JYVÄSKYLÄ STUDIES IN COMPUTING 45

Jian Zhang

QoS- and Revenue-Aware Resource Allocation Mechanisms in Multiclass IP Networks

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Agora-rakennuksessa (Alfa-sali)
joulukuun 10. päivänä 2004 kello 12.

Academic dissertation to be publicly discussed, by permission of
the the Faculty of Information Technology of the University of Jyväskylä,
in the Building Agora, Alfa Hall, on December 10, 2004 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2004

QoS- and Revenue-Aware Resource Allocation Mechanisms in Multiclass IP Networks

JYVÄSKYLÄ STUDIES IN COMPUTING 45

Jian Zhang

QoS- and Revenue-Aware Resource Allocation Mechanisms in Multiclass IP Networks

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2004

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN 9513920364
ISBN 951-39-2036-4 (PDF)

ISBN 951-39-1978-1 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä
and ER-Paino Ky, Lievestuore 2004

ABSTRACT

Zhang, Jian
QoS- and Revenue-Aware Resource Allocation Mechanisms in Multiclass IP Networks
Jyväskylä: University of Jyväskylä, 2004, 85 p.
(Jyväskylä Studies in Computing,
ISSN 1456-5390;45)
ISBN 951-39-2036-4
Diss.

The substantial changes transforming the Internet from a communication and browsing infrastructure to a medium for conducting personal business and e-commerce are making Quality of Service (QoS) an increasingly critical issue. However, QoS of networks by itself is not sufficient to support end-to-end QoS. To avoid high priority network traffic being dropped at the server, Web servers should have mechanisms and policies for delivering end-to-end QoS. More importantly, in the future multiclass Internet, each class of customers may have to pay their service providers for the received level of QoS based on the Service-Level-Agreements negotiated and committed between them.

In this dissertation, we first propose a novel *Arrival-Related Dynamic Partitioning* mechanism for enabling differentiated services in cluster-based Web server systems, which works well even when the Web cluster system is heavily loaded and does not have enough server resources to be allocated. Then, a scalable Web cluster architecture – the two-level cluster architecture, is proposed for implementing scalable service differentiation in cluster-based Web server systems, whose scalability is theoretically determined only by the scalability of its layer-4 switch.

Next, we link the issue of resource partitioning scheme with the pricing strategy in a Service-Level-Agreement (SLA) and address the problem of maximizing the SLA revenue obtained in a IP network node under a given amount of resources by optimally allocating the resources among the supported service classes. First, the revenue-aware resource allocation schemes, which can achieve the maximization of the SLA revenue obtained in a multiclass-supported network node under a given amount of network resources and linear pricing strategy, are proposed. Then we derive a novel upper bound on mean packet delay of GPS-based (Generalized Processor Sharing based) Fair Queueing (FQ) algorithms under the probabilistic traffic model of Poisson arrival and any general packet length distribution, which is much simpler and tighter than the known ones by M. Hawa *et al* and fits a class of GPS-based FQ algorithms including Weighted Fair Queuing (WFQ), Self-Clocked Fair Queuing (SCFQ) and Starting Potential-based Fair Queuing (SPFQ). Furthermore, based on this novel upper delay bound, the suboptimal resource allocation scheme is presented for maximizing the SLA revenue attained under a constrained amount of network resources and flat pricing strategy in a network node which deploys a GPS-based packetized fair queueing algorithm.

In the end, the problem of maximizing the SLA revenue attained for the hosting of an e-commerce Web site upon a cluster-based Web server system by optimally partitioning the server resources among all supported service classes is analyzed. The optimal resource partitioning scheme is derived, which can implement the maximization of the SLA revenue obtained for the hosting of an e-commerce site under a given amount of server resources and linear pricing strategy. Moreover, the suboptimal resource partitioning scheme is also proposed for achieving the highest SLA revenue in the hosting of an e-commerce site under a given amount of server resources and flat pricing strategy.

Keywords: Cluster-based Web Server System, IP Network Node, Quality of Service, Service Level Agreement, QoS- and Revenue-aware Resource Allocation Scheme

Author	Jian Zhang Department of Mathematical Information Technology University of Jyväskylä Finland
Supervisors	Professor Timo Hämaläinen Department of Mathematical Information Technology University of Jyväskylä Finland
	Professor Jyrki Joutsensalo Department of Mathematical Information Technology University of Jyväskylä Finland
Reviewers	Professor Tapani Ristaniemi Institute for Communications Engineering Tampere University of Technology Finland
	Professor Corneliu A.Marinov Department of Electrical Engineering Polytechnic University of Bucharest Romania
Opponent	Professor Pertti Raatikainen VTT Information Technology, Telecommunications VTT Finland

ACKNOWLEDGEMENTS

On this occasion, I give thanks to my advisors Prof. Timo Hämaläinen and Prof. Jyrki Joutsensalo for their valuable guidance and support during the entire course of this thesis. I also thank Dr. Kimmo Kaario for his useful comments in publishing our co-authored articles. Also, I give thanks to Prof. Tapani Ristaniemi (Tampere University of Technology) and Prof. Corneliu A.Marinov (Polytechnic University of Bucharest) for their careful review of the manuscript and give thanks to Dr. Kevin Rogovin for giving suggestions to improve the text.

I am also indebted to the COMAS Graduate School, the Dept. of Mathematical Information Technology at the University of Jyväskylä and Nokia Foundation for their financial support during my doctoral studies.

Finally, many thanks to my parents and my wife Jia Jia for their never-ending support and love.

Jyväskylä, November 22, 2004.

Jian Zhang

CONTENTS

LIST OF PUBLICATIONS AND CONTRIBUTIONS OF THE AUTHOR

In the publications below except publication **PIII**, the author has created simulation programs and run simulation tests. Moreover, whenever the author's name appears as the first one, he was mainly responsible for writing the paper with co-operation of the other authors.

In publications **PIV**, **PVI**, **PVIII**, **PXII** and **PXIV**, the author has solely developed the used algorithms while in publications **PIX**, **PX**, **PXI** and **PXIII**, the author has extended the idea of revenue-aware scheduling algorithm in a network node from the second and third author and derived the new mechanisms. In publication **PII**, the second and fourth author contributed their idea to the algorithm design and all the co-authors have participated in the written work. Additionally, the author has contributed in the written work in publications **PI** and **PIII**.

PI K. Kaario, T. Hämäläinen, and J. Zhang, "Tuning of QoS aware load balancing algorithm (QoS-LB) for highly loaded server clusters," In Proceedings of First International Conference on Networking (ICN'01), Part II, pp. 250-258, Jul. 11-13, 2001.

PII J. Zhang, T. Hämäläinen, J. Joutsensalo and K. Kaario, "QoS-Aware Load Balancing Algorithm for Globally Distributed Web Systems," In Proceedings of the International Conferences on Info-tech and Info-net (ICII2001), Oct. 29 - Nov.1, 2001.

PIII T. Hämäläinen, J. Joutsensalo and J. Zhang, "Optimal Link Allocation and Charging Model," In Proceedings of the International Conference on Advances in Infrastructure for E-business, E-education, E-science, and E-medicine on the Internet (SSGRR2002), Jul. 29-Aug. 4, 2002.

PIV J. Zhang, T. Hämäläinen and J. Joutsensalo, "Dynamic Partitioning: A Mechanism for Supporting Differentiated Services in Cluster-based Network Servers," In Proceedings of 16th Nordic Teletraffic Seminar (NTS16), pp. 185-193, August 21-23, 2002.

PV J. Joutsensalo, T. Hämäläinen and J. Zhang, "Resource Allocation for Differentiated QoS by Adaptive Weighted Fair Queue Technique," In Proceedings of 16th Nordic Teletraffic Seminar (NTS16), pp. 291-302, August 21-23, 2002.

PVI J. Zhang, T. Hämäläinen and J. Joutsensalo, "A New Mechanism for Supporting Differentiated Services in Cluster-based Network Servers," In Proceedings of 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02), pp. 427-432, Oct. 12-16, 2002.

PVII J. Joutsensalo, T. Hämäläinen and J. Zhang, "Revenue maximization-based adaptive WFQ," In Proceedings of SPIE symposium of Asia-Pacific Optical and Wireless Communication (APOC2002), Volume 4907, pp. 108-117, Oct. 13-18, 2002.

PVIII J. Zhang, K. Luostarinen, T. Hämäläinen and J. Joutsensalo, "Enabling QoS Support in Global Replicated Web Systems," In Proceedings of 9th Asia Pacific Conference on Communications (APCC2003), Volume 2, pp. 735-739, Sept. 21-24, 2003.

PIX J. Zhang, T. Hämäläinen and J. Joutsensalo, "Revenue-aware Resource Allocation Scheme in Multiservice IP Networks," WSEAS Transactions on Communications, Issue 1, Volume 3, pp. 277-281, Jan. 2004, ISSN 1109-2742.

PX J. Zhang, T. Hämäläinen and J. Joutsensalo, "Packet Scheduling for Maximizing Revenue in a Network Node," In Proceedings of 1st International Conference on E-Business and Telecommunication Networks (ICETE2004), Volume 2, pp. 134-139, Aug. 25-28, 2004.

PXI J. Zhang, T. Hämäläinen and J. Joutsensalo, "Optimal Resource Allocation Scheme for Maximizing Revenue in the Future IP Networks," In Proceedings of Joint Conference of 10th Asia Pacific Conf. on Communications and 5th International Symposium on Multi-Dimensional Mobile Communications (APCC2004 & MDMC2004), Aug. 29-Sept. 1, 2004.

PXII J. Zhang, T. Hämäläinen and J. Joutsensalo, "Maximizing Service-Level-Agreement Revenues in Clustered-based Web Server Systems," WSEAS Transactions on Information Science and Applications, Issue 3, Volume 1, pp. 861-866, Sept. 2004, ISSN 1790-0832.

PXIII J. Zhang, T. Hämäläinen and J. Joutsensalo, "Revenue-aware Resource Allocation in the Future Multi-Service IP Networks," In Proceedings of IFIP TC6 Conference on Network Control and Engineering for QoS, Security and Mobility (Net-Con'2004), Nov. 3-5, 2004.

PXIV J. Zhang, T. Hämäläinen and J. Joutsensalo, "Stochastic Analysis of Upper Delay Bound of GPS-based Packetized Fair Queueing Algorithms," In Proceedings of 12th IEEE International Conference On Networks (ICON2004), Nov. 16-19, 2004.

1 INTRODUCTION

Resource allocation in the multiservice communication networks presents a very important problem in the design of the future multiclass IP networks. The main motivation for the research in this field lies in the necessity for structural changes in the way the Internet is designed. The current Internet offers a single class of 'best-effort' service, although some traffic prioritization will be active in the new network router implementations. However, the Internet is changing and it is becoming an important channel for critical information and the fundamental technology for information systems of most advanced companies and organizations. Users are becoming increasingly reliant on the Internet for up-to-date personal, professional and business information. The substantial changes transforming the Internet from a communication and browsing infrastructure to a medium for conducting personal business and e-commerce are making Quality of Service (QoS) an increasingly critical issue. Meanwhile, to realize the above changes, the future IP networks must be able to support a wide range of different traffic types with different QoS requirements. For example, new sophisticated real-time applications such as Voice over IP (VoIP), Video-on-Demand (VoD) and Video-Conferencing require firm performance guarantees from the network where certain resources should be reserved for them. Hence, efficient resource allocation mechanisms are needed to distribute network resources among all competing service classes for achieving their QoS requirements.

As a result of the complexity of the Internet infrastructure, many factors affect the performance of Internet services. Hence, to guarantee the assessed service levels for QoS, actions on all components of the WWW would be required: from network technology and protocols, to hardware and software architectures of Web servers and proxies. Many efforts in the field of QoS provisioning have focused on the solutions at network side, e.g., Integrated Services (IntServ) architecture [8] and Differentiated Services (DiffServ) architecture [7] have been proposed by the Internet Engineering Task Force (IETF) and also novel Fair Queuing (FQ) algorithms have been proposed (e.g., [64, 29, 82]) to implement QoS in packet-switched networks. However, QoS of networks by itself is not sufficient to support end-to-end QoS. Furthermore, with network bandwidth increasing much faster than server capacities, it is more likely that the bottleneck will be on the server side. To avoid high priority network traffic being dropped at the server, Web servers should have mechanisms and policies for delivering end-to-end QoS.

As we know, the exponential growth of the Internet, coupled with the increasing popularity of dynamically generated content on the World Wide Web, has created the need for more and faster Web servers capable of serving over 100 million Internet users. The only solution for scaling server capacity in the past had been to completely replace the old server with a new one. Organizations must discard their investment in the old server and purchase a new one — an expensive, short-term solution. Nowadays, Web servers based on clusters of commodity PCs or workstations offer a cost-effective and scalable solution to the increasing performance demands placed on popular Web sites. Therefore, to enable QoS support in such cluster-based Web server systems is of practical importance.

However, many results about enabling QoS support in Web clusters consider Web server farms as the target cluster architecture, where multiple Web sites co-located on the same platform, e.g., [4]. Few results exist on the topic of providing QoS support in a cluster-based Web server system that hosts a single Web site. Chen *et al* evaluated the impact of the request admission control and dispatching mechanisms in [12]. Kanodia *et al* [43] have proposed a QoS policy that uses both admission control and performance isolation mechanisms to guarantee different classes of service to have latencies within pre-specified targets. Cardellini *et al* [9] described and evaluated some mechanisms that can be integrated into existing Web cluster architectures to support quality of Web services. A dynamic resource partitioning algorithm for static and dynamic Web requests is proposed in [98], where QoS principles are addressed as an optimization problem. But it has assumed that the processing rates of the incoming requests have exponential distribution in a Web cluster system, which is not true in the real situation. In this dissertation, we focus on the kind of Web cluster architecture where only a single Web site is hosted. The issue of enabling differentiated services in the above target Web cluster architecture is first addressed. Specifically, by extending the QoS research work in [9] and publications **PI** [42] and **PIV** [87], we proposed a new mechanism [88] for enabling differentiated services in the target cluster architecture, which dynamically partitions the back-end server nodes in a cluster-based Web server system into different server subsets so that each class of requests will be served only by its own assigned server set. More importantly, two different dynamic partitioning algorithms are designed in this mechanism to be used under different arrival load intensities so as to achieve better performances in the case that the system is heavily loaded and does not have enough server resources available. We call this novel mechanism the *Arrival-Related Dynamic Partitioning* mechanism (ARDP mechanism). The simulation results demonstrated that our proposed ARDP mechanism can provide different classes of customers with different QoS targets and succeed in achieving the goals of service differentiation in target cluster-based Web server systems.

Additionally, state-of-the-art cluster-based systems employ a specialized front-end node, which acts as the single input point of customer requests and is responsible for distributing the accepted requests among the back-end server nodes. Typically, all the server nodes in a Web cluster resides at a single network location and the front-end distributes requests such that the load among the back-end nodes remains balanced. To support the differentiated services in a cluster-based Web server system, the front-end node must be able to examine at least the content of the header field of each access to identify the requested service class. This additional overhead at the front-end node can dramatically limit the scalability of the existing cluster architectures. For instance, conventional PC/workstation based front-ends with QoS support can only scale to a small number of server nodes (less than ten on typical Web workloads [63]). In this dissertation, we extend the studies in publication **PII**, further examine the scalability problems of the existing cluster architectures and propose a much more scalable one – the two-level cluster architecture in Chapter 3. Moreover, the proposed QoS-aware resource partitioning mechanisms in publications **PIV** [87] and **PVI** [88] are generalized to achieve differentiated services in the two-level cluster architecture, which has been published in publication **PVIII** [89]. In Chapter 3, the issue of enabling scalable service differentiation in cluster-based Web server systems is addressed systematically and more simulation results are presented under different parameter settings to demonstrate the effectiveness and feasibility of our proposed scalable Web cluster architecture and scheduling algorithms.

On the other hand, in the future multiclass Internet, each class of customers may have to pay network service providers for their received level of QoS based on the pricing strategy agreed upon in the Service-Level-Agreements between them. A Service-Level-Agreement (SLA) defines the QoS metrics for each class of service, the anticipated per-class workload intensity and the pricing strategy by which the service payment will be determined. Obviously, the pricing strategy will specify the relationship between the QoS level offered to each class of customers and the relevant price which should be paid by them. For example, the service provider will receive a certain amount of revenue from a class of customers if the offered QoS level is more than the minimal requirement of that class and suffer another certain amount of penalty for failing to meet that. Thus, from service providers' point of view, the optimal resource allocation scheme, which can achieve the maximization of SLA revenues under a given amount of network resources (e.g., bandwidth) and a given pricing strategy, is very desirable.

The use of pricing as a means for allocating resources in communication networks has received much attention in recent years. A smart charging method for network usage is presented in [59]. This paper studies individual packet bid for transporting while the network only serves packets which bid above a certain (congestion-dependent) cutoff amount. Charges that increase with either realized flow rate or with the share of the network resources consumed by a traffic flow is studied in [47, 48]. Packet-based pricing schemes (e.g. [28]) has also been proposed as an incentive for more efficient flow control. The fundamental problem of achieving the system optimum that maximizes the aggregate user utility using only the information available at the end hosts is studied in [52]. They assume that the users are of elastic traffic and can adjust their rates based on their estimates of network congestion level. Pricing and link allocation for real-time traffic that requires strict QoS guarantees is studied e.g. in [67, 68]. Such QoS guarantees can often be translated into a preset resource amount that has to be allocated to a traffic flow at all links in its route through the network. If the resource is bandwidth, this resource amount can be some sort of effective bandwidth (see, e.g., [46] for a survey of effective bandwidth characterizations and [66] for similar notions in the multiclass case). In this setting, [45, 17] propose the pricing scheme of real-time traffic with QoS requirements in terms of its effective bandwidth. Their pricing scheme can also be called as a static one and it has clear implementation advantages: charges are predictable by end users, evolve in a slower time-scale than congestion phenomena, and no real-time mechanism is needed to communicate tariffs with the users.

However, none of the above papers addressed the issue of combining the pricing strategies in SLAs and the resource allocation mechanisms in multiclass IP networks to maximize service providers' revenues. J. Joutsensalo *et al* studied the problem of maximizing service providers' revenues obtained in a network node under linear pricing strategy in [38, 39, 40, 41]. In this dissertation, I first extended our previous QoS and revenue-maximization research in [38, 39, 40, 33, 41] and addressed the issue of maximizing SLA revenues in a IP network node under a constrained amount of network resources by novel revenue-aware resource allocation schemes. The linear and flat pricing strategies are deployed in a SLA, which have been demonstrated to be practical ones in the real world [22, 84]. The closed-form solution to the optimal network resource allocation scheme under linear pricing strategy is derived in publications **PIX** [90] and **PXI** [92] when the *packet delay* is chosen as the QoS metric in a SLA while publication **PX** [91] presented the closed-form solution under linear pricing strategy for the QoS metric of *mean packet delay* in a SLA. Furthermore, the suboptimal network resource allocation scheme was proposed **PXIII** [94] for achieving the highest SLA revenues under linear pricing strategy for the case that the firm QoS guarantees are required for all the supported service classes. Chapter 4 summarizes the contributions in these publications.

Next, a novel upper bound on mean packet delay of GPS-based Fair Queueing (FQ) algorithms was derived in publication **PXIV** [95] under the general probabilistic traffic model of Poisson arrivals and any general packet length distribution. The resulted upper delay bound is much simpler and tighter than the ones by M. Hawa *et al* [34]. Moreover, it fits a class of GPS-based FQ algorithms including Weighted Fair Queuing (WFQ) [21, 64], Self-Clocked Fair Queuing (SCFQ) [29] and Starting Potential-based Fair Queuing (SPFQ) [82]. In Chapter 5, the derivation of this novel upper bound on mean packet delay is summarized and then based on this upper delay bound, the suboptimal resource allocation scheme is derived for maximizing the SLA revenue obtained under a given amount of network resources and flat pricing strategy in a network node which deploys a GPS-based packetized fair queueing algorithm.

Cluster-based Web server systems have become a major means to hosting e-commerce sites. The issue of maximizing SLA revenues in the cluster platform of Web server farms was recently studied by Liu *et al* in [56]. A Web server farm is typically deployed to host several Web sites simultaneously on the same platform. Liu *et al* assumed that each back-end server node in a Web server farm can serve multiple service classes simultaneously. Then they tried to optimally allocate the resource (e.g., processing capacity) of each server node among its supported service classes to maximize the SLA revenues in [56], where the closed-form solution to the optimal resource allocation scheme (i.e., the optimal weights) in each back-end node did not be provided. Diao *et al* [22] proposed a profit-oriented feedback control system for maximizing SLA profits in Web server systems, which automated the admission control decisions in a way that balances the loss of revenue due to rejected work against the penalties incurred if admitted work has excessive response times by a fuzzy control algorithm. Additionally, the issue of maximizing the expected value of a given cluster utility function by allocating server resources of a cluster-based Web server system dynamically was studied in [54], where the closed-form solution was also not derived.

In this dissertation, we focus on the cluster platform which hosts a single Web site in a cluster-based Web server system, as was studied in Chapter 2. Chapter 6 analyzes the problem of maximizing the revenues attained in the hosting of a e-commerce site with a SLA contract by optimally partitioning the server resources of a cluster-based Web server system among the supported service classes. It first summarizes the optimal server resource partitioning scheme derived in publication **PXII** [93], which can achieve the maximization of SLA revenues under a constrained amount of server resources and linear pricing strategy in the above target cluster-based Web cluster system. Then the suboptimal server resource allocation scheme for revenue maximization under a given amount of server resources and flat pricing strategy is proposed in Chapter 6. Below the basic concepts in the area of this dissertation are discussed.

1.1 Cluster-based Web Server System

A cluster-based Web server system (briefly, *Web cluster*) consists of a number of commodity workstations or PCs connected by a network. Typically, the server nodes are connected by a high-speed LAN. In this dissertation, we focus on the kind of Web cluster architecture where a collection of all the server nodes work for only a particular Web site and refer to it as Web cluster. Whereas, the term "Web farm" denotes the architecture for the co-location or co-hosting of several Web sites.

Specifically, a basic Web cluster consists of a dedicated front-end server node and a number of back-end server nodes. The authoritative DNS server for the Web site built upon the Web cluster always translates the site name into the IP address of the front-end server node. The front-end node, hereafter called *Web switch*, receives all inbound packets destined for the Web site and then distributes them among the back-end server nodes for serving, thus making the distributed nature of the site architecture completely transparent to both the user and the client application.

Note that the key role in a Web cluster is played by its Web switch and each back-end server node in a cluster is identified uniquely by the Web switch through a private address that can be at different protocol levels (an IP address or a MAC address) depending on the specific cluster architecture. Moreover, the Web cluster architectures are normally first classified according to the OSI protocol stack layer at which the Web switch routes inbound packets to the back-end server node, that is layer-4 or layer-7 Web switches [75, 10].

Layer-4 Web switches determine the target back-end server node when the client asks for establishing a TCP/IP connection, upon the arrival of the first TCP SYN packet at the Web switch. As the client packets do not reach the application level, the routing mechanism is efficient but only content-blind/QoS-blind request distribution strategies can be deployed at layer-4 switches where the requested content/QoS is not examined. Nowadays, very fast layer-4 Web switches are available that can act as front-ends for Web clusters where content-aware request distribution or QoS support is not required [15, 36]. The hardware-based switch fabric of these layer-4 switches can support a very large number of back-end server nodes of layer-4 Web clusters.

In contrast to layer-4 Web switches, layer-7 Web switches first establish a complete TCP connection with the client, examine the HTTP request at application level and then relay it to the target back-end server node. This routing mechanism is much less efficient and introduces a severe process overhead at the layer-7 Web switch, which can significantly limit the scalability of a layer-7 Web cluster where a centralized request distribution strategy is deployed at the front-end [3, 79]. Conventional, PC/workstation based layer-7 switches can only scale to a relatively small number of back-end server nodes (less than ten on typical workloads [63]). On the other hand, however, layer-7 Web switches can efficiently support content-aware/QoS-aware dispatching policies by examining the content of inbound HTTP requests.

Web cluster architectures based on layer-4 and layer-7 Web switches can be further classified on the basis of whether the data from back-end server nodes to clients (outgoing data) go through the Web switch. The kind of cluster architecture, where the back-end nodes respond directly to the clients without passing through the front-end node as an intermediary, is called the *one-way* architecture whereas it is referred to as the *two-way* architecture if the outgoing data have to pass through its Web switch. Typically, one-way architectures are more scalable as the Web switch processes only inbound packets.

