JYVÄSKYLÄ STUDIES IN COMPUTING
75

Jarmo Siltanen

Quality of Service and Dynamic Scheduling for Traffic Engineering in Next Generation Networks

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 75

Jarmo Siltanen

Quality of Service and Dynamic Scheduling for Traffic Engineering in Next Generation Networks

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston liikunnan salissa L304
huhtikuun 25. päivänä 2007 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in Auditorium L304, on April 25, 2007 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2007

Quality of Service and Dynamic Scheduling for Traffic Engineering in Next Generation Networks

JYVÄSKYLÄ STUDIES IN COMPUTING 75

Jarmo Siltanen

Quality of Service and Dynamic Scheduling for Traffic Engineering in Next Generation Networks

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2007

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

URN:ISBN:9789513928117
ISBN 978-951-39-2811-7 (PDF)

ISBN 978-951-39-2792-9 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2007

ABSTRACT

Siltanen, Jarmo
Quality of Service and Dynamic Scheduling for Traffic Engineering in Next Generation Networks
Jyväskylä: University of Jyväskylä, 2007, 88 p.
(Jyväskylä Studies in Computing
ISSN 1456-5390; 75)
ISBN 978- 951-39-2811-7 (PDF), 978-951-39-2792-9 (nid.)
Finnish summary
Diss.

Quality of Service is important in modern networks for several reasons. Critical applications, such as real-time audio and video, should be given priority over less critical ones, such as file transferring and web surfing. Furthermore, many multimedia applications require delay or delay variation guarantees for acceptable performance.

This thesis presents simulation and analysis of adaptive resource allocation models. Presented models rely upon the Weighted Fair Queuing service policy and use the revenue criterion to adjust the weights. The purposes of the proposed models are to maximize a provider's revenue and at the same time ensure the required Quality of Service for the end-users. A network scenario, which consists of several intermediate-switching nodes, is considered. Besides, adaptive and non-adaptive approaches to the Weighted Fair Queuing in terms of obtained revenue and state of queues at intermediate nodes are compared. It is shown that the adaptive approach can improve the total revenue obtained by a provider when compared to a non-adaptive approach.

Keywords: Quality of Service, Accounting management, Performance management, Link allocation

Author's address	Jarmo Siltanen FIN-40100 Jyväskylä Finland Email: jarmo.siltanen@jamk.fi
Supervisors	Professor Timo Hämäläinen Department of Mathematical Information Technology University of Jyväskylä Finland
	Professor Jyrki Joutsensalo Department of Mathematical Information Technology University of Jyväskylä Finland
Reviewers	Prof. Gennady Yanovsky yanovsky <yanovsky@sut.ru> Head of Telecom. Netw. Dept. SPb State University of Telecom. Tel.: (+7812) 962 22 88 Fax: (+7812) 315 32 27
	Prof. Pekka Loula TTY/Porin yksikkö PL 300, 28101 Pori pekka.loula@tut.fi Puh. 02 627 2740
Opponent	Dr. Erkki Laitinen Matemaattisten tieteiden laitos Oulun yliopisto OULU

ACKNOWLEDGEMENTS

This thesis is based on work completed at the University of Jyväskylä during 1999-2007 while at the same time working for Jyväskylä University of Applied Sciences.

I wish to express my gratitude to professors Timo Hämäläinen and Jyrki Joutsensalo for their co-operation, advice, discussions, and help during this work. I also want to thank Kari Luostarinen of his time and effort during this dissertation project.

This thesis consists of publications that have been co-authored with Timo Hämäläinen, Jyrki Joutsensalo, Kari Luostarinen, Ari Viinikainen, Isto Kannisto, and Alexander Sayenko. Finally, many thanks to my wife Anita, my daughter Minna, and my son Juho for their patience and support on my dissertation project.

Jyväskylä 9th February 2007

Jarmo Siltanen

LIST OF SYMBOLS

$R(t)$	virtual time
P_i^α	transmission time for packet i in queue α
τ_i^α	arrival time for packet i in queue α
S_i^α	virtual start time for packet i in queue α
F_i^α	virtual finish time for packet i in queue α
w_α	assigned weight for queue α
$\hat{R}(t)$	simply extracted from the packet situated at the head of a queue in the SCFQ algorithm
$P(t)$	system potential
L_c	size of a cell
C	capacity of the server
$f^{\,j}$	service sequence in a round, flow to which slot j
ρ_i	allocated bandwidth for flow i
$\Re_i$	relative service share for flow i
Ω_i	accumulated relative service share for flow i
R	service rate at the shared outgoing link
L_c	constant packet length
C_i	price for the *i*th class
d_0	minimum processing time it takes the scheduler to transmit one byte of data from one of the input queues to the output
B_i^{user}	bandwidth allocated for each customer
N_i	active flows within the i th service class
σ	bucket depth
ρ	token rate
D_i	delay of the *i*th class
L_i^α	size of a packet

ϕ_α	weight for queue α
$E(b_i)$	average packet length in queue i
m	the number of service classes
N_i	number of connections in the ith queue
t	time index
F, R	revenue functions
w	weight
r	pricing function
d	delay
c_i	constant delay, covers insertion, transmission etc. delays
k_l	shifting factor related to pricing functions
λ	penalty factor
e_i	pricing factor for the service class i

ABREVIATIONS

ACK	Acknowledgement
AF	Assured Forwarding
ATM	Asynchronous Transfer Mode
BBRR	bit-by-bit round-robin
BE	Best Effort
BPDU	Bridge Protocol Data Unit
BRFQ	Bit-Round Fair Queue
CAC	Call Admission Control
CBQ	Class Based Queuing
CBR	Constant Bit Rate
CE	Customer Edge
CIM	Common Information Model
CoS	Class of Service
CPU	Central Processing Unit
DiffServ	Differentiated Services
DRR	Deficit Round Robin
DS	Differentiated Services
DSCP	Differentiated Services Code Point
DWRR	Deficit Weighted Round Robin
EF	Expedited Forwarding
FCFS	First Come First Served
FFQ	Frame-based Fair Queuing
FIFO	First In First Out
FQ	Fair Queuing
GPS	General Processor Sharing
H-VPLS	Hierarchical Virtual Private Lan Service
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IntServ	Integrated Services
LSP	Label Switched Path
L2VPN	Layer 2 Virtual Private Network
MAC	Medium Access Control
MPLS	Multi Protocol Label Switching
MTU	Multi Tenant Unit
NS2	Network simulator 2
PE	Provider Edge
PGPS	packet-by-packet generalized processor sharing
PHB	per-hop behavior
POP	Point of Presence
PQ	Priority Queue
PQWRR	Priority Queuing Weighted Round Robin
PWE3	Pseudowire Edge to Edge Emulation
P2MP	Point-to-Multi Point

