

Our aggregate scheduling service model of (GR with GRFA) is able to provide guaranteed rate service to individual flows in the presence of flow aggregation. The comparison shows that (GR with GRFA) can achieve better performance in end-to-end delay bound, especially at large α scenario. Furthermore, it is capable of utilizing fully the configured rate. The features are contributed by the design of GRFA fair aggregator. While FIFO-aggregation based approaches suffer very lower utilization to obtain reasonable end-to-end delay bound due to huge burstiness increase.

4.3 Aggregate Scheduling of FA-GR and P-shaper

In this section, we continue to develop a flow-aggregation capable guaranteed rate (FA-GR) scheduling and a priority shaper (P-shaper) for aggregate scheduling in order to provide end-to-end QoS in a more scalable way than the approaches of (GR with GRFA) and other two-level aggregate scheduling approaches.

4.3.1 FR-GR Scheduling

FA-GR scheduling consists of three functional components as shown in Figure 4.3. The first part calculates GRC value for each packet as a normal GR scheduler and encodes it into the packet header. The second part is a GR scheduler that transmits packets in the order of their GRC values. The last part performs flow aggregation and/or separation if needed. Note that the order of the three parts is not fixed. For

example, in actual implementation, the last part may be at the first position. We place it at the last position here just to emphasize that packets of a flow from the upstream node are queued in a single buffer and are scheduled as a normal single flow at the GR scheduler of the FA-GR. It does not matter whether the flow is to be aggregated with other flow(s) or be separated into its constituents at the FA-GR. This means the effect of flow aggregation or separation will only be reflected at the next node.

Figure 4.3: The architecture of FA-GR scheduling

The concept of fair aggregator and implementation example have been presented in Chapter 2 and Section 4.2. Here, to make our scheme available to a number of circumstances, we assume the aggregation/separation to be the last component of FA-GR as shown in Figure 4.3 and also, we define a more general concept of aggregator. Specially, an aggregator receives as input a number of flows and produces as output one or several aggregate flow(s) (aggregate for short). The set of flows forming the same output aggregate is said to be the immediate constituents of the output aggregate. The output aggregate is said to be the root of the constituents. The root of a flow may vary throughout the network with the occurrence of flow aggregation and/or separation. Note that the immediate constituent may disappear after flow aggregation. We illustrate this through an example. Let flow f and g be aggregated into aggregate e at node h_1 , flow i and j be aggregated into aggregate k at node h_2 . At node h_3 , flow e and k are further aggregated into flow u . There are four cases. The

first case is that e and k are directly aggregated into u . Thus, the immediate constituents of u after the FA-GR are e and k , which in turn contain f, g and i, j , respectively. The second case is that u directly contains f, g, i and j after the FA-GR. Logically, we can say that e and k are firstly separated into f, g and i, j respectively and then aggregated into u . However, we do not regard this circumstance as an example of flow separation because there is no flow being separated out. For this case, we still treat e and k as the immediate constituents of u at the FA-GR. After FA-GR, the immediate constituents become f, g, i, j . The third and fourth cases are that flow u contains e, i, j and f, g, k , respectively. The difference between the four cases is that after flow aggregation, the information that flows f, g and i, j ever come from different flows may be lost. Whether the information is needed or not is dependent on the real network configuration. In this chapter, we do not make any assumption on this as we only concern with the end-to-end QoS for individual flows in the aggregation environment. Our definition of flow aggregation covers four basic cases and can be easily used to produce more complex forms of “flow aggregation”. Hence, it well meets the needs of practical networks, such as MPLS networks. Such definition of flow aggregation is somewhat different from previous definitions where only the first case is assumed. With only the first case, it is inadequate to describe complex forms of “flow aggregation” and may cause extra delay latency.

4.3.2 P-shaper

In Chapter 2, we have reviewed the concept of shaper and its characteristics. A shaper generally transits packets in a FIFO manner, thus we call this shaper as “FIFO-

shaper". In this chapter, we present a new shaper called "*Priority-shaper*" or "*P-shaper*" for short. P-shaper transmits packets by the order of their priority which is determined by the GRC value of the packet. The packet with smaller GRC value has higher priority. Formally, P-shaper is a greedy shaper that transmits packets as soon as possible by the order of their carried GRC values in the packet header while not violating the shaping curve α . In this chapter, the shaping curve is assumed to be leaky bucket constraint in the form of $(R, L_{\max})$, where R is the rate.

By definition, P-shaper is able to reorder packets by their GRC values. As GR server is non-preemptive, a packet with smaller GRC value cannot preempt a packet in transmitting with higher GRC value. If there is no device to adjust the order after flow aggregation, the burstiness of each constituent will increase much after each flow aggregation and in the end will waste much bandwidth to obtain reasonable end-to-end QoS as in [Jia02, etc]. With P-shaper, the packet with smaller GRC value has a chance to exceed the packet with larger GRC value before getting to the next FA-GR server which delivers packets from the aggregate flow by FIFO manner. This helps refrain the increase of burstiness and improves the end-to-end QoS performance.

4.3.3 Node Model

A node is synonymous with a router in this chapter acting as the building block of a network. For each node, there is a FA-GR scheduler for each outgoing link to perform packet scheduling, encode GRC value into the packet header, and execute flow aggregation and/or separation if needed. Meanwhile, there are a number of P-shapers that control flows to be leaky bucket constraint ahead of the FA-GR scheduler. Note that after the P-shaper, the original GRC value in the packet header becomes useless

and will be replaced by a new GRC value calculated by the FA-GR scheduler at this node. This feature is different from the technique of Dynamic Packet State (DPS) in [SZ99,KV01,SS04] where the new packet state value of a packet is calculated based on the present one carried from the upstream server. GRC value calculation of FA-GR is thus much simpler.

4.3.4 Performance Analysis on a Network of FA-GR and P-shaper

In this section, we analyze end-to-end delay bound for a network of nodes that apply a combination of P-shaper and FA-GR scheduling. We first present an important delay bound property of GR scheduler that is derived in Theorem IV. 1 of [BC03].

Consider a GR node with rate r and latency β . Assume the input is constrained by an arrival curve $\alpha(\cdot)$. The queuing does not need to be FIFO. The delay for any packet is bounded by:

$$\sup_{t>0} \left[\frac{\alpha(t)}{r} - t \right] + \beta \quad (4.5)$$

where $\sup$ indicates the upper limit of the value.

Now, we consider a flow f through the network along a series of nodes $1, \dots, j, \dots, J$. The root of flow f is g_j after the FA-GR server j but before server $j+1$. Note that even there is no flow aggregation or flow separation occurring at node j , we still mark the root of f as g_j after the FA-GR j .

There are two cases for consideration. For the first case, we do not consider the circumstance of flow separation at core routers. The separation only occurs at the egress node, causing no effect on end-to-end delay bound. In this case, only flows

belonging to the same class and destining to the same egress node may be aggregated together at each node. For the second case, we consider a general situation where both flow aggregation and separation may occur at each node.

Now, we study the first case. We obtain the packet delay and buffer size needed at the FA-GR and P-shaper. The delay at the FA-GR (P-shaper) is between the arrival time at the buffer of the FA-GR (P-shaper) and the departure time from the FA-GR (P-shaper). In this chapter, we assume the operation of flow aggregation or separation causes no delay as the operation is generally simple in real implementation.

Theorem 4.2: The delay bound of flow g_{j-1} at FA-GR j and P-shaper j is $\beta_j + L_{\max} / R_{g_{j-1}}$ and $\beta_{j-1} + (2N-1)L_{\max} / R_{g_{j-1}}$, respectively. The queue size for g_{j-1} is bounded by $L_{\max} + R_{g_{j-1}}\beta_j$ and $(2N-1)L_{\max} + R_{g_{j-1}}\beta_{j-1}$, respectively at the FA-GR j and P-shaper j . β_j is the latency term of the FA-GR j , $R_{g_{j-1}}$ is the rate reserved for g_{j-1} , N is the number of immediate constituents of g_{j-1} at node $j-1$.

Proof. From the definition of P-shaper, g_{j-1} is α -constraint in the form of $R_{g_{j-1}}t + L_{\max}$ at the output of the P-shaper.

The FA-GR j offers the flow g_{j-1} a service of GR with rate $R_{g_{j-1}}$ and latency term β_j . By Eq. (4.5), we get the delay of any packet of flow g_{j-1} is bounded by:

$$\sup_{t>0} \left[\frac{R_{g_{j-1}}t + L_{\max}}{R_{g_{j-1}}} - t \right] + \beta_j = \beta_j + \frac{L_{\max}}{R_{g_{j-1}}}$$

On the other hand, by Theorem 2.1, the buffer size at the FA-GR is bounded by:

$$\sup_{t>0} [R_{g_{j-1}} t + L_{\max} - R_{g_{j-1}} \times (t - \beta_j)] = L_{\max} + R_{g_{j-1}} \beta_j$$

Consider an arbitrary constituent f of g_{j-1} at node $j-1$, let $R(\cdot)$ and $R^*(\cdot)$ be its cumulative input and output function, respectively. For any busy period (s, t) , we get:

$$\begin{cases} R^*(t) \leq R(t) \leq R_f t + L_{\max} & (\text{arrival curve constraint}) \\ R^*(s) \geq R_f (s - \beta_{j-1} - \frac{L_{\max}}{R_f}) & (\text{packet delay property}) \end{cases}$$

Thus, we have: $R^*(t) - R^*(s) \leq R_f (t - s) + 2L_{\max} + R_f \beta_{j-1}$.

Accordingly, the root g_{j-1} has arrival curve:

$$\sup_{t>0} (R_{g_{j-1}} t + 2NL_{\max} + R_{g_{j-1}} \beta_{j-1} - R_{g_{j-1}} t - L_{\max}) = (2N-1) \times L_{\max} + R_{g_{j-1}} \beta_{j-1}$$

Finally, the buffer size at P-shaper j for g_{j-1} is bounded by:

$$\sup_{t>0} (R_{g_{j-1}} t + 2NL_{\max} + R_{g_{j-1}} \beta_{j-1} - R_{g_{j-1}} t - L_{\max}) = (2N-1) \times L_{\max} + R_{g_{j-1}} \beta_{j-1}$$

and the maximum delay of g_{j-1} at P-shaper j is bounded by:

$$\frac{(2N-1) \times L_{\max} + R_{g_{j-1}} \beta_{j-1}}{R_{g_{j-1}}} = \beta_{j-1} + \frac{(2N-1) \times L_{\max}}{R_{g_{j-1}}}$$

Comment: Generally, β_j is a small value and is equal to $L_{\max} / C_j$ for the virtual clock and WFQ scheduler, where C_j is the link capacity of the node j . Hence, the delay bound is at most $L_{\max} / C_j + L_{\max} / R_{g_{j-1}}$. The buffer bound for each flow at FA-GR j is less than $2L_{\max}$ as the second term is always less than $L_{\max}$ for every flow ($R_{g_{j-1}} \beta_j = R_{g_{j-1}} \times L_{\max} / C_j < L_{\max}$). Correspondingly, the buffer bound at P-shaper for each flow is bounded by $2NL_{\max}$.

Next, we consider the departure time of a packet from the P-shaper in order to derive the concatenation property of GRC values of a packet at two consecutive FA-GR nodes, which leads to the end-to-end concatenation property from induction.

Lemma 4.2: For an arbitrary packet $p_{g_j}^i$ of flow g_j , the departure time at P-shaper S_{j+1} is bounded by: $d_{g_j,i}^{S_{j+1}} \leq GRC_{g_j,i}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$, where $I_j \geq 2$ is the number of immediate constituents of flow g_j at node j .

Proof. Let $p_{g_j}^i, \dots, p_{g_j}^K$ be the set of packets orderly departed at S_{j+1} during the same busy period. As $p_{g_j}^i$ is the first transmitted, we have: $d_{g_j,i}^{S_{j+1}} \leq a_{g_j,i}^{S_{j+1}} + L_{\max} / R_{g_j}$.

From the definition of GR, we have: $a_{g_j,i}^{S_{j+1}} = d_{g_j,i}^j \leq GRC_{g_j,i}^j + \beta_j$.

Hence, we get: $d_{g_j,i}^{S_{j+1}} \leq GRC_{g_j,i}^j + \beta_j + L_{\max} / R_{g_j} \leq GRC_{g_j,i}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$.

We now prove the lemma by contradiction. Assume $p_{g_j}^k$ is the first packet such that: $d_{g_j,k}^{S_{j+1}} > GRC_{g_j,k}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$.

Define $t_2 = GRC_{g_j,k}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$. Let t_1 be the least time less than t_2 during the busy period such that no packet with GRC value greater than t_2 is served in the interval $[t_1, t_2]$. Such a t_1 exists. Otherwise, $p_{g_j}^k$ is either the first packet served in the busy period or the first packet served after its arrival at the shaper (with the smallest GRC value among waiting packets). In either case, the departure time of the packet is bounded by:

$$d_{g_j,k}^{S_{j+1}} \leq a_{g_j,k}^{S_{j+1}} + l_{g_j}^k / R_{g_j} \leq GRC_{g_j,k}^j + \beta_j + (I_j - 1) L_{\max} / R_{g_j}$$

which violates our assumption. Therefore, such a t_1 exists.

On the other hand, packet served in the interval $[t_1, t_2]$ must also arrive in the interval. Otherwise the packet would have been served before t_1 , because packets served in $[t_1, t_2]$ are all with GRC value smaller than or equal $GRC_{g_j, k}^j$, which is smaller than the last packet served before t_1 . Define $AP_{g_j}(t_1, t_2)$ as the cumulative packet length of packets in flow g_j that arrive in interval $[t_1, t_2]$ and have GRC value no greater than $GRC_{g_j, k}^j$, then, we get:

$$\begin{cases} AP_{g_j}(t_1, t_2) > R_{g_j}(t_2 - t_1) + L_{\max} & t_1 \text{ is the beginning of the busy period} \\ AP_{g_j}(t_1, t_2) > R_{g_j}(t_2 - t_1) & \text{otherwise} \end{cases} \quad (4.6)$$

as the shaper has capacity R_{g_j} and packet $p_{g_j}^k$ is not served by time t_2 .

Second, assume g_j is formed by the set of immediate constituents $f_1, \dots, f_{I_j}$ at the node j (obviously, g_{j-1} is one of them). For an arbitrary flow f_i , assume packets $p_{f_i, m}, \dots, p_{f_i, n}$ are in $AP_{g_j}(t_1, t_2)$. From the definition of $AP_{g_j}(t_1, t_2)$, we get:

$$\begin{cases} GRC_{f_i, n}^j \leq GRC_{g_j, k}^j = t_2 - \beta_j - (I_j - 1) \times L_{\max} / R_{g_j} \\ t_1 \leq a_{f_i, m}^{S_{j+1}} = d_{f_i, m}^j \leq GRC_{f_i, m}^j + \beta_j \end{cases} \quad (4.7)$$

In addition, from the definition of GR, we can deduce:

$$\begin{aligned} GRC_{f_i, n}^j &= \max\{a_{f_i, n}^j, GRC_{f_i, n-1}^j\} + l_{f_i}^n / R_{f_i} \geq GRC_{f_i, n-1}^j + l_{f_i}^n / R_{f_i} \\ \dots &\geq GRC_{f_i, m}^j + \frac{\sum_{e=m+1}^n l_{f_i}^e}{R_{f_i}} \end{aligned}$$

Then, we get: $AP_{f_i}(t_1, t_2) = \sum_{e=m}^n l_{f_i}^e \leq R_{f_i}(GRC_{f_i, n}^j - GRC_{f_i, m}^j) + l_{f_i}^m$

Together with Eq. (4.7), we have: $AP_{f_i}(t_1, t_2) \leq R_{f_i} \left(t_2 - t_1 - (I_j - 1) \times L_{\max} / R_{g_j} \right) + l_{f_i}^m$.

As g_j is formed by flows $f_1, \dots, f_{I_j}$, then $R_{g_j} = \sum_{i=1}^{I_j} R_{f_i}$. Therefore, we obtain:

1) t_1 is the beginning of the busy period:

$$\begin{aligned} AP_{g_j}(t_1, t_2) &= \sum_{i=1}^{I_j} AP_{f_i}(t_1, t_2) \leq \left(t_2 - t_1 - (I_j - 1) \times L_{\max} / R_{g_j} \right) \sum_{i=1}^{I_j} R_{f_i} + I_j L_{\max} \\ &\leq R_{g_j} (t_2 - t_1) + L_{\max} \end{aligned}$$

2) t_1 is not the beginning of the busy period. From the definition, clearly t_1 is the

departure time of a packet with GRC value greater than $GRC_{g_j, k}^j$. This implies

there is at least one constituent of g_j that has no packets in $AP_{f_i}(t_1, t_2)$. Then,

$$\begin{aligned} AP_{g_j}(t_1, t_2) &= \sum_i AP_{f_i}(t_1, t_2) \leq (t_2 - t_1 - (I_j - 1) \times L_{\max} / R_{g_j}) \times \sum_i R_{f_i} + (I_j - 1) \times L_{\max} \\ &\leq R_{g_j} (t_2 - t_1) \end{aligned}$$

This contradicts with Eq. (4.6). Thus, $d_{g_j, k}^{S_{j+1}} \leq GRC_{g_j, k}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$.

Now, we present an important concatenation property of FA-GR.

Theorem 4.3: For a packet $p_{g_j}^i$ of flow g_j , its GRC values at two consecutive FA-GR server j and $j+1$ has the following relationship:

$$GRC_{g_{j,i}}^{j+1} \leq GRC_{g_{j-1,i'}}^j + \beta_j + I_j L_{\max} / R_{g_j}$$

where $p_{g_{j-1}}^{i'} = p_{g_j}^i$.i.e., the i' th packet of g_{j-1} is the i th packet of g_j .

Proof. From Lemma 4.2, we know: $d_{g_j, i}^{S_{j+1}} \leq GRC_{g_{j-1}, i'}^j + \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$.

Hence, we can treat the whole of the FA-GR server j and P-shaper $j+1$ as a single

GR scheduler for flow g_{j-1} , with latency $e_j = \beta_j + (I_j - 1) \times L_{\max} / R_{g_j}$.

At FA-GR $j+1$, $GRC_{g_j,i}^{j+1} = \max\{a_{g_j,i}^{j+1}, GRC_{g_j,i-1}^{j+1}\} + l_{g_j}^i / R_{g_j}$. There are two cases:

Case 1. $a_{g_j,i}^{j+1} \geq GRC_{g_j,i-1}^{j+1}$, we have $GRC_{g_j,i}^{j+1} = a_{g_j,i}^{j+1} + l_{g_j}^i / R_{g_j} \leq GRC_{g_{j-1},i'}^j + \beta_j + I_j L_{\max} / R_{g_j}$

Case 2. $a_{g_j,i}^{j+1} < GRC_{g_j,i-1}^{j+1}$. By the definition of GR, we can get: $GRC_{g_j,i}^{j+1} = a_{g_j,k}^{j+1} + \frac{\sum_{e=k}^i l_{g_j}^e}{R_{g_j}}$

where $p_{g_j}^k$ is the first packet in the busy period of g_j at FA-GR $j+1$.