Thus, according to the above classifications, these kinds of basic Web cluster architectures exist: layer-4 one-way architecture, layer-4 two-way architecture, layer-7 one-way architecture and layer-7 two-way architecture. In a layer-4 one-way architecture, each back-end server node in the cluster is configured with a unique private IP address which can be either at IP level (layer-3) or MAC level (layer-2); and routing to the target back-end node may be done using several mechanisms, such as packet single-rewriting [23], packet tunnelling [70] and packet forwarding. For a layer-4 two-way architecture, the private address of each back-end node is at IP level; both inbound and outbound packets are rewritten at TCP/IP level by

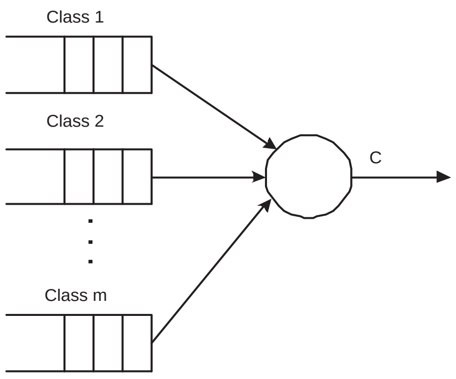


Figure 1: The model of our target multiclass-supported network node.

the Web switch based on the IP Network Address Translation approach [24]. On the other hand, in a layer-7 one-way architecture, the mechanism of TCP handoff [63] or TCP connection hop [72] may be used to relay the established TCP connection to the target back-end server node; and the layer-7 Web switch in a layer-7 two-way architecture may deploy TCP gateway or TCP splicing [16] approach to route inbound requests to the target back-end node. A survey on these basic Web cluster architectures can be found in [75, 10]

1.2 Multiclass-supported Network Node

In general, network node [1] architectures may be classified into two main categories based on whether packets are buffers at the inputs or the outputs of the node [44]. In an input-buffered network node, packets are stored at the inputs of the node and the traffic scheduling can be simplified by dividing the problem into two levels: (i) scheduling transmissions among the input ports of the node transmitting to a common output port, and (ii) scheduling a packet from the chosen input port. In an output-buffered network node, packets are immediately available for transmission as they arrive in the node, and can be regarded as being buffered at the output ports of the network node. In this dissertation, the output-buffered network node is our target node architecture.

If m service classes are supported in an output-buffered network node, packets with the same priority will be buffered at the same queue and each queue corresponds to one service class. Then, the function of a scheduling algorithm is to select, for each outgoing link of the network node, the packet to be transmitted in the next cycle from the available packets belonging to the queues sharing the output link. We assume that each outgoing link of the network node can support all m service classes, thus our target multiclass-supported network node may be abstracted as the model in Figure 1, where C is the bandwidth of the output link.

[1]In this work, we will use the term "network node" to designate either a router or an ATM switch, unless otherwise specified.

1.3 Quality of Service and Service Level Agreement

Over the last few years the Internet has undergone phenomenal growth and there are no signs that this will stop or abate. In fact, the convergence of other networks, such as radio, telephone and television, to the Internet will likely accelerate its growth. These events are not only providing the impetus for a change of the nature of the current Internet but are also the driving forces behind it [80].

It is relatively easy to understand why the Internet is facing these demands. The Net has become an important facet of many people's lives, whether that be for email, web surfing, online shopping or banking or any of the other possible uses. Also fuelling the growth of the Internet is the proliferation of commercial activities using it to both conduct and support business. Of course, both of these factors are placing strenuous requirements on the Internet, most of which can hardly be achieved because they fall on the border or outside of the Net's origin design goals and resultant capabilities.

The current architecture of the Internet is based on best-effort traffic model. In other words, it is able to provide only best effort delivery of data. This has sufficed for traditional Internet applications (e.g. email, file transfer and web surfing). However, the Internet is increasingly required to support many new IP-based applications which have widely differing operational requirements. Multimedia elements can be found in these new applications demonstrating the need for more bandwidth to carry audio and video. In addition, some of them, for instance, real-time applications, certainly have strict timing demands that the best effort service of the Internet is just not able to meet. The point is that the Internet's best effort is likely to result in unacceptable, if not completely unusable, service to some applications, particularly the next generation of Internet applications. The solution is that some intelligence (i.e. complexity) must be introduced into the Internet so that it can differentiate traffic with strict timing requirements from those that can tolerate delay and loss and enable different service levels for different users and applications. What this means is that the enhanced architecture of IP networks must first be able to provide different Classes of Service (CoS) to indicate the needed service level of individual packets or traffic flows being transported and then be able to allocate network resources among these service classes supported based on their QoS requirements.

On the other hand, in the future multiclass Internet, users will have to pay the service providers for their used resources based on the pricing strategies agreed upon in their Service-Level-Agreements (SLAs). Obviously, the pricing strategy will specify the relationship between the price paid by each class of customers and the QoS (e.g., delay, jitter) provided by a service provider. For example, the service provider will receive a certain revenue from a class of customers if the offered QoS is more than the minimal requirement of that class and suffer a certain penalty for failure to meet that. Hence, for a service provider to obtain as high SLA revenues as possible under a given amount of resources, the network elements in the future multiclass IP networks must deploy revenue-aware resource allocation mechanisms based on the SLAs negotiated and committed between the service provider and its customers.

In this work, the network architecture is assumed to support packet classification already and we concentrate on studying novel QoS- and revenue-aware resource allocation mechanisms in both cluster-based Web server systems and IP network nodes. The basic concepts about QoS and SLA are reviewed below.

1.3.1 QoS definitions

Quality of Service (QoS) can be defined in various ways, most of which are equivalent or complimentary. It is the ability of a network element (e.g. a router/switch or Web server) to have some degree of assurance that its traffic and service requirements can be satisfied [80]. It describes the assurance of sufficiently low delay and packet loss for certain types of applications or traffic [97]. The provision of QoS in a network, especially one as large as the global Internet, is not a trivial matter. The cooperation of all network layers from top-to-bottom (i.e. layer one to layer seven of the ISO-OSI model) in addition to every network element from end-to-end (i.e. from sender to receiver) is required.

The need for QoS is becoming progressively more evident daily. For example, many companies rely on the Internet for the day-to-day management of their global enterprises. Many more also utilize the Internet to conduct business (e.g. to place order with their suppliers, interact with customers, etc.). These companies are willing to invest a substantial amount of money for the best possible QoS from the Internet. After all, building and maintaining private high-speed global communication networks would represent a major capital investment for any multinational corporation. On a potentially much larger scale, there are myriads of individual users and organizations that are willing to pay a higher Internet service fee for better QoS in order to take advantage of demanding applications like IP-telephony, video-conferencing and online games. Of course, there will always be a large constituency of users who want to pay nothing at all (excluding Internet service provider fees) for the more traditional services such as email and Web surfing.

As is expected, applications differ in their QoS requirements. A loss-sensitive application is one that cannot accept the loss of data. For example, an ftp file transfer is loss-sensitive since every bit is important while a telephony application can handle the loss of an occasional packet (without retransmission). A delay-sensitive application is dependant on the timely arrival of its packets. File transfers are not delay sensitive although human patience places lower bounds on throughput. Multimedia applications like streaming video and audio can only handle small delays with predictable variation (jitter) and still maintain their utility.

QoS has two components: performance assurance and service differentiation. The former relates to bandwidth, delay, jitter and packet loss. Bandwidth is the fundamental resource since it affects the other three. Service differentiation addresses providing different QoS to different applications with differing requirements. In both cases, this is also a way for Internet Service Providers (ISPs) to increase revenues.

Absolute QoS vs Relative QoS

Performance assurance typically relies on absolute (quantitative) QoS specifications that in some way deal with loss or delay bounds. As an example consider the statements ”no packet will experience a delay greater than 200 ms” or ”packet loss will not exceed 1%”. In the absolute model, if the network can guarantee the user’s requested performance level, he or she will be admitted access to the network. Typically the user will be rejected if the network can not provide the requested assurances. It is an all or nothing proposition. This is typically between specific endpoints.

Relative QoS is precisely what service differentiation addresses. The only assurance from the network is that a higher class will receive better or at least no worse service than any lower class. This model can not offer hard guarantees on delay or packet loss because the amount of service received by a class and the resulting QoS perceived by an application depend on the current network load in each class. This model also requires integration with a pricing or policy-based scheme to make higher classes more costly than lower classes. Disincentives (e.g. high cost) must exist so that everyone does not prefer to use the higher priority classes since this would effectively shrink the relative QoS differences between the classes. The existence of QoS classes supports the dynamic change of classes by a user or application that is able to actively adapt based on the observed performance. For example, a telephony application can dynamically switch classes to find an acceptable QoS at the most economical cost (i.e. no need to pay for a high priority class if a lower priority, and hence cheaper, class meets its requirements). Typically, this model supports arbitrary endpoints.

End-to-End QoS

The QoS protocols of IntServ [8] and DiffServ [7] are capable of furnishing a QoS to flows or aggregates of flows in an end-to-end manner but it is unlikely that either one will ever be ubiquitous enough to do that. Instead, they will likely be used together in the real world to provide end-to-end QoS. By mixing and matching their capabilities in a variety of possible architectures, the goal of end-to-end QoS is nearing reality. Now at least MPLS (Multi-Protocol Label Switching [5, 73]) and the combination of RSVP and DiffServ have been proposed to provide end-to-end QoS. However, end-to-end QoS does not take place only between the network nodes of a network. The end hosts play an integral part in this. Most of the research on QoS is oriented towards network issues such as scheduling and routing. However, network QoS by itself is not sufficient to support end-to-end QoS because bandwidth management and congestion avoidance cannot resolve scheduling or bottleneck problems at the end hosts (Web servers). The FIFO scheduling performed by most servers can undermine any QoS improvements made by the network since a busy Web server can indiscriminately drop high priority network packets. Thus, a true end-to-end Internet QoS solution has as an essential component a Web server with QoS mechanisms to provide overload protection and to enable differentiated services.

In addition, satisfying the end-to-end QoS requirement (delay in particular) of an application traversing a number of network elements is usually addressed by partitioning the end-to-end QoS requirement into the local QoS requirements in each individual element involved and then achieving them respectively [62, 57]. Hence, the problem of allocating resources among multiple service classes based on the local QoS requirements in a single network element is of practical importance.

1.3.2 QoS parameters

The fundamental QoS parameters can be defined as a set of parameters where other sets can always be mapped to [25, 50]. Following this definition, a sufficient set of QoS parameters include delay, throughput, jitter, packet loss ratio and availability [83]. In this work, we put our emphasis particularly on the delay guarantee or differentiation in individual network elements.

Delay is the first QoS parameter that one usually considers. From the user's point of view, end-to-end delay is the measure that matters. Specifically, an end-to-end delay guarantee along a network path can usually be partitioned into the local delay guarantees in each individual network element across that network path [62, 57].

Throughput equals to the bandwidth guarantee of the traffic. It is defined as the effective amount of data transported per unit time. The traffic entitled to this bandwidth guarantee can be modelled simply as constant bit rate, or models like leaky buckets can be used to better capture the nature of traffic distribution. As examples of mechanisms to handle throughput, we refer to two equivalent versions for the Generic Cell Rate Algorithm (GCRA) as defined in ITU-T Recommendation I.371 [37], namely the Virtual Scheduling (VS) and continuous-state Leaky Bucket (LB) algorithm. They are defined for cell based networking, but the algorithms can also be enhanced for packet based networking.

Jitter or delay variation is a critical parameter in real-time applications dealing with applications like VoIP or online live-video. There is a number of ways to define how to calculate jitter. It is quite common to calculate it accumulatively and to have a smoothing equation to give more weight to the latest samples [76, 77].

Packet loss ratio indicates the probability for a packet to get lost in the network.

Availability is usually directly related to the services of the network. It has a close relation to reliability, and there are lots of mechanisms to enhance the availability of the Internet, such as router redundancy protocols VRRP [49] and HSRP [55].

1.3.3 Service Level Agreement

A Service-Level-Agreement (SLA), which is negotiated and committed between service providers and service consumers (either another service providers or common users or both), defines the QoS metrics for each class of service, the anticipated per-class workload intensity and the pricing strategy by which the service payment will be determined. The problem with SLAs is that the rules that are readable and understandable for human beings have to be translated into a form that is readable for machines. Differentiated Services framework [7] suggests that the negotiated and committed SLA may be represented by Service Level Objectives (SLOs) for machine readability. As a summary of terminology related to SLAs, we refer to the definitions of RFC 3198 [83] as follows:

- Service Level Agreement (SLA): The documented result of a negotiation between a customer/consumer and a provider of a service, which specifies the levels of availability, serviceability, performance, operation or other attributes of the service [7].
- Service Level Objective (SLO): Partitions an SLA into individual metrics and operational information to enforce and/or monitor the SLA. SLO may be defined as part of an SLA, an SLS, or in a separate document. It is a set of parameters and their values. The actions of enforcing and reporting monitored compliance can be implemented as one or more policies.
- Service Level Specification (SLS): Specifies the handling of customer's traffic by a service provider. It is negotiated between a customer and a service provider. For instance, in a DiffServ network environment, it defines parameters such as specific Code Points and the Per-Hop-Behavior, profile characteristics and the treatment of the traffic for those Code Points. An SLS is a specific SLA and its SLOs (the individual metrics and operational data to enforce) guarantee the QoS required by customer's traffic.

1.4 Outline of the Dissertation

The rest of the dissertation is organized as follows. Chapter 2 summarizes our contribution for enabling differentiated services in cluster-based Web server systems. The novel *Arrival-Related Dynamic Partitioning* mechanism proposed in publication **PVI** [88] is also presented in clearer format there. The issue of implementing scalable service differentiation in cluster-based Web server systems is addressed systematically in Chapter 3. Chapter 4 presents our contributions for maximizing SLA revenues obtained in a multiclass-supported network node under a given amount of network resources and linear pricing strategy by optimally allocating the resources among the supported service classes. Our derived upper bound on mean packet delay of GPS-based Fair Queueing (FQ) algorithms is first summarized in Chapter 5, then we utilize this upper delay bound to derive the suboptimal resource allocation scheme for maximizing the SLA revenue acquired in a GPS-based network node under a given amount of network resources and flat pricing strategy. Chapter 6 analyzes the problem of maximizing the SLA revenue attained in the hosting of an e-commerce Web site by cluster-based Web server systems. First, the optimal resource partitioning scheme proposed in publication **PXII** [93] under linear pricing strategy is summarized and then the suboptimal resource partitioning scheme is derived, which can achieve the highest SLA revenue for the hosting an e-commerce site under a given amount of server resources and flat pricing strategy. We make the conclusions of this dissertation in Chapter 7. Finally, all my publications are attached at the end of this dissertation.

2 SUPPORTING DIFFERENTIATED SERVICES IN CLUSTER-BASED WEB SERVER SYSTEMS

As we know, to avoid high priority network traffic dropped at the Web server side, a true end-to-end Internet QoS solution has as an essential component: a Web server with QoS mechanisms to provide overload protection and to enable differentiated services. Nowadays Web servers based on clusters of commodity PCs or workstations offer faster and more cost-effective solutions to the increasing performance demands placed on popular Web sites. Therefore, to enable QoS support in such cluster-based Web server systems is of practical importance.

However, many results about enabling QoS support in Web clusters consider Web server farm as the target architecture, where multiple Web sites are co-located on the same platform, e.g., [4]. Few results exist on the topic of providing QoS support in a cluster-based Web server system that hosts a single Web site. Chen et al. evaluated the impact of the request admission control and dispatching mechanisms in [12]. Kanodia et al. have proposed a QoS policy that uses both admission control and performance isolation mechanisms to guarantee different classes of service to have latencies within pre-specified targets [43]. Cardellini et al. described and evaluated some mechanisms that can be integrated into existing Web cluster architectures to support quality of Web services [9]. A dynamic resource partitioning algorithm for static and dynamic Web requests is proposed in [98], where QoS principles are addressed as an optimization problem. However, the work [98] has assumed that the processing rates of the incoming requests have exponential distribution in a Web cluster system, which is not true in the real situation.

This dissertation first addresses the issue of enabling differentiated services in cluster-based Web server systems built upon layer-7 one-way Web cluster architecture. In extending the QoS work in [9], publication **PI** [42] and publication **PIV** [87], we propose a new mechanism [88] for enabling differentiated services in layer-7 one-way cluster architecture, which dynamically partitions the back-end server nodes in a cluster-based Web server system into different server subsets so that each class of requests will be served only by its own assigned server set. More importantly, two different dynamic partitioning algorithms are designed in this mechanism to be used under different arrival load intensities so as to achieve better performances in the case that the system is heavily loaded and does not have enough server resources available. We call this novel mechanism the *Arrival-Related Dynamic Partitioning* mechanism (ARDP mechanism). The simulation results demonstrated that our proposed ARDP mechanism can provide different classes of customers with different QoS targets and succeed in achieving the goals of service differentiation in target cluster-based Web server systems. In this Chapter, the proposed ARDP mechanism and its target Web cluster architecture are explained more clearly. Then the contribution of publication **PVI** [88] is summarized.

2.1 Target Web cluster architecture

A Web cluster consists of a number of commodity workstations or PCs connected by a network. Typically, those server nodes in a Web cluster are connected by a high-speed LAN. As for the target Web cluster architecture in publication **PVI** [88], it consists of a front-end component called Web switch and a number of back-end server nodes, although our proposed mechanism may be applied to other cluster-based architectures. The Web switch acts as the network representative for a Web site built upon the target cluster architecture. In such a way, the authoritative DNS server for the Web site translates the site name into the IP address of its Web switch, which receives all incoming requests destined for the site and makes the distributed nature of the site architecture completely transparent to both the users and the client applications. The Web switch can use various mechanisms to distribute inbound requests among the back-end server nodes. The above Web cluster architecture can be further classified on the basis of whether the data from the back-end server nodes to clients (outgoing data) go through the Web switch. To provide the system with more scalability, the outgoing data will not pass through the Web switch in the target architecture. Figure 2 shows the main components of the target Web cluster architecture.

To enable QoS support in the target cluster architecture, the Web switch must be able to examine the content of a HTTP request and identify its requested service class. Therefore, the target Web cluster architecture is actually the layer-7 one-way architecture according to the classifications in Chapter 1. Additionally, the Web switch of the target cluster architecture will monitor the load information (based on the number of active connections) of each back-end server node and maintain the dynamic server-partitioning map for each supported service class.

2.2 Mechanism design and analysis

To provide service differentiation, we first refer to *service class* to denote the differentiation of incoming requests and users into classes. Given multiple request

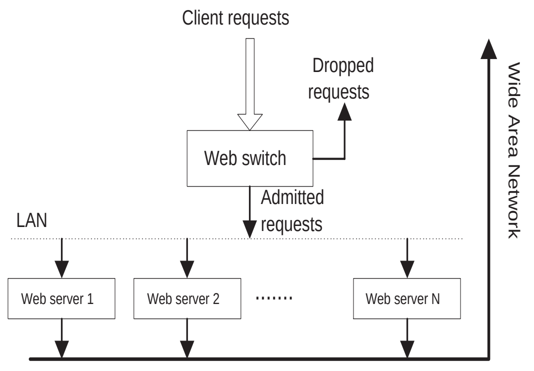


FIGURE 2: Target Web cluster architecture.

classes supported in a Web cluster with a number of homogeneous server nodes, we want to design a QoS-aware (server) resource allocation mechanism which can achieve some basic goals of service differentiation. First, it should be guaranteed that the requests from higher priority class will receive better services than the ones with lower priority, especially when the Web cluster is heavily loaded. Secondly, the admission control mechanism is needed to prevent the Web cluster from being overwhelmed by excessive user requests. These goals are reasonable because they ensure prioritized services in a cluster-based Web server system without overloading it. Admission control has been proposed as the first key mechanism to prevent performance degradation of Web services [13]. It is also considered and included in our ARDP mechanism. We select as a load index the number of active connections that in various implementations of Web clusters has been proven to be the most effective measure [75]. Normally, for a server subset assigned to service class i, the Web switch periodically gathers the number of active connections from each server node, calculates the current sum (denoted by $SumLoad_i$) of the server load in that server subset, and then determines if the incoming requests from that service class will be dropped or accepted. The rejection phase starts when $SumLoad_i$ exceeds a given threshold Thr_i and ends when the load sum returns below the same threshold value. This threshold may be set to $n_i \cdot MaxConn$, where n_i is the number of server nodes in that server subset, and $MaxConn$ denotes the maximum number of active connections which each server can sustain without performance degradation. In our ARDP mechanism, the server nodes from different server subsets have different values of $MaxConn$. More importantly, to provide a flexible mechanism and achieve better performances, especially when a Web system is heavily loaded, in our ARDP mechanism the dynamic partitioning algorithm of server resources can be adjusted based on the arrival rate λ of incoming requests.

To simplify the presentation, we consider without loss of generality three service classes named *High*, *Medium* and *Low* classes supported in a Web cluster. Obviously, the requests belonging to the High class hold the highest priority and the ones belonging to the Low class the lowest priority. Thus we need to partition the server resources in the Web cluster into three server subsets, denoted by

High Set(HS), *Medium Set*(MS) and *Low Set*(LS) and then the incoming requests belonging to one service class will be distributed only to the server nodes in the server subset assigned to that service class. In other words, the server nodes in HS will serve only the requests from the *High* class, the ones in MS will serve only the requests from the *Medium* class and the ones in LS will serve only the requests from the *Low* class. Our ARDP mechanism is concerned with how to dynamically partition the server resources among all supported classes to enable differentiated services in the target Web cluster architecture.

Below we present our ARDP mechanism in detail. Note that for our ARDP mechanism, there is a given threshold λ_{thr} (requests/second) of request arrival rate, whose value is determined by the parameter settings of a Web cluster. Two different dynamic partitioning algorithms may be chosen in our ARDP mechanism by comparing a real or estimated request arrival rate (λ) with the above threshold (λ_{thr}). Let's assume that there are a total of N homogeneous back-end server nodes in a Web cluster built upon our target Web cluster architecture. At time t, the resulted server subsets are denoted by *HS(t)*, *MS(t)* and *LS(t)* and the number of server nodes in the three subsets is denoted by $n_{HS}(t)$, $n_{MS}(t)$ and $n_{LS}(t)$, respectively, where $HS(t) \cap LS(t) = \oslash$, $MS(t) \cap LS(t) = \oslash$, $HS(t) \cap MS(t) = \oslash$ and $n_{HS}(t) + n_{MS}(t) + n_{LS}(t) = N$. Thus the initial number of server nodes in the three subsets is $n_{HS}(0)$, $n_{MS}(0)$ and $n_{LS}(0)$ and the chosen dynamic partitioning algorithm of our ARDP mechanism will update the values of $n_{HS}(t)$, $n_{MS}(t)$ and $n_{LS}(t)$ periodically based on Web switch's server load measurements.

First, the values of $n_{HS}(0)$, $n_{MS}(0)$ and $n_{LS}(0)$ should be determined. Specially, when λ is less than λ_{thr}, we think it indicates that the Web cluster is not heavily loaded and still has enough server resources to be allocated. In this case, the dynamic partitioning algorithm 1 of our ARDP mechanism is chosen and the values of $n_{HS}(0)$ and $n_{MS}(0)$ are set to $\lceil \rho_{HC} \cdot N \rceil$ and $\lceil \rho_{MC} \cdot N \rceil$, respectively, where ρ_{HC} represents the percentage of connection requests belonging to the High class while ρ_{MC} is the percentage of connection requests belonging to the Medium class and $\lceil x \rceil$ means the ceil of x. When λ is not less than λ_{thr}, it indicates that the Web cluster becomes heavily loaded and does not have enough server resources to be allocated. In this case, the dynamic partitioning algorithm 2 of our ARDP mechanism is used and $n_{HS}(0)$ and $n_{MS}(0)$ are set to $\rho_{HC} \cdot N$ and $\rho_{MC} \cdot N$, respectively, which shows that the number of server nodes assigned to a service class does not have to be an integer. That means that when the server resources in a Web cluster are not enough, a server node may actually be assigned to serve multiple service classes simultaneously and each class takes a portion of that server, i.e., the requests from multiple classes will be scheduled to that server in proportion to their share of that server node. In our simulations, we assume that higher class requests may share one server node only with the lowest class requests in order to reduce the complexity of implementation.

Next we explain our ARDP mechanism's method to update the partitioning of server resources among all supported classes periodically. Suppose that the last partitioning of server nodes among HS, MS and LS happened at time t_1 and the resulted number of server nodes in HS, MS and LS was $n_{HS}(t_1)$, $n_{MS}(t_1)$ and

$n_{LS}(t_1)$, respectively. Now at time t_2, the server partitioning will be updated again based on new load measures: the sum of the server loads in HS and MS at time t_2 are $SumLoad_{HS}(t_2)$ and $SumLoad_{MS}(t_2)$, respectively. As mentioned above, in our ARDP mechanism, *MaxConn* has different values for server nodes belonging to HS, MS and LS and they are denoted by $MaxConn_{HS}$, $MaxConn_{MS}$ and $MaxConn_{LS}$, respectively. Then, as for both dynamic partitioning algorithm 1 and 2 of our ARDP mechanism, the process of updating the partitioning of server resources at time t_2 goes as follows.

1. First compute the values of dynamic load thresholds H_{thr1}, H_{thr2} and M_{thr1}, M_{thr2} at time t_2:

$$\begin{aligned} H_{thr1}(t_2) &= n_{HS}(t_1) \cdot MaxConn_{HS}, \\ H_{thr2}(t_2) &= (n_{HS}(t_1) - 1) \cdot MaxConn_{HS}, \\ M_{thr1}(t_2) &= n_{MS}(t_1) \cdot MaxConn_{MS}, \\ M_{thr2}(t_2) &= (n_{MS}(t_1) - 1) \cdot MaxConn_{MS}; \end{aligned}$$

2. If $SumLoad_{HS}(t_2)$ is more than $H_{thr1}(t_2)$ and $n_{LS}(t_1) \geq 2$, the least loaded server node in LS is moved into HS, i.e., $n_{HS}(t_2) = n_{HS}(t_1) + 1$ and $n_{LS}(t_2) = n_{LS}(t_1) - 1$. Otherwise, if $SumLoad_{HS}(t_2) < H_{thr2}(t_2)$ and $n_{HS}(t_1) - n_{HS}(0) \geq 1$, the most loaded one among those serve nodes in HS which were moved-in from LS before will be returned back to LS, i.e., $n_{HS}(t_2) = n_{HS}(t_1) - 1$ and $n_{LS}(t_2) = n_{LS}(t_1) + 1$;

3. If $SumLoad_{MS}(t_2)$ is more than $M_{thr1}(t_2)$ and the resulted $n_{LS}(t_2)$ in Step 2 is more than 2, the least loaded server node in LS is moved into LS, i.e., $n_{MS}(t_2) = n_{MS}(t_1)+1$ and $n_{LS}(t_2) = N-n_{HS}(t_2)-n_{MS}(t_2)$. Otherwise, if $SumLoad_{MS}(t_2) < M_{thr2}(t_2)$ and $n_{MS}(t_1) - n_{MS}(0) \geq 1$, the most loaded one among those serve nodes in MS which were moved-in from LS before will be returned back to LS, i.e., $n_{MS}(t_2) = n_{MS}(t_1) - 1$ and $n_{LS}(t_2) = N - n_{HS}(t_2) - n_{MS}(t_2)$.