P2P	Point-to-Point
QoS	Quality of Service
RPS	Rate-Proportional Servers
RR	Round Robin
RSVP	Resource ReServation Protocol
SCFQ	Self-Clocked Fair Queuing
SFQ	Stochastic Fair Queuing
SLA	Service Level Agreement
SPFQ	Starting Potential based Fair Queuing
STP	Spanning Tree Protocol
TDM	Time Division Multiplexing
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URR	Uniform Round Robin
VBR	Variable Bit Rate
VLAN ID	Virtual Local Area Network Identifier
VPLS	Virtual Private Lan Service
VPN	Virtual Private Network
WF2Q	Worst-case Fair Weighted Fair Queuing
WFQ	Weighted Fair Queuing
WRR	Weighted Round Robin
WRR-RSS	Weighted Round Robin using Relative Service Share
VWRR	Variably Weighted Round Robin
WWW	World Wide Web

LIST OF PUBLICATIONS

This thesis includes nine research papers, which structurally follow the selected research approach:

[PI] A. Sayenko, J. Joutsensalo, T. Hämäläinen, and J. Siltanen: "An adaptive Approach to WFQ with the Revenue Criterion", Proceedings of Eighth IEEE International Symposium on Computers and Communication, (ISCC 2003), Turkey, pp. 181-186, June 2003.

[PII] A. Sayenko, T. Hämäläinen, J. Joutsensalo, and J. Siltanen: "The simulation and analysis of the revenue criterion based adaptive WFQ", Proceedings of 18th IEEE International Teletraffic Conference (ITC'18), Volume 5b, Germany, pp. 1071-1080, September 2003.

[PIII] A. Sayenko, T. Hämäläinen, J. Joutsensalo, and J. Siltanen: "On providing bandwidth and delay guarantees using the revenue criterion based adaptive WFQ", Proceedings of the 9th IEEE Asia-Pacific Conference on Communications (APCC), Penang, Malaysia, pp. 745-750, September 2003.

[PIV] T. Hämäläinen, J. Siltanen, A. Viinikainen, and J. Joutsensalo: ″Adaptive Tuning of Scheduling Parameters″, Proceedings of the WSEAS Transactions on Computers, Issue 1, Volume 3, pp. 75-78, January 2004.

[PV] J. Joutsensalo, A. Viinikainen, T. Hämäläinen, and J. Siltanen: ″Bandwidth Allocation Bit Rate and Pricing″, Proceedings of The IEEE International Conference on Communications (ICC), June 2005.

[PVI] J. Joutsensalo, T. Hämäläinen, J. Siltanen, and K. Luostarinen: ″Delay Guarantee and Bandwidth Allocation for Network Services″, Proceedings of conference on Next Generation Internet Networks (NGI), Italy, pp. 129-134, April 2005.

[PVII] J. Joutsensalo, I. Kannisto, J. Siltanen, and T. Hämäläinen: ″Fast Closed Form Approximation for Dynamic Network Resource Allocation″, ACTA Electrotechnica et Informatica, No.1, Vol. 6, pp. 5-10, April 2006.

[PVIII] J. Siltanen, I. Kannisto, J. Joutsensalo, and T. Hämäläinen: ″Bandwidth Broker And Pricing in Multinode Network″, Proceedings of The IASTED International Conference on Networks and Comunication Systems (NCS), March 29-31, Chiang Mai, Thailand, 2006.

[PIX] J. Joutsensalo, T. Hämäläinen, K. Luostarinen, and J. Siltanen: ″Adaptive scheduling method for maximizing revenue in flat pricing scenario″,

AEU International Journal of Electronics and Communications, vol. 60, issue 2, pp. 159-167, February 2006.

CONTRIBUTIONS OF THE AUTHOR IN JOINT PUBLICATIONS

In publications [PI], [PII] and [PIII] the author has designed the test cases. The author also participated in the result analysis. In [PI], [PII] and [PIII] the author has used his experience of the real network management, when designing test cases. In [PI] and [PII] the presented models enable operator to choose such weights for every class that the revenue is maximized. As can be seen from the author's simulation results the proposed model assigns higher weights for more expensive service classes whenever it is possible. At the same time, the QoS requirements are fulfilled.

In [PIII] test scenarios the author includes such QoS parameters as bandwidth and delay for service classes with different traffic characteristics. As an operator increases the total throughput of a router or the maximum amount of flows, which belong to expensive service classes, the adaptive WFQ gives better results when compared to the traditional WFQ.

In publication [PIV] the author proved the Theorem 1 and Theorem 4 and also ran all the simulation scenarios. In Publication [PIV], the author developed the algorithm that is based on the Theorem 4. He realized that although optimal closed form algorithm is difficult to develop, Theorem 4 yields non-optimal but practical and fast closed form rule. The author's main contribution in the publications [PV] and [PVI] was defining the bandwidth allocation part of the used optimal weight calculation algorithm. In [PVII], the author designed and ran the simulation scenarios.

The author invented Bandwidth Broker 1 [PVIII], and he was shown that it can be implemented in the closed form, when the pricing model is of power type, and bit rates of the different channels are approximated to be the same. However, he noted that in the non-power types of pricing models, adaptive algorithms, such as gradient or fixed point algorithms, must be used. In addition, he developed Bandwidth Broker 2 algorithm, which is computationally more complex but in the revenue-optimization sense more optimal than the approximate bandwidth broker 1. He was shown that by the suitable selected iteration formula, bandwidth broker 2 has quite well convergence properties. In addition, he was theoretically shown that the bandwidth broker 2 has the load balancing feature.

In publication [PIX], the first and second author contributed their idea to the algorithm design and all the co-authors participated in the result analyses and further development of the proposed method.

In each publication, all its authors are responsible for the written work.