On the other hand, g_j at the output of P-shaper is leaky bucket constraint in the

form of $R_{g_j}t + L_{\max}$, we have: $a_{g_j,i}^{j+1} - a_{g_j,k}^{j+1} \geq \left(\frac{\sum_{e=k}^i l_{g_j}^e}{R_{g_j}} - \frac{L_{\max}}{R_{g_j}} \right)^+$.

Combining Case 2, we get: $GRC_{g_j,i}^{j+1} \leq a_{g_j,i}^{j+1} + L_{\max} / R_{g_j} \leq GRC_{g_{j-1},i'}^j + \beta_j + I_j L_{\max} / R_{g_j}$.

In all cases, we have proved $GRC_{g_j,i}^{j+1} \leq GRC_{g_{j-1},i'}^j + \beta_j + I_j L_{\max} / R_{g_j}$. End of Proof.

Consider a flow f traverses a network along a series of nodes $1, \dots, J$ that couple FA-GR scheduler and P-shaper, the root of f at node j is g_{j-1} , we can directly get the end-to-end concatenation property of f by induction of Theorem 4.3 as:

$$GRC_{g_{j-1},i}^J \leq GRC_{f,i'}^1 + \sum_{j=1}^{J-1} \left(\beta_j + I_j L_{\max} / R_{g_j} \right) \quad (4.8)$$

If we further assume flow f is (σ_f, R_f) -constraint when entering the network, end-to-end delay bound can be derived like Eq. (4.4) for f as:

$$D_{\text{end-to-end}} \leq GRC_{f,i}^1 - a_{f,i}^1 + \sum_{j=1}^{J-1} \left(\beta_j + I_j L_{\max} / R_{g_j} \right) + \beta_J \quad (4.9)$$

Next, we consider the case where both flow aggregation and separation occur at core routers. For simplicity, we first consider a simple case as shown in Figure 4.4 to

present the key idea of modeling to obtain end-to-end delay bound in the presence of flow separation. Flow f and l are aggregated into flow g_i at node i , and after a number of nodes, the root g_{j-1} of f is separated into f , l and/or other flows at node j . Note that there is no flow separation between node i and j whereas flow aggregation may occur. Then at node $j+1$, f is aggregated with m into g_j . In this case, we explore the relationship of a packet the GRC values at node i and $j+1$ in the presence of flow separation between them.

Figure 4.4: Flow separation occurs at the network.

Eq. (4.8) is still applicable from node i to j because flow separation at node j has no effect on packet delay at node j . Hence, we get:

$$GRC_{f,k}^j \leq GRC_{f,k}^i + \sum_{m=i}^{j-1} (\beta_m + I_m L_{\max} / R_{g_m})$$

The departure time of the packet p_f^k is bounded by:

$$d_{f,k}^j \leq GRC_{f,k}^j + \beta_j \leq GRC_{f,k}^i + \sum_{m=i}^{j-1} (\beta_m + I_m L_{\max} / R_{g_m}) + \beta_j$$

Finally, we can get the end-to-end delay bound using Eq. (4.9) for a flow with leaky

bucket constraint as $\frac{\sigma_f}{R_f} + \sum_{m=i}^j (\beta_m + I_m L_{\max} / R_{g_m}) + \beta_{j+1}$.

Now, we consider the general case that flow aggregation and separation may occur at every node. By using Eq. (4.9) and the modeling method presented above, we give three steps to derive the end-to-end delay bound for the general scenario:

Step1. Using the modeling method to cover each flow separation. That is to model the set of nodes, where the roots of the considered flow at the start node and after the flow separation at last node are the same and there is no other flow separation except at the last node, as a single GR node.

Step2. Iteratively doing step 1 until there is no flow separation is “seen”.

Step3. Using Eq. (4.9) to obtain the end-to-end delay bound.

Below, we give the end-to-end delay bound for a flow through a general network with flow aggregation and separation may occur at every node.

Theorem 4.4: Assume f traverses a series of nodes $1...J$ through a network with flow aggregation and separation that may occur at every node, and f is (σ_f, R_f) -constraint by entering the network. The end-to-end delay bound of f satisfies:

$$D_{end-to-end} \leq \frac{\sigma_f}{R_f} + \sum_{j=1}^{J-1} \left(\beta_j + I_j L_{\max} / R_{g_j} \right) + \beta_J$$

Comment: Apparently, Theorem 4.4 has the same form as (4.9). However, as there is no flow separation except at the egress node in (4.9), the term R_{g_j} is wide-sense increasing with j . In Theorem 4.4, as flow separation may occur at each node along the flow path, the term R_{g_j} is varying widely.

4.3.5 Performance Comparison

In this section, we compare the performance of our approach with previous approaches. The performance metrics for comparison include end-to-end delay bound, scheduling complexity, and the implementation cost such as queue management, deadline calculation, etc.

4.3.5.1 End-to-end Delay Bound

For per-flow based approaches [GLV95,Zha95] in IntServ network, the end-to-end delay bound generally contains one rate-based term $\sigma_f / R_f + (J-1)L_{\max} / R_f$, where J is the number of nodes the flow f traverses. [SS03,SS04,SS05] achieves end-to-

end delay in the form of $\frac{\sigma_f}{R_f} + (\frac{\sum_{k \neq f} \sigma_k}{R_g} + \frac{\sum_{k \neq f} \phi_k^1 \cdot r_k + L_{\max}}{R_g}) + (J-3)\frac{L_{\max}}{R_g} + \frac{L_{\max}}{R_f}$ with

flow aggregation only at the ingress node. ϕ_k^1 is the latency of flow k at node 1. For two-level fair aggregation based approaches of (GR with GRFA) and [Cob02], the end-to-end delay bound has the form $\sigma_f / R_f + \sum_{j=1}^{J-1} (\beta_{v_j} + L_{\max} / R_{g_j})$, where β_{v_j} is the latency term of the fair aggregator at node j . For our FA-GR approach, as shown in Theorem 4.4, it covers the corresponding term $\sigma_f / R_f + \sum_{j=1}^{J-1} I_j L_{\max} / R_{g_j}$.

We can see that the approaches without flow aggregation achieve the worst end-to-end delay bound since R_f is used for calculating end-to-end delay bound, which is generally far smaller than R_{g_1} . Although, [SS03,SS04,SS05] make some improvement from non-aggregation approaches, it is clear from the form that the improvement is worse than that of FA-GR. This is because flow aggregation only happens at the ingress node. On the contrary, FA-GR approach allows flow

aggregation at every node and the end-to-end delay bound gains from flow aggregation at every node. Furthermore, the FA-GR working with P-shaper makes the aggregation fair among constituent flows and prevents flow aggregation from bringing extra latency. For scheduling disciplines with $\beta_{v_j} = L_{\max} / R_{g_j}$, the two-level fair aggregation approaches have better end-to-end delay bound than FA-GR as $I_j \geq 2$. For scheduling disciplines with $\beta_{v_j} = (I_j - 1)L_{\max} / R_{g_j}$, the two-level fair aggregation approaches have the same end-to-end delay bound as FA-GR.

Figure 4.5: An example network of configuration.

Next, we illustrate the performance of end-to-end delay bound for different approaches through an example network as shown in Figure 4.5. We consider a flow f with $R_f = 1Mbps$ through a network spanning 8 nodes. For non-aggregation approaches, the flow f is guaranteed 1Mbps bandwidth at each node. For ingress node aggregation approaches, we assume 4 other flows with the same arrival property as f are aggregated together with f at the ingress node and sharing the same end-to-end path. For two-level fair aggregation approaches and FA-GR, we make the same scenario as ingress aggregation approaches at the ingress node and at all other nodes j along the path of f (except the last node) where there is another flow with the

same rate as g_{j-1} aggregated together with g_{j-1} to produce g_j . For two-level fair aggregation approaches, assume $\beta_{v_j} = L_{\max} / R_{g_j}$. The link capacity is 500Mbps for all links and the scheduling latency is $L_{\max} / C$ for all scheduling disciplines with $L_{\max} = 1000\text{bytes}$. Figure 4.6 (a) shows the variation of end-to-end delay bound with changing burst size σ_f . The comparison shows that the approaches with flow aggregation outperform those without aggregation. Our approach as well as the two-level fair aggregation is better than the ingress node aggregation approach which is limited by flow aggregation only at ingress routers. For example, our approach achieves 19.23ms and 27.23ms at σ_f of 1000Bytes and 2000Bytes, respectively, which yields 40.2% and 41.5% improvement over that of ingress-aggregation approaches, and 70.0% and 62.3% improvement over that of non-aggregation approaches. The two-level fair aggregation approaches have better end-to-end delay bound than ours. Nevertheless, the difference is marginal (about 4.7ms difference for all cases), Moreover, the slope of ingress-aggregation approach is sharper than other approaches, indicating the ingress aggregation approach is more sensitive to σ_f .

Simulations are conducted to show the performance under different burst sizes and link utilizations. In order to simulate interferences by cross traffic, external traffic is injected at each node (except the last node) along the path. The cross traffic at each node is composed of two flows with the same characteristic and shares the path with the aggregated flows, i.e., flow g_x for only one hop before exiting the network at the next hop (except the interfering flows of g_7 depart from the same port as constituents of g_7 at node n_8). All traffic is generated using exponential distribution with fixed

(a) End-to-end delay bound with varying burst size.

(b) Simulative maximum end-to-end delay with varying burst size.

(c) Simulative maximum end-to-end delay with varying traffic load.

Figure 4.6: End-to-end delay comparison.

packet size of 1Kbytes. PSRG scheduling is used as the GR scheduler for both traffic scheduling and fair aggregator construction. Figure 4.6 (b) shows the maximum end-to-end delay with varying burst size under traffic load 0.98 (by manipulating cross traffic load). All the three aggregate scheduling achieve much lower maximum end-to-end delay than the non-aggregation approach, which is consistent with the analytical bound performance in Figure 4.6 (a). This is attributed to multiplexing gain of aggregate scheduling, i.e., the probability that all flows constituting an aggregate flow reach their peaks (of load) at the same time is very low [SS05]. Also, the fair aggregator design of two-level approach and FA-GR help to reduce delay bound with traffic aggregation [Cob02]. On the other hand, although ingress aggregation approach has larger end-to-end delay bound than two-level aggregation and FA-GR approaches, it achieves better simulation results. This is mainly because two-level aggregation and FA-GR employ non-work-conserving aggregator and thus can not take advantage of spare bandwidth. Figure 4.6 (c) presents the maximum end-to-end delay under varying traffic load (by manipulating traffic load of each flow). Both non-aggregation and ingress-aggregation approaches obtain much lower maximum end-to-end delay at non-saturation load than two-level aggregation and FA-GR approaches, which is due to non-work-conserving property of the aggregator designed in two-level aggregation and FA-GR. At saturation load, the maximum end-to-end delay of non-aggregation approach increases significantly, while ingress aggregation approach still achieves better performance due to multiplexing gain and no traffic aggregation at the core nodes. In summary, aggregate scheduling approaches achieve significantly lower maximum end-to-end delay at heavy load than non-aggregation approach. Two-level aggregation approach and FA-GR have lower end-to-end delay bound than ingress

aggregation approach due to further traffic aggregation at core nodes but generally obtain higher simulative maximum end-to-end delay due to the non-work-conserving property of the fair aggregator.

Figure 4.7: End-to-end delay with varying aggregated flows.

Now, we study how the number of flows for aggregation can affect on the end-to-end delay bound. We use a similar network configuration as in Figure 4.5, but with modification that $R_f = 200Kbps$ and σ_f fixed at 1200bytes. We consider two scenarios S1 and S2. In S1, the number of flows for aggregation at the ingress node varies from 5 to 25. Similarly at all other nodes except the last one, there is another flow with the same rate as g_{j-1} aggregated with g_{j-1} to produce g_j . In S2, the number of flows for aggregation at the ingress node is fixed at 5, while at node 2, the number of flows for aggregation varies from 1 to 9 (each flow has the same rate as g_1). In order to compare the performance of S1 and S2, at each point of S2 in Figure 4.7 the flow number is selected such that S1 and S2 will generate the same rate of g_2 . For example, when S1 has 15 flows aggregated at the ingress node, S2 will have 5 flows aggregated with g_1 . Thus both S1 and S2 will generate g_2 with rate 6Mbps.

Simulation results are also presented for S1 and S2 in Figure 4.7 which are conducted with total load at 0.98. We make three observations regarding the figure. First, the end-to-end delay bound decreases with the increasing number of flows for aggregation in both S1 and S2. Second, the better performance of S1 shows that aggregating flows at the ingress is more effective to lower delay bound than flow aggregation at core nodes, i.e., the earlier the aggregation, the better the performance. Third, the simulation result of S1 and S2 is consistent with the analytical bound performance. Also, both of the simulation results are much lower than the delay bound, which is attributed to the multiplexing gain of aggregate scheduling as discussed earlier.

4.3.5.2 Complexity

The scheduling complexity of flow-based approaches is $O(\log N)$ for deadline-based approaches [GLV95,Zha95] where N is very large. More precisely, in order to achieve guaranteed end-to-end performance, a flow needs to be allocated a separate packet queue and receives a specific scheduling treatment at each node along its path. Since there are tremendous individual flows that a link scheduler needs to be aware of and from which to select the next packet to send, the scheduling complexity of $O(\log N)$ is very high due to large N which brings implementation difficulty at high speed and in an efficient way. For [SS03,SS04,SS05], the complexity is $O(\log N)$ where N is significantly smaller than that of flow-based approaches after flow aggregation at the ingress node. More precisely, the ingress node is aware of individual flows and conducts flow aggregation according to some mechanism, for example, aggregating flows belonging to the same class and heading to the same

egress node into one single aggregate. This brings complexity to ingress node due to per-flow scheduling plus flow aggregation. The benefit is that the network core becomes simple and scalable since the number of flows is reduced significantly at core routers. Thus the scheduling complexity of $O(\log N)$ is marginal due to small N since the scheduling can be implemented by specific chips, like very large scale integration (VLSI) chip, at very high speed [JG02]. For two-level fair aggregation approaches, the framework is of two-level scheduling. The first level has the complexity $O(\log N)$ where N is the number of constituent flows at an aggregator. The second level has the complexity $O(\log M)$ where M is the number of flows at the link scheduler. Besides flow aggregation at ingress node, the core node can also implement flow aggregation using two-level fair aggregation model to further reduce flow number. Flow aggregation at core router further improves network performance while not bringing much complexity. First, the number of flows at a core router is very limited due to flow aggregation and thus the operation of flow aggregation can be implemented by specific chip. Second, the flow aggregation can be implemented by simple and efficient approaches, such as the label merging in MPLS protocol [FWD01]. For FA-GR approach, the complexity is $O(\log N)$ where N is the number of flows at FA-GR scheduler. Similarly, the complexity is very low due to flow aggregation at both ingress and core routers. P-shaper can be implemented by calendar queue [Zha95], which is shown to have $O(1)$ complexity. Furthermore, as constituent flows are shaped at the upstream node, packets at each P-shaper have GRC values in a short time range which makes the calendar queue simple to be implemented. Moreover, the number of packets at the P-shaper is limited due to small

bounded buffer size (refer to Theorem 4.2). Therefore, FA-GR is much simpler than that of two-level fair aggregation approaches.

As discussed above, FA-GR as well as other aggregation-based approaches can significantly reduce scheduling complexity and the operation of scheduling and flow aggregation can be implemented at high speed using specific hardware method. Therefore, FA-GR is applicable to aggregation-based networks, in particular MPLS networks, to provide guaranteed end-to-end QoS. At the ingress node, flow aggregation is conducted by techniques of traffic classification and labeling [FWD01]. At core node, flow aggregation can be further implemented by techniques such as label merging to improve network performance and scalability.

4.3.5.3 Implementation cost

Per-flow based approaches in IntServ network achieve the worst implementation cost. Per-flow management, classification, queuing and scheduling make the scalability very poor and the implementation very complex. For [SS04], packet reordering may occur at each node, the implementation can be very complex while [SS03,SS05] are relatively simple without reordering in the core. For two-level fair aggregation methods, the implementation is similar with a two-level scheduling approach and two times of deadline calculation are executed for each packet. As the technique of flow aggregation can greatly reduce the number of flows, the implementation of this approach can be very simple.

For our approach, the implementation includes several P-shapers and one FA-GR at each output port. The P-shaper can be implemented with a calendar queuing and the FA-GR is indeed a GR scheduler plus some operations for flow aggregation and/or

separation. For each packet, one time deadline calculation is needed. Thus, the overall implementation is simpler than two-level fair aggregation approaches. Moreover, there is no need to maintain per-constituent flow state at the P-shaper. For FA-GR approach, each flow needs at most $2L_{\max}$ buffer size at the FA-GR and $2NL_{\max}$ buffer size at the P-shaper. This property makes the implementation of P-shaper and FA-GR very simple due to small number of concurrent packets co-existing in the buffer. For all other approaches, the buffer size needed is unpredictable and the burstiness accumulation from node to node will make the queuing system very complex.

4.3.6 Summary

In this chapter, we have proposed two new techniques, namely flow aggregation capable guaranteed rate (FA-GR) scheduling and priority shaper (P-shaper). FA-GR performs packet scheduling as general GR scheduling, while encoding GRC value of the packet into the packet header and executing flow aggregation and/or separation if needed. P-shaper is a deadline-based shaper, which controls the output to obey an arrival curve while delivering packets by the order of their GRC values carried in their headers from the upstream FA-GR server.

We present the framework of the network with nodes applying P-shaper and FA-GR scheduler and analyze the per-node delay and buffer requirement. Then, we derive the end-to-end delay bound for the network. Consequently, we compare the performance of our approach with previous approaches. The results show that FA-GR approach is significantly better than non-aggregation approaches and ingress-aggregation approaches particularly in end-to-end delay bound. As compared with

two-level fair aggregation approaches, the FA-GR approach achieves similar end-to-end delay bound, but with better scheduling complexity and implementation cost.

Chapter 5

Hybrid Cycle Scheduling: A Fair Bandwidth

Allocation Algorithm with QoS Support in EPON

5.1 Introduction

In this chapter, we propose the design of DBA algorithm for upstream bandwidth allocation in EPON. The key performance metrics of a DBA algorithm includes efficiency and QoS-support. An efficient DBA will result in high link utilization. This is achieved through statistical multiplexing, reduced guard time and timeslot remainder, and reduced cost in round-trip time (RTT) and DBA processing in bandwidth scheduling decision. QoS-support refers to the capability of providing differentiated services in order to accommodate heterogeneous traffic with diverse QoS requirements. This is achieved through traffic classification by enumerating the types of applications supported in the network and accordingly delivering distinct services to each traffic class based on their QoS demanding.

In Section 5.2, we present a new DBA scheme. We adopt cycle partition strategy to separate the transmission of high priority traffic from medium/low priority traffic in order to provide premium service to high priority traffic. Meanwhile, an adaptive

approach is introduced for medium and low traffic transmission to improve network utilization and performance. A suitable local buffer management scheme is also proposed to facilitate QoS implementation. In order to improve delay and delay-variation performance, a novel feature providing potentially multiple transmission opportunities (M-opportunities) per-cycle has been developed for high priority traffic. Furthermore, we develop analytical performance analysis on the deterministic delay bound for high priority traffic and minimum throughput guarantees for both high and medium priority traffic.