Based on the above procedure, the new partitioning of server resources among the High, Medium and Low classes at time t_2 are calculated. Then for an incoming request of a given class, our ARDP mechanism will distribute it to the least loaded server node within its assigned server subset. Note that if a server node in LS is required to move into HS or MS at an updating time point, this server node will receive new requests only from the High or Medium class, while it will continue to serve the requests belonging to the Low class which were already accepted. More importantly, if $SumLoad_{LS}(t_2)$ is more than $n_{LS}(t_2) \cdot MaxConn_{LS}$, all incoming requests belonging to the Low class will be dropped at the Web switch from t_2 till the next updating time. We found in the simulations that only dropping Low class requests is not enough to achieve the above goals of service differentiation. So, in our ARDP mechanism, when $SumLoad_{MS}(t_2) > M_{thr1}(t_2)$ and there are not any more server nodes available in LS which could be moved into MS, the incoming requests belonging to the Medium class will also be dropped at the Web switch.

Now we know that the difference between the dynamic partitioning algorithm 1 and 2 of our ARDP mechanism is only the initial number of the server nodes in HS, MS and LS, i.e., the values of $n_{HS}(0)$, $n_{MS}(0)$ and $n_{LS}(0)$ are different.

2.3 Summarized Contribution

By extending the QoS research work in [9], publication **PI** [42] and publication **PIV** [87], we propose a new mechanism for enabling differentiated services in layer-7 one-way cluster architecture in publication **PVI** [88], which dynamically partitions the back-end server nodes in a cluster-based Web server system into different server subsets so that each class of requests will be served only by its own assigned server set. More importantly, two different dynamic partitioning algorithms are designed in this mechanism to be used under different arrival load intensities so as to achieve better performances in the case that the system is heavily loaded and has no enough server resources available. We call this novel mechanism the *Arrival-Related Dynamic Partitioning* mechanism (ARDP mechanism).

To evaluate the performance of our ARDP mechanism, a simulation environment was built including the queuing model and workload model of the target Web cluster architecture to conduct the simulations. The simulation results in publication **PVI** [88] demonstrate that our proposed mechanism (ARDP) can guarantee that the requests from higher priority class will receive better services than the ones from lower priority class and enable differentiated services in a Web cluster system built upon the target cluster architecture. Furthermore, the simulation results show that the ARDP mechanism works well when the Web cluster system is heavily loaded and does not have enough server resources to be allocated.

3 ENABLING SCALABLE SERVICE DIFFERENTIATION IN CLUSTER-BASED WEB SERVER SYSTEMS

Web servers based on clusters of commodity PCs or workstations offer a cost-effective and scalable solution to the increasing performance demands placed on popular Web sites. As described in Chapter 1.1, a Web cluster architecture typically consists of a specialized front-end server node Web switch and a number of back-end server nodes connected by a high-speed LAN. The Web switch receives all inbound requests destined for a Web site built upon the cluster architecture and then distributes them among the back-end server nodes for serving, thus making the distributed nature of the site architecture completely transparent to both the user and the client application.

To support the differentiated services in such a cluster-based Web server system, the Web switch must be able to examine the content of each inbound HTTP request and identify its requested service class when deciding which back-end node should handle the request. This additional overhead at the Web switch can dramatically limit the scalability of existing cluster architectures. For instance, conventional PC/workstation based front-ends, which support QoS-aware scheduling algorithms, can only scale to a small number of back-end server nodes (less than ten on typical Web workloads [63]). Whereas, extremely fast layer-4 switches [15, 36] are available which can support a large number of back-end server nodes in a single layer-4 Web cluster. However, layer-4 switches can not support QoS-aware request distribution because the latter requires the layer-7 (HTTP) process at the Web switch to examine the requested service class.

In this Chapter, by summarizing our previous QoS research work in publications **PII** [86], **PIV** [87], **PVI** [88] and **PVIII** [89], the issue of implementing scalable service differentiation in cluster-based Web server systems is addressed. First of all, the problems of the existing cluster architectures are examined and then a much more scalable one – the two-level cluster architecture is proposed based on the analysis in [3] and IBM Network Dispatcher [35], which includes a layer-4 switch (at the first-level) and a number of parallel layer-7 sub-clusters (at the second-level). The layer-4 switch acts as the single point of contact with the clients, which is just responsible for making the loads among those parallel sub-clusters balanced and does not perform any QoS-aware distribution. Whereas, the layer-7 front-end of each parallel sub-cluster performs QoS-aware dispatching mechanisms to enable differentiated services in the scalable cluster architecture. Thus, the process overhead at the Web switch of a conventional layer-7 Web cluster is distributed among a number of second-level layer-7 front-ends in the two-level cluster architecture, which results in much improved scalability at the expense of minimal additional latency penalty (the link delay between the layer-4 switch and the layer-7 front-end of each sub-cluster).

Then the relevant scheduling algorithms are investigated for enabling differentiated services in the two-level cluster architecture. Specifically, Pick2X [27] and Round-Robin (RR) algorithms are illustrated as the first-level scheduling algorithm and our Dynamic Partitioning algorithm (DP) is proposed to serve as the second-level scheduling algorithm. Through the simulations, it is demonstrated that the proposed two-level cluster architecture can be significantly more scalable compared to those existing ones by having a highly scalable layer-4 switch at the first-level and more parallel sub-clusters co-existing at the second-level. Theoretically the scalability of the two-level cluster architecture is determined only by the scalability of its layer-4 switch. Nowadays, hardware-based, highly scalable layer-4 switches are commercially available. Thus, our proposed scalable cluster architecture is feasible to implement. Furthermore, the simulation results show that our selected/designed scheduling algorithms (Pick2X-DP and RR-DP) can succeed in enabling differentiated services in the two-level cluster architecture under different workload intensities and parameter settings.

3.1 The design of a novel scalable cluster architecture

In this part, the bottlenecks of the existing cluster architectures are identified and then a new configuration – the two-level cluster architecture is proposed, which is significantly more scalable one with QoS-enabled capability.

M. Aron *et al* [3] separated a layer-7 front-end functionally into two components: dispatcher and distributor as shown in Figure 3. The *dispatcher* is the component that implements the content-aware or QoS-aware request distribution strategy. In other words, its task is to decide which back-end server node should handle an incoming request. The *distributor* component interfaces with the client and is responsible for routing the client requests to their target back-end nodes through either TCP handoff [63] or TCP splicing mechanism [26, 16]. The *server* component represents the server node running at the back-end and serving incoming requests.

Then, M. Aron *et al* argued that the bulk of the overhead at the layer-7 front-end is incurred by the distributor, not the dispatcher component, and the distributor component can be readily distributed as its individual tasks are com-

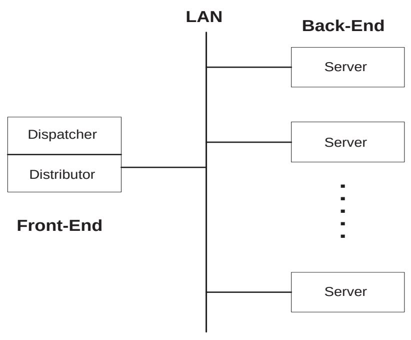


FIGURE 3: The main components of a basic layer-7 cluster architecture.

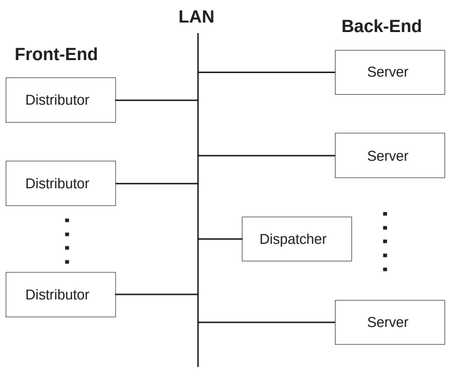


FIGURE 4: The cluster configuration with multiple front-ends.

pletely independent while it is the dispatcher component that typically requires centralized control. Thus, a more scalable content-aware/QoS-aware request distribution can be achieved by distributing the function of the distributor component of a layer-7 front-end over multiple cluster nodes while leaving the function of the dispatcher component on a dedicated node.

In [3], M. Aron *et al* first analyzed a cluster configuration where the distributor component is distributed across several front-end nodes while the dispatcher component resides on a dedicated node as shown in Figure 4. In such a configuration with multiple front-end nodes, a choice must be made as to which front-end should receive an incoming client request. This choice can be made either explicitly by the user with strategies like *mirroring*, or in a client transparent manner using DNS round-robin. However, these approaches are known to lead to poor load balancing [35], in this case among the front-end nodes. They argued that another drawback of the cluster configuration shown in Figure 4 is that efficient partitioning of cluster nodes into either front-end or back-end nodes depends upon the workload and is not known *a priori*.

Then, an alternate cluster design (in Figure 5) where the distributor components are co-located with the back-end server components was proposed by M. Aron *et al* in [3]. In the cluster configuration of Figure 5), the function of the distributor component is decentralized to each back-end server node, eliminating the bottleneck imposed by a centralized distributor, and the dispatcher component

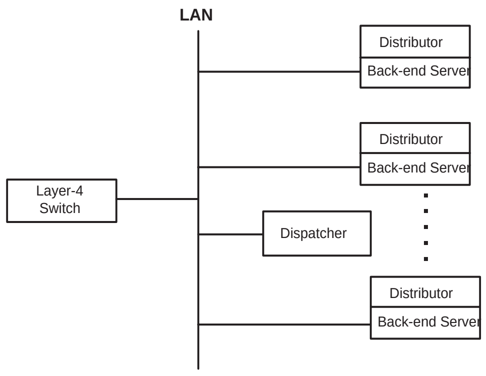


FIGURE 5: The cluster configuration with co-located distributors and back-end server nodes.

still resides on a dedicated node and perform the content-aware/QoS-aware request distribution policies. Moreover, a layer-4 switch acts as the front-end of the cluster configuration, making the load among those decentralized distributors balanced. The experimental results in [3] indicated that the cluster configuration in Figure 5) is much more scalable than conventional layer-7 cluster architectures. But, as the dispatcher component is still centralized, its performance determines the scalability of the cluster configuration of Figure 5). They discovered that the key to achieving greater cluster scalability is to reduce the communication overhead between the dispatcher node and other cluster nodes.

Actually, the dispatcher component of a layer-7 switch may also be decentralized especially in the scenario that the Web cluster hosts only a single Web site. Based on the above analysis and IBM Network Dispatcher [35], we construct a novel cluster configuration which can performs QoS-aware request distribution strategies while being much more scalable than the above one proposed by M. Aron *et al.* It is named the *two-level cluster architecture* shown in Figure 6. Specifically, the two-level cluster architecture consists of a very fast layer-4 switch at the first-level and a number of parallel layer-7 sub-clusters co-existing at the second-level. The layer-4 switch acts as the single point of access to the two-level architecture, which receives all inbound client requests and then distributes them among the parallel sub-clusters using a simple QoS-blind scheduling algorithm (referred to as the first-level scheduling algorithm) to prevent it from becoming the bottleneck of the architecture. Moreover, with this single switch the distributed nature of the two-level cluster architecture becomes completely transparent to the clients. Each parallel sub-cluster contains a layer-7 front-end and multiple back-end server nodes connected via a high-speed LAN. Within each sub-cluster, its layer-7 front-end assigns incoming requests across its back-end server nodes by a QoS-aware scheduling algorithm (referred to as the second-level scheduling algorithm) and furthermore the TCP handoff mechanism [63] is used to enable the back-end nodes respond to the clients directly without passing through the front-end nodes as an intermediary.

In this case, the layer-7 front-end is not separated into the distributor component and the dispatcher component. However, as multiple sub-clusters co-exist

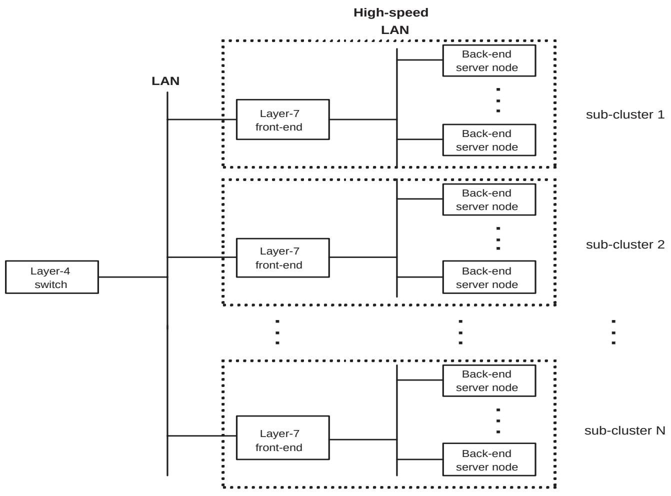


FIGURE 6: The two-level cluster architecture.

at the second-level of the cluster architecture, the proposed cluster configuration functions as if both the distributor component and the dispatcher component are decentralized. Since the layer-7 front-end in each sub-cluster distributes the assigned requests only across the back-end nodes of its sub-cluster, the communication overhead at the layer-7 front-end is significantly reduced and is no longer the key impact on the cluster's scalability. The scalability of the two-level cluster architecture will be determined only by the performance of its layer-4 switch. Nowadays hardware-based, highly scalable layer-4 switches are commercially available. Therefore, our proposed cluster architecture may become significantly more scalable compared with those existing ones by using a highly scalable layer-4 switch at the first-level and having more parallel sub-clusters co-existing at the second-level. Additionally, the performances of both the layer-4 switch and the layer-7 dispatcher will impact on the scalability of the cluster configuration proposed by M. Aron *et al* and its scalability is determined by the smaller one of the above performances. Hence, the two-level cluster architecture is also more scalable than the one proposed by M. Aron *et al* in [3].

Moreover, as each parallel sub-cluster co-existing at the second-level is actually a layer-7 cluster, QoS-aware request distribution can readily be supported in the two-level architecture by the second-level (QoS-aware) scheduling algorithms. Note that in the two-level cluster architecture, each sub-cluster may be required to report its load information to the layer-4 switch periodically. Moreover, the layer-7 front-end of each sub-cluster maintains the load information of its back-end server nodes based on the number of active client connections handled by each back-end node. Note also that although in this Chapter, homogeneous back-end server nodes are deployed in the two-level cluster architecture, heterogeneous back-end nodes can also be supported in the proposed cluster architecture by adjusting the parameters of both first-level and second-level scheduling algorithms.

3.2 Enabling service differentiation in the novel scalable cluster architecture

This section investigates the scheduling algorithms which can be used in the proposed scalable cluster architecture – the two-level cluster architecture to enable service differentiation. First of all, a *service class* is defined as a category of client requests that enjoy the same level of QoS support. Thus, client requests belonging to different service classes will receive differentiated services. Note that the service class of a client request may be classified based on client identities or service types.

Given multiple service classes supported in the two-level cluster architecture consisting of a layer-4 switch and a number of parallel layer-7 sub-clusters, the first-level and second-level scheduling algorithms should be chosen or designed to achieve the basic goals of service differentiation in the architecture. We recall that the layer-4 switch does not perform any QoS-aware request distribution and it just distributes the inbound requests among the parallel sub-clusters to have their loads balanced by a simple QoS-blind scheduling algorithm. Recall also that each parallel sub-cluster may be required to report its load information to the layer-4 switch periodically (the period referred to as T_{post}). Hence, the layer-4 switch may utilize the load information of sub-clusters to select the target sub-cluster for inbound requests.

In this section we investigate two kinds of algorithms which could be used as the first-level scheduling algorithm: the first kind of algorithms select the target sub-cluster without considering the load distribution among those parallel sub-clusters while the second kind will consider that. The *Round-Robin* algorithm is chosen as the representative of the first kind of algorithms to be studied. Although the least loaded approach is commonly used in commercial products, selection of the least loaded server can result in pathologically poor performance. This poor performance is due to the so-called "herd effect" that is well known in distributed systems [61, 20]. The Pick-KX method proposed in [27] is one solution to deal with the "herd effect". Here the Pick-2X algorithm [27] is investigated as the representative of the second kind of algorithms, which means that, in a T_{post} interval, whenever a client request reaches the two-level cluster architecture, the target sub-cluster of the request is selected from two randomly chosen sub-clusters by weighting their latest reported loads. In other words, given two randomly chosen sub-clusters, the probability P_1, which an incoming request selects the first sub-cluster of them, is $P_1 = L_2/(L_1 + L_2)$ and the probability P_2, which the incoming request selects the second one of them, is $P_2 = L_1/(L_1 + L_2)$, where L_i indicates the latest reported load of the *ith* one of the two randomly chosen sub-clusters. For instance, given two sub-clusters with loads of 10 and 5, the probabilities of selection would be 1/3 and 2/3, respectively. Note that the number of active connections is selected as the load index, which has been proven to be the most effective measure in various implementations of Web clusters [75]. The Round-Robin algorithm selects the target sub-cluster cyclically without considering the load distribution among those parallel sub-clusters.

As the layer-4 switch just performs a simple QoS-blind scheduling algorithm to prevent it from becoming the bottleneck of our proposed architecture, the front-end node of each parallel layer-7 sub-cluster should employ a QoS-aware scheduling algorithm to enable differentiated services in the two-level cluster architecture. As we know, the dynamic partitioning algorithms proposed in Chapter 2 can succeed in enabling service differentiation in a existing layer-7 cluster architecture. In this section, they are combined and extended to a more general one named *Dynamic Partitioning* algorithm, which is to be used by the front-end of each layer-7 sub-cluster for implementing differentiated services in the two-level cluster architecture. Furthermore, the Dynamic Partitioning algorithm can enable service differentiation under different workload intensities and different system parameter settings, which will be demonstrated in the following simulations. Since the Dynamic Partitioning algorithm is performed only within each parallel sub-cluster, below we explain how it works in the scope of one sub-cluster.

Let us assume that there are a total of M service classes supported in the two-level cluster architecture and each parallel sub-cluster in the architecture can provide all services which the whole architecture offers. Assume also that there are N homogeneous back-end server nodes in a layer-7 sub-cluster. Then, within this layer-7 sub-cluster, the main idea of the Dynamic Partitioning algorithm is to dynamically partition the N back-end server nodes into M disjoint server subsets so that each class of requests reaching this sub-cluster will be served only by the server subset assigned to that class.

Specifically, $s_i(t)$ is used to denote the server subset assigned to class i at time t (in this Chapter, i=1 indicates the highest class and i=M the lowest class) and $n_i(t)$ is used to denote the number of back-end server nodes in $s_i(t)$. Then the two equations, $s_i(t) \cap s_j(t) = \oslash$, $i, j \in [1, M]$, $i \neq j$ and $\sum_{i=1}^{M} n_i(t) = N$, hold for each time t. In addition, the Dynamic Partitioning algorithm sets the initial number of back-end server nodes in $s_i(t)$ as follows:

$$n_i(0) = \gamma_i \rho_i N$$

where $i \in [1, M]$, ρ_i represents the percentage of the client requests belonging to class i with $\sum_{i=1}^{M} \rho_i = 1$ and γ_i is a constant factor with $\sum_{i=1}^{M} \gamma_i \rho_i = 1$. Note that the number of back-end server nodes assigned to a service class does not have to be an integer. That means a back-end node may actually be assigned to multiple service classes and each class takes a portion of that node, i.e., the requests from the multiple classes will be scheduled to that node in proportion to their share of that back-end server node. Note also that the initial number of back-end server nodes in $s_i(t)$ is the guaranteed portion of server resources assigned to class i except for the lowest class, i.e., for each service class i other than the lowest one, $n_i(t) \geq n_i(0)$ for each time t. Whereas, for the lowest class M, a constant factor ϵ is set by our Dynamic Partitioning algorithm to denote the guaranteed minimum number of back-end server nodes in server subset $s_M(t)$, i.e., $n_M(t) \geq \epsilon$ at each time t. Obviously, the value of ϵ depends on the QoS requirements of the lowest class and $\epsilon \in [0, n_M(0)]$. Additionally, the value of the factor γ_i should also be set carefully based on the QoS requirements of class i.

Recall that the front-end node of each parallel sub-cluster maintains the load information of its back-end server nodes. Based on the load measurement, the Dynamic Partitioning algorithm will update the server partitioning among the M service classes periodically (this period referred to as T_{update}). Additionally, $MaxConn_i$ denotes the maximum number of active connections from class i that a back-end server node can handle without performance degradation. Then, $MaxConn_i \leq MaxConn_j$ needs hold to enable differentiated services in a layer-7 sub-cluster if class i is higher priority than class j. Note that the concrete value of $MaxConn_i$ should be considered together with γ_i and set carefully based on the QoS requirements of class i.

Below we present how our Dynamic Partitioning algorithm updates the server subsets periodically in a sub-cluster. Suppose that t_1 is the last updating time, the resulted server subsets in the layer-7 sub-cluster are $s_1(t_1)$,...,$s_i(t_1)$,...,$s_M(t_1)$ and the number of back-end server nodes in them is $n_1(t_1)$,...,$n_i(t_1)$,...,$n_M(t_1)$, respectively, for the M service classes. Now at time $t_2 = t_1 + T_{update}$, the server partitioning among the M classes will be updated based on new load measurements (the sum of server loads in server subset $s_i(t_1)$ at time t_2 is denoted by $SumLoad_i(t_2)$). Then the server subsets assigned to the M service classes are updated at time t_2 by using the following pseudo-code on next page.

Thus, at time t_2, M new disjoint server subsets are created: $s_1(t_2)$, ..., $s_i(t_2)$, ..., $s_M(t_2)$. For each service class i, the sum of the server loads in $s_i(t_2)$ at time t_2 can be calculated based on the above load measurements and denoted by $SumLoad_i^{'}(t_2)$. Then, during the time period $[t_2,\ t_2 + T_{update})$, the admission control mechanism of the Dynamic Partitioning algorithm operates on class i requests ($i \in [1, M]$) as follows: if $SumLoad_i^{'}(t_2) > \beta_i n_i(t_2) MaxConn_i$, the incoming requests belonging to class i will all be dropped by the layer-7 front-end of the sub-cluster; otherwise, all class i requests are accepted into the sub-cluster. As the M disjoint server subsets are updated periodically, the conditions in the admission control mechanism will also be updated periodically. Note that β_i is another constant factor, which value should also be set based on the QoS requirements of class i.

3.3 Simulations and results

3.3.1 Simulation environment

To evaluate the performances of our proposed scalable cluster architecture and relevant scheduling algorithms, a simulation environment was built to hold the simulations including the system model and workload model. First of all, in the simulations we focus on enabling differentiated services in the proposed two-level cluster architecture and the details of the external networks which connect the clients to the cluster architecture did not be modelled because that will not affect the main evaluation conclusions. As mentioned above, within each parallel sub-cluster of the two-level architecture, all the back-end server nodes are connected with the layer-7 front-end via a high-speed LAN. However, the layer-4 switch may connect to the layer-7 front-ends of those sub-clusters through a normal LAN. Hence, the transmission delays within each sub-cluster can be neglected due to high-speed LAN connections while the link delays between the layer-4 switch and

```
Initialize: $s_M(t_2) \leftarrow s_M(t_1)$;
            $n_M(t_2) \leftarrow n_M(t_1)$;
for $i \leftarrow 1$ to $M$ do
    if $i \neq M$ then
        calculate the values of the two dynamic load thresholds of class $i$ at
        time $t_2$;
        $Thr1_i(t_2) = n_i(t_1)MaxConn_i$;
        $Thr2_i(t_2) = (n_i(t_1) - 1)MaxConn_i$;
        if $SumLoad_i(t_2) > Thr1_i(t_2)$ and $n_M(t_2) \geq \epsilon + 1$ then
            the least loaded server node in $s_M(t_2)$ is moved into server subset
            $s_i(t_1)$ to form server subset $s_i(t_2)$ and server subset $s_M(t_2)$ is
            refreshed;
            $n_i(t_2) = n_i(t_1) + 1$;
            $n_M(t_2) = n_M(t_2) - 1$;

        else if $SumLoad_i(t_2) < Thr2_i(t_2)$ and $n_i(t_1) > n_i(0)$ then
            the most loaded one among those server nodes in server subset
            $s_i(t_1)$ which were moved in before is returned back to $s_M(t_2)$ to
            form server subset $s_i(t_2)$ and server subset $s_M(t_2)$ is refreshed;
            $n_i(t_2) = n_i(t_1) - 1$;
            $n_M(t_2) = n_M(t_2) + 1$;
        end
    else
        $n_M(t_2) = N - \sum_{i=1}^{M-1} n_i(t_2)$;
        thus server subset $s_M(t_2)$ is formed;
    end
end
```

Pseudo-code: For updating the server subsets assigned to the M service classes periodically.

the layer-7 front-ends must be considered.