CONTENTS

1 INTRODUCTION

Today, the Internet provides an access to services that have strict requirements for network performance. Each of these services may be characterized with parameters that specify its packet transmission across a set of nodes in the network. Collectively, these parameters are usually referred to as Quality-of-Service (QoS). To provide Quality-of-Service, several approaches have been proposed by the Internet Engineering Task Force (IETF). The best known are Integrated Services (IntServ) and Differentiated Services (DiffServ). In the IntServ architecture [7] the QoS requirements are provided on the per-flow basis. This architecture implies the presence of the signaling protocol and requires all intermediate nodes keep the state information. The DiffServ architecture, in turn, processes traffic aggregates eliminating the need to exchange with signaling data. Data is forwarded according to the DS codepoint associated with every traffic aggregate and written into the IP header. As a result, this architecture has better scalability and is becoming more popular. The Differentiated Services architecture [6] has become the preferred method to provide QoS in IP networks. This approach, which is based on packet marking, is attractive due to its simplicity and ability to scale. Even though the specifications of DiffServ and IntServ give several examples of use of different solutions, it is a responsibility of the service provider to choose appropriate technologies and their implementations. One of the key points in providing QoS is the implemented queuing policy at a routing node. The set of queuing policies, which are used to guarantee the required QoS, has been considered in number of works [73]. However, if a queuing discipline has input parameters then in most cases static configuration is used. An adaptive approach can more effectively share processing resources providing the required QoS and improving a provider's functioning in a certain manner.

The DiffServ architecture has a high-speed core, which is capable of streamlined data forwarding. Routers in the core of a network are referred to as *interior* or *core* routers. Using the minimal amount of classification actions and simple forwarding mechanisms provides streamlined data transmission. Client networks are connected to the core through a set of nodes that are located at the border. These nodes are referred to as *exterior* or *edge* routers. Depending on the direction of traffic, an exterior node may be referred to as an *ingress* router or as an *egress* router. Unlike core routers, border routers perform more sophisticated data classification and forwarding. The contiguous set of exterior and interior nodes, which operate with a common set of service provisioning policies, form the DiffServ domain. The differentiated services architecture is based on a simple model, in which traffic entering a network is classified and conditioned at the boundaries of the network. At edge routers complex multi-field classifiers are used to select packets from an incoming stream based on the content of some portion of the packet header. According to this information and behavior of the traffic stream, a packet is assigned with the DS codepoint (DSCP), which identifies how data must be treated in the core of a DiffServ network. Based on the DSCP value written in the IP header, distinctive traffic flows are grouped into behavior aggregates. Services are provided for these aggregates, instead of providing them for individual flows. At the core of the network, simple differentiation mechanisms are used to select traffic aggregates according to the DS codepoint only. Externally observable behavior of a core router forwarded data is referred to as the per-hop behavior (PHB), which has one to one correspondence with the DS codepoint. In other words, a packet belonging to a certain PHB is identified by an appropriate codepoint. PHBs are often specified in terms of their observable characteristics, such as bandwidth, delay, loss etc. It is important that PHBs are defined in terms of characteristics relevant to the provisioned service, and not in terms of the particular implementation mechanisms. Thus, a provider may use different technologies and implementations to achieve the required behavior.

Quality of Service in the future

The challenge with the multi-operator environment is to guarantee end-to-end quality of the service, and this requires a lot from the single service operators between the end points. There is a great demand for QoS from companies and different societies, which are globalising rapidly. These groups need reliable connections with trustable partners, because their customers are only interested in the product instead of networked technical solutions. Today, in these kinds of networked transactions the role of security is highly significant and thus increases the importance of QoS. One main component to success is the ability to make standards-based QoS available, preferably via SLAs.

Layer 3 VPN solutions based on MPLS have appeared during the last years. For example, there exist several network operators in Europe, who offer services with this L3 technology. Layer 2 solutions will be the next step in this development. These technologies can be divided into two categories, namely: point-to-point and point-to-multi-point. Point-to-point services are referred to as Pseudo-wire Emulation and point-to-multi-point as Virtual Private LAN Service (VPLS).

MPLS-VPN Point-to-Point Connection

The first and simplest MPLS VPN service was known as Point-to-Point (P2P) connection, which is configured between two sites. Before these implementations solutions recommended by Martini Drafts were used. These originate from the Internet Engineering Task Force (IETF) Pseudowire Edge to Edge Emulation (PWE3) working groups. Pseudowires can be made for totally different services or they can be part of other MPLS VPN techniques.

Virtual Private Lan Service (VPLS)

Virtual private LAN services are the latest evolution steps in MPLS VPN environments. VPLS technology offers a very scalable solution. It is based on point-point service offered by Ethernet Standards for VPLS, and is defined by the IETF Layer 2 Virtual Private Network (L2VPN) working group.

In the VPLS hierarchy every customer has a Customer Edge (CE) device, which is a small router or a switch. This CE device is connected to the operator's PE device, which functions as a higher level VPLS enabled multilayer device. The PE device has the intelligence of the whole VPLS network, and this means that only the PE device should have those VPLS functionalities in it. Normally the operator’s network also has other devices (label switch routers, LSR), which are not able to understand VPLS functionalities, and just forward packets between the PE devices. What is needed is a full mesh of Label Switched Paths (LSPs) between all PE routers. This is a fundamental requirement for any VPLS network.

To properly function with VPLS every PE needs to have a split horizon forwarding property. Using this configuration, all the traffic between two CEs crosses only two PEs, so no alternative routers, and no Spanning Tree Protocol (STP) are needed.

From the point of view of the end user, VPLS network resembles and operates like an Ethernet local area network segment. Customers can send Ethernet packets to the VPLS network. Packets arrive, unchanged, to the network destination, which can be either a local area network in the normal Ethernet network style (VLAN ID) or classified by the CoS field. From the point of view of the service provider, VPLS realizes all the functionalities of the traditional Ethernet network, such as learning of MAC address and broadcasting to the unknown addresses. However, STP is not needed, and MPLS heads are used in parallel with VLAN ID. Each PE router keeps VPLS data of all the customers that are connected. VPLS technology has been resorted to in order to alleviate the Metro Ethernet scalability problem using the efficiency and scalability of MPLS to that effect. For that reason, the processing of an Ethernet packet in the VPLS network is almost identical with what ordinary Ethernet switch does with it.

When an Ethernet frame arrives from the customer’s network to the VPLS network, the PE router performs MAC address checking (the PE device has its own MAC address table for each VPLS domain). If the destination address is found in this table, the PE device routes the frame to the corresponding LSP. If it is not found from the table, it is distributed to all VPLS domains that are in the area of the corresponding VLAN packet. On the other hand, each PE device, that gets the corresponding frame via a broadcast message, sends it directly to the CE device that has been connected to it. When the receiving MAC address answers, the PE device adds that to the address table, and from then on it knows the LSP it belongs to. The PE device keeps its MAC address table in a reasonable size by removing the older addresses first from the table.

H-VPLS

When the use of VPLS increases – due to the increasing interest for getting a full mesh MPLS – the limited resources of PE will become a bottleneck. It must be kept in mind that PE devices handle all unknown unicast, broadcast, and multicast traffic.