In Section 5.3, we conduct detailed simulation experiments to verify the effectiveness and efficiency of the proposed algorithm. The simulation analysis is performed using synthesis traces that exhibit the property of self-similarity [TWS97]. Using self-similarity traffic is vital important to obtain an accurate and realistic performance analysis. It has been studied extensively that most network traffic flows can be characterized by self-similarity [WTE96]. A detailed description on how to generate self-similarity traffic is introduced in [TWS97,Kra03].

5.2 Hybrid Cycle Bandwidth Allocation Scheme (HCS)

In this section, we present a new DBA algorithm, termed HCS, to support differentiated services in EPONs. We adopt the traffic classification recommendations in [BBC98] to classify traffic into three classes, i.e., expedited forwarding (EF), assured forwarding (AF), and best effort (BE). The objective of the scheme is to provide guaranteed services to EF traffic to satisfy the rigorous delay and delay-variation requirements, and at the same time offering guaranteed throughput for AF traffic as well as yielding maximum throughput for BE traffic. In particular, several

specific features are developed for HCS: 1) A novel M-opportunities per-cycle solution is designed for EF class, which can fundamentally improve EF delay and delay variation performance, 2) there is no RTT idling problem which substantially improve network throughput. Meanwhile, an adaptive extended AF cycle is designed that further reduces guard time cost to improve throughput performance. While previous QoS-support algorithms [AYD03, BSG05, XJJ04, JX06, etc] suffer RTT idling problem and some of them also have large guard time cost due to class-based transmission yielding lower network throughput.

5.2.1 EF Cycle and AF Cycle

In our design, we adopt a similar strategy used in [AHK03,XJJ04,BSG05] to partition a scheduling frame into two parts. The first part has N fixed and orderly distributed timeslots, termed EF timeslots, corresponding to N ONUs ($ONU_1 \sim ONU_N$). This is intended for steady EF traffic transmission. The size of each EF timeslot is determined according to the SLA between the subscriber and the service provider. Normally, the EF traffic requires a small percentage of the total bandwidth, for example 20%, and is not bursty in nature (e.g., constant bit rate, Poisson distribution), thus the cyclic fixed timeslot mechanism ensures the EF frames to be transmitted in short time delay and with little variation.

The second part is designed for AF and BE traffic. This is divided into a variable number of timeslots, termed as AF timeslots, in each frame depending on the instant network load. BE traffic has the lowest priority and has no QoS requirement, thus it is transmitted only when there are no EF and AF frames in an ONU. The AF traffic is not sensitive to delay and jitter but requires statistical throughput guarantees. We

propose a novel mechanism that is adaptive to the network load and tolerates the burstiness of AF traffic. It provides guaranteed throughput in a relatively large-time scale for AF traffic while maximizing the BE throughput at heavy load by fully exploiting the total network throughput. Each ONU is fairly allocated bandwidth in the second part according to its AF/BE weight, thus AF traffic is always guaranteed its share of bandwidth while BE throughput of an ONU is affected by its own EF and AF traffic but is immune to overload traffic of other ONUs.

Figure 5.1: Illustration of EF cycle and AF cycle

Now, we introduce the concept of multiple cycles. An EF cycle, given by T_{EF} , is defined as the time period between two consecutive beginnings of an ONU_1 EF timeslot. An EF cycle is equal to one scheduling frame size that is limited in a range of $[T_{EF}^{Min}, T_{EF}^{Max}]$. T_{EF}^{Min} is introduced to prevent the light load downstream consumption problem when the steady EF part is small. T_{EF}^{Max} is the upper limit of the EF cycle which is critical to delay performance of EF traffic. The scheme is illustrated in Figure 5.1. An EF cycle covers one EF timeslot for each ONU and one or more AF timeslot for some ONUs depending on the AF allocation. The other cycle is the AF cycle, given by T_{AF} , which is defined as the time period covering a complete set of AF timeslots. As shown in Figure 5.1, T_{AF} spans the interval between two consecutive beginnings of an ONU_1 AF timeslot. The cycle of an AF timeslot is dynamic and is

adaptive to network load. One AF cycle may span several consecutive EF cycles. Figure 5.1 shows an example that one AF cycle spans three EF cycles. Therefore, it is possible that not all ONUs are allocated an AF timeslot in an EF cycle. Furthermore, an ONU may be allocated two adjacent AF timeslots each in one of the two adjacent EF cycles. An example is illustrated in Figure 5.1 where ONU₄ has two adjacent timeslots with each in the two adjacent EF cycles.

An adaptive dynamic AF cycle with a large maximum value is significant in improving system performance. Principally, it serves two objectives. First, when the network is heavily loaded, a large value of T_{AF} can reduce guard time waste and unused timeslot remainder (i.e., timeslot is not fully utilized by an ONU due to prohibited packet fragmentation) in a cycle, especially in a network with a large number of ONUs, thus, effectively improving the system throughput. Second, the maximum value of T_{AF} also determines the maximum AF timeslot size allocated to an ONU. For a heavily loaded ONU, the grant size for its AF/BE traffic equals its maximum AF timeslot size. Since AF/BE traffic is generally generated by applications that exhibit bursty nature, such as variable-bit-rate (VBR) video streams, it needs a more elastic approach in distributing AF timeslot to combat the bursty property of AF/BE traffic. A larger maximum AF timeslot size will enable a larger timeslot to be allocated when the bursty traffic arrives. Meanwhile, a small timeslot is reserved when there is no bursty traffic. Therefore, a larger maximum T_{AF} which produces a larger maximum AF timeslot size for each ONU makes the HCS scheme well suited to accommodate AF/BE traffic. On the other hand, the delay performance of AF traffic also benefits from a large maximum T_{AF} . This is because more AF

frames of an ONU will be transmitted in one cycle instead of being delayed to the next cycle. This will reduce packet delay induced by guard time, thus improves AF delay performance.

In order to improve bandwidth utilization in light load situation where EF traffic cannot consume the entire EF timeslot, AF and BE frames will be transmitted when there is no EF frame queuing while the leftover timeslot is large enough to transmit the head-of-line frame of AF/BE queue. Similarly, EF frame can be transmitted in AF timeslot. Here, in order to ensure guaranteed bandwidth for AF traffic and improve the delay and delay-variation performance for EF traffic, we propose a new priority scheme as part of buffer management for the three classes in the AF timeslot. The AF class is made to have the highest priority for transmission, EF, the second and BE, the last. The proposal is based on the reasoning that EF traffic has already secured its lion share of bandwidth in an EF timeslot whereas AF traffic needs to provide guaranteed throughput demand and cannot be starved by EF traffic, while BE traffic has no QoS requirement. Transmitting AF/BE traffic in an EF timeslot will leave more space in an AF timeslot for EF traffic. Setting the priority of EF traffic ahead of BE traffic in an AF timeslot will further benefit EF traffic, especially at light load scenario. This is because the statistical cycle time for EF traffic is reduced by half at light load, i.e., EF frames will be sent out two times in one EF cycle. At a heavy load scenario, although an AF timeslot emerges once in multiple EF cycles for an ONU, the statistical cycle time of EF traffic is still much reduced. Therefore, statistically there are more than one opportunities for EF transmission in each EF cycle. We call this new style of timeslot scheduling for EF traffic as the M-opportunities per-cycle solution. This idea is inspired by the newly developed round robin scheduling algorithms, namely

smoothed round robin [Guo01] and stratified round robin [RP03], where the premium flows are scheduled more than one times in each cycle according to their reserved rate. This approach can significantly lower average delay. Previous approaches using S-opportunity per-cycle for EF traffic encounter a large delay due to long waiting time for each EF service and large delay variation due to long and fluctuating cycle time. Although the work in [BSG05] can provide accurate prediction for an EF timeslot, this is true only for constant rate EF traffic but not so for dynamic rate. Besides, it has the RTT idling problem.

The OLT maintains a table of ONUs, including the fixed EF timeslot size, maximum AF timeslot size and the latest total buffer occupancy information of AF and BE classes of each ONU. An ONU sends a REPORT message to the OLT at the end of each EF and AF timeslot, reporting the total queuing size of the AF and BE traffic. OLT determines the grant AF timeslot size for AF and BE traffic according to the latest buffer occupancy information and sends GATE messages to ONUs one maximum RTT time earlier than the start time of the first AF timeslot. AF timeslots are granted to ONUs orderly in an AF cycle and this requires the OLT to remember the latest ONU that has been granted AF timeslots. As the EF timeslot and its relative position for each ONU is fixed in each EF cycle, OLT sends the EF grant messages to each ONU ahead of the beginning of the next EF cycle to keep the ONU active. In order to save GATE message costs, the EF grant for the next EF cycle and the AF/BE grant for the current cycle can be delivered in one single GATE message to the destined ONU if the ONU has AF timeslot in the current EF cycle.

5.2.2 Performance Analysis for EF Traffic

Guaranteed bandwidth is provided to the EF class as a fixed timeslot is reserved for an ONU in each EF cycle ($\leq T_{EF}^{Max}$). The EF traffic of ONU_i in each EF cycle receives a fixed service in the timeslot size of

$$W_i^{EF} = (T_{fix} - NT_g) * \frac{w_i}{\sum_j w_j} * R_E \quad (5.1)$$

Table 5-1: Parameter used for performance analysis

N	Total number of ONUs
T_g	Guard time between timeslots
R_U, R_E	Line rate of user to ONU link and EPON link rate, respectively.
$T_{EF}^{Max}, T_{EF}^{Min}$	Maximum and minimum EF cycle
T_{fix}	Steady EF part in an EF cycle, i.e., total EF timeslot size plus NT_g
$T_{dynamic}^{Max}, T_{dynamic}^{Min}$	Maximum and minimum AF dynamic part in an EF cycle $T_{dynamic}^{Max}(T_{dynamic}^{Min}) = T_{EF}^{Max}(T_{EF}^{Min}) - T_{fix}$
w_i, ϕ_i	Weight of EF class and total weight of AF and BE classes for ONU_i
W_i^{EF}, W_i^{AF-max}	EF timeslot size and maximum AF timeslot size of ONU_i
RTT_i	Round trip time of ONU_i
W_i^{Min}	Minimum AF grant size for ONU_i
R_i, G_i	AF/BE bandwidth request and grant size of ONU_i

The parameters used in the performance analysis are listed in Table 5-1. Then, the service curve for ONU_i is:

$$\beta(t) \geq \left[R_E t \wedge R_E (t - T_{EF}^{Max})^+ + W_i^{EF} \wedge R_E (t - 2T_{EF}^{Max})^+ + 2W_i^{EF} \wedge \dots \right] \setminus \otimes \delta \left(t - \left(T_{EF}^{Max} - \frac{W_i^{EF}}{R_E} + \frac{l_{AF}^{max} + l_{EF}^{max}}{R_E} \right) \right) \quad (5.2)$$

where $\begin{cases} (\cdot)^+ = \max(\cdot, 0) \text{ and } \wedge \text{ is defined as } f \wedge g = \min(f, g) \\ \delta(t-T) = \begin{cases} +\infty & \text{if } t > T \\ 0 & \text{otherwise} \end{cases} \\ f(t) \otimes g(t) = \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\} \text{ where } \inf(S) \text{ represents the infimum of } S \end{cases}$.

$l_{AF}^{\max} / R_E$ is due to the effect of low priority traffic and $l_{EF}^{\max} / R_E$ is due to packetization. $l_{AF}^{\max}$ ($l_{EF}^{\max}$) is the maximum packet size of AF/BE (EF) traffic. If the arrival curve $\alpha(t)$ of the EF traffic is constrained by $\alpha(t \leq T_{EF}^{Max}) \leq \min(R_E t + l_{EF}^{\max}, W_i^{EF})$, i.e., at most W_i^{EF} bits of EF traffic arriving at any interval T_{EF}^{Max} , consequently, the result of network calculus on delay bound property in Theorem 2.2 can be applied. Accordingly, the maximum queuing delay for EF traffic of ONU_i is the maximum horizontal distance between the arrival curve $\alpha(t)$ and the service curve $\beta(t)$ and thus is proved to be bounded by:

$$T_{EF}^{Max} - \frac{W_i^{EF}}{R_E} + \frac{l_{AF}^{\max} + 2l_{EF}^{\max}}{R_E} \quad (5.3)$$

The arrival curve, service curve and queuing delay for ONU_i are illustrated in Figure 5.2.

Figure 5.2: Illustration of queuing delay for ONU_i.

Now, we consider the guaranteed EF traffic throughput and service rate of ONU_i .

The EF traffic of ONU_i receives a fixed service of W_i^{EF} in each EF cycle, thus during any backlogged period $(t, t + nT_{EF}^{Max}]$, the cumulative bits departure function W in $(t, t + nT_{EF}^{Max}]$ guarantees that:

$$W(t, t + nT_{EF}^{Max}) \geq n(W_i^{EF} - l_i^{EF-remainder}) \quad (5.4)$$

$$\text{where } l_i^{EF-remainder} = \begin{cases} W_i^{EF} \bmod l_{\max}^{EF} & \text{if all EF frames have equal size} \\ l_{\max}^{EF} - 1 & \text{otherwise} \end{cases}$$

is the unused EF timeslot remainder of ONU_i . Accordingly, the EF service rate of ONU_i is at least:

$$\frac{W_i^{EF} - l_i^{EF-remainder}}{T_{EF}^{Max}} \quad (5.5)$$

5.2.3 Dynamic Bandwidth Allocation Algorithm for AF and BE Traffic

The OLT maintains the weight of AF/BE class and the latest buffer occupancy information for each ONU. The buffer occupancy is updated when the OLT receives a REPORT message from the corresponding ONU. OLT determines the granted AF timeslot size according to the latest AF/BE buffer occupancy. AF timeslots are granted to ONUs orderly in each AF cycle which spans one or more EF cycles.

To limit the maximum T_{AF} , we define $T_{MAF} = M * T_{dynamic}^{Max}$ as the maximum size of the total AF dynamic parts in an AF cycle, where M is an integer. Accordingly, we define the maximum AF timeslot of ONU_i that can be granted in a single grant as:

$$W_i^{AF-\max} = (M * T_{dynamic}^{Max} - NT_g) * \frac{\phi_i}{\sum_j \phi_j} * R_E \quad (5.6)$$

Thus, we can get the general form of minimum grant size W_i^{Min} for ONU_i as:

$$W_i^{Min} = \min\{R_i, W_i^{AF-\max}\} \quad (5.7)$$

Figure 5.3: Example of AF distribution.

Nonetheless, there is a special case. Let $ONU_{current}$ be the first ONU that will be granted to transmit AF/BE traffic during the forthcoming AF interval. There are two cases. First, $ONU_{current}$ is not the last ONU that is allocated AF timeslot in the previous EF cycle. Then, the minimum grant size for $ONU_{current}$ is just the form of Eq. (5.7). Second, $ONU_{current}$ is the last ONU in the previous EF cycle. This means $ONU_{current}$ is allocated bandwidth less than that shown in Eq. (5.7) in the previous EF cycle in order to make the previous EF cycle maximum. As an example shown in Figure 5.3, ONU_{j+1} is partly allocated AF timeslot G_{j+1}^{Prev} in EF cycle n but $G_{j+1}^{Prev} < \min\{R_{j+1}, W_{j+1}^{AF-\max}\}$, thus in the next EF cycle, $ONU_{current}$ becomes ONU_{j+1} , i.e., ONU_{j+1} is the first one being allocated AF timeslot. Now, the maximum AF timeslot for ONU_{j+1} in the next EF cycle is left as $(W_{j+1}^{AF-\max} - G_{j+1}^{Prev})$. However, as AF/BE frame size is general very large, we define W_{AF}^{Min} as the minimum AF

guaranteed size. In the end, the minimum grant size for ONU_i which is $\text{ONU}_{\text{current}}$ in the current EF cycle and has G_i^{Prev} in the previous EF cycle is derived as:

$$W_i^{\text{Min}} = \min \left[R_i, \max \left(W_i^{\text{AF-max}} - G_i^{\text{Prev}}, W_{\text{AF}}^{\text{Min}} \right) \right] \quad (5.8)$$

Next, we try to find ONU_{end} that makes the number of ONUs in $[\text{ONU}_{\text{current}}, \text{ONU}_{\text{end}}]$ maximum and satisfies the inequality:

$$\begin{cases} \sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} \leq (T_{\text{dynamic}}^{\text{Max}} - (K-1)T_g) * R_E & \text{if current} = N \\ \sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} \leq (T_{\text{dynamic}}^{\text{Max}} - KT_g) * R_E & \text{otherwise} \end{cases} \quad (5.9)$$

where K is the number of ONUs in $[\text{ONU}_{\text{current}}, \text{ONU}_{\text{end}}]$ and $\text{current} = N$ means that ONU_N is the first ONU for AF timeslot allocation thus saving one guard time cost as its AF timeslot just follows its EF timeslot.

If $\text{ONU}_{\text{current}} \neq \text{ONU}_{\text{end}\%N+1}$ and $\sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} < (T_{\text{dynamic}}^{\text{Max}} - KT_g) * R_E$, this means the leftover AF interval in this cycle is not enough to accommodate $W_{\text{end}\%N+1}^{\text{Min}}$ of $\text{ONU}_{\text{end}\%N+1}$, i.e., the next ONU of ONU_{end} . For this situation, similarly, if the leftover AF interval deducted by one T_g is larger than $W_{\text{AF}}^{\text{Min}}$, then a total of:

$$G_i^{\text{Prev}} = (T_{\text{dynamic}}^{\text{Max}} - (K+1)T_g) * R_E - \sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}}$$

will be granted to $\text{ONU}_{\text{end}\%N+1}$ in current EF cycle to make the current EF cycle maximally utilized.

On the other hand, if $\text{ONU}_{\text{current}} = \text{ONU}_{\text{end}\%N+1}$ and $\sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} < (T_{\text{dynamic}}^{\text{Min}} - NT_g) * R_E$, this means all ONUs are allocated their desired bandwidth but the total bandwidth is

still smaller than the minimum AF interval $T_{\text{dynamic}}^{\text{Min}}$ in an EF cycle. In this case, the excess bandwidth will be fairly allocated to ONUs according to their weight. Finally, we get the grant size for ONU_i in $[\text{ONU}_{\text{current}}, \text{ONU}_{\text{end}}]$ and $\text{ONU}_{\text{end}\%N+1}$ if possible as:

$$G_i = \begin{cases} W_i^{\text{Min}} & \text{if } \sum_{j=1}^N W_j^{\text{Min}} \geq (T_{\text{dynamic}}^{\text{Min}} - NT_g) * B_{\text{total}} \text{ and } i \neq \text{end}\%N + 1 \\ (T_{\text{dynamic}}^{\text{Max}} - (K+1)T_g) * R_E - \sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} & \text{if } (T_{\text{dynamic}}^{\text{Max}} - (K+1)T_g) * R_E - \sum_{i=\text{current}}^{\text{end}} W_i^{\text{Min}} - W_{\text{AF}}^{\text{Min}} \geq 0 \\ & \text{and } i = \text{end}\%N + 1 \\ W_i^{\text{Min}} + ((T_{\text{dynamic}}^{\text{Min}} - NT_g) * B^{\text{total}} - \sum_{j=1}^N W_j^{\text{Min}}) * \frac{\varphi_i}{\sum_j \varphi_j} & \text{otherwise} \end{cases} \quad (5.10)$$

After each round of AF grants, $\text{ONU}_{\text{current}}$ is updated as:

$$\text{ONU}_{\text{current}} = \text{ONU}_{\text{end}\%N+1} \quad (5.11)$$

Comments: The definitions of T_{MAF} and $W_i^{\text{AF-max}}$ make sense in two cases. Firstly, at high load, a large T_{AF} is useful to improve network throughput by reducing the number of AF timeslots in an EF cycle to save guard time and potential timeslot remainder cost. Secondly, Eqs. (5.6) and (5.7) indicate that a large $W_i^{\text{AF-max}}$ is more adaptive to a bursty AF/BE traffic environment by raising the maximum timeslot size limitation. When an ONU has large AF/BE traffic for some time, it can get a minimum timeslot as big as $W_i^{\text{AF-max}}$. However, it cannot occupy too much bandwidth through the control of $W_i^{\text{AF-max}}$ to prevent a malicious user from unfairly occupying the large part of excess bandwidth and consequently lowering the performance of other users.