Recall that in the two-level cluster architecture, the TCP handoff mechanism [63] is used to enable the back-end server nodes respond to the clients directly without passing through the front-ends as an intermediary. Thus each parallel sub-cluster in the two-level architecture can be abstracted as a queuing system shown in Figure 7. Additionally, the processing delay at the layer-4 switch can also be neglected due to the fact that in a Web environment the client-to-server packets are typically much less than the server-to-client packets. Based on the analysis in Chapter 2, the service time of a client request at the back-end server nodes is proportional to the size of its requested Web object, i.e., $ServiceTime = L/C + \sigma$, where L denotes the size of the requested Web object in bytes, C is the effective processing capacity of a back-end Web server and σ a constant factor. In the following simulations, the homogeneous back-end server node is used and parameters C and σ were set to 5.95MB/s and 4.2ms, respectively, based on the

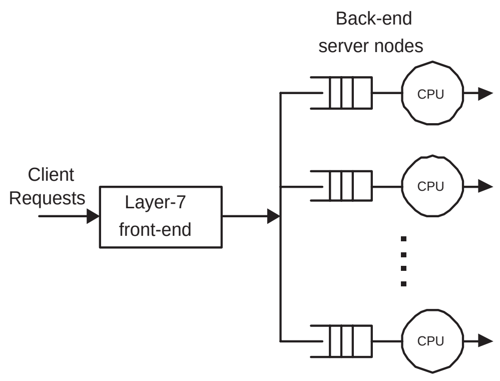


FIGURE 7: Queuing model of a parallel sub-cluster in the two-level cluster architecture.

experimental measurements.

Furthermore, throughout the following simulations, the Pick2X [27] and Round-Robin (RR) algorithms were used as the first-level scheduling algorithm, respectively, and our proposed Dynamic Partitioning algorithm (DP) was deployed as the second-level scheduling algorithm. The well-known Least Load algorithm (LL) was sometimes also used as the second-level scheduling algorithm to evaluate the performances of our DP algorithm. The Least Load algorithm just chooses the least loaded back-end server node within a parallel sub-cluster to serve the requests assigned to that sub-cluster without considering the requested service classes, which also does not include any admission control mechanism.

For actual Web workloads, it is recognized that Web object sizes are distributed with a heavy tail. Here the Bounded Pareto distribution ($BP(p, q, \alpha)$) [19] is used to model the heavy-tailed characteristic of Web objects. Specifically, the mean size of Web objects is set to 21KB as measured in [2] and p=1KB and q=10MB are chosen as the reasonable minimum and maximum Web object size, respectively. The resulting α=0.8037 is within the range of α values measured in [1] and [18]. The arrival process of client requests destined for the two-level cluster architecture was modelled by Poisson distribution.

3.3.2 Parameters and methods of evaluation

The parameters and methods of evaluation in the simulations are summarized in this part. Throughout the following simulations, the *High*, *Medium* and *Low* classes were illustrated and supported in the two-level cluster architecture; moreover, we deployed eight homogeneous back-end server nodes within each parallel sub-cluster of the two-level architecture and the values of parameters γ and $MaxConn$ were set as follows: $\gamma_{HS} = 1$, $\gamma_{MS} = 1$, $\gamma_{LS} = 1$, $MaxConn_{HS} = 15$, $MaxConn_{MS} = 40$ and $MaxConn_{LS} = 60$. To investigate the impacts of link delays between the layer-4 switch and parallel sub-clusters, two groups of link delays were deployed for each simulation. As we concentrate on enabling differentiated services, in particular, differentiated request delays, in the two-level cluster architecture, the *90-percentile*

Table 1: The parameters deployed in the first and second set of simulations.

	For the first set of simulations	For the second set of simulations
The number of parallel sub-clusters	4	8
The percentage of inbound requests belonging to different service classes	ρ_{HS}=20% ρ_{MS}=30%, ρ_{LS}=50%	ρ_{HS}=20% ρ_{MS}=30%, ρ_{LS}=50%
Group 1 of link delays	$LinkDelay_1$=50ms $LinkDelay_2$=55ms $LinkDelay_3$=60ms $LinkDelay_4$=57ms	$LinkDelay_1$=50ms $LinkDelay_2$=55ms $LinkDelay_3$=60ms $LinkDelay_4$=57ms $LinkDelay_5$=65ms $LinkDelay_6$=54ms $LinkDelay_7$=62ms $LinkDelay_8$=52ms
Group 2 of link delays	$LinkDelay_1$=100ms $LinkDelay_2$=105ms $LinkDelay_3$=110ms $LinkDelay_4$=115ms	$LinkDelay_1$=100ms $LinkDelay_2$=105ms $LinkDelay_3$=110ms $LinkDelay_4$=115ms $LinkDelay_5$=112ms $LinkDelay_6$=103ms $LinkDelay_7$=108ms $LinkDelay_8$=120ms
Arrival rates (requests/s)	3200, 4000, 4800, 5600	6400, 8000, 9600, 11200
T_{post} (ms)	500	500
T_{update} (ms)	100	100
β	β_{HS}=3, β_{MS}=1, β_{LS}=1	β_{HS}=3, β_{MS}=1, β_{LS}=1

of request delay [51, 9] and the *percentage of dropped requests* were chosen as the performance metrics. Note that the notion of request delay in this section means the delay which an accepted request experiences in the proposed cluster architecture, including link delay between the layer-4 switch and its target sub-cluster, possible queuing delay and its service time. For example, in this case, *90-percentile of request delay* of the Low class means the value which 90% among the delays of all accepted Low class requests are exactly less than.

Specifically, the first set of simulations were carried out with the parameters in Table 1 to investigate whether our selected/designed scheduling algorithms can enable differentiated services in the proposed cluster architecture under different workload intensities. Then the number of the parallel sub-clusters in the two-level cluster architecture was doubled in the second set of simulations to evaluate the ability of our proposed cluster architecture and selected/designed scheduling algorithms supporting scalable service differentiation.

Next, the third and fourth sets of simulations were made to investigate the impacts of parameters T_{post} and T_{update} on system performances, respectively, as

TABLE 2: The parameters deployed in the third and fourth set of simulations.

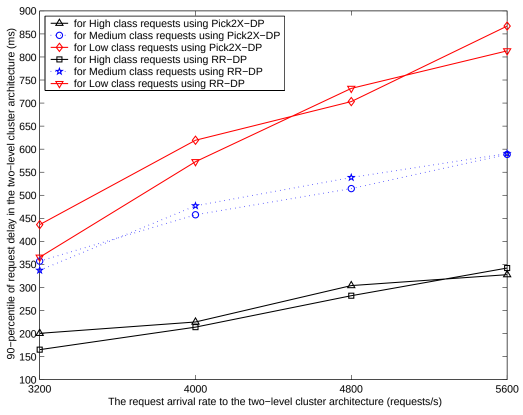
	For the third set of simulations	For the fourth set of simulations
The number of parallel sub-clusters	4	4
The percentage of inbound requests	ρ_{HS}=20%, ρ_{MS}=30%, ρ_{LS}=50%	ρ_{HS}=20%, ρ_{MS}=30%, ρ_{LS}=50%
Group 1 of link delays	$LinkDelay_1$=50ms, $LinkDelay_2$=55ms $LinkDelay_3$=60ms, $LinkDelay_4$=57ms	$LinkDelay_1$=50ms, $LinkDelay_2$=55ms $LinkDelay_3$=60ms, $LinkDelay_4$=57ms
Group 2 of link delays	$LinkDelay_1$=100ms, $LinkDelay_2$=105ms $LinkDelay_3$=110ms, $LinkDelay_4$=115ms	$LinkDelay_1$=100ms, $LinkDelay_2$=105ms $LinkDelay_3$=110ms, $LinkDelay_4$=115ms
Arrival rates (requests/s)	4000, 5600	4000, 5600
T_{post} (ms)	400, 800, 1200, 1600	500
T_{update} (ms)	100	80, 160, 240, 320
β	β_{HS}=3, β_{MS}=1, β_{LS}=1	β_{HS}=3, β_{MS}=1, β_{LS}=1

TABLE 3: The parameters deployed in the fifth set of simulations.

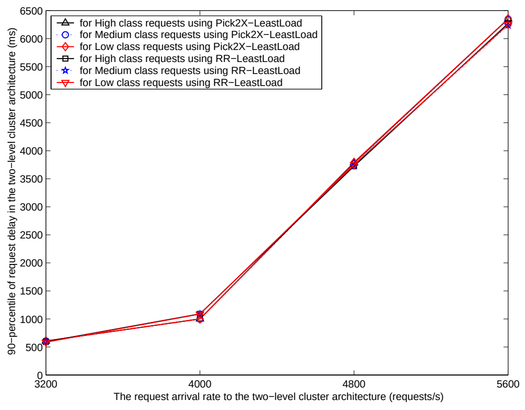
The number of parallel sub-clusters	4
The percentage of the High class requests	ρ_{HS}=15%, 30%, 45%
Group 1 of link delays	$LinkDelay_1$=50ms, $LinkDelay_2$=55ms $LinkDelay_3$=60ms, $LinkDelay_4$=57ms
Group 2 of link delays	$LinkDelay_1$=100ms, $LinkDelay_2$=105ms $LinkDelay_3$=110ms, $LinkDelay_4$=115ms
Arrival rates (requests/s)	4000, 5600
T_{post} (ms)	500
T_{update} (ms)	100
β	β_{HS}=2, β_{MS}=1, β_{LS}=1

they are crucial for the implementation of service differentiation in the two-level architecture. Note that parameter T_{post} is not valid when the Round-Robin algorithm is used as the first-level scheduling algorithm. Their deployed simulation parameters were summarized in Table 2, where it is noticed that a normal workload intensity (4000 requests/s) and a stressing one (5600 requests/s) were both used to further evaluate whether the impacts of T_{post} and T_{update} on system performances are consistent under different workload levels.

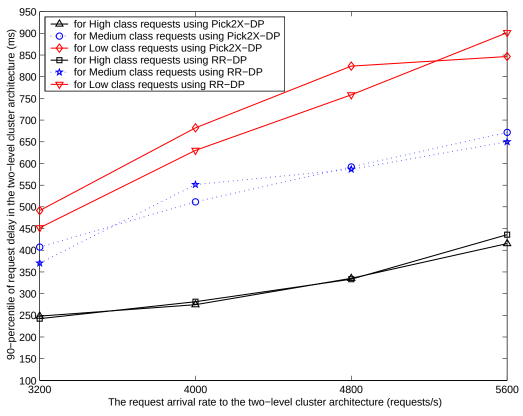
In the end, the fifth set of simulations was carried out to evaluate the sensitivity of system performances to the percentage of High class requests. The deployed simulation parameters are presented in Table 3. It is shown that the percentage of High class requests increases from 15% to 45%, which is reasonable in the real world. To reduce the degrees of freedom, the ratio between the Medium class requests and the Low class requests was kept at 3/5. Similarly, in the fifth set of simulations, a normal workload intensity (4000 requests/s) and a stressing one



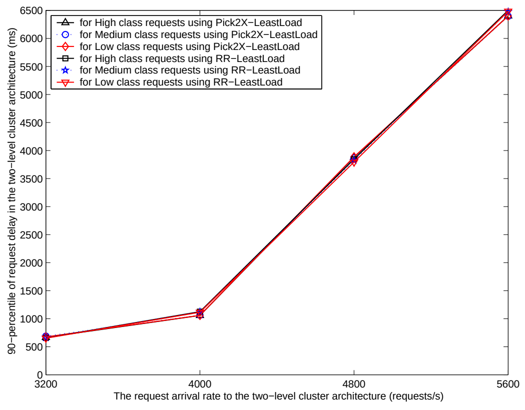
(a)



(b)



(c)



(d)

FIGURE 8: For the first set of simulations: (a) 90-percentile of request delay when using Pick2X-DP, RR-DP and Group 1 of link delays; (b) 90-percentile of request delay when using Pick2X-LeastLoad, RR-LeastLoad and Group 1 of link delays; (c) 90-percentile of request delay when using Pick2X-DP, RR-DP and Group 2 of link delays; (d) 90-percentile of request delay when using Pick2X-LeastLoad, RR-LeastLoad and Group 2 of link delays.

(5600 requests/s) were both used to further investigate whether the sensitivity of system performances to the percentage of High class requests is consistent under different workload levels.

3.3.3 Simulation results

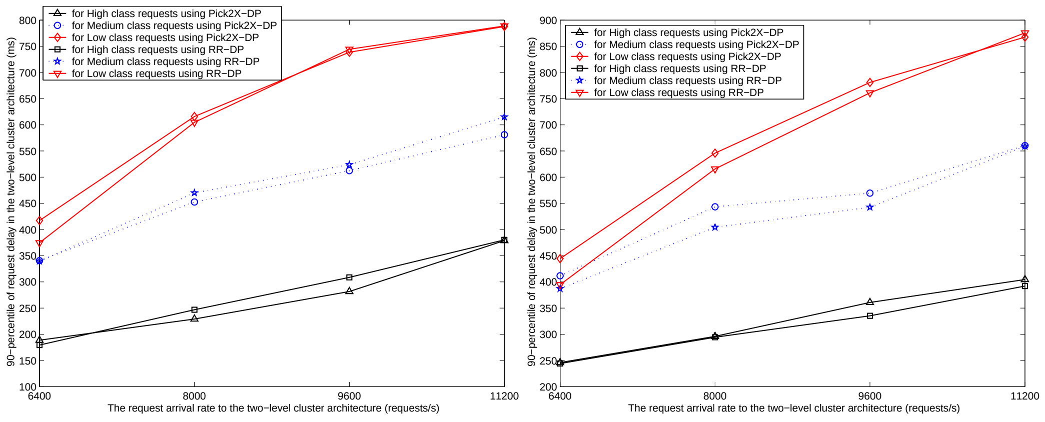
The simulation results for the first set of simulations are presented in Figure 8 and Table 4. It is shown that differentiated services are enabled in the two-level cluster architecture as long as the Dynamic Partitioning algorithm is used as the second-level scheduling algorithm whereas the Least Load algorithm can not achieve any service differentiation at all. In other words, both Pick2X-DP and RR-DP combinations can succeed in enabling QoS support in our proposed scalable cluster

Table 4: The percentage of dropped requests in the first set of simulations.

	The group 1 of link delays is deployed					
	when using the Pick2X-DP combination			when using the RR-DP combination		
the offered arrival rate (request/s)	for the High class	for the Medium class	for the Low class	for the High class	for the Medium class	for the Low class
3200	0	0.44%	1.87%	0	0	0.56%
4000	0	2.26%	10.89%	0.28%	3.71%	11.58%
4800	0	7.87%	29.78%	1.66%	9.06%	32.34%
5600	1.55%	15.59%	49.48%	2.57%	18.94%	46.76%
	The group 2 of link delays is deployed					
	when using the Pick2X-DP combination			when using the RR-DP combination		
the offered arrival rate (request/s)	for the High class	for the Medium class	for the Low class	for the High class	for the Medium class	for the Low class
3200	0	0.34%	2.16%	0	0.24%	1.23%
4000	0	3.03%	13.28%	0	3.04%	12.71%
4800	0.23%	9.55%	34.05%	0	10.38%	30.81%
5600	1.57%	18.07%	48.60%	1.61%	20.32%	47.45%

architecture, however Pick2X-LL and RR-LL combinations both failed to achieve that. Furthermore, we notice that although the link delays in group 2 are almost doubled compared with the ones in group 1, the service differentiation is still enabled in the two-level architecture when the group 2 of link delays are deployed. Hence, we can conclude that the differentiated services can always be achieved in the two-level cluster architecture by our selected/designed scheduling algorithms as long as the link delay between the layer-4 switch and the layer-7 front-end of each parallel sub-cluster is not the dominant contributor to the request delay, otherwise, new first-level scheduling algorithms must be designed for the layer-4 switch. Fortunately, the above assumption about the link delays is true for most cluster-based Web server systems built upon the two-level cluster architecture as the layer-4 switch is connected to all parallel sub-clusters also via a fast LAN in most cases nowadays.

In addition, the sub-figures (b) and (d) in Figure 8 indicate that when the arrival rate of client requests reaches 4800 requests/s or more, the simulation-generated request delays by the Least Load algorithm will become much larger (over 3700 ms), which shows that the two-level cluster architecture is overwhelmed by excessive client requests. Whereas, the sub-figures (a) and (c) in Figure 8 show that the proposed cluster architecture works well and differentiated services is enabled by our Dynamic Partitioning algorithms (DP) even under the workload intensity of 5600 requests/s. It is due to the admission control mechanism in our Dynamic Partitioning algorithm, which guarantees the implementation of service differentiation and makes the request delay within acceptable ranges even under overloaded situations. By having more parallel sub-clusters co-existing in the two-



(a) (b)

Figure 9: For the second set of simulations: (a) 90-percentile of request delay when using Pick2X-DP, RR-DP and Group 1 of link delays; (b) 90-percentile of request delay when using Pick2X-DP, RR-DP and Group 2 of link delays.

Table 5: The percentage of dropped requests in the second set of simulations.

	The group 1 of link delays is deployed					
	when using the Pick2X-DP combination			when using the RR-DP combination		
the offered arrival rate (request/s)	for the High class	for the Medium class	for the Low class	for the High class	for the Medium class	for the Low class
3200	0	0.07%	0.60%	0	0.45%	0.96%
4000	0	2.29%	11.24%	0	2.55%	13.57%
4800	0.60%	9.08%	31.36%	0.49%	8.36%	31.53%
5600	2.43%	16.81%	46.05%	2.43%	17.96%	46.05%
	The group 2 of link delays is deployed					
	when using the Pick2X-DP combination			when using the RR-DP combination		
the offered arrival rate (request/s)	for the High class	for the Medium class	for the Low class	for the High class	for the Medium class	for the Low class
3200	0	0.28%	0.44%	0	0	1.21%
4000	0	3.68%	11.38%	0.69%	2.15%	11.60%
4800	0.88%	10.46%	30.07%	0.75%	8.27%	29.80%
5600	1.92%	16.96%	44.85%	2.47%	17.47%	47.45%

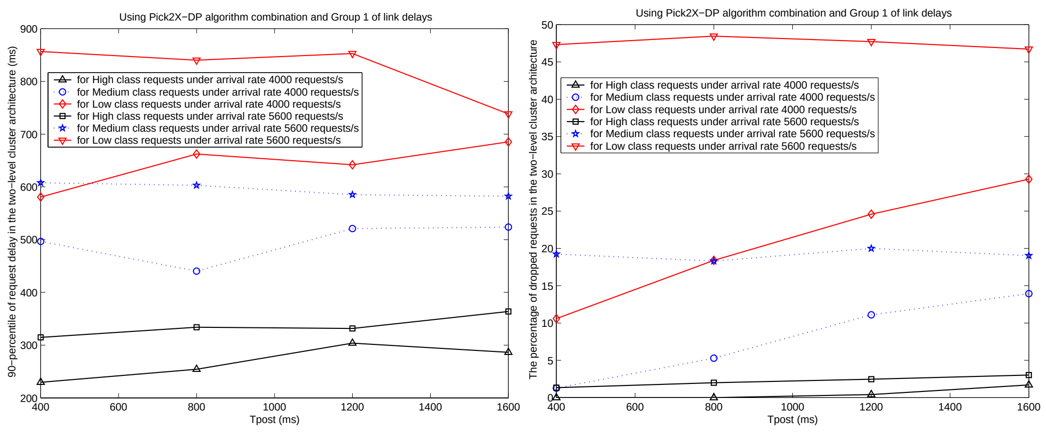
level cluster architecture, it will be able to serve even much heavier workloads while differentiated services can still be enabled using our selected/designed scheduling algorithms, which is shown below.

The simulation results of the second set of simulations are presented in Figure 9 and Table 5, which clearly indicate that the two-level cluster architecture with doubled parallel sub-clusters (i.e., the number of parallel sub-clusters is increased from 4 to 8) works well and service differentiation is also successfully achieved in it under the doubled workload intensities (up to 11200 requests/s) by both Pick2X-DP and RR-DP algorithm combinations no matter which group of link delays are used. Therefore, it is demonstrated that the proposed two-level cluster architecture can be significantly more scalable compared to those existing ones by having more parallel sub-clusters co-existing at the second-level of the architecture. Moreover, theoretically the scalability of the two-level cluster architecture is determined only by the scalability of its layer-4 switch.

Figures 10-11 show the simulation results about the impacts of T_{post} and Figures 12-15 show the ones about the impacts of T_{update}, which demonstrate that the differentiated services can be enabled in the two-level cluster architecture by our selected/designed scheduling algorithms (i.e., Pick2X-DP or RR-DP algorithm combination) under different settings of T_{post} and T_{update}, different workload intensities and different group of link delays. Moreover, Figures 10-11 show that the simulation-generated request delays and dropped rates of High, Medium and Low classes all fluctuate within small ranges when T_{post} increase from 400ms to 1600ms. The above small fluctuation of service performances is achieved due to the admission control mechanism in our Dynamic Partitioning algorithm, otherwise, the service performances (i.e., request delay and dropped rate) may vary a lot and the service differentiation can not be guaranteed.

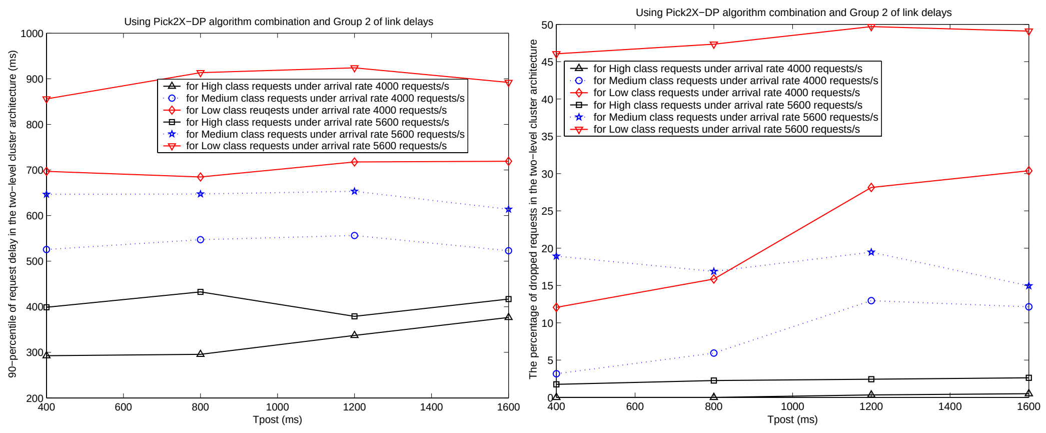
In addition, Figures 12-15 indicate that the simulation-generated request delays and dropped rates of High, Medium and Low classes almost always increase along with the increment of T_{update}. The reason is that for the Dynamic Partitioning algorithm, T_{update} determines the updating rate of the server subsets in each parallel sub-cluster and thus affects the request delays and dropped rates directly. A faster updating rate of those server subsets in a parallel sub-cluster means that the server resources in the sub-cluster can be better utilized to serve the incoming requests from different service classes, hence T_{update} should be set to a smaller value to achieve better QoS performances. As within each parallel sub-cluster, the front-end is connected to all back-end server nodes via a high-speed LAN, it is feasible for T_{update} to have a small value in the two-level cluster architecture. Furthermore, from the simulation figures, we notice that the above impacts of T_{post} and T_{update} on system performances are almost consistent under different workload intensities.

Finally, the simulation results of the fifth set of simulations, which investigated the sensitivity of the performances of our proposed cluster architecture and scheduling algorithms to the percentage of High class requests, are shown in Figures 16-19. From these figures, it is clear that the support of service differentiation can be enabled by both Pick2X-DP and RR-DP combinations under different percentage of High class requests, different workload intensities and different group of link delays, which is implemented by adjusting the value of the factor β_{HS}. Specifically, the value of β_{HS} in this set of simulations is decreased to 2 especially when the percentage of High class requests is increased to 45%, which affects the condition of the admission control mechanism in our DP algorithm to drop more High class requests for guaranteeing the support of differentiated services. Note that when the percentage of High class requests grows higher than 45%, we should also adjust



(a) (b)

FIGURE 10: For the third set of simulations when using Pick2X-DP and Group 1 of link delays: (a) 90-percentile of request delay vs T_{post}; (b) The percentage of dropped requests vs T_{post}.



(a) (b)

FIGURE 11: For the third set of simulations when using Pick2X-DP and Group 2 of link delays: (a) 90-percentile of request delay vs T_{post}; (b) The percentage of dropped requests vs T_{post}.

the value of β of other service classes (e.g. β_{MS}) carefully to enable service differentiation in the two-level cluster architecture especially under heavier workload intensities as our DP algorithm allots the initial server resources to a certain class based on the percentage of its requests and thus fewer percentage of incoming requests from that class lead to fewer server resources assigned to that class, which make the issue of QoS support more sophisticated. Hence, under different QoS performance targets and different workload intensities, different settings of those constant factors should be deployed carefully for different service classes.