For solving the scalability and efficiency problem, the VPLS standard has been enlarged to become the hierarchic VPLS (H-VPLS). In this kind of H-VPLS network, the number of VPLS domains remains the same, but the number of PE devices has been decreased, which decreases the number of LSPs in a full-mesh configuration. This makes the MPLS network more scalable.

H-VPLS brought a new term, "spoke", to a wider use. This refers to what is known as MTU (Multi Tenant Unit) in the standard, because it serves many end-users at the same time. MTU behaves like an edge device for a PE device. This makes it possible to use two-level hierarchy, where there are MTU-PE connections as well as PE-to-PE connections.

MTU is connected only to one PE device (via a predetermined LSP). MTU does not know the topology of the network behind the corresponding PE device, and thus it can function using a simpler configuration than the PE device. With the help of this kind of hierarchy, one can obtain remarkable savings[14].

1.1 Problem Statement

The problem of implementing PHB's has recently gained significant attention. It includes the creation of new scheduling policies and the combination of existent ones. Another direction is the dynamic adaptation of parameters of existent queuing disciplines to the varying network environment.

The choice of the queuing policy is important for the implementation of behavior aggregates in a DiffServ network. The parameters, which a queuing policy makes use of, are important as well. On the one hand, it is possible to provide as much resources for each aggregate as it is necessary when the maximum amount of flows is reached. Such approach requires only minimum management and configuration efforts at edge routers and, in certain cases, no efforts at core routers at all. On the other hand, it is much more efficient and resource conserving to perform dynamic allocation of resources according to the current amount of active flows. Free resources, in turn, can be allocated for those aggregates that improve a service provider's functioning in a certain manner. In this case a DiffServ router would be smarter and implement a certain *adaptive model* that enables them to adapt to the varying network environment. This requires structural changes in a DiffServ router.

In this work we present different resource allocation algorithms, and these adaptive methods work at different traffic scenarios. We have implemented our model to the Network Simulator 2 (NS2), and analyzed how well it can guarantee capacity and delay to different traffic classes. The question, how to put these two issues together still remains. Pricing research has been quite intensively dealt with during the last years, and novel queuing algorithms have been proposed, but these tow aspects have has not been analyzed together widely.

1.2 Related research work

Packet scheduling discipline is an important factor affecting a network node. The choice of the discipline impacts the allocation of restricted network resources among contending flows of the communication network. Network operators can handle resource reservations by using traffic differentiation and by designing different kinds of pricing strategies. We will present summary of the recent pricing work, and after that we will highlight the most commonly used queuing disciplines.

A smart market charging method for network usage is presented in [47]. This paper studies individual packets bids for transport while the network only serves packets with bids above a certain (congestion-dependent) cutoff amount. Charges that increase with either the realized flow rate or with the share of the network consumed by a traffic flow is studied in [34], [37]. Packet-based pricing schemes [18], [42] have also been proposed to act as an incentive for more efficient flow control. The fundamental problem of achieving the a system optimum that maximizes the aggregate utility of the users, using only the information available at the end hosts is studied in [43]. The authors assume that the users consist of elastic traffic and can adjust their rates based on their estimates of network congestion level. Equilibrium properties of bandwidth and buffer allocation schemes are analyzed in [46]. Pricing and link allocation for real-time traffic that requires strict QoS guarantees are studied e.g. in [57], [56]. Such QoS guarantees can often be translated into a preset resource amount that has to be allocated to a call at all links in its route through the network. If the resource is bandwidth, this resource amount can be some sort of an effective bandwidth [35] for a survey of effective bandwidth characterizations and [55] for similar notions in the multiclass case. In this setting, [36], [11] propose the pricing of real-time traffic with QoS requirements, in terms of its effective bandwidth. Their pricing scheme can also be called a static one. It has clear implementation advantages, charges are predictable by the end users, evolve in a slower time-scale than congestion phenomena, and no real-time mechanism is needed to communicate tariffs to the users.

There is also a lot of research, in which the game-theoretic models of routing and flow control in communication networks are applied. The related papers [53], [40], [41], [44], [2], [1] show conditions for the existence and uniqueness of equilibrium. This has allowed, in particular, the design of network management policies that induce efficient equilibrium [40]. This framework has also been extended to the context of repeated games in which cooperation can be enforced by using policies that penalize users who deviate from the equilibrium [44]. A revenue-maximizing pricing scheme for the service provider is presented in [4]. There, a non-cooperative (Nash) flow control game is played by the users (followers) in a Stackelberg game where the goal of the leader is to set a price to maximize revenue.

Two well-known scheduling algorithms are the packet-by-packet generalized processor sharing (PGPS) [54] and the worst case fair weighted fair queuing (WF2Q) [5]. The WF2Q has been proposed to eliminate PGPS burstiness problem exhibited in a flow packet departure process. Based on the fluid traffic model, the generalized processor sharing discipline provides the delay and buffer occupancy bounds for guaranteeing the QoS. The delay bound for the PGPS is provided, e.g., in [54], which is equivalent to the weighted fair queuing (WFQ) [13]. As outlined in [5], the departure process resulting from packet assignment by a PGPS server could be bursty. To avoid this problem, a new packet approximation algorithm of the GPS (i.e., WF2Q) was proposed in [5]. The queuing disciplines such as PGPS and WF2Q are based on a timestamp mechanism to determine the packet service sequence. The timestamp mechanism for all packets, however, entails implementation complexity. If a fixed length packet is used, the implementation complexity due to the timestamp mechanism can be reduced, by employing a round robin (RR) discipline such as the weighted round robin (WRR). Although simple to implement by avoiding the use of timestamp mechanism, the WRR has a larger delay bound. To solve this problem, several modification approaches of the WRR have been proposed. As seen in [50] and [9], the uniform round robin (URR) discipline and the WF2Q interleaved WRR discipline emulate the WF2Q to determine the packet service sequence. These scheduling disciplines result in a more uniform packet departure and a smaller delay bound than those provided by the conventional round robin. An extension to WRR algorithm for fixed length packets is studied in [36] where the authors present a scheduling algorithm for fixed length packets that does not emulate the WF2Q. As the timestamp mechanism is not necessary, the proposed algorithm can be implemented with a low complexity and low processing delay for high-speed networks.

Our research differs from the above studies by linking pricing and queuing issues together. In addition our model does not need, unlike most pricing and game-theoretic ones, any additional information about user behavior, utility functions etc. The work we have done extends our previous pricing and QoS research [27], [26], [23], [28] to take into account queuing scheduling issues by introducing dynamic weight tracking algorithm in the scheduler.