Now, we investigate the guaranteed AF traffic throughput of ONU_i . From the above formulation, we know that the maximum AF transmission part T'_{MAF} in an AF cycle (i.e., deduct the fixed EF intervals from the AF cycle) is as large as $M * T_{dynamic}^{Max} + (M + 1) * T_g + (M + 1) * W_{AF}^{Min} / R_E$. For any ONU_i , the minimum AF throughput is experienced when all other ONUs are also in the busy period and the cumulative AF transmission part in every AF cycle is just T'_{MAF} . In this case, each EF cycle is at the maximum value of T_{EF}^{Max} . Note that the EF cycle may not reach the maximum due to the aforementioned W_{AF}^{Min} requirement to fulfill the leftover AF interval. This is ignored here for simplicity since the effect is less than 0.34% for $T_{EF}^{Max} \geq 1.2 \text{ ms}$. Consider any AF backlogged period $(t, t + nT_{EF}^{Max}]$ of ONU_i , the total AF transmission part in $(t, t + nT_{EF}^{Max}]$ is $(nT_{EF}^{Max} - nT_{fix}) = nT_{dynamic}^{Max}$. Thus, the AF throughput guaranteed for ONU_i is:

$$W(t, t + nT_{EF}^{Max}) \geq (W_i^{AF-max} - l_i^{AF-remainder}) * \left\lfloor \frac{nT_{dynamic}^{Max}}{T'_{MAF}} \right\rfloor \quad (5.12)$$

Accordingly, the AF service rate of ONU_i is at least:

$$\frac{(W_i^{AF-max} - l_i^{AF-remainder}) * \left\lfloor \frac{nT_{dynamic}^{Max}}{T'_{MAF}} \right\rfloor}{nT_{EF}^{Max}} \xRightarrow{n \text{ is large}} (W_i^{AF-max} - l_i^{AF-remainder}) * \frac{T_{dynamic}^{Max}}{T'_{MAF} * T_{EF}^{Max}} \quad (5.13)$$

where $l_i^{AF-remainder} = l_{max}^{AF} - 1$ is the maximum unused AF timeslot remainder.

Clearly, AF throughput is guaranteed in a relatively large-time scale of the AF cycle while EF traffic throughput is guaranteed in a short-cycle scale of the EF cycle. This is in line with the different features and requirements of EF and AF traffic. EF traffic

is smooth and needs urgent transmission, thus a service of short-cycle scale is expected. AF traffic is bursty and is not sensitive to delay, thus a long-time scale is more suitable for it. This hybrid cycle mechanism produces high throughput performance at heavy load as compared with approaches of S-opportunity per-class-per-cycle solution.

5.3 Performance Evaluation

In this section, we calculate the performance of the proposed HCS scheme and compare the results with previous representative schemes such as DBA1 (the first dynamic bandwidth allocation algorithm) in [AYD03] and HGP (hybrid granting protocol) in [BSG05]. For this purpose, an event-driven frame-based simulation model was developed using the C programming language. We consider an EPON architecture with 16 ONUs, each at a distance of 18 km from the OLT, corresponding to an RTT of 0.18 ms. The data rate R_U and R_E are set at 1 Gbps. The guard time and the REPORT transmission time are grouped together to be 2 μs . EF traffic is generated from a Poisson process, and EF frame size is fixed at 70 bytes. To closely emulate the self-similar property of AF and BE traffic, we generate AF and BE traffic for each ONU by aggregating multiple Pareto-distributed ON-OFF sources [TWS97]. According to IEEE 802.3, the size of AF and BE frames is uniformly distributed between 64 and 1518 bytes. The traffic profile is designed to be as follows: 20% EF traffic, 40% AF traffic and 40% BE traffic. Without loss of generality, the weights of ONUs are orderly distributed as $\{1,1,1,2,2,2,3,3,3,4,4,4,5,5,5,5\}$. Each ONU has a finite buffer of 8 Mbit with maximum 560 Kbit for EF class. T_{EF}^{Max} is 1.50 ms. In

addition to the common parameters, we assume for HCS, $T_{EF}^{Min} = 0.56$ ms, $T_{fix} = 0.332$ ms, $W_{AF}^{Min} = 500$ bytes and $M = 3$ giving rise to $T_{MAF} = 3T_{EF}^{Max}$.

Furthermore, in order to overcome the uncertainty of self-similar traces and get conclusive results, each point of the results is an average of multiple repeated experiments. The confidence interval is computed for each point using t-distribution approach since standard deviation for the results is unknown. In particular, for a set of n samples, the $100(1-a)\%$ confidence interval is $\bar{x} \pm t_{a/2, n-1} s / \sqrt{n}$ [LK99], where $\bar{x}$, s , $t_{a/2, n-1}$ denote sample mean, estimated standard deviation, and $a/2$ critical value of t-distribution with $n-1$ degrees of freedom, respectively. In our simulation, the 95% confidence interval range is very small (less than 2% of the sample mean in most cases), thus it has little effect on the performance comparison and is not indicated in the Figures.

5.3.1 Delay and Delay Variation

The average queuing delay (AD) for EF class is shown in Figure 5.4 (a). HCS achieves the best performance at all loads. The improvement is in the range of 43.0% to 69.3% when compared with DBA1 and 20.0% to 95.6% when compared with HGP. In particular, even at light load below 0.4, HCS still outperforms DBA1 by 43.0% to 50.1% and HGP by 20.0% to 33.2% despite that DBA1 and HGP have a short cycle time at light load. This is attributed to the feature of M-opportunities per-cycle, i.e., statistically more than one EF transmission opportunities in each EF cycle is provided which helps to shorten the actual EF cycle nearly by half at light load despite the minimum specified EF cycle of 0.56 ms. At medium and high load, HCS is better than

(a) AD for EF and AF Class.

(b) AD for BE Class.

Figure 5.4: Simulation results of AD for EF, AF, and BE Class.

DBA1 over 46.8%, and HGP over 33.8%. This is because HCS offers fixed timeslot for EF traffic in each EF cycle to deliver all or most of the queuing EF frames. It also benefits from the solution of M-opportunities per-cycle which shortens the EF transmission cycle by statistically providing more than one transmission opportunities at each EF cycle. DBA1 and HGP face fast increasing average delay (AD) with the

increasing cycle time as there is only one EF timeslot in each cycle. There is a surge in AD for HGP at load 1.0 due to its inaccurate EF arrival estimation by using a constant rate EF traffic estimation model. In conclusion, the HCS design of fixed EF timeslot per-cycle and M-opportunities per-cycle significantly improves AD performance as compared with S-opportunity per-ONU-per-cycle featured in DBA1 and S-opportunity per-class-per-cycle featured in HGP where there is only one EF transmission opportunity in each cycle.

The average queuing delay (AD) for the AF class is also presented in Figure 5.4 (a). HCS achieves slightly higher AD than HGP (load below 0.2) and DBA1 (load below 0.4) at light load due to its bounded minimum EF cycle time, while HGP and DBA1 benefit from the short cycle time at light load. At high load, HCS performs better than HGP and DBA1. This is attributed to the efficient AF cycle design. Precisely, an AF cycle may span a number of EF cycles at heavy load by defining a large and flexible AF cycle. This helps to reduce guard time waste to improve system efficiency and enables more AF frames of an ONU to be transmitted in an AF cycle. In conclusion, the efficient adaptive AF cycle design improves system efficiency and lowers AF AD.

The AD for the BE class is depicted in Figure 5.4 (b). Similar with AF AD, HCS achieves slightly higher BE AD than HGP (load below 0.4) and DBA1 (load below 0.6) at light load while obtaining significantly lower BE AD at high load. Again, this attributes to efficient AF cycle design where an ONU can receive a large AF timeslot in an extended AF cycle that yields improved network throughput and thus lowering BE AD at high load.

Figure 5.5: Simulation results of DV for EF and AF Class.

Figure 5.5 shows the performance of delay variation (DV) for EF traffic. HCS presents excellent performance at non-saturated loads. It lowers DV by 17.9% to 52.4% as compared with DBA1 and by 12.8% to 40.8% as compared with HGP at load 0.1 to 0.9. The improvement is attributed to the mechanism of providing fixed timeslot at fixed position for EF traffic in each EF cycle and the feature of statistically offering more than one EF transmission opportunities in each EF cycle (M-opportunities per-cycle). The design of fixed timeslots at fixed position in each EF cycle smoothes the delay of EF frames and the M-opportunities per-cycle solution helps to shorten the EF transmission cycle. In particular, HCS outperforms DBA1 by 37.7% to 52.4% for load below 0.85. This is because DBA1 suffers from the severely fluctuating cycle time at light load due to high bursty AF/BE traffic as a result of using a single timeslot to accommodate all traffic classes. At high load, the cycle time tends to be stable as it is capped at the maximum value, thus resulting in good performance. On the contrary, HGP achieves relatively better performance at load

below 0.8 with 28.2% to 69.0% higher than that of HCS. This is because HGP benefits from the short cycle time at light load. Also, it allocates EF timeslots by assuming the EF traffic arrives in full speed. This is adequate to deplete all EF queues at light load. However, HGP still suffers from a severely fluctuating cycle time and thus yields much worse DV than that of HCS. Although the benefit of M-opportunities feature reduces at heavy load due to long AF cycle, HCS is still better than DBA1 over 17.9% at load 0.9, and better than HGP over 12.8% and 85.4% at load 0.9 and 1.0, respectively. At load 0.95-1.0, both HCS and HGP have a surge in DV. This is attributed to the long AF cycle of HCS and full separation of EF and AF/BE transmission of HGP, respectively. DBA1 achieves better DV at load 1.0, this is because that all traffic of an ONU is transmitted in one timeslot and EF has the highest priority. Thus, it achieves better DV performance at heavy load due to stable cycle time and timeslot size of each ONU. Nonetheless, as a whole, HCS achieves excellent improvement in DV performance by its design of fixed timeslot allocation method and M-opportunities per-cycle solution.

The DV for AF class is also illustrated in Figure 5.5. HCS obtains higher DV than HGP (load below 0.5) and DBA1 (load below 0.7) at light / medium load due to bounded minimum EF cycle time, while HGP and DBA1 benefit from the short cycle time. At heavy load, HCS outperforms HGP and DBA1 owing to the efficient AF cycle design such that more AF frames can be accommodated in an AF timeslot. This helps to improve network efficiency and smooth AF AD resulting in better AF AD.

Figure 5.6: Throughput

5.3.2 Throughput

Figure 5.6 shows the throughput performance for the EF, AF, and BE classes. Among them, the EF (lower line) and AF (upper line) throughputs are identical for all the three schemes. However, BE throughput differs at high load. HCS performs the best as compared with HGP and DBA1. For example, HCS offers 356.8 Mbps and 341.1 Mbps at loads 0.9 and 1.0, respectively for BE traffic. This improves that of DBA1 over 38.2 Mbps and 82.5 Mbps as well as that of HGP over 115.9 Mbps and 137.0 Mbps, respectively at the high load of 0.9 and 1.0. For a cost sensitive EPON system, the improvement is significant as it means that using the HCS scheme, the EPON can accommodate a lot more customers than the other two schemes. The poor performance of DBA1 and HGP is mainly due to the RTT idling problem. The cost of $2N$ guard times further aggravates the poor result of HGP. On the contrary, HCS has no RTT idling problem. It also benefits from the introduction of T_{MAF} to alleviate the waste of guard time and unused timeslot remainder at heavy load situation.

Figure 5.7: Loss Ratio

5.3.3 Loss Ratio

Figure 5.7 presents the loss ratio performance. All three schemes experience no data loss of EF and AF class except a marginal 0.1% EF loss for HGP at load 1.0 due to inaccurate EF arrival estimation. This is because priority-based dropping mechanism is adopted in all three schemes such that a new higher priority frame will displace a lower priority frame if the buffer is full (unless EF queue length reaches the limit). As for BE class, HGP suffers frame loss from load 0.8 and yields the worst results at load 0.8-1.0. This is due to its poor network throughput performance that the BE throughput becomes smaller than the BE arrival speed from load 0.8 and deteriorates with increasing load, thus yielding increased data loss ratio. DBA1 is better than HGP which starts frame loss at load 0.85 and has over 10% less than HGP at each heavy load. This is because DBA1 has relatively better throughput performance with only N guard times waste. HCS obtains the best results. It begins frame loss from load 0.9 and has over 20% and 30% less data loss than DBA1 and HGP, respectively. This

attributes to its efficient AF cycle design and RTT idling free property that yield high network throughput and subsequently reduce frame loss.

5.3.4 Delay Bound Comparison and Throughput for ONUs

For the purpose of comparison and validation, we derive the deterministic maximum queuing delay for various ONUs, i.e., ONU₃, ONU₅, ONU₇, ONU₁₁, and ONU₁₅ by two means. One is obtained from mathematical analysis using equation (5.3). Another is obtained from the simulation using the entire network traffic load from 0.1 to 1.2 while EF traffic is shaped by the arrival curve $\alpha(t \leq T_{EF}^{Max}) \leq \min(R_E t + l_{EF}^{\max}, W_i^{EF})$.

The results are tabulated and shown in Table 5-2.

Table 5-2 Maximum queuing delay under HCS

	Analysis result (μs)	Simulation result (μs)
ONU ₃	1507.264	1500.271
ONU ₅	1502.264	1500.462
ONU ₇	1495.264	1494.922
ONU ₁₁	1489.264	1488.801
ONU ₁₅	1483.264	1482.828

From the table, we observe that the simulation result is quite close to the analytical result but marginally lower. This is consistent. It shows that the analytical result can provide a reliable and tight upper bound on the maximum queuing delay. This is significant as it provides an assurance that if the source generates traffic in line with its arrival curve constraint, then all frames will be delivered in a guaranteed short time delay. This is useful for delay sensitive applications. Delay bound estimation on the access network is essential as it enables the calculation of a true end-to-end delay over

the entire network. Traditionally, many investigations on end-to-end delay bounds have focused mainly on wide area networks. Little studies have been done on access networks. Our pioneering effort is an attempt in this direction.

In order to validate the tightness of the analytical throughput guarantees on EF and AF traffic represented by formulas (5.5) and (5.13), simulation results for two different ONUs, namely ONU_7 and ONU_{15} , are carried out with the background network load varying from 0.8 to 1.2. The simulation environment is similar as before but with some minor modification. First, the BE traffic is removed in this simulation. Thus, the AF traffic is generated at twice of its original rate. This is because the EF traffic has higher priority than BE traffic in an AF timeslot, thus to prevent EF traffic occupying part of the BE bandwidth, the background traffic is set to all AF traffic to produce the worst-case EF throughput. On the other hand, as AF traffic always has higher priority than BE traffic, there is no effect of BE traffic on the worst-case AF throughput. Second, the average generating rate (called as source rate) of the considered traffic is set slightly larger than the guaranteed rate and keeps the same for all the simulations. This is needed for the test of throughput performance.

Table 5-3 shows the comparison of simulation and analytical results. At loads 0.8 and 0.9, the simulation throughput is obviously higher than the analytical throughput. This is expected as the considered ONU traffic will occupy excessive bandwidth if the network is not fully utilized. For load between 1.0 and 1.2, the simulated EF throughput is nearly the same as the analytical guaranteed throughput, with a difference of 0.025%. This shows that the analytical formula (5.5) is able to provide the minimum EF throughput guarantee. As for AF throughput at load between 1.0 and

1.2, the simulation result of ONU_7 and ONU_{15} is higher than the corresponding analytical result by 5.124% and 3.412%, respectively. The analytical formula (5.13) is able to provide reliable lower bound on AF throughput, but it is on conservative side. This is because we use the maximum AF remainder to calculate the throughput guarantee. The maximum AF remainder could be large due to the large maximum AF frame size. In [KMP01], a formula was derived for estimating the mean remainder. If we use the mean remainder of $\overline{l_i^{AF-remainder}} = 506.52$ bytes in place of $l_i^{AF-remainder}$ in formula (5.13), then the analytical result gives 45.084 for ONU_7 and 75.736 for ONU_{15} , respectively. In this case, the simulated throughput will be quite close to the analytical lower bound, with 0.963% and 0.976% difference only. Therefore, formula (5.13) describes the worst-case AF throughput guarantee, but for practical use, it is reasonable to use the following formula to estimate the AF throughput guarantees:

$$(W_i^{AF-max} - \overline{l_i^{AF-remainder}}) * \frac{T_{dynamic}^{Max}}{T_{MAF}' * T_{EF}^{Max}} \quad (5.14)$$

where $\overline{l_i^{AF-remainder}}$ is the mean AF remainder.

Table 5-3: Throughput guarantee validation

	Source Rate (Mbps)	Throughput under different network loads (Mbps)					Analytical Guaranteed rate (Mbps)
		0.8	0.9	1.0	1.1	1.2	
ONU_7 -EF	14	14.015	13.806	11.950	11.950	11.950	11.947
ONU_{15} -EF	22	21.995	21.700	19.791	19.791	19.791	19.787
ONU_7 -AF	52.8	51.807	51.814	45.518	45.287	45.219	43.300
ONU_{15} -AF	88	85.780	87.620	76.475	75.923	75.886	73.952

5.4 Summary

In this chapter, we have studied dynamic bandwidth allocation with differentiated services support in EPON. We propose a new Hybrid Cycle Scheme (HCS) for bandwidth allocation. In this scheme, the high priority traffic is transmitted in fixed timeslots at fixed positions in an EF cycle while the medium and low priority traffic is transmitted in variable timeslots in an adaptive dynamic AF cycle. Premium service is guaranteed to EF traffic through fixed timeslot approach while the adaptive design of the AF cycle effectively improves network utilization and performance by reducing the cost of guard time and timeslot remainder. A suitable local buffer management scheme has been developed, which supports a novel feature of M-opportunities per-cycle. Consequently, HCS offers potentially multiple transmission opportunities (timeslots) for EF traffic in each EF cycle which significantly improves delay and delay variation performance for EF traffic. Performance analyses on maximum queuing delay and throughput have been presented for EF and AF traffic.

In addition, we have conducted extensive simulation experiments. The simulation results agree well with the analytical results. Moreover, the results show that the HCS scheme is able to provide excellent performance in terms of average delay, delay-variation and throughput as compared with previous approaches.