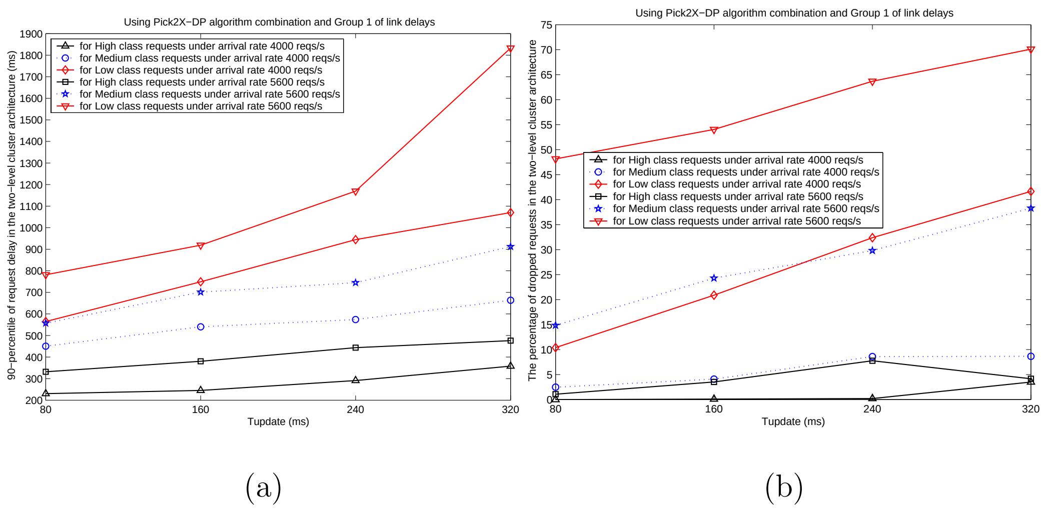


FIGURE 12: For the fourth set of simulations when using Pick2X-DP and Group 1 of link delays: (a) 90-percentile of request delay vs T_{update}; (b) The percentage of dropped requests vs T_{update}.

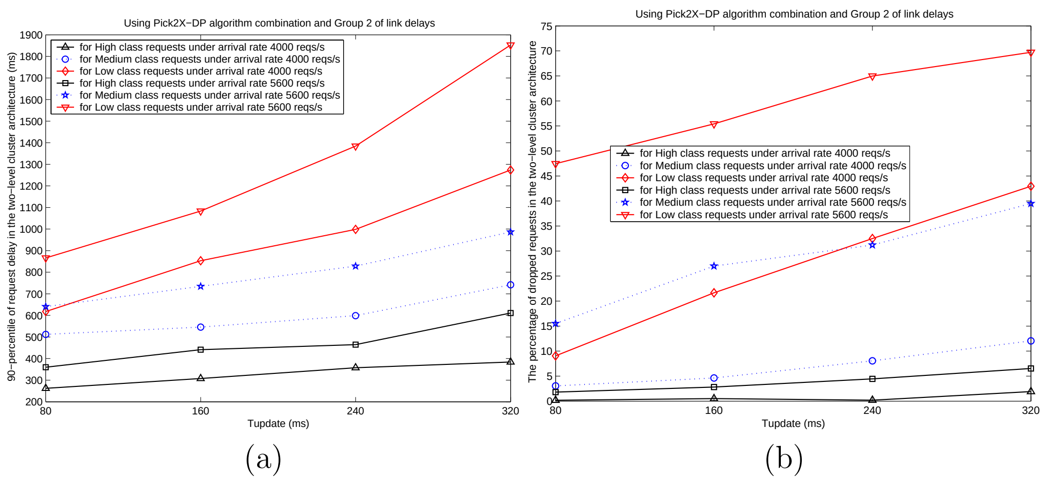
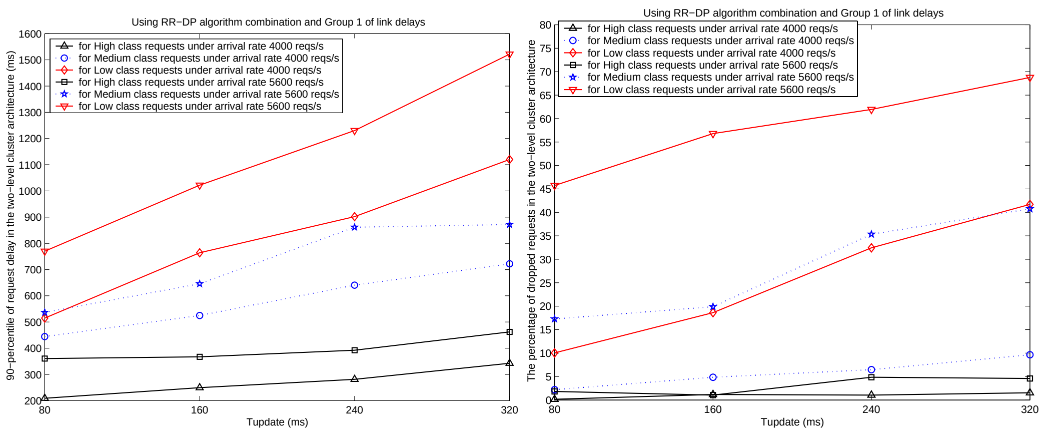


FIGURE 13: For the fourth set of simulations when using Pick2X-DP and Group 2 of link delays: (a) 90-percentile of request delay vs T_{update}; (b) The percentage of dropped requests vs T_{update}.

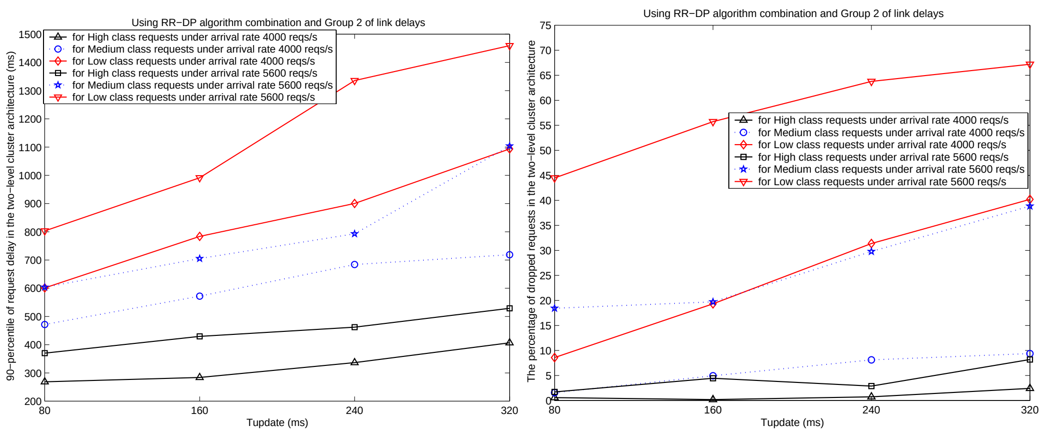
3.4 Summarized Contributions

In this Chapter, the problem of how to enable scalable service differentiation in cluster-based Web server systems is addressed. First of all, the problems of the existing cluster architectures are examined and then a much more scalable one – the two-level cluster architecture is proposed, which includes a layer-4 switch (at the first-level) and a number of parallel layer-7 sub-clusters (at the second-level). Thus, in the two-level cluster architecture, the expensive operations of TCP connection establishment, the identification of requested service classes as well as TCP



(a) (b)

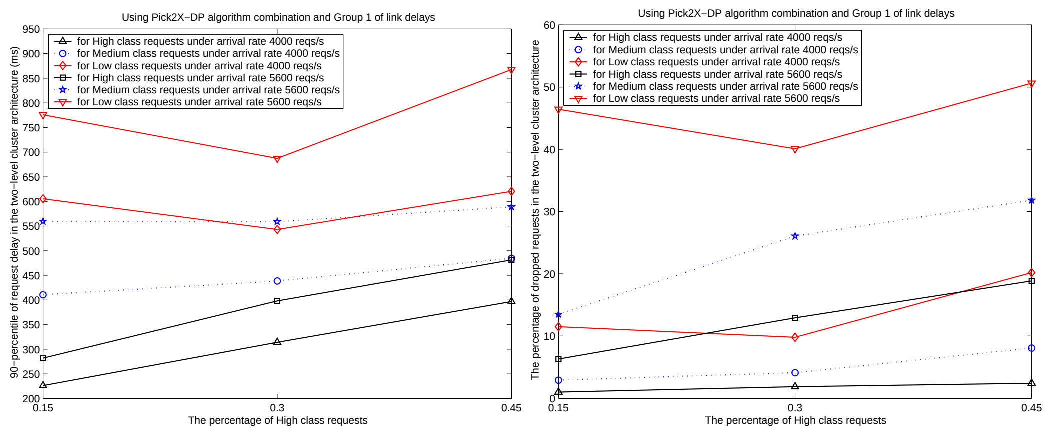
FIGURE 14: For the fourth set of simulations when using RR-DP and Group 1 of link delays: (a) 90-percentile of request delay vs T_{update}; (b) The percentage of dropped requests vs T_{update}.



(a) (b)

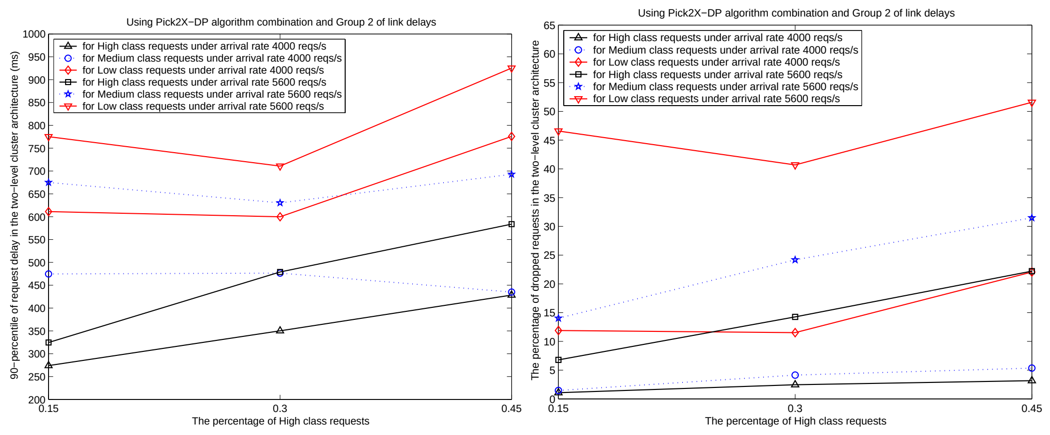
FIGURE 15: For the fourth set of simulations when using RR-DP and Group 2 of link delays: (a) 90-percentile of request delay vs T_{update}; (b) The percentage of dropped requests vs T_{update}.

handoff are distributed among a number of parallel layer-7 front-ends, rather than being centralized in a single front-end node. Only a minimal additional latency penalty (the link delay between the layer-4 switch and the layer-7 front-end of each sub-cluster) is paid for much improved scalability. Moreover, the relevant scheduling algorithms are investigated for enabling differentiated services in the proposed cluster architecture. Specifically, Pick2X [27] and Round-Robin (RR) algorithms are illustrated as the first-level scheduling algorithm and our Dynamic Partitioning algorithm (DP) is proposed to serve as the second-level scheduling algorithm. Through the simulations, it is demonstrated that the proposed two-level



(a) (b)

FIGURE 16: For the fifth set of simulations when using Pick2X-DP and Group 1 of link delays: (a) Sensitivity of request delay to the percentage of High class requests; (b) Sensitivity of dropped rate to the percentage of High class requests.



(a) (b)

FIGURE 17: For the fifth set of simulations when using Pick2X-DP and Group 2 of link delays: (a) Sensitivity of request delay to the percentage of High class requests; (b) Sensitivity of dropped rate to the percentage of High class requests.

cluster architecture can be significantly more scalable compared to those existing ones by having a highly scalable layer-4 switch at the first-level and more parallel sub-clusters co-existing at the second-level. And theoretically the scalability of the two-level cluster architecture is determined only by the scalability of its layer-4 switch. Nowadays, hardware-based, highly scalable layer-4 switches are commercially available. Thus, our proposed scalable cluster architecture is feasible to be implemented. Furthermore, the simulation results show that our selected/designed scheduling algorithms (Pick2X-DP and RR-DP) can succeed in enabling differentiated services in the two-level cluster architecture under different workload inten-

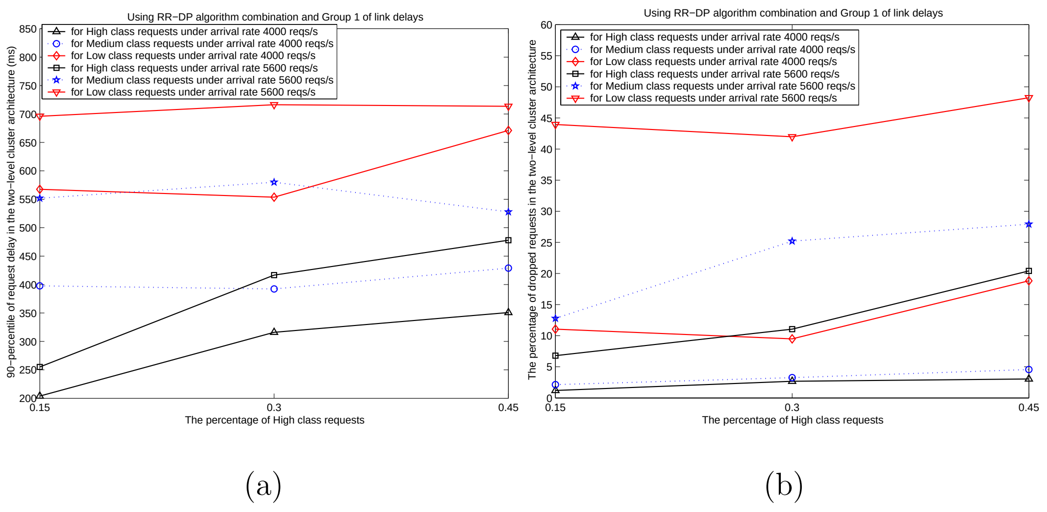


FIGURE 18: For the fifth set of simulations when using RR-DP and Group 1 of link delays: (a) Sensitivity of request delay to the percentage of High class requests; (b) Sensitivity of dropped rate to the percentage of High class requests.

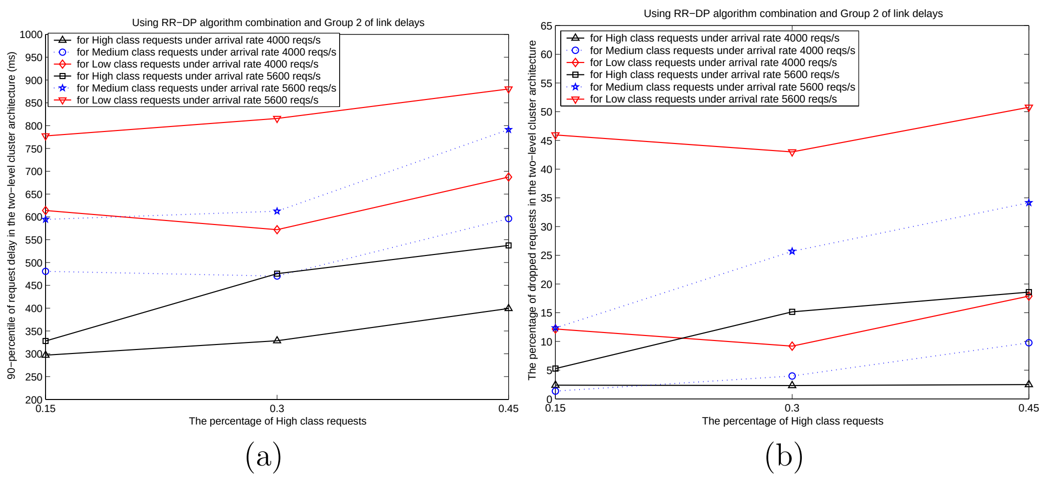


FIGURE 19: For the fifth set of simulations when using RR-DP and Group 2 of link delays: (a) Sensitivity of request delay to the percentage of High class requests; (b) Sensitivity of dropped rate to the percentage of High class requests.

sities and parameter settings.

4 REVENUE-AWARE RESOURCE ALLOCATION SCHEMES IN A MULTICLASS-SUPPORTED NETWORK NODE

Resource allocation in the multiservice communication networks presents a very important problem in the design of the future Internet. The main motivation for the research in this field lies in the necessity for structural changes in the way the Internet is designed. The current Internet offers a single class of 'best-effort' service. However, the Internet is changing and a diverse set of services should be provided in the future Internet to support the requirements of various applications and customers, which result in the definition of different service classes with different requirements of QoS levels. For instance, new sophisticated real-time applications (video conferencing, video on demand, distance learning, etc) demand a better and more reliable network performance. Moreover, these applications require firm performance guarantees from the network where certain resources should be reserved for them. On the other hand, in the future multiclass Internet, each class of customers may have to pay network service providers for their received level of QoS based on the pricing strategy agreed upon in the Service-Level-Agreements between them. A Service-Level-Agreement (SLA) defines the QoS parameters for each class of service, the anticipated per-class workload intensity and the pricing strategy by which the service payment will be determined. Obviously, the pricing strategy will specify the relationship between the QoS level offered to each class of customers and the relevant price which should be paid by them. For example, the service provider will receive a certain amount of revenue from a class of customers if the offered QoS level is more than the minimal requirement of that class and suffer a certain amount of penalty for failing to meet that. Thus, from service providers' point of view, the optimal resource allocation scheme, which can achieve the maximization of SLA revenues under a given amount of network resources (e.g., bandwidth) and a given pricing strategy, is very desirable.

Pricing research in the communication networks has been quite intensive during the last few years (e.g., [47, 17, 67, 68]). Also a great deal of work (e.g., [78, 58, 60]) has been done concerning the issues of resource allocation and fairness in a single-service environment. However, the combination of pricing research and multiclass resource allocation mechanisms have not been analyzed widely. A number of works [71, 11, 74, 53] recently used end-users' utility as the maximizing objective for resource allocation schemes. All of these approaches have a common objective of maximizing the network performance in terms of the users' utility. Our research differs from these studies by linking the resource allocation scheme with the pricing strategy in a SLA and allocating a certain amount of network resources (e.g., bandwidth) among the supported service classes optimally to maximize SLA revenues. A revenue-maximizing pricing scheme for the service provider is presented in [6], where a noncooperative (Nash) flow control game is played by the users (followers) in a Stackelberg game with the goal of setting a price to maximize revenue. Our study focuses on deriving the optimal resource allocation scheme under the given pricing strategy in a SLA for revenue maximization.

Specifically, we focus on the delay performance of each service class and *mean packet delay* and *packet delay* are chosen as the QoS metric in a SLA, respectively, in this Chapter. When the *mean packet delay* is used as the QoS metric in a SLA, the measurement period of packet delays should also be specified in the SLA so that the SLA revenues can be collected periodically based on the deployed pricing strategies and the periodical QoS performance measurements (in this case, the *mean packet delay*). Whereas, with *packet delay* as the QoS metric, the SLA revenues are collected based on the used pricing strategies and the delay of each inbound packet; in other words, a certain amount of revenue or penalty is obtained as long as an inbound packet is served.

J. Joutsensalo *et al* studied the problem of maximizing service providers' revenues obtained in a network node under linear pricing strategy in [38, 39, 40, 41]. In this dissertation, I first extended our previous QoS and revenue-maximization research in [38, 39, 40, 33, 41] and addressed the issue of maximizing SLA revenues in a multiclass-supported network node under a given amount of network resources by novel revenue-aware resource allocation schemes. The closed-form solution to the optimal network resource allocation scheme under linear pricing strategy is derived in publications **PIX** [90] and **PXI** [92] where the *packet delay* is chosen as the QoS metric in a SLA while publication **PX** [91] presented the closed-form solution under linear pricing strategy for the QoS metric of *mean packet delay* in a SLA. Furthermore, the suboptimal network resource allocation scheme was proposed **PXIII** [94] for achieving the highest SLA revenues under linear pricing strategy for the case that the firm QoS guarantees are required for all the supported service classes. This chapter summarizes the contributions in these publications.

4.1 Linear pricing strategy

In this section, the linear pricing strategy, which has been demonstrated to be of practical importance in the real world [84], is deployed in the derivation of optimal resource allocation schemes for maximizing SLA revenues. Consider a multiclass-supported network node with network bandwidth C bits/s, where m service classes are supported and the inbound packets are queued in a multi-queue system (each queue corresponds to one service class). Moreover, the delay of class i packets in the network node is denoted by d_i and the mean delay of class i packets during one measurement period is denoted by $\bar{d}_i$, $i \in [1, m]$. Then, when *mean packet delay* is chosen as the QoS metric, the linear pricing strategy for class i is characterized by the following definition of the linear pricing function.

Definition 1: *The function*

$$r_i(\bar{d}_i) = b_i - k_i\bar{d}_i, \quad i = 1, 2, ..., m, \ b_i > 0, \ k_i > 0 \tag{4.1}$$

is called the *linear pricing function* of class i under the QoS metric "*mean packet delay*", where b_i and k_i are positive constants and $b_i \geq b_j$ and $k_i \geq k_j$ should hold to ensure differentiated pricing if class i has higher priority than class j (in this dissertation, we assume that class 1 is the highest priority and class m is the lowest one).

With *packet delay* as the QoS metric, the linear pricing strategy of class i is characterized by the linear pricing function defined below.

Definition 2: *The function*

$$r_i(d_i) = b_i - k_i d_i, \quad i = 1, 2, ..., m, \ b_i > 0, \ k_i > 0 \tag{4.2}$$

is called the *linear pricing function* of class i under the QoS metric "*packet delay*", where b_i and k_i are positive constants and $b_i \geq b_j$ and $k_i \geq k_j$ should also hold to ensure differentiated pricing if class i has higher priority than class j.

4.2 Optimal resource allocation schemes for maximizing SLA revenues under linear pricing strategy

Consider the above multiclass-supported network node fed by m Poisson packet streams with arrival rates λ_1, λ_2, ..., λ_m, respectively. We assume that in this section the packet length distribution is exponential and use $\bar{L}_i$ to denote the mean packet length of class i in bits. Let the weight allotted to class i be w_i, i=1,2,...,m, which means that the reserved bandwidth for class i packets is w_iC (bits/s). Without loss of generality, only non-empty queues are considered, and thus $w_i \neq 0$. Therefore, the natural constraint for the weights is $\sum_{i=1}^{m} w_i = 1, w_i \in (0, 1]$. As class i is guaranteed to use a portion of the network resource w_iC and the packets of class i arrive at queue i with rate λ_i, the analytic mean delay of class i packets in the network node (referred to as $\hat{\bar{d}}_i$) can be denoted as:

$$\hat{\bar{d}}_i = \frac{1}{\frac{w_iC}{\bar{L}_i} - \lambda_i} = \frac{\bar{L}_i}{w_iC - \lambda_i\bar{L}_i} \tag{4.3}$$

based on queueing theory. The natural constraint of Eq. (4.3) is $w_iC > \lambda_i\bar{L}_i$ due to the fact that delay can not be negative.

With the *mean packet delay* chosen as the QoS metric in a SLA, we use $\hat{\bar{d}}_i$ to estimate the real mean delay of class i packets $\bar{d}_i$ in the network node during one measurement period. Thus, the SLA revenues F obtained in the network node during one measurement period is defined by the following equation under linear pricing strategy:

$$F = \sum_{i=1}^{m} r_i(\bar{d}_i) = \sum_{i=1}^{m} r_i(\hat{\bar{d}}_i) = \sum_{i=1}^{m} (b_i - \frac{k_i\bar{L}_i}{w_iC - \lambda_i\bar{L}_i}). \tag{4.4}$$

Based on the above definition, publication **PXI** [91] derived the closed-form solution to the optimal resource allocation scheme (the optimal weight) for this case under the linear pricing strategy:

$$w_i = \frac{\sqrt{k_i\bar{L}_i}(C + \frac{\sum_{j=1}^{m}\sqrt{k_j\bar{L}_j}}{\sqrt{k_i\bar{L}_i}}\lambda_i\bar{L}_i - \sum_{j=1}^{m}\lambda_j\bar{L}_j)}{C\sum_{j=1}^{m}\sqrt{k_j\bar{L}_j}}, \quad i = 1, 2, ..., m. \tag{4.5}$$

When *packet delay* is the QoS metric in a SLA, the real delay of class i packets d_i in the network node is also estimated by $\hat{\bar{d}}_i$. As in this case a certain amount of revenue or penalty is attained as long as one inbound packet is served, we try to maximize the SLA revenues obtained in the network node per time unit (also denoted by F). Thus, in this case, F is defined as follows under linear pricing strategy:

$$F = \sum_{i=1}^{m} \lambda_i r_i(d_i) = \sum_{i=1}^{m} \lambda_i r_i(\hat{\bar{d}}_i) = \sum_{i=1}^{m} \lambda_i(b_i - \frac{k_i\bar{L}_i}{w_iC - \lambda_i\bar{L}_i}). \tag{4.6}$$

And the closed-form solution to the optimal resource allocation scheme for this case is presented in publications **PX** [90] and **PXII** [92]:

$$w_i = \frac{\sqrt{\lambda_i k_i\bar{L}_i}(C + \frac{\sum_{j=1}^{m}\sqrt{\lambda_j k_j\bar{L}_j}}{\sqrt{\lambda_i k_i\bar{L}_i}}\lambda_i\bar{L}_i - \sum_{j=1}^{m}\lambda_j\bar{L}_j)}{C\sum_{j=1}^{m}\sqrt{\lambda_j k_j\bar{L}_j}}, \quad i = 1, 2, ..., m. \tag{4.7}$$

Furthermore, publication **PXIII** [94] proposed the suboptimal resource allocation scheme for the case that all the supported service classes have their firm QoS (mean delay) requirements, which can satisfy those required QoS guarantees while still being able to achieve very high revenue close to the analytic maximum one under linear pricing strategy.