Structure of the thesis

Theoretical basis of the techniques of traffic qualification as well as the effect of the former on the Quality of Service will be discussed in Chapter 2. In the same chapter, the most common techniques to carry out the above processes in modern network systems will be introduced. This thesis is based on nine publications presented in Chapter 3 (theory based) and 4 (simulation results). Chapter 5 contains the summary of the study results of these publications.

2 SCHEDULING METHODS

To provide QoS to streams in a network, the packet scheduling discipline provided by a node (a switch or a router) should accommodate the various bandwidth requirements of incoming flows that share the same outgoing link. Based on the fluid traffic model, the generalized processor sharing (GPS) [54] discipline provides the delay and buffer occupancy bound for guaranteeing QoS. A packet-by-packet version of the algorithm, known as PGPS or Weighted Fair Queuing (WFQ), is defined in terms of the GPS system [54], [13]. A problem with this approach is its computational complexity. Various variants of WFQ have been developed to address this problem. On the other hand, if a fixed length packet is used, a simple round robin discipline such as the weighted round robin (WRR) could be used. Since the WRR has a larger delay bound, to alleviate this, several modification approaches to the WRR have been proposed.

Scheduler algorithms can be divided into two groups, work-conserving and non-work-conserving. There are two types of packet schedulers. When no packets are waiting for being transmitted, we speak of work-conserving schedulers. In the opposite case, we speak of non-work-conserving schedulers. Non-work-conserving schedulers are a practical choice – even an optimal one - when jitter control and/or predictive delay are sought. When the goal is to achieve a minimum average queuing delay, the work-conserving approach is preferred. The delay value is invariant with respect to all work-conserving approaches. Thus the average queuing delay is always the same for different functions enabling selection of the best packet to be transmitted [59].

2.1 FIFO (First In First Out)

FIFO is the traditional scheduling algorithm deployed in the Internet. It is also known as FCFS (First Come First Served). FIFO sends data out in the same order as it receives packets, and it has a very low complexity [59]. Figure 1 presents n flows time division multiplexed (TDM) to one FIFO queue.

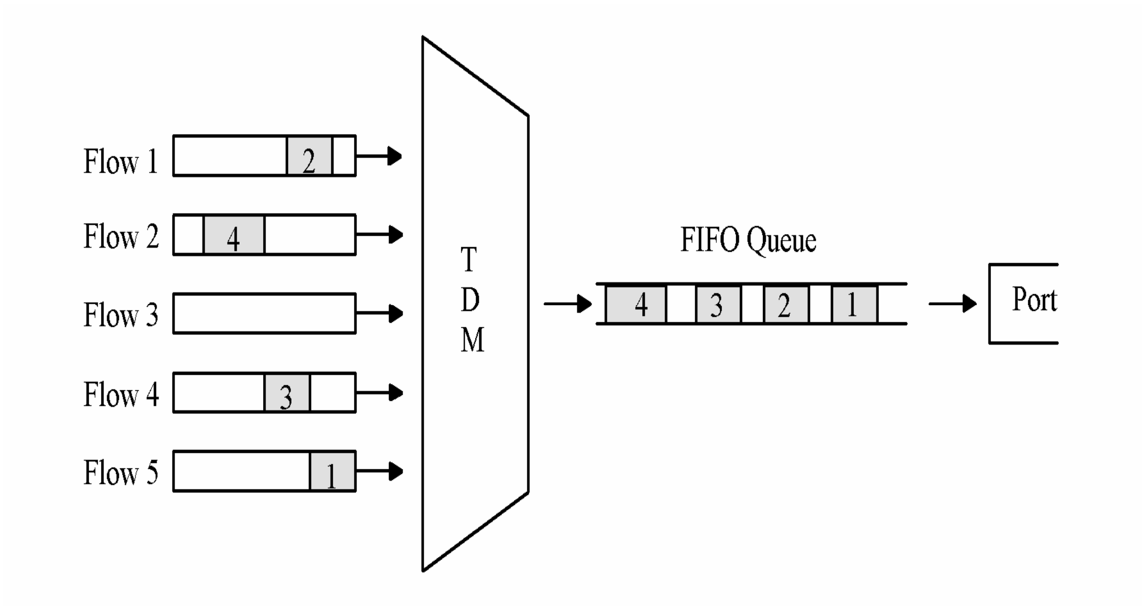


FIGURE 1 FIFO Queuing

The benefit using FIFO queuing is that it is computationally very inexpensive in software-based routers. Maximum delay is determined by maximum depth of the queue. However, routers are not allowed to organize buffering by a single FIFO. All flows are impacted by a single FIFO. The reason for that is that when congestion in the FIFO queue increases, the mean queuing delay for all the flows increases. Therefore, for real time applications, jitter, delay and packet loss can be increased. When congestion occurs, FIFO benefits the UDP flows over TCP flows. When changing network conditions are adapted, TCP applications slow their transmission rate. Then delay and jitter may be increased. FIFO queue buffer space may be consumed by a burst flow. In these cases, before the burst is served, all other flows may be denied [63].

2.2 Priority Queuing

Priority Queuing (PQ) is able to isolate the sessions among different services. Figure 2 shows the functionality of PQ. There are two different queues where traffic is inserted. They are served according to their priority. The highest priority queues are served as soon as possible. In this mechanism, the lowest priority classes may have a very high delay.

In the PQ mechanism, the highest priority class must not get too much bandwidth. Therefore, it requires an admission control mechanism. This decreases the robustness of PQ. In several cases, the service guaranteed by PQ is much better than the service required for the session. Thus PQ can not delay a high priority packet in order to increase the service pattern for the other session. Due to the above reasons, priority queuing is not optimal for a network where best effort and real-time traffic are used at the same time [10] [59].

This work-conserving algorithm performs well when the network has a small amount of high priority traffic. Of this, real-time sessions can serve as an example. The reason is that in this situation the network gives a high quality service to high priority traffic without a notable worsening in the best effort traffic performance. The output link bandwidth that is available is maximal for the highest priority class. The next class has the entire link bandwidth decreased by the amount used by the first class etc. This means, that only the higher class influences the traffic of the lower priority class.