Chapter 6

User-oriented Hierarchical Scheduling for

EPON

6.1 Introduction

In Chapter 5, we presented a hybrid cycle scheme (HCS) for upstream bandwidth allocation among different ONUs in EPON and highlighted some of the properties that contribute to improved network performance. HCS distinguishes a small number of traffic classes at an ONU and accordingly supplies differentiated service to best satisfy distinct QoS requirements. However, it does not concern whether the traffic at an ONU originates from single or multiple users and thus does not provide protection and fairness between different users under an ONU. In this chapter, we move forward to design user-oriented hierarchical bandwidth allocation algorithms that aim to provide QoS delivery not only to aggregate traffic (i.e., traffic classes) of an ONU, but also to individual traffic of different users hosted by an ONU.

For this purpose, we adopt the main ideas of HCS for inter-ONU scheduling design but with modifications to adapt to the requirements of finer QoS support. While for intra-ONU scheduling, we design a credit-based scheduling approach to guarantee

fairness among different end-users. We also adopt the traffic classification recommendations in [BBC98] to classify network traffic into three classes: EF, AF, and BE.

In Section 6.2, we present different user-oriented hierarchical bandwidth scheduling algorithms (UHSA) that support DiffServ and guarantee fairness among end-users. Just like HCS, a transmission cycle is separated into a static part for high priority (HP) traffic for EF, to ensure guaranteed services and better performance, and an adaptive and dynamic part for low priority (LP) traffic, namely AF and BE, to adapt to the bursty nature of AF/BE traffic.

As discussed in Chapter 2, most algorithms of user-oriented hierarchical scheduling tend to use per-class-per-user queuing mechanism, i.e., queuing traffic of each class of a user in a separate buffer. This brings large scheduling overhead and queuing complexity with increased number of queues. In order to overcome this deficiency, we design a credit-based common queue (CCQ) for each class to enhance scheduling architecture and minimize average number of queues in an ONU. Meanwhile, we propose a transmission priority scheme for different queue groups in both EF and AF/BE timeslot. Together with the CCQ mechanism, it improves delay and delay variation performance of EF traffic by developing a feature of multiple transmission opportunities (M-opportunities) per-cycle for EF traffic.

In Section 6.3, comprehensive simulation experiments are presented to validate the effectiveness and efficiency of the proposed algorithms. Self-similar traffic is generated in the simulation as introduced in [TWS97,Kra03] to obtain an accurate and

realistic performance analysis. In all performance diagrams, the network load values are normalized to the link capacity.

6.2 User-oriented Hierarchical Bandwidth Scheduling Algorithms (UHSAs)

In an attempt to enhance EF performance, UHSAs adopt two-part frame partition strategy similar with that of HCS. A two-level hierarchical queuing architecture is designed to provide fairness and a CCQ mechanism is proposed for high-level class-based queuing in order to minimize average queue number. In particular, fairness here refers to guaranteed bandwidth allocation or excess bandwidth sharing in proportion to predetermined weights.

Figure 6.1: Illustration of EF cycle and AF cycle (a) UHSA-S. (b) UHSA-M.

6.2.1 Two-part Frame Partition Strategy

A frame (cycle) is partitioned into two parts as shown in Figure 6.1. The initial part has N fixed and orderly distributed timeslots, termed EF timeslots, corresponding to N ONUs ($ONU_1 \sim ONU_N$). We call this part as EF part or EF interval. This design of EF timeslots is the same as that of HCS where the size of each timeslot is determined

according to the maximum frame size and Service Level Agreement (SLA) between the subscriber and the service provider.

The second part is designed for AF and BE traffic, termed AF part or AF interval. We adopt two different distribution approaches in this part. The first approach divides this part into N orderly distributed timeslots, termed AF timeslots, each for an ONU. A whole set of AF timeslot is thus covered in a single frame, like that of EF timeslot. We call this approach as UHSA with single cycle (UHSA-S for short). For the other approach, the AF part is divided into a variable number of AF timeslots dependent on instantaneous queuing status of all ONUs. In this case, a whole set of AF timeslot may span multiple frames, i.e., an ONU gets an AF timeslot in multiple cycles. We name this approach as UHSA with multiple cycles (UHSA-M). UHSA-S and UHSA-M have different advantages in providing QoS. This will be discussed subsequently.

Now, we introduce the concept of multiple cycles. EF cycle, termed as T_{EF} , is defined as the time period between two consecutive beginnings of ONU₁ EF timeslot. This definition of EF cycle is the same as that in HCS where an EF cycle is equal to one frame size that is limited in a range of $[T_{EF}^{Min}, T_{EF}^{Max}]$. T_{EF}^{Min} is introduced to prevent light load bandwidth consumption problem while T_{EF}^{Max} is the upper limit of the EF cycle which is critical to delay performance of EF traffic. A reasonable T_{EF}^{Min} and T_{EF}^{Max} is needed in EPON configuration. Figure 6.1 illustrates the architecture of EF cycle for both UHSA-S and UHSA-M. An EF cycle comprises one EF timeslot for each ONU, and includes one AF timeslot for each ONU in UHSA-S while one or more AF timeslots for some ONUs in UHSA-M.

The other cycle is AF cycle, termed as T_{AF} , which is defined as the time period covering a complete set of AF timeslots. As shown in Figure 6.1, a T_{AF} spans the interval between two consecutive beginnings of ONU₁ AF timeslot. For UHSA-S, the n -th AF cycle gives the same length as the n -th EF cycle although the time coverage is different in the EF part. While for UHSA-M, an AF cycle may span several consecutive EF cycles. Figure 6.1 (b) shows an example that an AF cycle spans three EF cycles. Therefore, it is possible that not all ONUs are allocated an AF timeslot in an EF cycle for UHSA-M. This design of AF cycle aims to improve network utilization at heavy load for a large network of ONUs by reducing guard time wastage. This improves bandwidth but trades off delay performance of EF traffic as compared with UHSA-S approach.

Guaranteed bandwidth is provided to the EF class as a fixed timeslot is reserved for an ONU in each EF cycle ($\leq T_{EF}^{Max}$). The EF traffic of ONU_{*i*} in each EF cycle receives a fixed service in the timeslot size of:

$$W_i^{EF} = (T_{fixed} - NT_g) \times \frac{\phi_i^H}{\sum_k \phi_k^H} \times R_E = (T_{fixed} - NT_g) \times \phi_i^H \times R_E \quad (6.1)$$

Accordingly, each user j in ONU_{*i*} also receives a guaranteed EF service in each EF cycle in the timeslot size of:

$$W_{i,j}^{EF} = W_i^{EF} \times \frac{\phi_{i,j}^H}{\sum_k \phi_{i,k}^H} = W_i^{EF} \times \phi_{i,j}^H \quad (6.2)$$

The parameters used are listed in Table 6-1.

Since AF traffic requires statistical throughput guarantee and generally subscribers sign up network service with a bandwidth requirement, thus, fairness guarantee is also

an important issue in AF timeslot distribution. The minimum guaranteed AF timeslot size of ONU_i in each AF cycle is defined as:

$$W_i^{AF} = \begin{cases} (T_{dynamic}^{Max} - NT_g) \times \frac{\phi_i^{AF} + \phi_i^{BE}}{\sum_k (\phi_k^{AF} + \phi_k^{BE})} \times R_E \\ \quad = (T_{dynamic}^{Max} - NT_g) \times (\phi_i^{AF} + \phi_i^{BE}) \times R_E & \text{for UHSA-S} \\ M \times (T_{dynamic}^{Max} - NT_g) \times \frac{\phi_i^{AF} + \phi_i^{BE}}{\sum_k (\phi_k^{AF} + \phi_k^{BE})} \times R_E \\ \quad = M \times (T_{dynamic}^{Max} - NT_g) \times (\phi_i^{AF} + \phi_i^{BE}) \times R_E & \text{for UHSA-M} \end{cases} \quad (6.3)$$

where M is an integer. UHSA-M augments the minimum guaranteed AF timeslot of an ONU by M times of that of UHSA-S. In other words, the total size of AF timeslots in an AF cycle is maximized by M times. Accordingly, the AF cycle is fluctuating in a larger range and it may span several EF cycles at heavy network load. This feature makes UHSA-M more adaptive for bursty AF and BE traffic. It also improves bandwidth utilization since the guard time waste is reduced in an EF cycle.

Table 6-1: Parameters used for UHSAs formulization

N	Total number of ONUs
R_U, R_E	Line rate of user to ONU link and EPON link rate, respectively
$T_{EF}^{Max}, T_{EF}^{Min}$	Maximum and Minimum EF cycle, respectively
T_{fixed}	Steady EF part in an EF cycle, i.e., total EF timeslot plus NT_g
$T_{dynamic}^{Max}, T_{dynamic}^{Min}$	Maximum and minimum AF dynamic part in an EF cycle $T_{dynamic}^{Max} (T_{dynamic}^{Min}) = T_{EF}^{Max} (T_{EF}^{Min}) - T_{fixed}$
$\phi_i^H, \phi_i^M, \phi_i^L$	Weight of EF, AF, and BE of ONU _i among all ONUs, $\sum_k \phi_k^H = 1$ and $\sum_k (\phi_k^M + \phi_k^L) = 1$
$\phi_{i,j}^H, \phi_{i,j}^M, \phi_{i,j}^L$	Weight of EF, AF and BE of user j in ONU _i $\sum_k \phi_{i,k}^H = 1$ and $\sum_k (\phi_{i,k}^M + \phi_{i,k}^L) = 1$
$C_{i,j}^H, C_{i,j}^M, C_{i,j}^L$	Credit value of EF, AF, and BE class of user j in ONU _i
Q_i^H, Q_i^M, Q_i^L	CCQ of EF, AF, and BE, respectively of ONU _i . It also represents the queue size.
$Q_{i,j}^H, Q_{i,j}^M, Q_{i,j}^L$	Private queue of EF, AF, and BE, of user j in ONU _i , respectively. It also represents queue size.

$CoQ_{i,j}^H, CoQ_{i,j}^M, CoQ_{i,j}^L$	Number of bits in CCQ of EF, AF, and BE, of user j in ONU _{i} , respectively
$W_i^{EF}, W_{i,j}^{EF}$	EF timeslot size of ONU _{i} and Minimum EF timeslot guarantee of user j in ONU _{i}
$W_i^{AF}, W_{i,j}^{AF}$	Minimum guaranteed AF timeslot size of ONU _{i} and Minimum AF timeslot of user j in ONU _{i}
T_g	Guard Time between timeslots

Accordingly, we define a minimum guaranteed AF timeslot size for each user j of ONU _{i} as follows:

$$W_{i,j}^{AF} = W_i^{AF} * \frac{\phi_{i,j}^M + \phi_{i,j}^L}{\sum_k (\phi_{i,k}^M + \phi_{i,k}^L)} = W_i^{AF} * (\phi_{i,j}^M + \phi_{i,j}^L) \quad (6.4)$$

It is indicated in formulas (6.1) - (6.4) that minimum guaranteed bandwidth is provided to an ONU and its users in terms of EF timeslot and AF timeslot. Nonetheless, to provide better QoS and fairness to ONUs and end users, it is essential to design effective inter-ONU bandwidth scheduling and intra-ONU queuing and scheduling to deal with the unpredictable network traffic.

6.2.2 Credit-based Common Queue and Transmission Priority Scheme

The UHSA architecture is illustrated in Figure 6.2. The inter-ONU scheduling is executed in OLT to manage bandwidth (timeslot) distribution amongst ONUs, and the intra-ONU scheduling is implemented at each ONU to further distribute bandwidth allocated to an ONU amongst end users. Queuing and scheduling at low-level layer plays a crucial role in ultimate QoS fulfillment and fairness guarantees to end users. For ONU local queuing, a new two-level hierarchical queuing architecture is designed. The high-level queuing consists of three credit-based common queues

(CCQs) each for a traffic class, while the low-level queuing comprises a set of private queues (PQs) created in the manner of per-class-per-user. These queues are put into six groups: EF CCQ group, AF CCQ group, BE CCQ group, EF PQ group, AF PQ group, and BE PQ group. Each CCQ group includes a single high-level CCQ while each PQ group includes a set of user PQs.

Figure 6.2: Architecture of the User-oriented Hierarchical Scheduling.

Principally, the CCQ is introduced to serve two objectives. First, it enhances scheduling architecture and substantially improves scheduling efficiency. When the right timeslot arrives at an ONU, it firstly serves packet from a CCQ in FIFO manner before picking packet from the corresponding PQs using a credit-based fair scheduling method. As FIFO scheduling is much simpler, the CCQ mechanism saves much time for the ONU. In particular, a large percentage of packets is served using the common queues. This will make the scheduling efficient. Packets from PQs are

also scheduled. However, the number of active PQs is small. Second, the queuing management complexity is lowered by using less number of queues. CCQ uses credit-based method to queue the packets. The number of a user's packets being queued in a common queue is determined by the user's credit value of the corresponding class, not by traditional FIFO approach. This prevents the common queue being monopolized by a single user and effectively guarantees fairness to end users. Besides, this approach of common queue sharing helps to minimize the number of PQs in the ONU and thus lowers the queuing management complexity.

To maximize system efficiency, the maximum CCQ length needs to be deliberately specified for each traffic class. For EF CCQ Q_i^H , it is limited by:

$$Q_i^{H-Max} = W_i^{EF} = (T_{fixed} - NT_g) \times \phi_i^H \times R_E \quad (6.5)$$

Accordingly, each user j in ONU _{i} could queue theoretically as many as $W_{i,j}^{EF} (W_i^{EF} \times \phi_{i,j}^H)$ bits in the EF CCQ. The maximum EF common queue size Q_i^{H-Max} is equal to the fixed EF timeslot size of the ONU, thus all packets in EF CCQ are guaranteed to be served by the next EF cycle. Each user theoretically can queue as many as $W_{i,j}^{EF}$ bits in the common queue to obtain its share of EF bandwidth. As frame fragmentation is not allowed in EPON, a user may not always get its share of bandwidth in each cycle. To fairly share the common queues by end users, we develop a credit-based queuing mechanism for common queue sharing by all users as shown in Figure 6.3. Specifically, for EF CCQ, the operations are summarized as follows:

- Initialization. Set credit value $C_{i,j}^H = 0$ for each user j in every ONU _{i} .

<pre> Initialize(){ $C_{i,j}^H = 0$; $C_{i,j}^M = C_{i,j}^L = Q_i^{ML-Max}$; //Queues are classified into six groups: Q1, Q2, Q3 for common queue of EF, AF and BE Q4, Q5, Q6 for private queues of EF, AF and BE // Queue priority for the six groups. // EF timeslot: Q1>Q4>Q2>Q3>Q5>Q6 // AF timeslot Q2>Q1>Q4>Q3>Q5>Q6 } </pre>	<pre> Pick up a packet P from Qi (Start with higher priority and assure the leftover timeslot could carry this packet. For Q1~Q3, the head packet; for Q4~Q6, the head packet of the PQ with the largest credit value) If(such P exists) { Transmit P; t_current += trans_time(P); if(P from Q4) $C_{i,j}^H -= L(P)$ else if (P from Q5 or Q6) $C_{i,j}^{M(L)} -= L(P) / (\phi_{i,j}^M + \phi_{i,j}^L)$ } else t_current = t_nextEvent; } </pre>
<pre> ArrivalHandler(int i, int j, int class) {//packet arrival of User j in ONU i First check whether there is enough space for this packet. If not, drop packet(s) from queue(s) exceeding its guaranteed space, then drop lower priority packet(s) from private queues of j. Otherwise drop this packet and return. //For EF packet arrival if($Q_{i,j}^H == 0$ && $CoQ_{i,j}^H < W_{i,j}^H$ && $L(P) + Q_{i,j}^H \leq Q_i^{H-Max}$) { //Queue this packet in common queue $Q_{i,j}^H$ Q_CoEF(P); If($CoQ_{i,j}^H > W_{i,j}^H$) $C_{i,j}^H = (CoQ_{i,j}^H - W_{i,j}^H)$; } else//Queue in user j private queue $Q_{i,j}^H$ Q_EF(P); //For AF or BE packet arrival if($Q_{i,j}^{M(L)} == 0$ && $C_{i,j}^{M(L)} > 0$ && $L(P) + Q_{i,j}^{M(L)} \leq Q_i^{ML-Max}$) { //Queue this packet in common queue Q_CoAF(P);// Q_CoBE(P) $C_{i,j}^{M(L)} = L(P) / (\phi_{i,j}^M + \phi_{i,j}^L)$; } else //queue in private queue Q_AF(P);//Q_BE(P) } </pre>	<pre> //Update credit value and packet shifting //For EF class if($Q_{i,j}^H != 0$ && $CoQ_{i,j}^H + HeadP \leq W_{i,j}^H$ && $L(headP) + Q_{i,j}^H \leq Q_i^{H-Max}$) Shift(P); //shift the EF packet from PQ to CCQ; else if($Q_{i,j}^H != 0$ && $C_{i,j}^H == 0$ && $CoQ_{i,j}^H < W_{i,j}^H$ && $L(headP) + Q_{i,j}^H \leq Q_i^{H-Max}$) { Shift(P); $C_{i,j}^H = (CoQ_{i,j}^H - W_{i,j}^H)$; } if($Q_{i,j}^H == 0$ $CoQ_{i,j}^H \leq W_{i,j}^H$) $C_{i,j}^H = 0$; else $C_{i,j}^H = \max\{C_{i,j}^H\}$ //AF or BE class after EF timeslot transmission if($C_{i,j}^{M(L)} > 0$) { Shift(P); $C_{i,j}^{M(L)} = L(P) / (\phi_{i,j}^M + \phi_{i,j}^L)$; } //AF or BE class after AF timeslot transmission while($Q_{i,j}^{M(L)}$ not full) { shift(P); //a head AF/BE packet from PQ with the highest credit is shifted to the common queue. $C_{i,j}^{M(L)} = L(P) / (\phi_{i,j}^M + \phi_{i,j}^L)$; } //AF and BE credit update after AF timeslot transmission $C_{i,j}^{M(L)} = Q_i^{ML-Max} - CoQ_{i,j}^{M(L)} / (\phi_{i,j}^M + \phi_{i,j}^L)$; } </pre>
<pre> Schedule(){ t_start=t_current=GATE.start_time; t_end=t_start+GATE.trans_time; WHILE(t_current<=t_end) { </pre>	<pre> } </pre>

Figure 6.3: Pseudo code for UHSA.

- EF packet arrival. When an EF packet P of user j of ONU_i arrives, if j 's EF PQ $Q_{i,j}^H$ is null and its EF bits in the common queue $CoQ_{i,j}^H$ is less than $W_{i,j}^{EF}$ as well as the space of the EF CCQ is available, P queues in the CCQ, otherwise, it queues in $Q_{i,j}^H$. If P queues

in CCQ and leads to $CoQ_{i,j}^H$ exceeding $W_{i,j}^{EF}$, then $C_{i,j}^H$ deducts the extra part $(CoQ_{i,j}^H - W_{i,j}^{EF})$.