4.3 Summarized contributions

By extending our previous QoS and pricing research work in extended our previous QoS and revenue-maximization research in [38, 39, 40, 33, 41], we addressed the problem of maximizing SLA revenues under a given amount of network resources (e.g., bandwidth) and the linear pricing strategy in a SLA using novel revenue-aware resource allocation schemes. Specifically, publication **PX** [91] derived the closed-form solution to the optimal resource allocation scheme for revenue maximization in a multiclass-supported network node under linear pricing strategy when *mean packet delay* is chosen as the QoS metric in the SLA. Moreover, with packet delay as the SLA QoS metric, the similar closed-form solution to the optimal resource allocation scheme under linear pricing strategy is presented by our publications **PIX** [90] and **PXI** [92]. The simulation results in these publications demonstrated the effectiveness of our derived optimal resource allocation schemes for maximizing SLA revenues under a given amount of network resources and the linear pricing strategy.

Furthermore, publication **PXIII** [94] proposed the suboptimal resource allocation scheme for the case that all the supported service classes have their firm QoS (mean delay) requirements and it can not only satisfy those required QoS level guarantees but also achieve very high SLA revenue close to the analytic maximum one under linear pricing strategy, as is demonstrated by the simulations in [94].

5 STOCHASTIC ANALYSIS OF UPPER DELAY BOUND OF GPS-BASED PACKETIZED FAIR QUEUEING ALGORITHMS

The provision of Quality-of-Service (QoS) guarantees such as bandwidth, delay, jitter and cell loss to applications with widely different characteristics is a primary objective in the design of next-generation networks. One important issue in the provision of QoS guarantees is the study of the queueing algorithms employed at network nodes. Among the queuing algorithms that have been proposed, the class of algorithms which aim at approximating the Generalized Processor Sharing (GPS) policy are most popular. GPS [64, 65] is an idealized fluid discipline with a number of very desirable properties: (i) it provides minimum QoS guarantees to each traffic session [2], regardless of the behavior of other sessions; (ii) it provides the deterministic worse-case delay bound to each session whose traffics are leaky-bucket constrained; and (iii) it ensures fairness in the amount of service provided by a network node to competing sessions. Since this policy is a fluid model, it is not adapted for packet-by-packet transmission. Therefore, its packet-based extensions that we call GPS-based Fair Queueing algorithms have been proposed, well-known examples of which include Weighted Fair Queuing (WFQ) [21, 64], Self-Clocked Fair Queuing (SCFQ) [29], Frame-based Fair Queuing (FFQ) [82] and Starting Potential-based Fair Queuing (SPFQ) [82].

A significant volume of work in the literature [21, 64, 29, 30, 31, 81, 82, 14] has been concerned with evaluating the deterministic worst-case delay guarantees that GPS-based Fair Queueing (FQ) algorithms can provide when the burstiness of the traffic feeding them is bounded (mostly shaped by a leaky bucket). However, little work has been reported on analyzing the stochastic delay bounds of such packetized policies under a general probabilistic traffic model. This has been mainly due to the difficulty of stochastically modelling the complex behavior of a GPS-based FQ

[2]in this Chapter and publication **PXIV** [95], the notion of session actually means the aggregate of packet streams which require the same QoS level and thus it is exchangeable with the notion of service class.

algorithm. Indeed an important advantage of stochastic modelling of FQ systems as compared to worst-case deterministic analysis is that statistical analysis takes into account the actual dynamics of the packet arrival process, thus being more accurate in predicting the QoS guarantees provided to each traffic session and also being able to derive more efficient revenue-aware resource allocation scheme for maximizing SLA revenues in a network node deploying GPS-based FQ algorithms.

Zhang *et al* [96] studied the statistical behavior of GPS discipline using exponentially bounded burstiness processes [85] as the session source traffic model and derived upper bounds on the tail distributions of session backlog and delay. However, no simulation results were provided to verify the quality of those bounds. Pekergin [69] derived stochastic bounds on the delay distribution of GPS-related FQ algorithms fed by a Switched Bernoulli Batch process. The analysis in [69] is quite complex and does not result in explicit analytical equations, thus limiting its usefulness for back-of-the-envelope calculations and comparisons. The analysis also makes some limiting assumptions such as the use of fixed packet lengths and the need to set all sessions other than the tagged one to be greedy all the time. M. Hawa *et al* used the bounded fairness criterion of FQ algorithms to derive the upper and lower bounds on mean packet delay of those algorithms under Poisson arrivals [34]. However, the analysis in [34] assumes that all sessions have the exact same packet length distribution, which limits its application scope.

In publication **PXIV** [95], we extend the notion of *feasible partition* introduced by Zhang *et al* [96] for the analysis of idealized GPS discipline and apply it to the stochastic bound analysis of GPS-based FQ algorithms. With the help of this notion, we derive our new upper bound on mean packet delay under the general probabilistic traffic model of Poisson arrivals and any general packet length distribution. The resulted upper bound is much simpler and tighter than the ones by M. Hawa *et al* [34]. Moreover, it fits a class of GPS-based FQ algorithms including WFQ, SCFQ and SPFQ, which aims at approximating GPS discipline and thus validate the notion of *feasible partition.* Finally, in this Chapter our new upper bound on mean packet delay is utilized to derive the suboptimal resource allocation scheme under a given amount of network resources and flat pricing strategy for maximizing SLA revenues in a network node deploying a GPS-based packetized FQ algorithm.

5.1 Upper Bound on Mean Packet Delay of GPS-based Packetized FQ Algorithms

Consider a single-server GPS-based FQ system with capacity C bits/s, which multiplexes N sessions with Poisson arrival rates $\lambda_1, \lambda_2, ..., \lambda_N$ (packets/s). In the FQ system, each session has its own queue. Assume that the queues corresponding to different sessions are infinite in length and the packets in the same queue are served in the order they arrive. We use L_i to denote session i packet length (in bits). As mentioned above, the distribution of L_i can be any general distribution and $\bar{L_i}$ is used to denote session i mean packet length, i.e., $E[L_i] = \bar{L_i}$. As a necessary

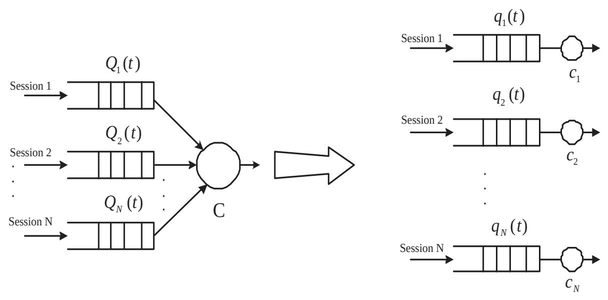


FIGURE 20: Decomposition of a GPS-based FQ system.

stability condition, $\sum_{i=1}^{N} \lambda_i \bar{L}_i < C$ is required. Furthermore, the service weight w_i determines the minimum guaranteed share of capacity assigned to session i when session i is backlogged. Obviously, $\sum_{i=1}^{N} w_i = 1$ and $0 < w_i \leq 1$, $i \in [1, N]$.

Let $\mathcal{N}= \{1, 2, ..., N\}$. In publication **PXIV** [95], first a sequence of disjoint sets, $\mathcal{H}= \{H_k\}_{1 \leq k \leq m}$, where $m \geq 1$ and $H_1 \cup \cdots \cup H_m = \{1, 2, ..., N\}$, is defined as the *feasible partition* of $\mathcal{N}$ with respect to the given sets of arrival rates $\{\lambda_i \bar{L}_i\}_{i \in \mathcal{N}}$ (bits/s) and service weights $\{w_i\}_{i \in \mathcal{N}}$. Then, using the *decomposition approach* introduced in [96], the above GPS-based FQ system can be analyzed through a set of N separate single-queue systems, each of which has a dedicated server with capacity c_i (referred to as the *virtual decomposed system*) (see Figure 20).

Let $X_i = L_i/(w_i C)$ denote the service time of session i packets in the corresponding virtual decomposed system where a dedicated server with capacity $c_i = w_i C$ serves session i packets and $E[X_i] = \bar{X}_i$ and $E[X_i^2] = \bar{X_i^2}$. Moreover, the mean delay of session i packets in the FQ system is denoted by $\bar{d}_i$, which equals the mean waiting time in queue plus the mean service time. Publication **PXIV** [95] derives the following upper bound on mean packet delay $\bar{d}_i$ as long as $\lambda_i \bar{L}_i < w_i C$ (i.e., session i in the GPS-based FQ system is a session in H_1):

$$\bar{d}_i \leq \bar{X}_i + \frac{\lambda_i \bar{X_i^2}}{2(1 - \lambda_i \bar{X}_i)}. \tag{5.1}$$

It can be noticed that the above upper bound on mean packet delay is derived under the general probabilistic traffic model of Poisson arrivals and any general packet length distribution. Moreover, the simulation results in Publication **PXIV** [95] demonstrates that the derived upper bound is much tighter than the ones by M. Hawa *et al* [34] and it fits a class of GPS-based packetized FQ algorithms including WFQ [21, 64], SCFQ [29] and SPFQ [82].

5.2 Revenue-aware resource allocation scheme in a GPS-based network node under flat pricing strategy

Similarly, when *mean packet delay* is chosen as the QoS metric in a SLA, a network server provider may receive SLA revenues or penalties in a GPS-based network node based on the offered QoS (mean packet delay here) guarantees and the deployed pricing strategy. In this part, the above upper bound on mean packet delay in Eq. (5.1) is utilized to derive the revenue-aware resource allocation scheme in a GPS-based network node under flat pricing strategy. Specifically, first the flat pricing function which characterizes the flat pricing strategy is generally defined. Then the suboptimal resource allocation scheme is presented for maximizing SLA revenues in a GPS-based network node under flat pricing strategy, whose performances are investigated in the following simulation part.

5.2.1 Flat pricing strategy

Consider the above GPS-based FQ system. As *mean packet delay* is deployed as the QoS metric in the SLA, the flat pricing strategy for class [3] i is characterized by the following definition of a flat pricing function.

Definition 3: *The function*

$$r_i(\bar{d_i}) = \begin{cases} R_i & \texttt{if } \bar{d_i} \leq D_i \\ -P_i & \texttt{if } \bar{d_i} > D_i \end{cases}, \quad i = 1, 2, ..., N \tag{5.2}$$

is called the *flat pricing function* of class i, where D_i is the QoS (mean packet delay) guarantee requested by class i packets and R_i and P_i are both positive constants. Obviously, the above flat pricing function specifies that if the real mean packet delay of class i is less than D_i, the network service provider receives a revenue R_i, otherwise a penalty P_i is incurred for failing to meet that QoS guarantee D_i. Moreover, $R_i \geq R_j$ and $P_i \geq P_j$ should hold to ensure differentiated pricing if class i has higher priority than class j, which are actually what we expect based on the SLA requirement. Figure 21 presents a example of the flat pricing functions of Gold, Silver and Bronze classes.

5.2.2 Suboptimal resource allocation scheme in a GPS-based network node under flat pricing strategy

Consider a GPS-based network node with capacity C bits/s and a total of N service classes supported. As the QoS metric in the SLA is *mean packet delay*, obviously the SLA revenue F obtained in the GPS-based network node under one charging period is defined as follows.

$$F = \sum_{i=1}^{N} r_i(\bar{d_i}). \tag{5.3}$$

Based on the definition of flat pricing function in Eq.(5.2), it is clear that if $\bar{d_i} \leq D_i$ holds for each service class $i \in [1, N]$, F achieves its maximum value $\sum_{i=1}^{N} R_i$. Since the mean packet delay of class i is bounded tightly by our derived

[3]The notion of class is used instead of session hereafter in this Chapter as they are exchangeable.

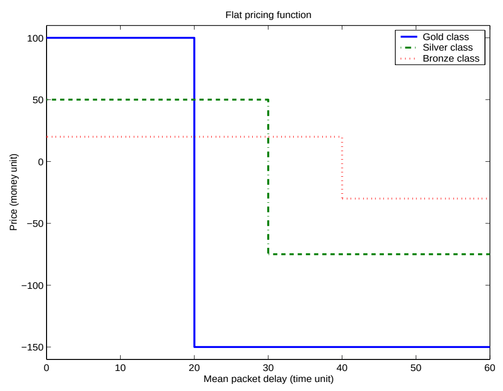


FIGURE 21: The flat pricing functions of Gold, Silver and Bronze classes.

upper delay bound as long as $\lambda_i \bar{L}_i < w_i C$, the problem of deriving the suboptimal resource allocation scheme for maximizing F under flat pricing strategy can be formulated as follows by linking parameter D_i in the flat pricing function of class i with the upper delay bound in Eq. (5.1).

$$\texttt{Solve} \quad \bar{d}_i \le \bar{X}_i + \frac{\lambda_i \bar{X_i^2}}{2(1-\lambda_i \bar{X}_i)} \le D_i \tag{5.4}$$

$$\texttt{s.t.} \quad \sum_{i=1}^{N} w_i \le 1, \;\; 0 < w_i \le 1, \tag{5.5}$$

$$w_i C > \lambda_i \bar{L}_i. \tag{5.6}$$

The formula of our upper delay bound in Eq. (5.1) is presented based on a general packet length distribution. Below, we illustrate how to derive the suboptimal resource allocation scheme by solving the right inequality in (5.4) under some practical packet length distributions in IP networks. First, under exponential packet length distribution, $E[L_i] = \bar{L}_i$ and $E[X_i^2] = 2(\bar{L}_i)^2$, resulting in $E[X_i] = \bar{X}_i = \bar{L}_i/(w_i C)$ and $E[X_i^2] = \bar{X_i^2} = 2[\bar{L}_i/(w_i C)]^2$. Substitute the above $\bar{X}_i$ and $\bar{X_i^2}$ into (5.4), then the right inequality in (5.4) becomes:

$$\frac{\bar{L}_i}{w_i C - \lambda_i \bar{L}_i} \le D_i$$

leading to the following solution under the exponential packet length distribution:

$$w_i \ge \frac{\bar{L}_i}{C}(\lambda_i + \frac{1}{D_i}), \;\; i = 1, 2, ..., N. \tag{5.7}$$

That is to say that we should allocate at least the capacity of $C_{i,min} = w_{i,min} C = \bar{L}_i(\lambda_i + 1/D_i)$ to class i so that its mean packet delay can be guaranteed to be less than D_i during the charging period. As a result of the above solution to w_i in (5.7), we present the suboptimal resource allocation scheme for maximizing the SLA revenue obtained in a GPS-based network node under a given amount of network resources and flat pricing strategy as follows:

1. Set $C_{i,min} = w_{i,min}C = \bar{L}_i(\lambda_i + 1/D_i)$, i=1,2,...,N,

2. If $\sum_{i=1}^{N} C_{i,min} \le C$, to reserve capacity $C_{i,min}$ for any class $i \in [1, N]$ is the suboptimal resource allocation scheme in this case. By the suboptimal scheme, the network service provider will receive the maximum SLA revenue $\sum_{i=1}^{N} R_i$ during one charging period under flat pricing strategy,

3. Otherwise, it means that the total capacity C of the GPS-based network node is not enough to satisfy the reservation of capacity $C_{i,min}$ for all the supported service classes. Hence, we should first guarantee the reservation of capacity $C_{i,min}$ for a set of selected service classes so that the obtained SLA revenue is the highest in this situation. Then the remaining resources are allocated to all the supported service classes other than the above selected ones uniformly.

Note that the above set of selected service classes in Step 3 is acquired by the comparison of the analytic SLA revenues under all possible resource allocation schemes, which makes its calculation complexity increases quickly with larger values of N. Hence, instead we may first reserve capacity $C_{1,min}$ for the highest priority class, then reserve capacity $C_{2,min}$ for the second highest priority class until the remaining capacity is not enough to satisfy the reservation of capacity $C_{i,min}$ for class i (we have assumed that class 1 is the highest priority class and class N is the lowest one), which also tries to achieve the highest SLA revenue in this case as the incurred penalty due to failing to meet the QoS guarantee of higher class is larger.

For the network traffics which exhibit self-similarity nature, we use *Bounded Pareto* to model the heavy-tailed distribution as used in [19]. In this Chapter, $BP(p_i, q_i, \alpha_i)$ is used to denote Bounded Pareto packet length distribution of class i, where p_i is the smallest length of class i packets, q_i the largest ($p_i \le L_i \le q_i$), and α_i the shape parameter. Then, $E[L_i] = \bar{L}_i = \alpha_i p_i^{\alpha_i}(p_i^{1-\alpha_i} - q_i^{1-\alpha_i})/[(\alpha_i - 1)(1 - (p_i/q_i)^{\alpha_i})]$ and $E[L_i^2] = \bar{L_i^2} = \alpha_i p_i^{\alpha_i}(p_i^{2-\alpha_i} - q_i^{2-\alpha_i})/[(\alpha_i - 2)(1 - (p_i/q_i)^{\alpha_i})]$, resulting in $E[X_i] = \bar{X}_i = \bar{L}_i/(w_i C)$, $E[X_i^2] = \bar{X_i^2} = \bar{L_i^2}/(w_i C)^2$. Thus, the right inequality in (5.4) becomes as follows:

$$\frac{2\bar{L}_i(w_i C - \lambda_i \bar{L}_i) + \lambda_i \bar{L_i^2}}{2w_i C(w_i C - \lambda_i \bar{L}_i)} \le D_i$$

leading to the solution under the Bounded Pareto packet length distribution:

$$w_i \ge \frac{(1 + \lambda_i D_i)\bar{L}_i + \sqrt{[(1 + \lambda_i D_i)\bar{L}_i]^2 - 2\lambda_i D_i(2\bar{L}_i^{\,2} - \bar{L_i^2})}}{2D_i C}, \; i = 1, 2, ..., N, \quad (5.8)$$

i.e., $C_{i,min} = w_{i,min}C = \left[(1 + \lambda_i D_i)\bar{L}_i + \sqrt{[(1 + \lambda_i D_i)\bar{L}_i]^2 - 2\lambda_i D_i(2\bar{L}_i^{\,2} - \bar{L_i^2})}\right] /2D_i$ in this case. Then the suboptimal resource allocation scheme under the Bounded Pareto packet length distribution can also be derived based on the above approach.

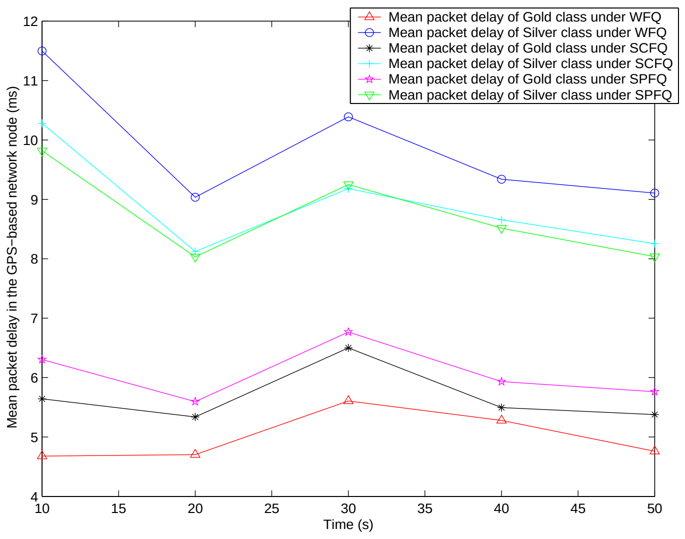


FIGURE 22: Mean packet delay in the first simulation where exponential packet length distribution is deployed.

5.2.3 Simulation results

In this section, we present some simulation results to illustrate the effectiveness of the above derived suboptimal resource allocation scheme in GPS-based network node under flat pricing strategy. Throughout the following simulations, three representative GPS-based FQ algorithms were used in a GPS-based network node: WFQ [21, 64], SCFQ [29] and SPFQ [82].

In the first simulation, we consider a GPS-based network node which has initial capacity C=1Mb/s and supports two service classes (namely Gold class and Silver class) with Poisson arrivals. The arrival rates of Gold and Silver classes are $\lambda_1 = 350$ packets/s and $\lambda_2 = 200$ packets/s, respectively (class 1 indicates Gold class and class 2 means Silver class). The packet length distributions of the two classes are both exponential with $\bar{L_1} = 1000$ bits and $\bar{L_2} = 2000$ bits. Moveover, the parameters deployed in the two flat pricing functions are summarized as follows: $D_1 = 10$ms, $R_1 = 10$ money units, $P_1 = 15$ money units and $D_2 = 20$ms, $R_2 = 5$ money units, $P_2 = 8$ money units.

By the calculation of the inequality in (5.7), it is shown that at least capacity 0.45Mb/s should be reserved for Gold class (i.e., $C_{1,min}$=0.45Mb/s) and at least capacity 0.5Mb/s for Silver class (i.e., $C_{2,min}$=0.5Mb/s) to guarantee the satisfaction of both $\bar{d_1} \leq D_1 = 10$ms and $\bar{d_2} \leq D_2 = 20$ms. As $C_{1,min} + C_{2,min} < C$, the suboptimal resource allocation scheme is that the reserved capacity for Gold class is 0.45Mb/s, the reserved capacity for Silver class is 0.5Mb/s and only 0.95Mb/s of capacity is needed. Hence, the real capacity of the GPS-based node is set to 0.95Mb/s in the first simulation and the simulation results are presented in Figure 22, where it can be seen that $\bar{d_i} \leq D_i$ holds for each class $i \in [1, 2]$ during each charging period (20s here) when any one of WFQ, SCFQ and SPFQ is deployed. Hence, it is demonstrated that the derived suboptimal resource allocation

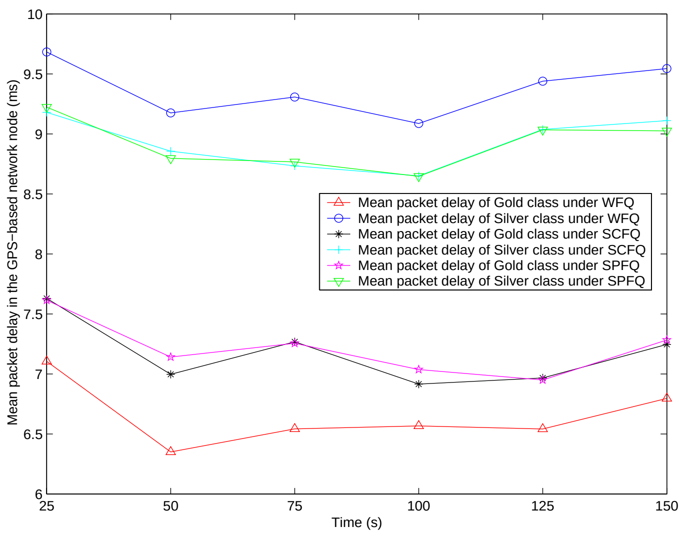


FIGURE 23: Mean packet delay in the second simulation where Bounded Pareto packet length distribution is deployed.

scheme achieves the maximum value $\sum_{i=1}^{2} R_i$=15 money unit of SLA revenue in the GPS-based network node under flat pricing strategy.

In the second simulation, the above GPS-based network node is fed by two classes of Poisson packet streams with arrival rates $\lambda_1 = 100$ packets/s, $\lambda_2 = 120$ packets/s and a heavy-tailed packet length distributions. The packet length distributions of the two classes are both modelled by Bounded Pareto with parameters $b_1 = 40$ bits, $p_1 = 12800$ bits, $\alpha_1 = 0.137$ (thus $\bar{L_1} = 2000$ bits), and $b_2 = 320$ bits, $p_1 = 16000$ bits, $\alpha_2 = 0.164$ (thus $\bar{L_2} = 3360$ bits). The parameters of the two flat pricing functions in this case are: $D_1 = 15$ms, $R_1 = 20$ money units, $P_1 = 30$ money units and $D_2 = 20$ms, $R_2 = 10$ money units, $P_2 = 15$ money units.

Similarly, by the calculation of the inequality in (5.8), it is gotten that $C_{1,min} = 0.3694Mb/s$, $C_{2,min} = 0.5857Mb/s$ and $C_{1,min} + C_{2,min} = 0.9551Mb/s < C = 1Mb/s$. Hence, the suboptimal resource allocation scheme in this case is that the reserved capacity for Gold class is 0.3694Mb/s, the reserved capacity for Silver class is 0.5857Mb/s and only 0.9551Mb/s of capacity is needed. Hence, the real capacity of the GPS-based node is set to 0.9551Mb/s in the second simulation and Figure 23 shows the simulation results.

The results in Figure 23 also indicate that $\bar{d_i} \leq D_i$ is satisfied for each class $i \in [1, 2]$ during each charging period (50s in this case), leading to the achievement of the maximum SLA revenue $\sum_{i=1}^{2} R_i$=30 money unit obtained within one charging period in this case.