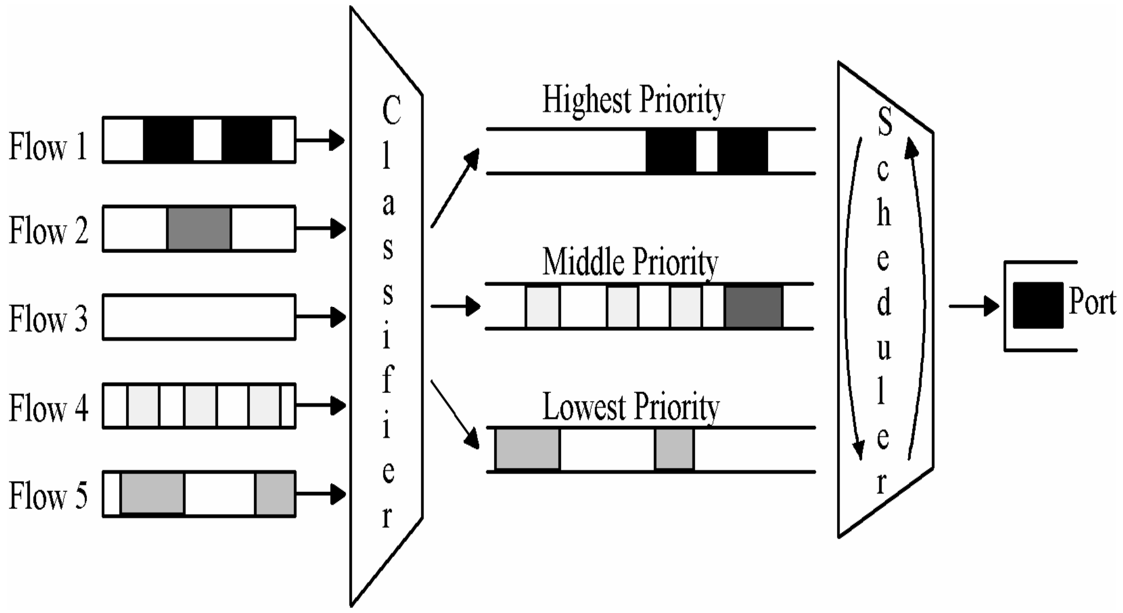


FIGURE 2 Priority Queuing Scheduler

When using priority queuing with the software based routers, quite a small of processing power is needed. Routers will handle different traffic classes independently, and different priority levels can be set for the different kinds of traffic flows (in applications such as VoIP, VoD), as can be seen from Figure 2. Priority queuing has also some limits. One of the biggest problems can arise, when the amount of the high priority traffic is high. In this case the lower priority traffic does not get any service and packet drops may occur. This kind of situation could be avoided by policing the high priority classes e.g. with some intelligent CAC mechanism. Priority Queuing can create a network environment where a reduction in the quality of service delivered to the highest priority service is delayed until the entire network is devoted to processing only the highest priority service class. A misbehaving high priority flow can increase significantly the amount of delay and jitter experienced by other high priority flows sharing the same queue [59].

2.3 Fair scheduling algorithms

Fair Scheduling algorithms are defined fair, because they allow fair sharing of bandwidth among all the users. In [52] the first ideas of the Fair Queuing algorithm were presented. The need for it came from misbehaving applications using TCP. The problem arises, because some applications are capable of using a lot of bandwidth with a cost to the other users. This kind of situation can arise when an application uses a protocol that does not share, unlike TCP, capacity fairly among the all other applications. The goal of FQ is to avoid this situation by guaranteeing to the each application fairly network resources, even thought some rate than the allocated. So PQ forces misbehaving traffic to use only the resources allocated to it [59].

2.3.1 Weighted Fair Queuing

The first known proportional share-scheduling algorithm is Weighted Fair Queuing [13], which emulates the behavior of a General Processor Sharing system using the concept of *virtual time*. Let us assume that $R(t)$ denotes a virtual time, P_i^α transmission time for packet i in queue α, τ_i^α arrival time for packet i in queue α, S_i^α value of $R(t)$ when packet i in queue α begins transmission, F_i^α value of $R(t)$ when packet i in queue α ends transmission. Then the virtual start time S_i^α and virtual finish time F_i^α for packet i in queue α are defined as follows.

$$S_i^\alpha = \max\{R(\tau_i^\alpha), F_{i-1}^\alpha\}, \qquad (1)$$

$$F_i^\alpha = S_i^\alpha + \frac{P_i^\alpha}{w_\alpha}, \qquad (2)$$

where w_α is the assigned weight for queue α based on its required resources. Since P_i^α is not known a priori, it is assumed equal to the maximum value among the packets in queue α. Packets of all queues are scheduled in order of increasing virtual finish time. In the virtual time domain each request will finish before its virtual finish time. In static systems, where all the queues are always active, WFQ provides the shortest delay bound and fairness by bounding the allocation error. The basic functionality of WFQ is presented in Figure 3.

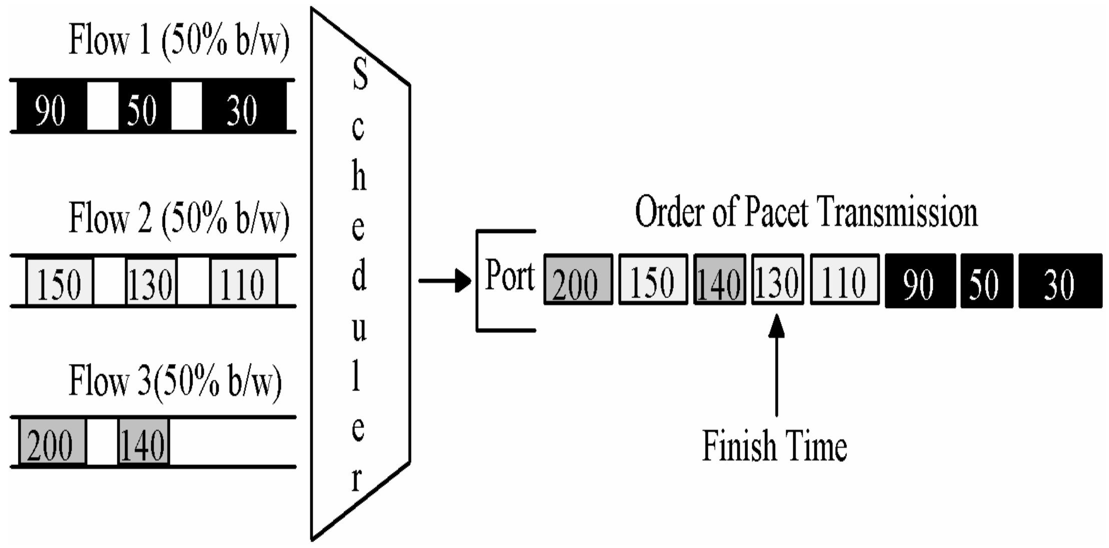


FIGURE 3 WFQ Service According to Packet Finish Time

2.3.2 WFQ Benefits and Limitations

With the use of weighted fair queuing each traffic class will get a minimum level of the capacity of the outgoing link, and this is done independently, i.e., the behavior of the other traffic classes does not effect the reservations of the others. Also, WFQ can guarantee a bounded delay for each traffic class.