- EF packet departure. 1) first, packets in EF CCQ are served. 2) When EF CCQ is null, an EF packet P is selected from the PQ with the largest $C_{i,j}^H$. $C_{i,j}^H$ deducts the packet length $L(P)$ after the packet departure. This is called as credit-based fair scheduling for PQ packet scheduling.
- Packet Shifting (a). After each EF/AF timeslot transmission, packet shifting from PQ to CCQ is performed if possible. A HoL packet P of $Q_{i,j}^H$ will be shifted from the PQ to the EF CCQ, if two conditions are satisfied 1) the CCQ has available space 2) $CoQ_{i,j}^H + L(P)$ is not larger than $W_{i,j}^{EF}$ or $CoQ_{i,j}^H$ is less than $W_{i,j}^{EF}$ and $C_{i,j}^H = 0$. If the condition is satisfied and $CoQ_{i,j}^H + L(P) > W_{i,j}^{EF}$, then $C_{i,j}^H$ is updated as $C_{i,j}^H = C_{i,j}^H - (CoQ_{i,j}^H + L(P) - W_{i,j}^{EF})$.
- Packet Shifting (b). If the EF CCQ is still large enough, an EF packet P is selected from the PQ with the largest $C_{i,j}^H$. $C_{i,j}^H$ deducts $L(P)$ after packet shifting.
- Finally, $C_{i,j}^H$ is updated. For user with $CoQ_{i,j}^H \leq W_{i,j}^{EF}$ or null PQ, set $C_{i,j}^H = 0$. For all other users, the credit value deducts the largest credit counter among all non-zero EF PQs.

For AF and BE CCQ, $Q_i^M (Q_i^L)$ is limited by:

$$Q_i^{ML-Max} = W_i^{AF} \quad (6.6)$$

That is Q_i^{ML-Max} is as large as the minimum guaranteed AF timeslot size of the ONU.

Similarly, each user j of ONU_i could also queue as many as $W_{i,j}^{AF}$ ($W_{i,j}^{AF} = W_i^{AF} * (\phi_{i,j}^M + \phi_{i,j}^L)$) bits in the AF and BE CCQ. To enable fair sharing of CCQ, we also develop a credit-based queuing mechanism for AF and BE CCQ sharing as

shown in Figure 6.3. AF and BE have the same CCQ sharing approach. The AF CCQ operations are described as follows.

- Initialization. Set $C_{i,j}^M = Q_i^{ML-Max}$.
- AF packet arrival. When an AF packet P of user j of ONU_i arrives, it queues in AF CCQ if $Q_{i,j}^M = 0$, $C_{i,j}^M > 0$, and AF CCQ could accommodate the AF packet. Otherwise, it queues in j 's AF PQ $Q_{i,j}^M$. In this case, $C_{i,j}^M$ deducts $L(P)/(\phi_{i,j}^M + \phi_{i,j}^L)$ to account for the credit expense of the packet since $C_{i,j}^M$ is initialized Q_i^{ML-Max} at the beginning. This guarantees fairness among end users.
- AF packet departure. 1) AF CCQ is served first. 2) When AF CCQ is null, an AF packet P is selected for service from PQ with the largest $C_{i,j}^M$, $C_{i,j}^M$ deducts $L(P)/(\phi_{i,j}^M + \phi_{i,j}^L)$ accordingly.
- Packet shifting. After each EF/AF timeslot transmission, packet shifting from PQ to CCQ is executed if possible. 1) After EF timeslot transmission, a HoL AF packet P from j 's PQ is shifted to the AF CCQ if $C_{i,j}^M > 0$ and the CCQ could accommodate it. 2) After AF timeslot transmission, an AF packet P will be selected to shift to CCQ from the PQ with the largest $C_{i,j}^M$. In both cases, $C_{i,j}^M$ deducts $L(P)/(\phi_{i,j}^M + \phi_{i,j}^L)$.
- Finally, $C_{i,j}^M$ is updated as $C_{i,j}^M = Q_i^{ML-Max} - CoQ_{i,j}^M / (\phi_{i,j}^M + \phi_{i,j}^L)$.

To improve bandwidth utilization at light load when EF traffic cannot consume the entire EF timeslot, AF and BE frames will be transmitted if the leftover EF timeslot is large enough to transmit the head-of-line frame of AF/BE queues. The transmission priority of different queue groups in EF timeslot is defined in descending order as: EF CCQ, EF PQ, AF CCQ, BE CCQ, AF PQ, BE PQ. Both of the two EF groups always

have higher priority than other class queue groups in EF timeslot. This is in line with the design objective of EF timeslot in order to provide guaranteed service to EF traffic. Thus, EF traffic should not be delayed by other class traffic in EF timeslot.

Similarly, EF frame can be transmitted in AF timeslot. Besides utilizing fully the bandwidth available, it is also crucial to improve delay and delay variation performance for EF traffic. To ensure both objectives, we propose transmission priority for queue groups in AF timeslot in descending order as follows: AF CCQ, EF CCQ, EF PQ, BE CCQ, AF PQ, BE PQ. This proposal is based on several considerations. First, AF traffic demands guaranteed throughput and cannot be starved by EF traffic, thus AF CCQ should have the highest priority in AF timeslot. According to AF CCQ design, the maximum size of AF CCQ is just as large as the minimum guaranteed AF timeslot size, thus packets in AF CCQ will be delivered in limited time. This ensures guaranteed throughput for AF traffic to meet its statistical bandwidth requirement. Second, EF traffic has already secured its lion share of bandwidth in EF timeslot. There is no need for it to be assigned the highest priority in AF timeslot. Nonetheless, the two EF traffic groups still occupy the second and the third priority, respectively in AF timeslot. Unless AF CCQ is full and there is no extra bandwidth allocated for the ONU, EF traffic has a high chance to transmit in AF timeslot. This means EF traffic statistically has more than one transmission opportunities in each EF cycle, which substantially shortens the statistical transmission cycle of EF traffic yielding much improved delay and delay variation performance. We call this style of timeslot scheduling for EF traffic as M-opportunities per-cycle solution which is similar with that in HCS. For EPON system, since the cycle time cannot be too small to prevent huge bandwidth cost by guard time

and GATE message, it is desirable that EF traffic has more than one transmission opportunity in one cycle to improve its performance. The algorithms of UHSAs provide M-opportunities per-cycle for EF traffic to meet this requirement. This is different from previous approaches which make use of S-opportunity per-cycle for EF traffic, thus encountering large delay due to long waiting time for each EF service and large delay variation due to long and fluctuating cycle time. Finally, BE CCQ has higher priority than AF PQ but lower than EF PQ. Since BE traffic has no QoS requirement, it is reasonable to put EF traffic ahead of it for urgent EF transmission. Putting AF PQ behind BE CCQ is intended to alleviate AF traffic monopolizing the whole AF timeslot. From the design of AF CCQ, it is possible that AF traffic may dominate the AF timeslot as the maximum size of AF CCQ is equal to the minimum guaranteed AF timeslot size of the ONU. On the other hand, traffic policing can be used to control the amount of traffic of each user to ensure it conforms to its SLA.

6.2.3 Report and Grant Mechanism

The OLT maintains a table of ONUs, including the fixed timeslot size, minimum guaranteed AF timeslot size and the latest buffer occupancy information of AF and BE classes, of each ONU. An ONU sends REPORT message to the OLT at the end of each EF and AF timeslot, reporting the queue size information of the AF and BE traffic. OLT determines the AF timeslot size according to the latest buffer occupancy information and sends GATE messages to ONUs maximum RTT time earlier than the start time of the first AF timeslot. This is to prevent RTT idle time and at the same time use the most updated buffer occupancy information. AF timeslots are granted to ONUs orderly in an AF cycle. For UHSA-M, this requires the OLT to remember the

latest ONU that has been granted AF timeslot since a whole set of AF timeslot may span several EF cycles. For UHSA-S, a whole set of AF timeslot is always included in the same EF cycle, there is no need to remember the latest served ONU. As EF timeslot and its relative position for an ONU is fixed in each EF cycle, OLT sends EF grant messages to each ONU ahead of the beginning of the next EF cycle to keep the ONU active. To save GATE message costs, the EF grant for the next EF cycle and the AF/BE grant for the current cycle can be delivered in a single GATE message to the destined ONU if the ONU has AF timeslot in current EF cycle.

As discussed earlier, an ONU_{*i*} is reserved a fixed timeslot of W_i^{EF} for EF traffic transmission. So, a fixed timeslot size of W_i^{EF} is granted to ONU_{*i*} in each EF cycle. The ONU further fairly distributes the bandwidth to end users, which is implemented by using the proposed EF CCQ mechanism and credit-base fair scheduling.

Next, we discuss REPORT and GRANT mechanism for AF and BE class. At the end of each EF and AF timeslot, a REPORT message is generated that contains three reports, i.e., AF report R_i^{AF} , BE report R_i^{BE} , and extra report R_i^{Extra} , respectively. They are determined using the following rule:

$$\begin{cases} R_i^{AF} = \min\{Q_i^{ML-Max}, Q_i^{AF} + \sum_k Q_{i,k}^{AF}\} \\ R_i^{BE} = \min\{Q_i^{ML-Max}, Q_i^{BE} + \sum_j Q_{i,k}^{BE}\} \\ R_i^{Extra} = Q_i^{AF} + Q_i^{BE} + \sum_k (Q_{i,k}^{AF} + \sum_k Q_{i,k}^{BE}) - R_i^{AF} - R_i^{BE} \end{cases} \quad (6.7)$$

R_i^{AF} represents AF traffic request but is capped by Q_i^{ML-Max} , the maximum size of AF CCQ (also the minimum guaranteed AF timeslot size). Thus, it represents the maximum queue size of AF CCQ at the moment. This part of request is always satisfied to guarantee AF bandwidth if an ONU is allocated AF timeslot in the next EF

cycle. Similarly, R_i^{BE} represents the maximum BE CCQ size. R_i^{Extra} represents the left-over request.

OLT determines the granted AF timeslot size according to the latest buffer occupancy information, i.e., R_i^{AF} , R_i^{BE} , and R_i^{Extra} of all ONUs. For UHSA-S, the grant algorithm works in following steps:

Step1. Initialize left-over available bandwidth $B_{avail} = (T_{dynamic}^{Max} - NT_g) * R_E$, and set grant size

$$G_i = 0 \text{ for all ONUs.}$$

Step2. All R_i^{AF} is fully satisfied, i.e., $G_i = G_i + R_i^{AF}$, $B_{avail} = B_{avail} - \sum_k R_k^{AF}$.

Step3. If $B_{avail} \geq \sum_k R_k^{BE}$, all R_i^{BE} is fully satisfied, then $G_i = G_i + R_i^{BE}$,

$$B_{avail} = B_{avail} - \sum_k R_k^{BE} \text{ otherwise } G'_i = \min\{R_i^{BE}, W_i^{AF} - R_i^{AF}\}, G_i = G_i + G'_i, B_{avail} = B_{avail}$$

$$- \sum_k G'_k, R_i^{BE} = R_i^{BE} - G'_i, \text{ the left-over } B_{avail} \text{ is further distributed among ONUs}$$

according to R_i^{BE} in Max-Min allocation method, and break;

Step4. If $B_{avail} \geq \sum_k R_k^{Extra}$, $G_i = G_i + R_i^{Extra}$, $B_{avail} = B_{avail} - \sum_k R_k^{Extra}$, otherwise, Max-Min

allocation among ONUs according to R_i^{Extra} , and break;

Step5. All requests are satisfied, then check whether the minimum EF cycle is satisfied, i.e.,

$$B_{avail} \leq (T_{EF}^{Max} - T_{EF}^{Min}) * R_E. \text{ If not, then distribute excess bandwidth fairly to ONUs}$$

according to their weight to ensure minimum EF cycle:

$$G_i = G_i + (B_{avail} - (T_{EF}^{Max} - T_{EF}^{Min}) * R_E) * (\phi_i^M + \phi_i^L) \quad (6.8)$$

R_i^{AF} is always satisfied for all ONUs to provide guaranteed bandwidth for AF traffic. For R_i^{BE} , if not all ONUs are satisfied, each ONU firstly gets its guaranteed

share of G_i' . This is to fully satisfy under-loaded ONUs while the over-loaded ONUs fairly share the savings from under-loaded ONUs using max-min allocation scheme.

For UHSA-M, the grant algorithm works as follows:

Step1. Initialize left-over available bandwidth $B_{avail} = T_{dynamic}^{Max} * R_E$, and set grant size $G_i = 0$

for all ONUs. Assume $i = ONU_{current}$ is the first ONU for service.

Step2. $G_i += R_i^{AF} + R_i^{BE}$ and $B_{avail} -= (R_i^{AF} + R_i^{BE} + T_g * R_E)$.

Step3. $i++$, if $B_{avail} \geq (R_i^{AF} + R_i^{BE} + T_g * R_E)$, go to step 2. Otherwise, break;

Step4. If all ONUs are satisfied, then check whether the minimum EF cycle is satisfied like the step 5 of UHSA-S and distribute excessive bandwidth if needed.

Step5. Update current ONU, $ONU_{current} = i$.

In UHSA-M, R_i^{AF} and R_i^{BE} are always satisfied if ONU_i is accepted in the current EF cycle. Since R_i^{AF} and R_i^{BE} can be very large dependent on the value of M in equation (6.3), it is likely that only part of the ONUs are allocated AF timeslot in an EF cycle. Note that, R_i^{Extra} is not used in UHSA-M. R_i^{Extra} in UHSA-S is helpful to improve system utilization and total traffic delay by allowing more packets being transmitted in the current cycle rather than delaying them to the next cycle. In UHSA-M, as R_i^{AF} and R_i^{BE} is augmented M times, more traffic of an ONU is allowed in an EF cycle. Thus the benefit of R_i^{Extra} is marginal.

6.3 Performance Evaluation

In this section, we evaluate the performance of the proposed UHSAs through extensive simulation experiments. For the purpose of comparison and to show the

advantages of M-opportunities per-cycle solution and CCQ mechanism, experiments are also carried out using traditional approaches. An event-driven frame-based simulation model was developed using the C programming language. We consider an EPON architecture with 16 ONUs, each at a distance of 20Km from the OLT, corresponding to an RTT of 0.2 ms. Each ONU hosts 8 end users. The link rate R_U and R_E are set to be 1 Gbps. The guard time and the REPORT message transmission time are grouped together to be 1 μs . EF traffic is generated from a Poisson process, and EF packet size is fixed at 70 bytes. To closely emulate the self-similar property of AF and BE traffic, we generate AF and BE traffic for each user by using a Pareto-distributed ON-OFF source [TWS97]. The traffic arriving at an ONU is thus an aggregation of 8 Poisson process sources and 16 Pareto-distributed ON-OFF sources. According to IEEE 802.3, the size of AF and BE packet is uniformly distributed between 64 and 1518 bytes. The traffic profile is designed to be as follows: 20% EF traffic, 40% AF traffic and 40% BE traffic. Without loss of generality, the weights of ONUs are made equal and the weights of users in each ONU are made uniformly and orderly distributed as {1,1,2,2,4,4,5,5}. Each ONU has a finite buffer of 8 Mbits. T_{EF}^{Max} is 2.00ms and T_{EF}^{Min} is 0.56ms. For UHSA-M, M is set to be 3. Furthermore, in order to overcome the uncertainty of self-similar traces and get conclusive results, each point of the results is obtained as an average of multiple repeated experiments.

6.3.1 Delay and Delay Variation

The average queuing delay (AD) for EF class is illustrated in Figure 6.4 (a). To demonstrate the effect of M-opportunities per-cycle solution in UHSAs in improving

(a) AD for EF traffic class.

(b) AD for AF and BE class.

Figure 6.4: Simulation results of AD for EF, AF, and BE Class.

EF performance, we design a S-opportunity per-cycle solution based on Regular scheduling algorithm for comparison. The Regular scheduling is derived from UHSA-S but not allowing EF traffic to be transmitted in AF timeslot and not allowing AF/BE traffic to be transmitted in EF timeslot. The Regular scheduling algorithm represents

previous approaches [XJJ04,BSG05] of employing S-opportunity per-cycle solution. Both UHSA-S and UHSA-M achieve significant improvement in delay performance than that of Regular scheduling, with UHSA-S showing over 41.6% advantage and UHSA-M showing over 40.2% advantage at all network loads, respectively. We conclude that the UHSAs algorithms featuring M-opportunities per-cycle for EF traffic can significantly improve EF AD performance as compared with previous approaches featuring S-opportunity per-cycle. This is attributed to allowing EF traffic being transmitted in AF cycle which substantially shortens the statistical EF transmission cycle. On the other hand, UHSA-S achieves better performance by 0.1%~10.5% at low load (0.1~0.85) and by 15.5~42.2% at high load (0.9~1.0) than that of UHSA-M. This is because the maximum AF cycle in UHSA-M is extended much by setting $M = 3$ such that an AF cycle may span several EF cycles, especially at heavy load. Therefore, UHSA-M generates much less frequent EF transmission opportunity than UHSA-S where AF cycle has the same frequency as EF cycle. This makes UHSA-S performing better than UHSA-M in EF delay, particularly at heavy load.

The average queuing delay (AD) for AF class is presented in Figure 6.4 (b). UHSA-S and UHSA-M achieve very close AF AD at load 0.1~0.95 since the average AF cycle has little difference at nonsaturated load. However, at saturated load 1.0, UHSA-S is 61.8% lower than that of UHSA-M. This is largely due to the sharp increase of average AF cycle in UHSA-M at saturated load condition for its extended maximum AF cycle. UHSA-S and UHSA-M outperform Regular scheduling in the range of 12.6%~40.0% and 22.5%~40.3% at load 0.1~0.9, respectively. At load 0.95-1.0, UHSA-S is slightly higher by 6.3%~1.1% while UHSA-M scores marginal 0.5%

higher at load 0.95 but 164.6% higher at load 1.0 as compared with Regular scheduling. This indicates that allowing EF and AF/BE to be transmitted in both EF and AF timeslot is beneficial not only EF AD but also AF AD at nonsaturated load. This is attributed to efficient utilization of EF timeslot. At light load, EF traffic cannot consume the whole EF timeslot; the leftover is used to carry AF/BE traffic so the AF delay is reduced. At heavy load, allowing EF traffic to transmit in AF timeslot and with transmission priority lower than AF traffic in CCQ but higher than in PQ bring worse AF AD for UHSAs than Regular scheduling. For UHSA-M, the sharp increase of average AF cycle further deteriorates its AF AD at saturated load.

The AD for BE class is also shown in Figure 6.4 (b). UHSA-S and UHSA-M produce similar BE AD and outperform Regular scheduling over 20.8% at load 0.1~0.85. Again, this is attributed to efficient utilization of EF timeslot. At heavy load 0.95-1.0, UHSA-S is higher over 114.9% than Regular scheduling while UHSA-M is still lower by 44.3%~65.7% than Regular scheduling. This is because part of BE bandwidth is preempted by EF traffic in UHSAs at heavy load but not in Regular scheduling. Thus UHSA-S achieves worse BE AD than Regular scheduling. While for UHSA-M, it benefits from the extended AF cycle method by reducing guard time cost and unused timeslot remainder and thus obtains much higher throughput at heavy load. Therefore, UHSA-M has minimal congestion situation and results in better BE AD performance.

In conclusion, the M-opportunities per-cycle solution of UHSAs benefits greatly EF AD, achieving its major objective. Meanwhile, AF and BE AD are also improved substantially in most cases largely attributing to efficient EF timeslot utilization.

Figure 6.5: Simulation results of DV for EF and AF Class.