Finally, the third simulation was made to evaluate the performance of the suboptimal resource allocation scheme derived by the above approach for the case that the capacity C of a GPS-based network node is not enough to guarantee the reservation of $C_{i,min}$ for all its supported service classes. Hence, the same parameter settings and exponential packet length distributions as the ones in the first sim-

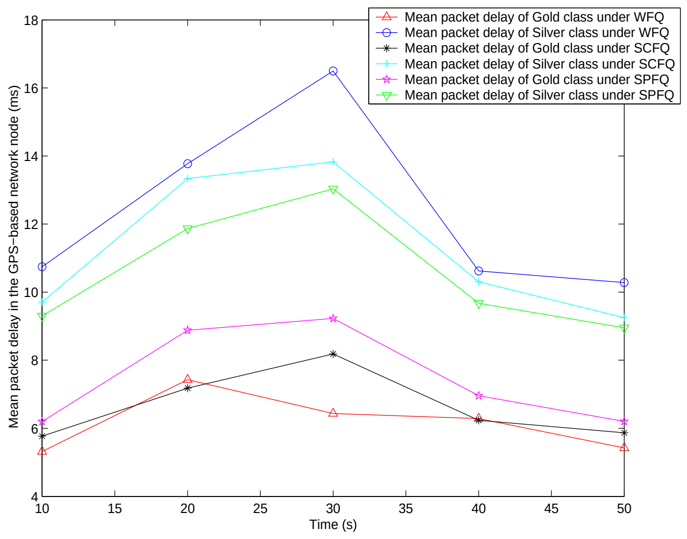


FIGURE 24: Mean packet delay in the third simulation where exponential packet length distribution is deployed.

ulation are deployed in the third simulation except that the arrival rates of Gold and Silver classes are $\lambda_1 = 450$ packets/s and $\lambda_2 = 200$ packets/s, respectively. By the inequality in (5.7), we know that $C_{1,min}$=0.55Mb/s, $C_{2,min}$=0.50Mb/s and $C_{1,min} + C_{2,min} = 1.05$Mb/s> C =1Mb/s. Thus, according to our proposed approach, the suboptimal resource allocation scheme under flat pricing strategy in this case is as follows: the reserved capacity for Gold class is 0.55Mb/s, the reserved one for Silver class is 0.45Mb/s and all of the capacity C=1Mb/s of the GPS-based node is used. The simulation results are presented in Figure 24. It is shown in Figure 24 that both $\bar{d}_1 \leq D_1$ and $\bar{d}_2 \leq D_2$ are satisfied when any one of WFQ, SCFQ and SPFQ is deployed, which also results in the achievement of the maximum SLA revenue $\sum_{i=1}^{2} R_i$=15 money units, although the reserved capacity for Silver class is less than $C_{2,min}$ in this case. The reason is that in a network node which deploys any GPS-based FQ algorithm, the reserved resources (the capacity in this case) for a certain class indicate only the minimum guaranteed resources allotted to that class and the actual real resources received by that class may exceed the reserved one. However, as we do not know the exact amount of resources received by a certain class in the real situation due to the characteristic of GPS-based FQ algorithms, it is the best to derive the suboptimal resource allocation scheme by the tightest upper delay bound of GPS-based FQ algorithms.

Therefore, based on the above simulation results, we can conclude that the above approach of deriving the suboptimal resource allocation scheme in a GPS-based network node is effective and the derived resource allocation scheme can achieve the highest SLA revenue under a given amount of network resources and flat pricing strategy. More importantly, it also fits a class of GPS-based packetized FQ algorithms.

5.3 Summarized contributions

Fair Queueing (FQ) algorithms which aim at approximating the Generalized Processor Sharing (GPS) policy remain most popular for the provision of Quality-of-Service guarantees in IP networks. In publication **PXIV**, we extends the notion of *feasible partition* introduced by Zhang *et al* for the analysis of idealized GPS discipline and apply it to the stochastic delay bound analysis of GPS-based packetized FQ algorithms. A novel upper bound on mean packet delay is derived there under the general probabilistic traffic model of Poisson arrivals and any general packet length distribution and it is much simpler and tighter than the known ones by M. Hawa *et al.* Moreover, the derived upper bound on mean packet delay fits a class of GPS-based FQ algorithms including WFQ, SCFQ and SPFQ. Furthermore, this Chapter utilizes our upper bound on mean packet delay to analyze the problem of deriving the suboptimal resource allocation scheme under a given amount of network resources and flat pricing strategy for maximizing SLA revenues in a network node deploying a GPS-based packetized FQ algorithm. The simulation results presented in this Chapter demonstrate the effectiveness of our derived suboptimal resource allocation scheme, which also fits a class of GPS-based packetized FQ algorithms.

6 MAXIMIZING SLA REVENUES IN CLUSTER-BASED WEB SERVER SYSTEMS

The Web is changing from a sole communication and browsing infrastructure to an important medium for conducting personal business and e-commerce, which makes the Quality of Service (QoS) an increasingly critical issue. A fundamental characteristic of e-commerce environments is the diverse set of services provided to support the requirements of various businesses and customers, which result in the definition of different service classes. In a typical e-commerce environment, an e-business operator contracts with a Web service provider to provide applications and services to its business customers, which can be consumers (B2C) or other businesses (B2B); in other words, a Web service provider hosts an e-commerce Web site via a contract with the e-business operator. In many e-commerce contracts, the Web service provider agrees to offer a certain level of QoS to each class of service in the hosting of the e-commerce site, and in return the e-business operator agrees to pay the service provider based on the QoS levels received by its customers. These contracts are based on a SLA (Service-Level-Agreement) between the e-business operator and the Web service provider that defines the QoS metrics for each class of service, the anticipated workload intensity of per-class requests from the e-business and the pricing strategy by which the SLA payment will be determined.

The exponential growth in Internet usage, much of which is fueled by the growth and requirements of various aspects of e-commerce, has created the demand for more and faster Web servers capable of serving over 100 million Internet users. As mentioned in Chapter 2, server clustering has emerged as a promising technique to build faster, scalable and cost-effective Web servers [75] during recent years, which has made cluster-based Web server systems a major means to hosting e-commerce sites. A state-of-the-art cluster-based Web server system consist of a number of back-end server nodes and a specialized front-end node, which acts as the single input point of customer requests and is responsible for distributing the inbound requests among the back-end nodes. That is to say that the back-end server nodes of a cluster-based Web server system which hosts an e-commerce Web site are shared by the inbound requests of different service classes.

As discussed in Chapter 2, a number of papers [43, 9, 98, 87, 88] have proposed the mechanisms of partitioning the back-end server nodes among the supported service classes to enable differentiated services in the above cluster-based Web server system. In this Chapter, we further analyze the problem of maximizing the revenue attained in the hosting of an e-commerce site with a SLA contract under a given amount of server resources by optimally partitioning the server resources among all the supported classes. Little work has been reported on this topic. The issue of maximizing SLA revenues in the cluster platform of Web server farms was recently studied by Liu *et al* in [56]. A Web server farm is typically deployed to host several Web sites simultaneously on the same platform. Moreover, they assumed that each back-end server node in a Web server farm can serve multiple service classes; they then tried to optimally allocate the resource (e.g., processing capacity) of each server node among its supported service classes to maximize the resulted SLA revenues. However, the closed-form solution to the optimal resource allocation scheme (i.e., the optimal service weights) in each back-end node was not derived in [56]. Diao *et al* [22] proposed a profit-oriented feedback control system for maximizing SLA profits in Web server systems, which automated the admission control decisions in a way that balances the loss of revenue due to rejected work against the penalties incurred if admitted work has excessive response times by a fuzzy control algorithm. Additionally, the issue of maximizing the expected value of a given cluster utility function by allocating server resources of a cluster-based Web server system dynamically was studied in [54], where the closed-form solution was also not derived.

In this dissertation, we focus on the cluster platform which hosts a single e-commerce site in a cluster-based Web server system, as was studied in Chapter 2. Publication **PXII** [93] has addressed the problem of maximizing the SLA revenue obtained for the hosting of an e-commerce site by such a Web cluster system under a given amount of server resources and linear pricing strategy, where the closed-form solution to the optimal resource partitioning scheme is derived from the revenue target function by a Lagrangian optimization approach. This Chapter first presents the target Web cluster architecture upon which an e-commerce site is built and then summarizes the contributions in publication **PXII**. Finally, the suboptimal resource partitioning scheme is proposed, whose simulation results demonstrate its performance of maximizing the SLA revenues obtained in the hosting an e-commerce site under a given amount of back-end server nodes and flat pricing strategy.

6.1 Target Web cluster architecture for the hosting of an e-commerce Web site

The target Web cluster architecture consists of a front-end component called Web switch and a number of homogeneous back-end server nodes connected by a high-

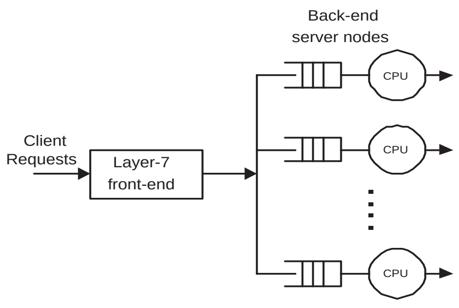


FIGURE 25: Queuing model of the target Web cluster architecture upon which an e-commerce Web site is built.

speed LAN. The Web switch acts as the network representative for an e-commerce Web site built upon the target cluster architecture, making the distributed nature of the site architecture completely transparent to its customers. In such a way, the authoritative DNS server for the e-commerce site translates the site name into the IP address of its Web switch, which receives all inbound requests destined for the site and then distribute them across the back-end nodes. Moreover, to enable QoS support in the e-commerce site, the Web switch must be able to examine the content of a HTTP request and identify its requested service class, i.e., it is the so-called layer-7 Web switch [75]. The above Web cluster architecture can be further classified on the basis of whether the data from the back-end server nodes to clients (outgoing data) go through the Web switch. In our target cluster architecture, the TCP handoff mechanism [63] is deployed to enable the back-end nodes respond to the clients directly without passing through the front-end nodes as an intermediary. Thus the target cluster architecture can be abstracted as a queuing system shown in Figure 25.

Suppose that an e-commerce Web site built upon the target cluster architecture consists of a layer-7 Web switch and N homogeneous back-end server nodes, each of which has the processing capacity C bits/s; there are a total of m service classes supported in the site. The idea of partitioning server resources among the supported service classes is to partition the N back-end server nodes into m disjoint server subsets so that each class of requests will be served only by its own server subset assigned. Specifically, the server subset assigned to class i is denoted by S_i and the number of server nodes in S_i is denoted by n_i, then $S_i \cap S_j = \oslash$, for $i \neq j$ and $i, j \in [1, m]$, and $\sum_{i=1}^{m} n_i = N$. Thus, the problem of deriving the optimal/suboptimal resource partitioning scheme is actually to find the optimal/suboptimal value of n_i, $i \in [1, m]$. Note that n_i does not have to be an integer, which means that a back-end server node may actually be assigned to multiple service classes with each class taking a portion of it. In this case, we have that back-end node serve those service classes by WFQ algorithm and the WFQ weights equal their shares in that node, respectively.

Based on the analysis in [88], the service time of a class i request at a back-end node is proportional to the size of its requested Web object, i.e., $X_i = L_i/C$, where

L_i denotes the size (Bytes) of the Web object requested by class i customers and C (Bytes/s) is the processing capacity of the back-end node. Thus, $\bar{L}_i = E[L_i]$, $\bar{L_i^2} = E[L_i^2]$ and $\bar{X}_i = E[X_i] = \bar{L}_i/C$, $\bar{X_i^2} = E[X_i^2] = \bar{L_i^2}/C^2$. In our scheme, the layer-7 Web switch distributes the inbound requests from class i uniformly among the back-end server nodes within server subset S_i to make the server loads balanced. In other words, if the overall arrival rate of class i requests to the e-commerce site is λ_i requests/s and a back-end server node in S_i is used exclusively by class i requests, the mean arrival rate of class i requests to that back-end node can be estimated as λ_i/n_i. Furthermore, in this dissertation, the processing delay at the layer-7 switch is neglected due to the fact that in a Web environment the client-to-server packets are typically much less than the server-to-client packets and the chosen QoS metric in a SLA is the *mean request delay* in the e-commerce Web site. Hence, according to the queuing theory of M/G/1, the analytic mean delay of class i requests $\hat{\bar{d}}_i$ in the e-commerce site can be calculated as follows.

$$\hat{\bar{d}}_i = \bar{X}_i + \frac{\frac{\lambda_i}{n_i}\bar{X_i^2}}{2(1 - \frac{\lambda_i}{n_i}\bar{X}_i)} = \frac{\bar{L}_i}{C} + \frac{\lambda_i \bar{L_i^2}}{2C(n_iC - \lambda_i)\bar{L}_i}. \tag{6.1}$$

The natural constraint of Eq. (6.1) is $n_iC > \lambda_i\bar{L}_i$ due to the fact that delay can not be negative.

6.2 Optimal resource partitioning scheme for the hosting of an e-commerce site under linear pricing strategy

Consider an e-commerce Web site built upon the target cluster architecture with N back-end server nodes and m service classes supported. The processing capacity of each back-end node is C bytes/s. Additionally, the arrival rate of class i requests is denoted by λ_i requests/s and the mean delay of class i requests in the e-commerce site denoted by $\bar{d}_i$. The linear pricing strategy for class i requests is characterized by the following definition of the linear pricing function $r_i(\bar{d}_i)$.
Definition 4: *The function*

$$r_i(\bar{d}_i) = b_i - k_i\bar{d}_i, \;\; i = 1, 2, ..., m, \;\; b_i > 0, \;\; k_i > 0 \tag{6.2}$$

is called the *linear pricing function* of class i, where $\bar{d}_i$ is the mean request delay of class i, b_i and k_i are both positive constants and $b_i \geq b_j$ and $k_i \geq k_j$ hold to ensure differentiated pricing if class i has a higher priority than class j.

As the QoS metric considered in the SLA is the *mean request delay*, the mean delay $\bar{d}_i$ of class i requests in the e-commerce site will be measured periodically and the SLA revenue due to serving class i requests can be determined also periodically based on class i pricing function and the above QoS (mean request delay) measurement. Specifically, we use Eq. (6.1) to estimate $\bar{d}_i$. Thus, the SLA revenue F obtained in the hosting of the e-commerce site during one measurement period is defined as follows under the linear pricing function in Eq. (6.2):

$$F = \sum_{i=1}^{m} r_i(\bar{d}_i) = \sum_{i=1}^{m} [b_i - k_i(\frac{\bar{L}_i}{C} + \frac{\lambda_i \bar{L_i^2}}{2C(n_i C - \lambda_i \bar{L}_i)})]. \tag{6.3}$$

Moreover, the issue of maximizing SLA revenue in the hosting of an e-commerce Web site under linear pricing strategy can be formulated as follows:

$$\max \quad F = \sum_{i=1}^{m} [b_i - k_i(\frac{\bar{L}_i}{C} + \frac{\lambda_i \bar{L_i^2}}{2C(n_i C - \lambda_i \bar{L}_i)})] \tag{6.4}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} n_i = N, \; 0 < n_i < N, \tag{6.5}$$

$$n_i C > \lambda_i \bar{L}_i. \tag{6.6}$$

By the Lagrangian optimization approach, we derived the following optimal resource partitioning scheme in publication **PXII** [93], which can achieve the maximum SLA revenue F under linear pricing strategy when hosting an e-commerce site built upon the above target cluster architecture.

$$n_i = \frac{(CN - \sum_{j=1}^{m} \lambda_j \bar{L}_j)\sqrt{\frac{k_i \lambda_i \bar{L_i^2}}{2}}}{C \sum_{j=1}^{m} \sqrt{\frac{k_j \lambda_j \bar{L_j^2}}{2}}} + \frac{\lambda_i \bar{L}_i}{C}, \; i \in [1, m]. \tag{6.7}$$

Publication **PXII** [93] made two sets of simulations to evaluate the effectiveness of the derived optimal resource partitioning scheme for maximizing the SLA revenue under linear pricing strategy, where the Bounded Pareto distribution ($BP(p, q, \alpha)$) [19] is used to model the heavy-tailed characteristic of Web objects. The simulation results there demonstrate that our derived optimal resource partitioning scheme can succeed in implementing the maximization of SLA revenues under a given amount of server resources and linear pricing strategy when a Web service provider hosts an e-commerce Web site by the above target Web cluster architecture.

6.3 Suboptimal resource partitioning scheme for the hosting of an e-commerce site under flat pricing strategy

The suboptimal resource partitioning scheme is proposed here for maximizing SLA revenue in the hosting of an e-commerce Web site under flat pricing strategy.

6.3.1 Flat pricing strategy for the hosting of an e-commerce site

As *mean request delay* is deployed as the QoS metric in the SLA, the definition of flat pricing strategy in this case is almost the same as the one defined in Chapter 5.2.1 except that $\bar{d}_i$ denotes the mean request delay of class i here. Specifically, consider an e-commerce Web site built upon the above target cluster architecture with N back-end server nodes and m service classes supported. The flat pricing strategy for class i is characterized by the following definition of flat pricing function.

Definition 5: *The function*

$$r_i(\bar{d}_i) = \begin{cases} R_i & \text{if } \bar{d}_i \le D_i \\ -P_i & \text{if } \bar{d}_i > D_i \end{cases}, \quad i = 1, 2, ..., m \tag{6.8}$$

is called the *flat pricing function* of class i in the hosting of an e-commerce Web site, where D_i is the QoS (mean request delay) guarantee required by class i requests and R_i and P_i are both positive constants. The above flat pricing function specifies that if the real mean delay offered to class i requests is less than D_i during one charging period, the Web service provider will receive a revenue R_i, otherwise a penalty P_i is incurred for failing to meet that D_i. Moreover, $R_i \ge R_j$ and $P_i \ge P_j$ should hold to ensure differentiated pricing if class i has higher priority than class j, which are expected under the SLA requirement.

6.3.2 Suboptimal resource partitioning scheme under the flat pricing strategy

Suppose that an e-commerce Web site built upon the above target cluster architecture consists of N homogeneous back-end server node, each of which has the processing capacity C bits/s, and supports a total of m service classes with Poisson arrival rate λ_1, λ_2, ..., λ_m, respectively. As the analytic mean delay $\hat{\bar{d}}_i$ of class i requests in Eq. (6.1) can be used to estimate the real mean request delay $\bar{d}_i$, $\hat{\bar{d}}_i$ has to be less than D_i so that the SLA revenue R_i can be obtained during one charging period for serving class i requests. The minimum number of back-end server nodes which has to be assigned to class i to meet its QoS guarantee D_i can be derived from the inequality $\hat{\bar{d}}_i \le D_i$. However, the real mean request delay $\bar{d}_i$ of class i will definitely differ by a small amount from $\hat{\bar{d}}_i$ as shown in publication **PXII** [93]. Hence, to guarantee $\bar{d}_i \le D_i$ in the real situation, we may deploy a small constant parameter ϵ_i for class i and then construct the following inequality:

$$\hat{\bar{d}}_i + \epsilon_i = \frac{\bar{L}_i}{C} + \frac{\lambda_i \bar{L_i^2}}{2C(n_i C - \lambda_i)\bar{L}_i} + \epsilon_i \le D_i. \tag{6.9}$$

By solving this inequality, the solution to n_i for a general distribution of Web object size can be calculated as follows:

$$n_i \ge \frac{\lambda_i \bar{L_i^2}}{2C[(D_i - \epsilon_i)C - \bar{L}_i]} + \frac{\lambda_i \bar{L}_i}{C} \tag{6.10}$$

$$\text{under the constraint } \sum_{i=1}^{m} n_i \le N, \;\; 0 < n_i \le N. \tag{6.11}$$

Note that the parameter ϵ_i determines how well the inequality $\bar{d}_i \le D_i$ will be guaranteed. We should set the value of ϵ_i carefully based on the system parameter settings and the burstiness of class i traffic. As a result of the above solution to n_i

in (6.10), we present the suboptimal resource partitioning scheme for maximizing the SLA revenue in the hosting of an e-commerce site built upon the target cluster architecture under a given amount (N back-end server nodes here) of server resources and flat pricing strategy as follows:

1. Set $n_{i,min} = \frac{\lambda_i \bar{L_i^2}}{2C[(D_i - \epsilon_i)C - \bar{L_i}]} + \frac{\lambda_i \bar{L_i}}{C}$, i=1,2,...,m,

2. If $\sum_{i=1}^{m} n_{i,min} \leq N$, then the above $n_{i,min}$, i=1,2,...,m, is the suboptimal resource partitioning scheme in this case,

3. Otherwise, it means that the N back-end server nodes are not enough to guarantee that $\bar{d_i} \leq D_i, i \in [1, m]$ for all the supported service classes. Hence, we should first satisfy the QoS guarantees of a set of selected service classes so that the obtained SLA revenue (actually revenue plus penalty) is the highest in this situation. The remaining server resources are allocated to all the supported service classes other than those selected ones uniformly.

Note that the above set of selected service classes in Step 3 is acquired by the comparison of the resulted SLA revenues under all possible resource partitioning schemes, which makes its calculation complexity increases quickly with larger values of m. Hence, instead we may first assign $n_{1,min}$ back-end server nodes to class 1, $n_{2,min}$ nodes to class 2,..., and $n_{j,min}$ nodes to class j ($j \in [1, m-1]$) until $\sum_{i=1}^{j+1} n_{i,min} > N$. The derivation of suboptimal resource partitioning scheme is illustrated in the next section.

6.3.3 Simulation results

Here some simulation results are presented to illustrate the effectiveness of our above approach by which the suboptimal resource partitioning scheme can be derived for the hosting of an e-commerce Web site under flat pricing strategy. Throughout this section, we study an e-commerce site built upon the target Web cluster architecture which consists of a layer-7 Web switch and 16 homogeneous back-end server nodes (N=16) that support three service classes (m=3, namely, Gold, Silver and Bronze classes). Moreover, each back-end server node has the processing capacity of C=5.95MB/s and the parameters of the three flat pricing functions for Gold, Silver and Bronze classes are summarized below: D_1=18ms, R_1=10 money units, $P1$=15 money units, D_2=25ms, R_2=5 money units, $P2$ =8 money units, and D_3=45ms, R_3=2 money units, $P3$=4 money units. Furthermore, ϵ_1=3ms, ϵ_2=4ms and ϵ_3=6ms are set for Gold, Silver and Bronze classes, respectively.

For actual Web workloads, it is recognized that Web object sizes are distributed with a heavy tail. Here the Bounded Pareto distribution ($BP(p, q, \alpha)$) [19] is used to model the heavy-tailed characteristic of Web objects. Specifically, the mean size of Web objects is set to 21KB as measured in [2] with p=1KB and q=10MB are chosen as the reasonable minimum and maximum Web object size, respectively. The resulting α=0.8037 is within the range of α values measured in

TABLE 6: For the first simulation: mean request delay in the e-commerce Web site.

	Mean request delay of Gold class ($\bar{d_1}$)	Mean request delay of Silver class ($\bar{d_2}$)	Mean request delay of Bronze class ($\bar{d_3}$)
By the suboptimal scheme	16.2579 ms	22.4538 ms	36.1440 ms
By the proportional scheme	25.7174 ms	24.1147 ms	23.8124 ms

[1] and [18]. The arrival process of client requests destined for the e-commerce site was modelled by Poisson distribution. Additionally, for each of the following simulations, we first derive the suboptimal resource partitioning scheme by our above approach and then deploy it as well as another proportional resource partitioning scheme in the simulation for comparison. Specifically, the *proportional* resource partitioning scheme assigns the following number of back-end server nodes to class i requests: $n_{i,proportional} = \lambda_i \bar{L_i} / \sum_{j=1}^{m}(\lambda_j \bar{L_j}), i \in [1, m]$.

In the first simulation, λ_1=100 requests/s, λ_2=150 requests/s and λ_3=250 requests/s. Thus, we see that $n_{1,min}$=5.3899, $n_{2,min}$=5.4919 and $n_{3,min}$=4.9559 by Eq. (6.10). As $\sum_{i=1}^{3} n_{i,min}$=15.8377 $<$ N=16, the e-commerce site has enough server resources to satisfy the QoS guarantees of Gold, Silver and Bronze classes and the above set of $n_{1,min}$, $n_{2,min}$ and $n_{3,min}$ is the suboptimal resource partitioning scheme in this case. Moreover, because the remaining server resources ($N - \sum_{i=1}^{3} n_{i,min}$=0.1623) is less than 1 and in a back-end server node which will serve multiple service classes, the requests from the multiple classes share the back-end node by WFQ algorithm, we allotted the remaining server resources to Gold class requests in the simulation. Hence, the deployed suboptimal resource partitioning scheme is as follows: $n_{1,suboptimal} = 5.5522$, $n_{2,suboptimal} = 5.4919$ and $n_{3,suboptimal} = 4.9559$. The simulation results of the mean request delays in the e-commerce site are presented in Table 6.

It can be seen from Table 6 that $\bar{d_1} \leq D_1 = 18ms$, $\bar{d_2} \leq D_2 = 25ms$ and $\bar{d_3} \leq D_3 = 45ms$ are all satisfied by the derived suboptimal resource partitioning scheme. Hence, the suboptimal resource partitioning scheme achieves the maximum value $\sum_{i=1}^{3} R_i$=17 money units of the SLA revenue during one charging period (400s here) for the hosting of the e-commerce site under the flat pricing strategy. Whereas, the proportional resource partitioning scheme can not have $\bar{d_1} \leq D_1$ hold, which results in $-P_1 + R_2 + R_3 = -8$ money units of the SLA revenue during the same charging period. In other words, the service provider will get the loss of 8 money units due to failing to satisfy the QoS guarantee of Gold class requests when using the proportional scheme.