Weighted fair queuing has also some drawbacks. Even today WFQ implementations are CPU based, not hardware based. This means that WFQ can not be used with fast network interface cards at the network edges. WFQ uses a complex algorithm that requires maintenance of a significant amount of pre-service class and iterative scans of state on each packet arrival and departure. The scalability problem arises due to this computational complexity when attempting to support a large number of service classes on fast interfaces. On fast network interfaces, minimizing delay to the granularity of a single packet transmission may not be worth the computational expense if one considers the insignificant amount of serialization delay introduced by high-speed links and lower computational requirements of other queue scheduling disciplines. Although the guaranteed delay bounds supported by weighted fair queuing are usually better than for other fair queue scheduling disciplines, the bounds can still be quite large [63].

Different enhancements have been proposed to the basic WFQ. These will be considered next.

2.3.3 Worst-case Fair Weighted Fair Queuing (WF2Q)

To solve the problem that the departure process resulting from packet assignment by a WFQ server could be bursty, a new packet approximation algorithm for GPS (i.e., WF2Q) was proposed in [5]. WF2Q like WFQ uses the virtual time concept. Virtual finish time is the time a packet would take if sent under the GPS discipline. Instead of searching for the smallest virtual finish time for all the packets waiting in the system, WF2Q looks for a packet with the smallest virtual finishing time among packets waiting in the system that have started service under GPS. The service provided by WF2Q is quite close to that of GPS, differing by no more than one maximum size packet. WF2Q+ is a simpler implementation of WF2Q with a relatively low asymptotic complexity of $O(\log N)$.

2.3.4 Self-Clocked Fair Queuing (SCFQ)

In order to reduce the complexity of WFQ/PGPS for updating its virtual time on a packet arrival, an approximate implementation is proposed and analyzed in [19] under the name of Self-Clocked Fair Queuing (SCFQ). The SCFQ algorithm can achieve easier implementation while maintaining the fairness property by introducing a new virtual time function. The complexity in the PGPS scheduler arises from the fact that the scheduler defines fairness in reference to the events in a hypothetical GPS scheduler, which creates the need for simulating events and computing the corresponding virtual time $R(t)$. The SCFQ scheme reduces the complexity by adopting a self-contained approach to the fairness definition. Similar to the WFQ/PGPS scheduler, the SCFQ scheduler is also based on the notion of system's virtual time, viewed as the indicator of work progress in the system, except that the measure of virtual time here is found in the actual queuing system itself, rather than being derived from a hypothetical system. Unlike the extensive computations needed to evaluate $R(t)$ in the PGPS algorithm, the virtual time $\hat{R}(t)$ is simply extracted from the packet situated at the head of queue in the SCFQ algorithm instead. Therefore, the service tag is computed as: $F_i^\alpha = \frac{L_i^\alpha}{\phi_\alpha} + \max\{F_{i-1}^\alpha, \hat{R}(\tau_i^\alpha)\}$. It reduces the complexity of computing the virtual finishing time to $O(1)$. However, a price is paid in terms of the end-to-end delay bounds that grow linearly with the number of sessions sharing the outgoing link [67].

2.3.5 Frame-based Fair Queuing (FFQ) and Starting Potential based Fair Queuing (SPFQ)

As described above so far, an algorithm that would combine the delay and fairness bounds of Weighted Fair Queuing with $O(1)$ timestamp computations remains desirable. Two novel scheduling algorithms are proposed in [66]: Frame-based Fair Queuing and Starting Potential based Fair Queuing, which have *O*(1) complexity for timestamp computations, and provide the same bounds on end-to-end delay and buffer requirements as Weighted Fair Queuing does. The two algorithms may be used in both general packet networks with variable packet sizes and in ATM networks. They are both based on the analytical framework of *rate-proportional servers* (RPS) introduced in [68].

Schedulers in the RPS class use the concept of *potential* to track the state of the system. Each connection is associated with a *connection potential* that keeps track of the amount of normalized service actually received by the connection during the current system-busy period, plus any normalized service it missed during the period when it was not backlogged. The connection potential is a non-decreasing function of time during a system-busy period. The basic system is defined in terms of a fluid model, and the corresponding packet-by-packet server is obtained by computing a timestamp for each arriving packet that represents the value of the connection potential at the instant the last bit of the packet leaves the fluid system, and scheduling the packets in the order of increasing timestamps. The basic objective of a rate-proportional server is to *equalize* the potential of all backlogged connections at each instant. This is achieved in a fluid server as follows. At any instant t, the scheduler services only the subset of connections with the minimum potential, and each connection in this subset receives service in proportion to its reserved rate ρ_i. Thus, the scheduler can be seen to increase the potentials of the connections in this subset at the same rate. At the time that a connection becomes backlogged, its potential is updated based on a *system potential* function that keeps track of the progress of the total work done by the scheduler. The system potential $P(t)$ is a non-decreasing function of time. If $P_i(t)$ denotes the potential of connection i at time t, when an idle session i becomes backlogged at time t, its potential $P_i(t)$ is set as $P_i(t) = \max\{P_i(t-), P(t)\}$, to account for the service it missed. Schedulers use different functions to maintain the system potential, giving rise to widely different delay and fairness behaviors.

The fundamental difficulty in designing a practical rate-proportional server (RPS) resides in the need to maintain the system potential function. Tracking the global state of the system precisely requires simulating the corresponding fluid-model RPS in parallel with the packet-by-packet system. However, the definition of the system potential function allows considerable flexibility in approximating the global state of the system. FFQ and SPFQ both maintain the system potential function only as an approximation of the actual global state in the fluid model, but re-calibrate the system potential periodically to correct any discrepancies. This re-calibration is the key for providing bounded fairness, where fairness is defined as the maximum difference in normalized service received by any two backlogged sessions during any arbitrary interval. Both FFQ and SPFQ are timestamp-based algorithms. However, FFQ uses a framing approach similar to that used in frame-based schedulers to re-calibrate the system potential periodically. This makes the fairness of the algorithm depend on the frame size chosen by the implementation. SPFQ avoids this sensitivity to the frame size by re-calibrating the system potential at the end of transmission of every packet. This gives rise to two algorithms with the same delay bound, but with slightly different fairness properties. In comparison to FFQ, SPFQ requires more state information to be maintained, resulting in a more complex hardware implementation. However, this increased hardware complexity does not affect its asymptotic time-complexity. Thus, SPFQ is more attractive than FFQ in applications where its improved fairness properties justify the additional hardware cost. The main limitation of the two algorithms is that due to their assumption of constant rate servers, they are unfair over variable rate servers.