Figure 6.5 shows the performance of delay variation (DV) for EF traffic. Again, UHSA-S and UHSA-M present superior performance, with UHSA-S lowering DV over 23.5% and UHSA-M lowering DV over 22.6% as compared with Regular scheduling at all network loads. The improvement is attributed to the M-opportunities per-cycle solution in UHSAs, that statistically offers more than one EF transmission opportunities in each EF cycle. This helps to shorten EF transmission cycle and thus smoothes EF delay to produce small DV. In addition, UHSA-S achieves better DV than UHSA-M, with about 0.1%~12.7% improvement at low load (0.1~0.85) as well as with 19.6%~37.9% big improvement at high load (0.9-1.0), respectively. This is because the average AF cycle is quite close for UHSA-S and UHSA-M at light load, while at heavy load the AF cycle of UHSA-M sharply increases due to its extended maximum AF cycle which yields averagely less EF transmission opportunity per EF cycle. Thus, the EF delay fluctuates more in UHSA-M resulting in large DV. In

conclusion, the UHSAs achieve significant improvement in EF DV performance by the feature of M-opportunities per-cycle solution.

The DV for AF class is also presented in Figure 6.5. At light load 0.1~0.2, the three approaches obtain approximate AF DV due to similar and small EF cycle. While at medium load 0.3-0.85 UHSA-S shows advantage by 9.6%~21.9% and at load 0.3-0.9 UHSA-M shows advantage by 14.7%~28.3% as compared with Regular scheduling. This attributes to efficient EF timeslot utilization that leads to reduced EF cycle and less fluctuated AF delay. At load 0.95-1.0, UHSA-S obtains slightly higher AF DV than Regular scheduling due to lower AF PQ transmission priority in AF timeslot. While UHSA-M obtains slightly better result at load 0.95 owing to its adaptive AF cycle and much worse result at load 1.0 due to sharp increase of AF cycle that yields fluctuated AF AD.

Figure 6.6: Throughput of different classes.

6.3.2 Throughput and Fairness

Now we study the effect of extended AF cycle on network throughput performance.

Figure 6.6 shows the normalized throughput with respect to theoretical total

throughput for EF, AF, and BE classes of UHSA-S and UHSA-M. Among them, the normalized throughputs of EF (lower line) and AF (upper line) are nearly identical for UHSA-S and UHSA-M. EF class has reserved bandwidth in EF timeslot and has higher priority than AF PQ and BE class in AF timeslot, thus EF class fully receives its required bandwidth and scores 0.2 of normalized throughput at all loads in both algorithms. The design of AF CCQ is to provide guaranteed bandwidth for AF traffic. The AF traffic could take up the whole guaranteed bandwidth (intended for both AF and BE), thus AF class also receives total required bandwidth and scores 0.4 of normalized throughput at all loads. The main difference is on BE throughput at load 1.0 where the network is congested. In this case, BE class cannot fully get its required bandwidth. The normalized BE throughput is 0.355 for UHSA-S while 0.390 for UHSA-M, with real throughput difference around 34.6 Mbps. The better BE throughput of UHSA-M is due to its extended AF cycle approach which saves guard time waste and unused timeslot remainder by reducing AF timeslot number in an EF cycle. In summary, UHSA-M could provide more bandwidth by extending AF cycle, but with the tradeoff of producing larger EF delay and delay variation as compared with UHSA-S.

Next, we demonstrate the fairness performance of the UHSA protocols. According to the designs of EF and AF timeslot by formulas (6.1) - (6.4), CCQ, and credit-based design of common queue and PQ fair scheduling, we measure fairness by two scenarios: 1) whether a user can always secure its guaranteed throughput regardless of the behavior of other users. 2) whether two users of an ONU share excessive bandwidth by their weights under saturated condition. For this purpose, we measure

and compare the throughput of user 3 in ONU_1 (termed $U_{1,3}$), user 8 in ONU_1 ($U_{1,8}$), and user 3 in ONU_2 ($U_{2,3}$). $U_{1,3}$ and $U_{1,8}$ are configured to generate 50 Mbps fixed bandwidth, all other users generate traffic in the normal way, i.e., varying according to the load. We consider fairness of EF and AF bandwidth among different users and also fairness between EF and AF class. BE class is ignored here as it has no guaranteed bandwidth. Thus, traffic profile is 20% EF traffic and 80% AF traffic for each user. Furthermore, we use Poisson process model for AF traffic generation, like that of EF traffic. This is to eliminate the burstiness effect on bandwidth and obtain more accurate result for each user. The simulation is based on UHSA-S. Nonetheless, the result is applicable to UHSA-M as similar scheduling algorithm is used.

Figure 6.7: Throughput of test users.

We observe in Figure 6.7 that EF and AF throughputs of $U_{2,3}$ grow linearly with its demand at the loads of 0.7~1.0. This verifies that the OLT high-level scheduling is fair in distributing bandwidth among different ONUs. An ONU is assured to get its guaranteed bandwidth without being affected by other congested ONUs. At the loads

of 0.7~0.8, both EF and AF throughputs of $U_{1,3}$ and $U_{1,8}$ are nearly the same as their base generating rate, as the network is not fully loaded and ONU_1 could utilize the excessive bandwidth saved from other under-loaded ONUs. At the load of 0.9, the network is congested due to high demand of ONU_1 . In this case, all other under-loaded ONUs will get fully their demands, while ONU_1 will get bandwidth less than its demand. From the design of AF CCQ and transmission priority scheme in AF timeslot, AF traffic in CCQ has higher priority for bandwidth at most Q_i^{ML-Max} in an AF timeslot (guaranteed bandwidth) while EF traffic has higher priority for the left-over bandwidth. This is confirmed by the simulation result that EF throughput of $U_{1,3}$ and $U_{1,8}$ maintains around 10 Mbps at load 0.9. On the other hand, since the weight of $U_{1,8}$ is 5 while it is 2 for $U_{1,3}$, $U_{1,8}$ could get up to 5/7 of the left-over excessive bandwidth. As in Figure 6.7, $U_{1,8}$ obtains fully its demand with nearly 40 Mbps throughput, while $U_{1,3}$ obtains partially its demand at about 21.80 Mbps. This shows AF fairness among different users. At the loads of 1.0 and 1.1, the network is fully loaded, ONU_1 could not get excessive bandwidth from other ONUs, thus EF and AF throughput of each user obtains only its guaranteed bandwidth.

6.3.3 Scheduling Efficiency and Queuing Number

One of the objectives of CCQ mechanism is to improve scheduling efficiency. Since the scheduling complexity of picking up a packet from CCQ is $O(1)$ while it is $O(n)$ from a private queue, where n is the number of active queues of the class in service. Thus, we examine the percentage of packets served from CCQs among all packets (including from CCQs and PQs) to evaluate CCQ behavior. The higher percentage

that packets are served from CCQs, the more efficient the CCQ mechanism is. It is shown in Figure 6.8 that nearly 100% of EF packets are served from EF CCQ for both UHSA-S and UHSA-M at the loads of 0.1~0.7 while at the loads of 0.8~0.1 the percentage drops a little but still over 95.1% and 97.4% for UHSA-M and UHSA-S, respectively. Hence, the scheduling efficiency is improved greatly for EF traffic. The percentage drop at high load is due to very limited EF frames allowed to inject CCQ for each user in an EF cycle that causes more frames served from PQ with increasing load. For AF traffic, the percent is 77.9%~93.2% for UHSA-M and 76.0%~97.8% for UHSA-S, respectively, while for BE traffic, it is 92.1%~95.7% for UHSA-M and 82.6%~100.0% for UHSA-S. Thus, the scheduling efficiency improvement is also significant for AF and BE traffic. Note that, UHSA-M's performance dips at saturated load. This is due to sharp increase of AF cycle such that more frames will be served from PQ besides limited frames from CCQ. In summary, the CCQ mechanism substantially improves scheduling efficiency in ONU, which facilitates ONU to transmit packets at high speed matching the rate of 1Gbps.

Figure 6.8: Percent of packets served from CCQs.

The other objective of CCQ is to lower queue management complexity by reducing average number of queues (AQ) in the ONU system. For comparison, we also conduct experiments for a Regular scheduling which is derived from UHSA-S but without CCQ. Figure 6.9 presents the AQ performance. UHSA-S shows a reduction of 40.6%~54.9% at the loads of 0.3~0.9. UHSA-M shows a reduction of 41.1%~58.9% at the loads of 0.3~0.9. This shows that CCQ mechanism substantially reduces average number of queues in an ONU, thus effectively lowering queue management complexity.

Figure 6.9: Average number of queues (AQ).

6.4 Summary

This chapter studies dynamic bandwidth allocation with differentiated services support to end users in EPON networks. We propose a collection of innovative credit-based hierarchical scheduling algorithms (UHSA-S and UHSA-M) for the purpose. Two-part frame partition approach is adopted for inter-ONU scheduling. While for intra-ONU scheduling, a credit-based scheduling approach is proposed to guarantee

fairness among end users. On the other hand, to improve scheduling efficiency and lower queuing management complexity, we design a novel credit-based common queue (CCQ) for each class to enhance scheduling architecture and reduce average number of queues. We also propose new transmission priority schemes for different queue groups in EF and AF timeslots, in order to improve EF delay and delay variation performance by offering M-opportunities per-cycle solution for EF class, while guaranteeing bandwidth for AF class. Extensive simulation experiments have been conducted which show: 1) the UHSA algorithms with M-opportunities per-cycle solution significantly improve delay and delay variation performance by reducing AD over 40.2% and DV over 22.6% as compared with traditional S-opportunity per-cycle solutions. 2) the CCQ mechanism substantially enhances scheduling efficiency with a large percent of packets served from CCQ groups and lowers queue management complexity with a reduction over 40.6% of AQ at the loads of 0.3-0.9. These improvements are significant.

Chapter 7

Conclusion and Future Work

In this thesis, our focus is to study QoS provisioning over IP/MPLS high speed networks and EPON access networks. In particular, we design and analyze various aggregate scheduling algorithms for end-to-end QoS delivery for aggregation-based networks and dynamic bandwidth allocation (DBA) algorithms for EPON networks. In this chapter, we conclude the thesis by summarizing the major contributions and suggest some potential directions for future research.

7.1 Summary

We study the aggregate scheduling in Chapter 3 for providing deterministic end-to-end QoS using the concept of route interference (*RI*). There has been some work done on RI-based aggregate scheduling. However, previous works tend to confine to a simple scheduling scheme and study the traffic in a single-class environment or the highest priority traffic in a multi-class environment. This is rather restrictive. The assumption of a simple network model restricts the application of the results to complex networks, where packets belonging to the same class may queue in a number of different buffers.

In order to extend RI-approach into practical networks, we first propose a new service scheme named as Latency-Rate Max-Min Service Curve (LRMMSC) to provide service. The general nature of LRMMSC enables a wide range of implementations covering many known scheduling disciplines and thus makes the results applicable to heterogeneous scheduling networks. Second, we introduce a general network model with no constraints on whether packets of the same class queue in one or more buffer(s) and how they queue in the buffer, except that each buffer receives a service of LRMMSC. These new features make our results applicable to practical and complex networks, such as MPLS networks. In addition, we have derived a concise expression of end-to-end delay bound. The simplicity of the results implies its usefulness as minimum information is needed to make use of the results. This is highly attractive. Simulation experiments are conducted to verify the LRMMSC model. Good general agreements are obtained between analytical and simulation results. The advantage of LRMMSC scheme in providing maximum end-to-end delay is also demonstrated.

In Chapter 4, we investigate aggregate scheduling by designing and analyzing various fair traffic aggregators. The motivation is that aggregate scheduling of FIFO aggregation induces poor network utilization due to the increase of traffic burstiness from hop to hop, while aggregate scheduling with fair aggregation could eliminate this problem. We first propose a guaranteed rate fair aggregator (GRFA) for traffic aggregation and derive end-to-end delay bound for a network of aggregate scheduling that incorporates a guaranteed rate (GR) scheduler and a GRFA (termed GR+GRFA). By analytical comparison, it shows that (GR+GRFA) achieves better performance in end-to-end delay bound contributed by the design of GRFA. Also, the GRFA which is

constructed from a GR scheduler inherits the advantage of wide-range implementations of GR that make it suitable for a heterogeneous network environment.

We then move forward to develop a fair aggregation capable guaranteed rate (FA-GR) scheduling and a priority shaper (P-shaper) for aggregate scheduling. FA-GR performs packet scheduling as general GR scheduling, while executing flow aggregation and/or separation if needed. P-shaper is a deadline-based shaper, which controls the output to obey an arrival curve while delivering packets by the order of their deadline carried in their headers from the upstream FA-GR server. FA-GR approach achieves the same level of end-to-end delay bound as that of the (GR+GRFA). However, it has relatively lower complexity and implementation cost.

In Chapter 5, we address dynamic bandwidth allocation with differentiated services (DiffServ) support in EPON networks. In order to provide guaranteed service to high priority traffic while accommodating bursty medium and low priority traffic, we propose a new Hybrid Cycle Scheme (HCS). The high priority traffic is transmitted in fixed timeslots at fixed positions in an EF cycle while the medium and low priority traffic is transmitted in variable timeslots in an adaptive dynamic AF cycle. Premium service is guaranteed to EF traffic through fixed timeslot approach. Meanwhile, the adaptive design of the AF cycle effectively improves network utilization and performance by reducing the cost of guard time and timeslot remainder. A novel feature of multiple transmission opportunities (M-opportunities) per-cycle for high priority traffic is developed using a suitable local buffer management scheme. This is significant in improving delay and delay variation performance. In particular, HCS

reduces average delay by a range of 20.0% to 95.6% and delay variation by a range of 12.8% to 85.4% at different network loads as compared with traditional approaches. This is a notable improvement.

In Chapter 6, we deal with user-oriented hierarchical bandwidth scheduling in order to support DiffServ and fairness to end-users in EPON. We propose a collection of credit-based hierarchical scheduling algorithms (UHSA-S and UHSA-M) for the purpose. Two-part frame partition approach is adopted for inter-ONU scheduling. As for intra-ONU scheduling, a credit-based scheduling is proposed to guarantee fairness among end users. On the other hand, to improve scheduling efficiency and lower queuing management complexity, a novel credit-based common queue (CCQ) for each class is proposed. We further propose a transmission priority scheme for different queue groups, which together with CCQ mechanism substantially improve delay and jitter performance of high priority traffic, while maintaining high throughput for medium priority traffic. Simulation experiments show: 1) the UHSA algorithms improve delay and delay variation performance by reducing AD over 40.2% and DV over 22.6% as compared with traditional solutions. 2) the CCQ mechanism substantially enhances scheduling efficiency with a large percent of packets served from CCQ groups and lowers queue management complexity with a reduction over 40.6% of AQ at the loads of 0.3-0.9. These improvements are significant.

7.2 Future Research

Aggregate scheduling and DBA protocols emerge as two new branches of scheduling family. The former provides scalable end-to-end QoS for next generation high speed

networks, while the latter offers efficient QoS support for access networks. This thesis contributes to both of the topics. Looking ahead, there are a few interesting research directions that are either direct extension of our work or motivated by more general problem of designing scheduling disciplines for QoS support.

- **Statistical End-to-end QoS Guarantees with Aggregate Scheduling** – In Chapter 3 and 4, we have studied aggregate scheduling for deterministic end-to-end QoS delivery. Another research direction is to investigate statistical end-to-end QoS performance with aggregate scheduling. Although, deterministic QoS is crucial for real-time and performance critical applications, statistical QoS also plays an important role for many applications, such as the assured forwarding (AF) traffic in DiffServ networks. On the other hand, the simulation experiments in chapter 3 show that for a complex network the mean maximum end-to-end delay is far lower than that of the analytical results. Therefore, statistical end-to-end QoS guarantee is another interesting direction in the study of aggregate scheduling. Little work has been on this topic. [LPB02,LPB03,LTJ07] give some references.
- **Work-conserving Fair Aggregator with Aggregate Scheduling** – A work-conserving system is considered to be better than a non-working conserving system in terms of throughput and average delay in a networking environment. Unfortunately, in aggregate scheduling, the circumstance becomes complex. In Chapter 2 and from literature [CB00,Jia02,etc], it shows that aggregate scheduling of FIFO aggregation brings poor utilization in order to derive guaranteed end-to-end delay bound. This is despite that FIFO aggregation is work-conserving. The

RI-based approaches also have strict source rate condition restrictions as shown in Chapter 3, which could limit network utilization when the network is large. Therefore, FIFO aggregation is not ideal with regard to utilization. On the other hand, fair aggregators introduced in Chapter 4 and [Cob02,Jia06] are non-work conserving systems. In [SS05], a work-conserving fair aggregator is introduced. However, it is not a real networking system since aggregation is done only once at the ingress node. Therefore, up-to-date, there is still no good result in fair aggregator design. This poses another interesting topic for future research.

- **Performance Bounds Analysis in DBA Algorithms** – Access network that connects local network with wide area network plays a necessary role in completing end-to-end QoS delivery. A lot of work has been done in deriving end-to-end QoS performance for local and core networks while research on access network has just attracted attention. Although, there are many DBA algorithms developed in literature, few have conducted deterministic or statistical performance analysis. In Chapter 5 and in [KBS04], some pioneer analytical work is done and in [MZC03] some statistical work is presented. However, DBA algorithms design with performance bounds analysis deserves more systematic and extensive research.
- **End-to-end QoS Delivery to End Users in DBA Algorithms** – Current research work in DBA design is focused on efficient bandwidth allocation and different QoS support to aggregation traffic of an ONU. Since in general an ONU hosts multiple users, not a single one. There is a need to distinguish traffic in an ONU and provide protection and fairness between different users in subscriber-based

access networks. In Chapter 6, we present two user-oriented hierarchical scheduling algorithms (UHSAs) and also there is another work in [CCH06] that considering QoS delivery to end users. More investigations and performance analysis are required on this topic.

Author's Publications

A. Journal and Conference Papers Published

1. Y. Yin, and G. S. Poo, "Preserving quality of service guarantees with guaranteed rate fair aggregation (GRFA)," in *Proc. IEEE ICON'04*, pp. 381-385, Nov.2004.
2. Y. Yin, and G. S. Poo, "End-to-end QoS guarantees for a network based on Latency-Rate Max-Min Service Curve," in *Proc. IEEE ICC'05*, pp. 255-259, May, 2005.
3. Y. Yin, and G. S. Poo, "End-to-end QoS guarantees for a network based on Latency-Rate Max-Min Service Curve," *Comp. Commu.*, vol. 29, pp. 3833-3843, 2006.
4. Y. Yin, and G. S. Poo, "A hybrid cycle bandwidth allocation scheme with differentiated services support in Ethernet passive optical networks," *Photonic Network Commun.*, vol. 15, no. 3, pp. 263-274, Jun. 2008.

B. Journal Papers Submitted

1. Y. Yin, and G. S. Poo, "User-oriented hierarchical bandwidth scheduling for Ethernet passive optical networks," submitted for publication.