Next, the workload intensity with λ_1=120 requests/s, λ_2=180 requests/s and λ_3=300 requests/s is fed into the e-commerce Web site in the second simulation, which leads to $n_{1,min}$=6.4679, $n_{2,min}$=6.5903 and $n_{3,min}$=5.9471 by Eq. (6.10). As $\sum_{i=1}^{3} n_{i,min}$ = 19.0053 $>N$=16, it means that the e-commerce site does not have enough server resources available to satisfy the QoS guarantees of all supported service classes (Gold, Silver and Bronze classes here). However, it is noticed

TABLE 7: For the second simulation: mean request delay in the e-commerce Web site.

	Mean request delay of Gold class ($\bar{d_1}$)	Mean request delay of Silver class ($\bar{d_2}$)	Mean request delay of Bronze class ($\bar{d_3}$)
By the suboptimal scheme	16.7959 ms	21.8557 ms	90.8417 ms
By the proportional scheme	26.3930 ms	27.4331 ms	30.4962 ms

that $n_{1,min} + n_{2,min} < 16$ holds in this case. Hence, we can derive the following suboptimal resource partitioning scheme by first guaranteeing the satisfaction of QoS requirements of both Gold and Silver classes: $n_{1,suboptimal}$=$n_{1,min}$=6.4679, $n_{2,suboptimal}$=$n_{2,min}$=6.5903 and $n_{3,suboptimal}$=$N - n_{1,min} - n_{2,min}$=2.9418. The simulation results for this case are presented in Table 7.

Table 7 shows that $\bar{d_1} \leq D_1 = 18ms$ and $\bar{d_2} \leq D_2 = 25ms$ both hold although $\bar{d_3} > D_3 = 45ms$ when the above suboptimal resource partitioning scheme is deployed. Hence, the suboptimal scheme can achieve the highest revenue in this case, i.e., $R_1 + R_2 - P3$=11 money units. Whereas, although the proportional resource partitioning scheme satisfies the QoS guarantee of Bronze class ($\bar{d_3} <$ 45 ms), it sacrifices the QoS performances of Gold and Silver classes (i.e., $\bar{d_1} >$ 18 ms and $\bar{d_2} >$ 25 ms). Thus, the Web service provider will get the loss of $P_1 + P_2 - R_3$=21 money units during each charging period for the hosting of the e-commerce site by the proportional resource partitioning scheme. Therefore, based on the above simulation results, it can be concluded that the suboptimal resource partitioning scheme derived by our proposed approach is able to achieve the highest SLA revenue for the hosting of an e-commerce site under flat pricing strategy.

6.4 Summarized contributions

Cluster-based Web server systems have become a major means to hosting e-commerce sites. In this Chapter, we link the issue of resource partitioning scheme with the pricing strategy in a Service-Level-Agreement (SLA) and analyze the problem of maximizing the revenues obtained in the hosting of an e-commerce site with a SLA contract by optimally partitioning the server resources among all supported service classes. In publication **PXII** [93], the optimal resource partitioning scheme is derived under a given amount of server resources and linear pricing strategy when the QoS metric in the SLA is *mean request delay*, which has the closed-form solution to the optimal number of the back-end server nodes assigned to each service class. Moreover, the closed-form solution can apply to any general distribution of requested Web object size. This Chapter first presents the target Web cluster architecture upon which an e-commerce site is built and then summarizes the contributions in publication **PXII**. Finally, the suboptimal resource partitioning scheme is proposed based on the analysis of the target cluster architecture when the *mean request delay* is chosen as the QoS metric in the SLA, which can achieve the highest SLA revenue obtained for the hosting an e-commerce site under a given amount of back-end server nodes and flat pricing strategy. The sub-optimal resource partitioning scheme can also apply to any general size distribution of requested Web object.

7 CONCLUSIONS

In this dissertation, we studied the QoS- and Revenue-aware resource allocation mechanisms in both cluster-based Web server systems and IP network nodes.

Chapter 1 contained the introduction, where the cluster architectures for building cluster-based Web server systems, the target multiclass-supported network node and the basic concepts of QoS and SLA were described.

Chapter 2 summarized our contribution for enabling differentiated services in cluster-based Web server systems. The novel *Arrival-Related Dynamic Partitioning* mechanism proposed in publication **PVI** was also presented in a clearer format there.

Chapter 3 was dedicated to addressing the issue of implementing scalable service differentiation in such a cluster-based Web server system where a single Web site is hosted.

In Chapter 4, the revenue-aware resource allocation schemes for a multiclass-supported network node were presented, which aims at the maximization of SLA revenue obtained in the node under a given amount of network resources and linear pricing strategy by optimally allocating the resources among the supported service classes.

Chapter 5 first summarized our proposed upper bound on mean packet delay of GPS-based FQ algorithms and then utilized this upper delay bound to derive the suboptimal resource allocation scheme for maximizing the SLA revenues acquired in a GPS-based network node under a given amount of network resources and flat pricing strategy.

In Chapter 6, the problem of maximizing the SLA revenue attained in the hosting of an e-commerce Web site by cluster-based Web server systems was analyzed. First, the optimal resource partitioning scheme proposed in publication **PXII** under linear pricing strategy was summarized and then the suboptimal resource partitioning scheme was derived, which can achieve the highest SLA revenue for the hosting an e-commerce site under a given amount of server resources and flat pricing strategy.

The main contributions of this dissertation are highlighted shortly as follows:

- A novel *Arrival-Related Dynamic Partitioning* mechanism was proposed to enable differentiated services in cluster-based Web server systems, which works well even when the Web cluster system is heavily loaded and has no enough server resources to be allocated.

- A scalable Web cluster architecture – the two-level cluster architecture, was proposed, whose scalability is theoretically determined only by the scalability of its layer-4 switch. Moreover, the relevant scheduling algorithms for enabling scalable service differentiation in the two-level cluster architecture were selected and designed.

- The revenue-aware resource allocation schemes were proposed which can achieve the maximization of the SLA revenue obtained in a multiclass-supp-
  orted network node under a given amount of network resources and linear pricing strategy. More importantly, in some cases, the closed-form solutions to the optimal resource allocation schemes were derived from the revenue target function by Lagrangian optimization approach under linear pricing strategy.

- A novel upper bound on mean packet delay of GPS-based FQ algorithms was derived under the probabilistic traffic model of Poisson arrival and any general packet length distribution and it is much simpler and tighter than the known ones by M. Hawa *et al.* Moreover, the derived upper bound fits a class of GPS-based packetized FQ algorithms including WFQ, SCFQ and SPFQ.

- The suboptimal resource allocation scheme was proposed for maximizing the SLA revenue in a GPS-based network node under a given amount of network resources and flat pricing strategy.

- The optimal resource partitioning scheme was derived, which can implement the maximization of the SLA revenue attained for the hosting of an e-commerce Web site by cluster-based Web server systems under a given amount of server resources and linear pricing strategy. Additionally, the suboptimal resource partitioning scheme was also proposed for achieving the highest SLA revenue in the hosting of an e-commerce site under a given amount of server resources and flat pricing strategy.

REFERENCES

[1] M. F. Arlitt and T. Jin, "A workload characterization study of the 1998 World Cup Web site," IEEE Network, 14(3):30-37, May/June 2000.

[2] M. Arlitt and C. Williamson, "Web server workload characterization: the search for invariants," In Proceedings of ACM SIGMETRICS, pp. 126-137, 1996.

[3] M. Aron, D. Sanders, P. Druschel, and W. Zwaenepoel, "Scalable content-aware request distribution in cluster-based network servers," In Proc. of USENIX 2000 Conf., San Diego, CA, June 2000.

[4] M. Aron, P. Druschel, and W. Zwaenepoel, "Cluster reserves: a mechanism for resource management in cluster-based network servers," In Proc. of ACM Sigmetrics 2000, pp. 90-101, Santa Clara, CA, June 2000.

[5] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for Traffic Engineering over MPLS," RFC 2702, IETF, September 1999.

[6] T. Basar, R. Srikant, "Revenue-maximizing rpicing and capacity expansion in a many-users regime," In Proc. of IEEE INFOCOM2002, Vol.1, 2002, pp. 294-301.

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

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

[9] V. Cardellini, E. Casalicchio, M. Colajanni, and S. Tucci, "Mechanisms for quality of service in web clusters," Computer Networks, Elsevier Science, 36(6):759-769, Nov. 2001.

[10] V. Cardellini, E. Casalicchio, M. Colajanni, and P. S. Yu, "The state of the art in locally distributed Web-server systems," IBM research report, RC22209 (W0110-048), Oct. 16, 2001.

[11] Z. Cao, E. W. Zegura, "Utility Max-Min: An Application-Oriented Bandwidth Allocation Scheme," In Proc. of IEEE INFOCOM99, New York, USA, 1999.

[12] X. Chen and P. Mohapatra, "Providing differentiated service from an Internet server," In Proc. IEEE Int'l Conf. on Computer Comm. and Networks, Boston, MA, Oct. 1999.

[13] L. Cherkasova and P. Phaal, "Session based admission control: a mechanism for improving performance of commercial Web sites," In Proc. of Int'l Workshop on Quality of Service, London, UK, June 1999.

[14] F. M. Chiussi and A. Francini, "Implementing Fair Queueing in ATM switches - Paart 1: A Practical Methodology for the Analysis of Delay Bounds," In Proc. of GLOBECOM'97, Vol. 1, pp. 509-518, November 1997.

[15] Cisco Systems Inc. LocalDirector. http://www.cisco.com.

[16] A. Cohen, S. Rangarajan, and H. Slye, "On the performance of TCP splicing for URL-aware redirection," In Proc. of USENIX Symp. on Internet Technologies and Systems, Boulder, CO, Oct. 1999.

[17] C. Courcoubetis, F. P. Kelly, and R. Weber, "Measurement-based usage charges in communication networks," Oper. Res., Vol.48, No.4, pp. 535-548, 2000.

[18] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes," IEEE/ACM Trans. on Networking, 5(6):835-846, Dec. 1997.

[19] M. Crovella, M. Harchol-Balter, and C. Murta, "Task assignment in a distributed system: improving performance by unbalancing load," Performance Evaluation Review, Vol. 26, No. 1, pp. 268-269, 1998.

[20] M. Dahlin, "Interpreting stale load information," IEEE Trans. on Parallel and Distributed Systems, 11(10):1033-1047, Oct. 2000.

[21] A. Demers, S. Keshav and S. Shenker, "Analysis and Simulation of a Fair Queuing Algorithm," Journal of Internetworking Research and Experience, pp. 3-26, October 1990.

[22] Y. Diao, J. L. Hellerstein and S. Parekh, "Using fuzzy control to maximize profits in service level management," IBM Systems Journal, Vol. 41, No. 3, pp. 403-420, 2002.

[23] D. M. Dias, W. Kish, R. Mukherjee, and R. Tewari, "A scalable and highly available Web server," In proceedings of 41st IEEE Computer Society Int'l Conf., pp. 85-92, Feb. 1996.

[24] K. Egevang and P. Francis, "The IP Network Address Translator (NAT)," RFC 1631, May 1994.

[25] M. A. ElGendy, A. Bose and K. G. Shin. "Evolution of the Internet QoS and support for soft real-time applications," Proceedings of the IEEE, Issue 7, Volume 91, pp. 1086 - 1104, July 2003.

[26] K. Fall and J. Pasquale, "Exploiting In-Kernel Data Paths to Improve I/O Throughput and CPU Availability," In Proceedings of the IEEE International Computer Conference, San Jose, CA, Feb. 1996.

[27] Z. Genova and K. Christensen, "Challenges in URL Switching for Implementing Globally Distributed Web Sites," In Proceedings of the Workshop on Scalable Web Services, pp. 89-94, August 2000.

[28] R. J. Gibbens and F. P. Kelly, "Resource pricing and the evolution of congestion control", Automatica, Vol.35, No.12, pp. 1969-1985, 1999.

[29] S. J. Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," In Proc. of INFOCOM'94, pp. 636-646, April 1994.

[30] S. J. Golestani, "Network Delay Analysis of a Class of Fair Queueing Algorithms," IEEE Journal on Selected Areas in Communications, Vol. 13, No. 6, pp. 1057-1070, August 1995.

[31] P. Goyal, S. S. Lam and H. M. Vin, "Determining End-to-End Delay Bounds in Heterogeneous Networks," In Proc. of 5th International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 287-298, April 1995.

[32] T. Hämäläinen, J. Joutsensalo and J. Zhang, "Optimal Link Allocation and Charging Model," In Proceedings of the International Conference on Advances in Infrastructure for E-business, E-education, E-science, and E-medicine on the Internet (SSGRR2002), Jul. 29-Aug. 4, 2002.

[33] T. Hämäläinen and J. Joutsensalo, "Link Allocation and Revenue Optimization for Future Networks", Proc. of IEEE Globecom 2002, Nov. 2002, Taipei.

[34] M. Hawa and D. W. Petr, "M/G/FQ: Stochastic Analysis of Fair Queueing Systems," In Proc. of IEEE 2nd International Conference on Networking, pp. 368-381, Aug. 2002.

[35] G. S. Hunt, G. D. H. Goldszmidt, R. P. King, and R. Mukherjee, "Network Dispatcher: A connection router for scalable Internet services," Computer Networks, 30(1-7):347-357, 1998.

[36] IBM Corporation. IBM interactive network dispatcher. http://www.ibm.com/software/network/dispatcher.

[37] ITU-T Recommendation I.371. "Traffic control and congestion control in BISDN," ITU-T Study Group 13, April 1996, Geneva.

[38] J. Joutsensalo and T. Hämäläinen, "Optimal Link Allocation and Revenue Maximization," Journal of Communications and Networks, Vol.4, No.2, pp. 136-147, June 2002.

[39] J. Joutsensalo, T. Hämäläinen and J. Zhang, "Resource Allocation for Differentiated QoS by Adaptive Weighted Fair Queue Technique," In Proceedings of 16th Nordic Teletraffic Seminar (NTS16), pp. 291-302, August 21-23, 2002.

[40] J. Joutsensalo, T. Hämäläinen and J. Zhang, "Revenue maximization-based adaptive WFQ," In Proceedings of SPIE symposium of Asia-Pacific Optical and Wireless Communication (APOC2002), Volume 4907, pp. 108-117, Oct. 13-18, 2002.

[41] J. Joutsensalo, T. Hämäläinen, M. Pääkkönen, and A. Sayenko, "Adaptive Weighted Fair Scheduling Method for Channel Allocation," In Proceedings of IEEE ICC2003, Alaska, May 2003.

[42] K. Kaario, T. Hämäläinen, and J. Zhang, "Tuning of QoS aware load balancing algorithm (QoS-LB) for highly loaded server clusters," In Proceedings of First International Conference on Networking (ICN'01), Part II, pp. 250-258, Jul. 11-13, 2001.

[43] V. Kanodia and E. W. Knightly, "Multi-class latency-bounded web services," In Proc. Int'l Workshop on Quality of Service, Pittsburgh, PA, June 2000.

[44] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input versus output queueing on a space-division packet switch," IEEE Transactions on Communications, Vol.35, pp. 1347-1356, December 1987.

[45] F. P. Kelly, "On tariffs, policing and admission control for multiservice networks," Oper. Res. Lett., Vol. 15, pp. 1-9, 1994.

[46] F. P. Kelly, "Notes on effective bandwidths," in Stochastic Networks: Theory and Applications, S. Zachary, I. B. Ziedins, and F. P. Kelly, Eds. London, U.K.: Oxford Univ. Press, Vol.9, pp. 141-168, 1996.

[47] F. P. Kelly, "Charging and rate control for elastic traffic," European Transaction on Telecommunication, Vol.8, pp. 33-37, 1997.

[48] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," Journal of the Operational Research Society, Vol.49, pp. 237-252, 1998.

[49] S. Knight, D. Weaver, D. Whipple, R. Hinden, D. Mitzel, P. Hunt, P. Higginson, M. Shand and A. Lindem, "Virtual Router Redundancy Protocol," RFC 2338. April 1998.

[50] H. Knoche, H. De Meer, "QoS parameters: A comparative study," University of Hamburg, Tech. Rep. 1997.

[51] D. Krishnamurthy and J. Rolia, "Predicting the QoS of an electronic commerce server: Those mean percentiles," In Proc. of Workshop on Internet Server Performance, Madison, WI, June 1998.

[52] R. J. La and V. Anantharam, "Utility-based Rate Control in the Internet for Elastic Traffic," IEEE/ACM Transactions on Networking, Vol.10, Issue: 2, pp. 272-286, April 2002.

[53] C. Lee, J. Lehoczky, R. Rajkumar, D. Siewiorek, "On Quality of Service Optimization with Discrete QoS Options," In Proc. of IEEE Real-Time Technology and Application Symposium, June 1999.

[54] R. Levy, J. Nagarajao, G. Pacifici, M. Spreitzer, A. Tantawi, and A. Youssef, "Performance management For Cluster Based Web Services," In Proc. of 8th IFIP/IEEE International Symposium on Integreted Network Management, Mar. 24-28, 2003.

[55] T. Li, B. Cole, P. Morton and D. Li, "Cisco Hot Standby Router Protocol (HSRP)," RFC 2281, March 1998.

[56] Z. Liu, M. Squillante, and J. Wolf, "On Maximizing Service-Level-Agreement Profits," In Proceedings of the 3rd ACM conference on Electronic Commerce, pp. 213¨C223, 2001.

[57] D. H. Lorenz and A. Orda, "Optimal partition of QoS requirements on unicast paths and multicast trees," In Proc. of IEEE INFOCOM'99, pp. 246-253, March 1999.

[58] S. H. Low, "Equilibrium Allocation of Variable Resources for Elastic Traffics," In Proc. of IEEE INFOCOM'98, San Francisco, USA, 1998.

[59] J.K. Mackie-Mason, H.R. Varian, "Pricing the Internet: Public Access to the Internet," Prentice-Hall, 1995.

[60] L. Massoulie, J. Roberts, "Bandwidth Sharing: Objectives and Algorithms," In Proc. of IEEE INFOCOM99, New York, USA, 1999.

[61] M. Mitzenmacher, "How useful is old information," IEEE Trans. on Parallel and Distributed Systems, 11(1):6-20, Jan. 2000.

[62] R. Nagarajan, J. F. Kurose and D. Towsley, "Allocation of Local Quality of Service Constraints to Meet End-to-End Requirements," IFIP Workshop on the Performance Analysis of ATM Systems, Martinique, Jan. 1993.

[63] V. S. Pai, M. Aron, G. Banga, M. Svendsen, P. Druschel, W. Zwaenepoel, and E. Nahum, "Locality-Aware Request Distribution in Cluster-based Network Servers," Proceedings of the 8th Conference on Architectural Support for Programming Languages and Operating Systems, San Jose, CA, Oct. 1998.

[64] 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 Trans. on Networking, Vol.1, No.3, pp. 344-357, June 1993.

[65] 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 Trans. on Networking, Vol.2, No.2, pp. 137-150, April 1994.

[66] I. Ch. Paschalidis, "Class-specific quality of service guarantees in multimedia communication networks," Automatica, Vol.35, No.12, pp. 1951-1968, 1999.

[67] I. Ch. Paschalidis and J. N. Tsitsiklis, "Congestion-dependent pricing of network services," IEEE/ACM Transactions on Networking, Vol.8, pp. 171-184, April 2000.

[68] I. Ch. Paschalidis and Yong Liu, "Pricing in multiservice loss networks: static pricing, asymptotic optimality and demand subsitution effects," IEEE/ACM Transactions on Networking, Vol.10, Issue:3, pp. 425-438, June 2002.

[69] N. Pekergin, "Stochastic Bounds on Delays of Fair Queueing Algorithms," INFOCOM'99, pp. 1212-1219, 1999.

[70] C. Perkins, "IP encapsulation within IP," IETF RFC 2003, Oct. 1996.

[71] R. Rajkumar, C. Lee, J. Lehoczky, D. Siewiorek, "A Resource Allocation Model for QoS Management," IEEE Real-Time Systems Symposium, December 1997.

[72] Resonate Inc. http://www.resonate.com/.

[73] E. Rosen, A Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," April 1999. http://www.ietf.org/ids.by.wg/mpls.html.

[74] S. Sarkar, L. Tassiulas, "Fair Allocation of Utilities in Multirate Multicast Networks," In Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing, 1999.

[75] T. Schroeder, S. Goddard, and B. Ramamurthy, "Scalable Web server clustering technologies," IEEE Network, 14(3):38-45, May/June 2000.

[76] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," Audio-Video Transport Working Group, RFC 1889, January 1996.

[77] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550, July 2003.

[78] S. Shenker, "Fundamental Design Issues for the Future Internet," IEEE Journal on Selected Areas in Telecommunications, Vol.13, No.7, September 1995.

[79] J. Song, E. Levy-Abegnoli, A. Iyengar, and D. Dias. Design alternatives for scalable Web server accelerators. In Proc. of 2000 IEEE Int'l Symp. on Performance Analysis of Systems and Software, pp. 184-192, Austin, TX, Apr. 2000.

[80] Stardust.com. White Paper - The Need for QoS. July 1999. http://www.qosforum.com.

[81] D. Stiliadis and A. Varma, "Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms," In proc. of INFOCOM'96, pp. 111-119, March 1996.

[82] D. Stiliadis and A. Varma, "Efficient Fair Queueing Algorithms for Packet-Switched Networks," IEEE/ACM Trans. on Networking, Vol.6, No.2, pp. 175-185, April 1998.

[83] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry and S. Waldbusser, "Terminology for Policy-Based Management," RFC 3198, November 2001.

[84] K. Yamori and Y. Tanaka, "Relation between Willingness to Pay and Guaranteed Minimum Bandwidth in Multiple-Priority Service," In Proceedings of Joint Conference of 10th Asia Pacific Conf. on Communications and 5th International Symposium on Multi-Dimensional Mobile Communications (APCC2004 & MDMC2004), Aug. 29-Sept. 1, 2004.

[85] O. Yaron and M. Sidi, "Performance and Stability of Communication Networks via Robust Exponential Bounds," IEEE/ACM Trans. on Networking, Vol.1, No.3, pp. 372-385, 1993.

[86] J. Zhang, T. Hämäläinen, J. Joutsensalo and K. Kaario, "QoS-Aware Load Balancing Algorithm for Globally Distributed Web Systems," In Proceedings of the International Conferences on Info-tech and Info-net (ICII2001), Oct. 29 - Nov.1, 2001.

[87] J. Zhang, T. Hämäläinen and J. Joutsensalo, "Dynamic Partitioning: A Mechanism for Supporting Differentiated Services in Cluster-based Network Servers," In Proceedings of 16th Nordic Teletraffic Seminar (NTS16), pp. 185-193, August 21-23, 2002.

[88] J. Zhang, T. Hämäläinen and J. Joutsensalo, "A New Mechanism for Supporting Differentiated Services in Cluster-based Network Servers," In Proceedings of 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02), pp. 427-432, Oct. 12-16, 2002.

[89] J. Zhang, K. Luostarinen, T. Hämäläinen and J. Joutsensalo, "Enabling QoS Support in Global Replicated Web Systems," In Proceedings of 9th Asia Pacific Conference on Communications (APCC2003), Volume 2, pp. 735-739, Sept. 21-24, 2003.

[90] J. Zhang, T. Hämäläinen and J. Joutsensalo, "Revenue-aware Resource Allocation Scheme in Multiservice IP Networks," WSEAS Transactions on Communications, Issue 1, Volume 3, pp. 277-281, Jan. 2004, ISSN 1109-2742.

[91] J. Zhang, T. Hämäläinen and J. Joutsensalo, "Packet Scheduling for Maximizing Revenue in a Network Node," In Proceedings of 1st International

Conference on E-Business and Telecommunication Networks (ICETE2004), Volume 2, pp. 134-139, Aug. 25-28, 2004.

[92] J. Zhang, T. Hämäläinen and J. Joutsensalo, ”Optimal Resource Allocation Scheme for Maximizing Revenue in the Future IP Networks,” In Proceedings of Joint Conference of 10th Asia Pacific Conf. on Communications and 5th International Symposium on Multi-Dimensional Mobile Communications (APCC2004 & MDMC2004), Aug. 29-Sept. 1, 2004.

[93] J. Zhang, T. Hämäläinen and J. Joutsensalo, ”Maximizing Service-Level-Agreement Revenues in Clustered-based Web Server Systems,” WSEAS Transactions on Information Science and Applications, Issue 3, Volume 1, pp. 861-866, Sept. 2004, ISSN 1790-0832.

[94] J. Zhang, T. Hämäläinen and J. Joutsensalo, ”Revenue-aware Resource Allocation in the Future Multi-Service IP Networks,” In Proceedings of IFIP TC6 Conference on Network Control and Engineering for QoS, Security and Mobility (Net-Con’2004), Nov. 3-5, 2004.

[95] J. Zhang, T. Hämäläinen and J. Joutsensalo, ”Stochastic Analysis of Upper Delay Bound of GPS-based Packetized Fair Queueing Algorithms,” In Proceedings of 12th IEEE International Conference On Networks (ICON2004), Nov. 16-19, 2004.

[96] Z.-L. Zhang, D. Towsley and J. Kurose, ”Statistical Analysis of Generalized Processor Sharing Scheduling Discipline,” IEEE Journal of Selected Areas in Communications, 13(6): 1071-1080, August 1995.

[97] W. Zhao, D. Olshefski, and H. Schulzrinne, ”Internet Quality of Service: an Overview,” IEEE Network, September/October 1999.

[98] H. Zhu, H. Tang, and T. Yang, ”Demand-driven service diffentiation in cluster-based network servers,” In Proc. of IEEE INFOCOM’01, Apr. 2001.