Bibliography

- [802] IEEE 802.ah draft at <http://www.ieee802.org/3/efm/>
- [ABH03] F. An, H. Bae, Y. Hsueh, M. Rogge, and L. Kazovsky, "A new media access control protocol guaranteeing fairness among users in Ethernet-based passive optical networks," in *Proc. IEEE OFC'03*, pp. 134-135, Mar. 2003.
- [AHK03] F. An, Y. Hsueh, K. S. Kim, I. M. White, and L.G. Kazovsky, "A new dynamic bandwidth allocation protocol with quality of service in Ethernet-based passive optical networks," in *Proc. International conference on wireless and optical commun. (WOC 2003)*, Banff, Canada, Jul. 2003.
- [AR96] R. Agrawal, and R. Rajan, "Performance bounds for guaranteed and adaptive services," IBM Tech. Rep. RC 20649, Dec. 1996.
- [AYD03] C. M. Assi, Y. Ye, S. Dixit and M. A. Ali, "Dynamic bandwidth allocation for quality-of-service over Ethernet PONs," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 9, pp. 1467-1477, Nov. 2003.
- [BBC98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An architecture for differentiated services," IETF RFC 2475, Dec. 1998.
- [BBC02] J. C. R. Bennett, K. Benson, A. Charny, W. F. Courtney, and J.-Y. Le Boudec, "Delay jitter bounds and packet scale rate guarantees for expedited forwarding," *IEEE/ACM Transactions on networking*, vol. 10, pp. 529-540, Aug. 2002.
- [BC03] J.-Y. Le Boudec, and A. Charny, "Packet scale rate guarantee for non-FIFO nodes," *IEEE/ACM Transactions on networking*, vol. 11, pp. 810-820, Oct. 2003.
- [BCO92] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat, "Synchronization and linearity, An algebra for discrete event systems," New York: Wiley, 1992.

- [BCS94] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet architecture: an overview," IETF RFC 1633, Jun. 1994.
- [BH00] J.-Y. Le Boudec, and G. Hébuterne, "Comments on a deterministic approach to the end-to-end analysis of packet flows in connection oriented networks," *IEEE/ACM Transactions on networking*, vol. 8, pp. 121-124, Feb. 2000.
- [BNL03] H. Byun, J. Nho, and J. Lim, "Dynamic bandwidth allocation algorithm in Ethernet passive optical networks," *Electronic Letters*, vol. 39, pp. 1001-1002, Jun. 2003.
- [Bou98] J.-Y. Le Boudec, "Application of network calculus for guaranteed service networks," *IEEE Transactions on Information Theory*, vol. 44, no. 3, May 1998.
- [BSG05] X. Bai, A. Shami, N. Ghani, and C. Assi, "A hybrid granting algorithm for QoS support in Ethernet passive optical networks," in *Proc. IEEE ICC'05*, pp. 1869-1873, May 2005.
- [BT03] J.-Y. Le Boudec, and P. Thiran, "Network calculus: a theory of deterministic queuing systems for the Internet," Springer Verlag Lecture Notes in Computer Science volume 2050 (available online at <http://lcawww.epfl.ch>), May, 2003.
- [BZ96a] J. C. R. Bennett, and H. Zhang, "WF2Q: Worst-case fair weighted fair queueing," in *Proc. IEEE INFOCOM*, vol. 1, pp. 120-128, Mar. 1996.
- [BZ96b] J. C. R. Bennett, and H. Zhang, "Hierarchical packet fair queueing algorithms," *IEEE/ACM Transaction on networking*, vol. 5, no. 5, pp. 675-689, Oct. 1997.
- [CB00] A. Charny, and J.-Y. Le Boudec, "Delay bounds in a network with aggregate scheduling," in *First International Workshop on Quality of Future Internet Services (QOFIS'2000)*, Berlin, German, vol. 1922, pp. 1-13, Sep. 2000.
- [CCH05] B. Chen, J. Chen, and S. He, "A novel algorithm for Intra-ONU bandwidth allocation in Ethernet passive optical networks," *IEEE Commun. Lett.*, vol. 9, no. 9, pp. 850-852, Sep. 2005.

- [CCH06] B. Chen, J. Chen, and S. He, "Efficient and fine scheduling algorithm for bandwidth allocation in Ethernet passive optical networks," *IEEE J. Sel. Topics Quantum Electronics*, vol. 12, No. 4, pp. 653-660, Jul-Aug. 2006.
- [CFZ98] I. Chlamtac, A. Faragó, H. Zhang, "A deterministic approach to the end-to-end analysis of packet flows in connection-oriented networks," *IEEE/ACM Transactions on networking*, vol. 6, no. 4, pp. 422-431, Aug. 1998.
- [CH02] S. I. Choi, and J. D. Huh, "Dynamic bandwidth allocation algorithm for multimedia services over Ethernet PONs," *ETRI Journal*, vol. 24, no. 6, pp. 465-468, Dec. 2002.
- [Cha00] C.-S. Chang, "Performance guarantees in communications networks," Springer Verlag, 2000, ISBN: 1852332263.
- [Cho04] S. I. Choi, "Cyclic polling-based dynamic bandwidth allocation for differentiated classes of service in Ethernet passive optical networks," *Photonic Network Commun.*, vol. 7, no. 1, pp. 87-96, 2004.
- [CO96] R. L. Cruz, and C. M. Okino, "Service guarantees for window flow control," in *34th Allerton Conf. Communication, Control, and Computing* (Monticello, IL), Oct. 1996.
- [Cob02] J. A. Cobb, "Preserving quality of service guarantees in spite of flow aggregation," *IEEE/ACM Transactions on networking*, vol. 10, no. 1, pp. 43-53, Feb. 2002.
- [Cru91a] R. L. Cruz, "A calculus for network delay, part I: network elements in isolation," *IEEE Transactions on Information Theory*, vol. 37, pp. 114-131, Jan. 1991.
- [Cru91b] R. L. Cruz, "A calculus for network delay, part II: network analysis," *IEEE Transactions on Information Theory*, vol. 37, pp. 132-141, Jan. 1991.
- [Cru95] R. L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE J. Sel. Areas Commun.*, vol. 13, pp. 1048-1056, Aug. 1995.

- [DCB02] B. Davie, A. Charny, J. C. R. Bennett, K. Benson, J.-Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis, "An expedited forwarding PHB (Per-Hop-Behavior)," IETF RFC 3246, Mar. 2002.
- [DKS90] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," *J. Internetworking Res. Experience*, pp.3-26, Oct. 1990.
- [FWD01] F. L. Faucheur, L. Wu, B. Davari, P. Vaananen, R. Krishnan, P. Cheval, and J. Heinanen, "Multi-protocol label switching (MPLS) support of differentiated services," IETF RFC3270, Apr. 2001.
- [GLV95] P. Goyal, S. S. Lam, and H. M. Vin, "Determining end-to-end delay bounds in heterogeneous networks," in *Proc. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95)*, pp. 287-298, Apr. 1995.
- [Gol91] S. Golestani, "A framing strategy for congestion management," *IEEE J. Sel. Areas Commun.*, vol. 9, no. 7, pp. 1064-1077, Sep. 1991.
- [Gol94] S. Golestani, "A self-clocked fair queueing scheme for broadband applications," in *Proc. IEEE INFOCOM'94*, Toronto, CA, pp. 636-646, Jun. 1994.
- [GSA04] N. Ghani, A. Shami, C. Assi, and M. Y. A. Raja, "Intra-ONU bandwidth scheduling in Ethernet passive optical networks," *IEEE Commun., Lett.*, vol. 8, no. 11, pp. 683-685, Nov. 2004.
- [Guo01] C. Cuo, "SSR: an $O(1)$ time complexity packet scheduler for flows in multi-service packet networks," in *Proc. SIGCOMM'01*, pp. 211-222, San Diego, CA, Sep. 2001.
- [GV97] P. Goyal, and H. M. Vin, "Generalized guaranteed rate scheduling algorithms: a framework," *IEEE/ACM Transactions on networking*, vol. 5, no. 4, pp. 561-571, Aug. 1997.
- [GVC96] P. Goyal, H. M. Vin, and H. Cheng, "Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks," in *Proc. ACM SIGCOMM'96*, pp. 157-168, Aug. 1996.
- [HO87] J. L. Hammond, and P. P. O'Reilly, "Performance analysis of local computer networks," Addison-Wesley, 1986. ISBN: 0-201-11530-1.

- [JB06] J. Joo, Y. Ban, "Dynamic bandwidth allocation algorithm for next generation access network," in *Proc. IEEE OFC'06*, Mar. 2006.
- [JG02] H. Jonathan chao, and X. Guo, "Quality of service control in high-speed networks," book, New York: John Wiley & Sons, Inc. 2002, ISBN: 0471003972.
- [Jia02] Y. Jiang, "Delay bounds for a network of guaranteed rate servers with FIFO aggregation," *Comput. networks*, vol. 40, pp. 683-694, Dec. 2002.
- [Jia03] Y. Jiang, "Relationship between guaranteed rate server and latency rate server," *Comput. networks*, vol. 43, no. 3, pp. 307-315, Oct. 2003.
- [Jia04] Y. Jiang, "Link-based fair aggregation: a simple approach to scalable support of per-flow service guarantees," *networking 2004*, pp. 1084-1095, 2004.
- [Jia06] Y. Jiang, "Per-domain packet scale rate guarantee for expedited forwarding," *IEEE/ACM Transactions on networking*, vol. 14, no. 3, pp. 630-643, Jun. 2006.
- [JX06] S. Jiang, and J. Xie, "A frame division method for prioritized DBA in EPON," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 4, pp. 83-94, Apr. 2006.
- [KBS04] G. Kramer, A. Banerjee, N. Singhal, B. Mukherjee, S. Dixit, and Y. Ye, "Fair queuing with service envelopes (FQSE): a cousin-fair hierarchical scheduler for subscriber access networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 8, pp. 1497-1513, Oct. 2004.
- [KKK90] C. Kalmanek, H. Kanakia, and S. Keshav, "Rate controlled servers for very high-speed networks," in *Proc. IEEE GLOBECOM*, San Diego, CA, USA, pp. 12-20, Dec. 1990.
- [KM02] G. Kramer and B. Mukherjee, "Interleaved polling with adaptive cycle time (IPACT): a dynamic bandwidth distribution scheme in an optical access network," *Photonic Network Commun.*, vol. 4, no. 1, pp. 89-107, 2002.
- [KMD02] G. Kramer, B. Mukherjee, S. Dixit, Y. Ye, and R. Hirth, "Supporting differentiated classes of service in Ethernet passive optical networks," *J. Opt. Networks*, vol. 1, pp. 280-298, Aug. 2002.

- [KMP01] G. Kramer, B. Mukherjee, G. Pesavento, "Ethernet PON (ePON): Design and Analysis of an Optical Access Network," *Photonic Network Commun.*, vol. 3, no. 3, pp. 307-319, Jul. 2001.
- [KPK05] H. Kim, H. Park, D. Kang, C. Kim, and G. Yoo, "Sliding cycle time-based MAC protocol for service level agreeable Ethernet passive optical networks," in *Proc. IEEE ICC'05*, Seoul, Korea, pp. 1848-1852, May 2005.
- [Kra03] G. Kramer, "Design of next-generation subscriber access systems based on Ethernet passive optical networks (EPON)," thesis, 2003.
- [KSC91] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in general-purpose ATM switch chip," *IEEE J. Sel. Areas Commun.*, vol. 9, pp. 1265-1279, Oct. 1991.
- [Kur92] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks," in *Proc. ACM Sigmetrics/Performance'92*, 1992.
- [KV01] J. Kaur, and H. Vin, "Core-stateless guaranteed rate scheduling algorithms," in *Proc. IEEE INFOCOMM'01*, vol. 3, pp. 1484-1492, Apr. 2001.
- [LK99] A. M. Law, and W. D. Kelton, "Simulation modeling & analysis," McGraw Hill, 1999.
- [LPB02] J. Liebeherr, S. D. Patek, and A. Burchard, "A calculus for end-to-end statistical service guarantees," *Technical Report of University of Virginia Department of Computer Science*, CS-2001-19, May 2002.
- [LPB03] J. Liebeherr, S. D. Patek, and A. Burchard, "Statistical per-flow service bounds in a network with aggregate scheduling," in *Proc. IEEE INFOCOMM'03*, vol. 3, pp. 1680-1690, Apr. 2003.
- [LTJ07] Y. Liu, C. K. Tham, and Y. Jiang, "A calculus for stochastic QoS analysis," *performance evaluation*, in press, 2007.
- [LWD06] W.W.K. Lin, A.K.Y. Wong, and T.S. Dillon, "Application of soft computing techniques to adaptive user buffer overflow control on the Internet," *IEEE Trans. System, Man, and Cyberntics – Part C: Application and Reviews* 36(3) (2006) pp. 397-410.

- [MMR04] M. P. McGarry, M. Maier, and M. Reisslein, "Ethernet PONs, a survey of dynamic bandwidth allocation (DBA) algorithms," *IEEE Commun., Magazine*, pp. S8-15, vol. 42, Aug. 2004.
- [MZC03] M. Ma, Y. Zhu, and T. H. Cheng, "A bandwidth guaranteed polling MAC protocol for Ethernet passive optical networks," in *Proc. IEEE INFOCOM*, pp. 22-31, Apr. 2003.
- [OB03] F.-D. Otel, and J.-Y. Le Boudec, "Deterministic end-to-end delay guarantees in a heterogeneous route interference environment," in *Proc. Int. Workshop Quality of Future Internet Services (QoFIS'03)*, Stockholm, pp. 21-30, Oct. 2003.
- [PG93] 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 Transaction on networking*, vol. 1, no. 3, pp. 344-357, Jun. 1993.
- [PG94] 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 Transaction on networking*, vol. 2, no. 2, pp. 137-150, Apr. 1994.
- [RP03] S. Ramabhadran, and J. Pasquale, "Stratified round robin: a low complexity packet scheduler with bandwidth fairness and bounded delay," in *Proc. ACM SIGCOMM'03*, pp. 239-250, Karlsruhe, Germany, Aug. 2003.
- [RVC01] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," *IETF RFC3031*, Jan. 2001.
- [Sar96] H. Sariowan, "A service curve approach to performance guarantees in integrated service networks," Ph.D dissertation, Univ. Calif. San Diego, 1996.
- [SBA05] A. Shami, X. Bai, C. Assi, and N. Ghani, "Jitter performance in Ethernet passive optical networks," *IEEE J. lightwave technology*, vol. 23, no. 4, pp. 1745-1753, Apr. 2005.
- [SBG05] A. Shami, X. Bai, N. Ghani, and H. T. Mouftah, "QoS control schemes for two-stage Ethernet passive optical access networks," *IEEE J. Sel. Areas Commun.*, vol 23, no. 8, Aug. 2005.

- [SRC04] K. Son, H. Ryu, S. Chong, and T. Yoo, "Dynamic bandwidth allocation schemes to improve utilization under non-uniform traffic in Ethernet passive optical networks," in *Proc. IEEE ICC'04*, pp. 1766-1770, Jun. 2004.
- [SS03] W. Sun, and K. G. Shin, "Delay bounds for end-to-end traffic aggregate under guaranteed rate scheduling algorithms," Dept. of EECS, Univ. of Michigan, Tech. Rep. CSE-TR-484-03, 2003.
- [SS04] W. Sun, and K. G. Shin, "Coordinated aggregate scheduling for improving end-t-end delay performance," in *Proc. IWQoS'04*, pp. 77-86, Jun. 2004.
- [SS05] W. Sun, and K. G. Shin, "End-to-end delay bounds for traffic aggregates under guaranteed-rate scheduling algorithms," *IEEE/ACM Transactions on networking*, vol. 13, no. 5, pp. 1188-1201, Oct. 2005.
- [SV96a] M. Shreedhar, and G. Varghese, "Efficient fair queuing using deficit round robin," *IEEE/ACM Transactions on networking*, vol. 4, no. 3, pp. 375-385, Jun. 1996.
- [SV96b] D. Stiliadis, and A. Varma, "Design and analysis of frame-based fair queuing: a new traffic scheduling algorithm for packet switched networks," In *Proc. ACM Sigmetrics'96*, pp. 104-115, May 1996.
- [SV98] D. Stiliadis, and A. Varma, "Latency-rate servers: a general model for analysis of traffic scheduling algorithms," *IEEE/ACM Transactions on networking*, vol. 6, no. 5, pp. 611-624, Oct. 1998.
- [SZ99] I. Stoica, and H. Zhang, "Providing guaranteed services without per-flow management," in *Proc. ACM SIGCOMM'99*, pp. 81-94, Sep. 1999.
- [TWS97] M. S. Taqqu, W. Willinger, and R. Sherman, "Proof of a fundamental result in self-similar traffic modeling," *ACM/SIGCOMM Comput. Commun. Rev.*, vol. 27, pp. 5-23, Apr. 1997.
- [VZF91] D. Verma, H. Zhang, and D. Ferrari, "Guaranteeing delay jitter bounds in packet switching networks," in *Proc. Tricomm'91*, Chapel Hill, NC, pp. 35-46, Apr. 1991.

- [WTE96] W. Willinger, M. S. Taqqu, and A. Erramilli, "A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks," in *Stochastic Networks*, F. P. Kelly, S. Zachary, and I. Ziedins, eds. (Oxford University, Oxford, UK, 1996), pp. 339-366.
- [XJJ04] J. Xie, S. Jiang, and Y. Jiang, "A dynamic bandwidth allocation scheme for differentiated services in EPONs," *IEEE optical commun.*, pp. 532-539, Aug. 2004.
- [XL95] G. G. Xie, and S. S. Lam, "Delay guarantee of virtual clock server," *IEEE/ACM Transactions on networking*, vol. 3, no. 6, pp. 683-689, Dec. 1995.
- [YD05] X. Yuan, and Z. Duan, "FRR: a proportional and worst-case fair round robin scheduler," in *Proc. IEEE INFCOMM'05*, vol. 2, pp. 831-842, Mar. 2005.
- [ZDH01] Z.-L. Zhang, Z. Duan, and Y. T. Hou, "Fundamental trade-offs in aggregation packet scheduling," in *Proc. 9th Int. Conf. Network Protocols (ICNP'01)*, pp. 129-137, Nov. 2001.
- [ZF94] H. Zhang, and D. Ferrari, "Rate-controlled service disciplines," *J. high speed networking*, vol. 3, no. 4, pp. 389-412, 1994.
- [Zha95] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," in *Proc. IEEE*, vol. 83, no. 10, pp. 1374-1396, Oct. 1995.
- [Zha99] H. Zhang, "A note on deterministic end-to-end delay guarantees in a heterogeneous route interference environment," in *Proc. IEEE ICC'99*, Vancouver, Canada, pp. 1223-1127, 1999.
- [Zhe06] J. Zheng, "Efficient bandwidth allocation algorithm for Ethernet passive optical networks," in *Proc. IEE Commun.*, vol. 153, pp. 464-468, Jun. 2006.
- [ZM05] J. Zheng and H. T. Mouftah, "Adaptive scheduling algorithms for Ethernet passive optical networks," in *Proc. IEE Commun.*, vol. 152, pp. 643-647, October 2005.
- [ZM06] J. Zheng, and H. T. Mouftah, "Media access control for Ethernet passive optical networks: an overview," *IEEE Commun., Magazine*, pp. 145-150, Feb. 2006.

- [ZMC05] Y. Zhu, M. Ma, and T. H. Cheng, "Hierarchical scheduling to support differentiated services in Ethernet passive optical networks," *Comput. networks*, vol. 50, pp. 350-366, Feb. 2006.