2.3.6 Weighted Round Robin (WRR)

Packet Round Robin is the basis for the weighted round robin [33]. In WRR different traffic flows (queue) will get a weight, which is usually some percentage of the total bandwidth [59].

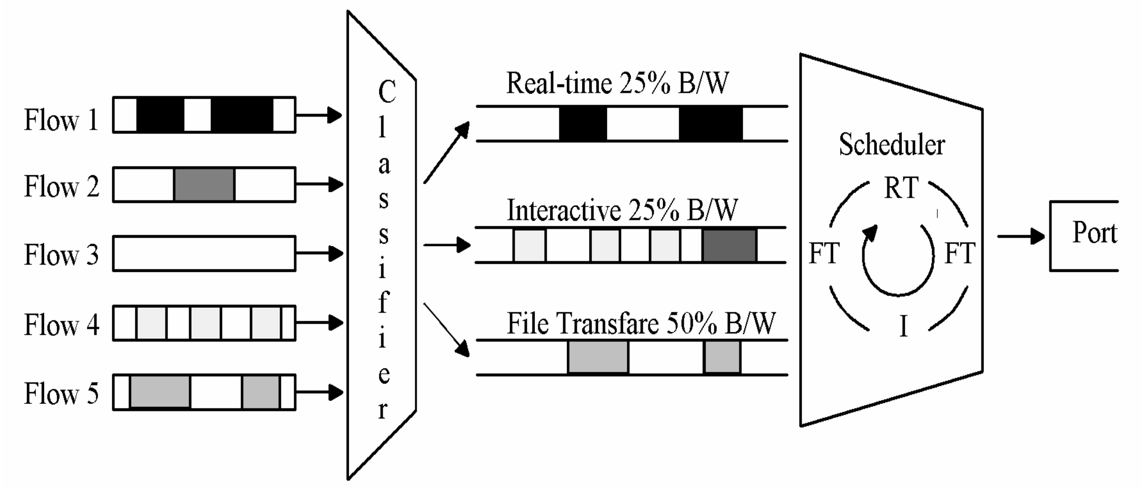


FIGURE 4 WRR Queuing

The basic behavior of WRR can be seen in Figure 4. First, the classifier in the router puts each incoming packet into the queue of the corresponding flow. Next, the scheduler accesses each queue in a round-robin fashion. The maximum number of packets allowed to be transmitted from a queue in a round is specified by the weight for the queue. The weight is typically a predetermined constant integer according to the requested resources. If the packets from different queues have different sizes, a WRR scheduler divides each queue's weight by its mean packet size to obtain a normalized set of weights. However, in practice, a source's packet size may be unpredictable, and so a WRR server may not allocate bandwidth fairly. Although WRR has the advantage of requiring only $O(1)$ processing per packet, it is known that its delay property gets worse as N increases [67], N denoting the number of connections sharing the link. Besides, the WRR scheduling is fair only over time scales longer than a round time. At a shorter time scale, some connections may get more service than others.

2.3.7 Stochastic Fair Queuing

Stochastic Fair Queuing (SFQ) [19] is a modification to Fair Queuing schemes, and tries to override the limitations related to its configuration.

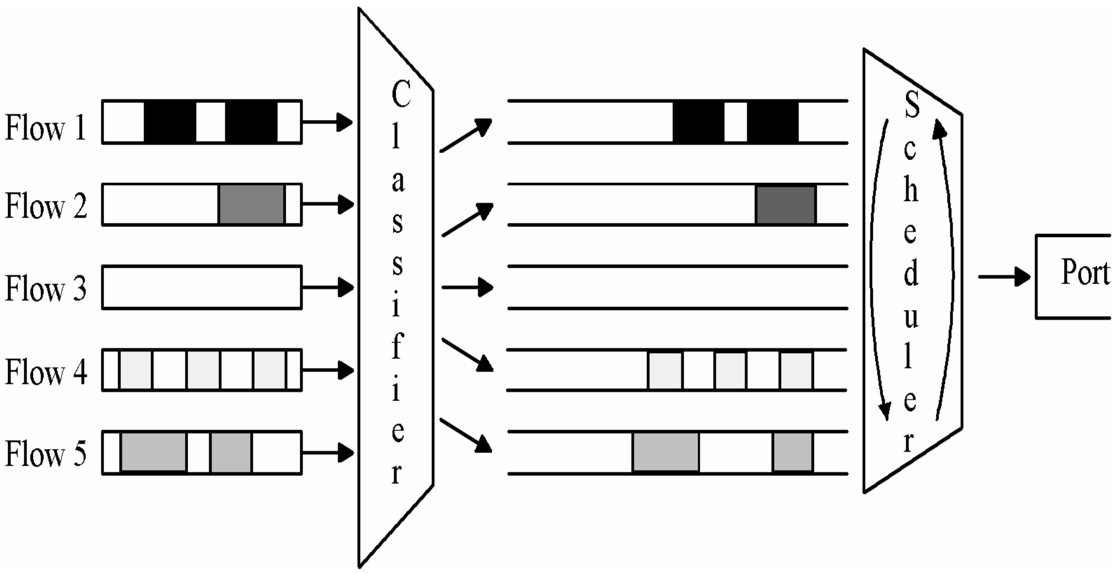


FIGURE 5 SFQ Queuing

Figure 5 shows its "two level" working principle. The packets are mapped using a hash table. For that reason, queue number may be less than the possible flow number. Those flows that are hashed in the same queue are handled in the same way, just as if they were the same flow. The complexity of this procedure is O(1). However, if some flows are colliding with other flows, they are treated unfairly. The name of SFQ becomes from the fact that the fairness guarantees are probabilistic. However, the probability of unfairness is small when the hash index size is larger than the number of active flows [59].

2.3.8 Deficit Round Robin

Because the WRR has its limitations, DWRR was developed. In that approach, the weighted fair distribution of bandwidth is accurately supported when servicing queues that contain variable-length packets. DWRR is allowed to support the output port bandwidth arbitration on high-speed network interfaces. Queues are configured with a number of parameters (see Figure 6).

DWRR uses a Deficit Counter, which tells the total number of bytes the queue is permitted to transmit each time that it is visited by the scheduler. This counter allows a queue, which was not permitted to transmit in the previous round because the packet at the head of the queue was large, that the value of the counter to transmission credits and can be be used during the next service round The Deficit Counter for a queue is incremented by the quantum each time that the queue is visited by the scheduler. A quantum of services is an unit that is proportional to the weight of the queue and is expressed in terms of bytes. If quantum [*i*] = 2*quantum[*x*], then queue *i* will receive twice the bandwidth of queue x when both queues are active [63].

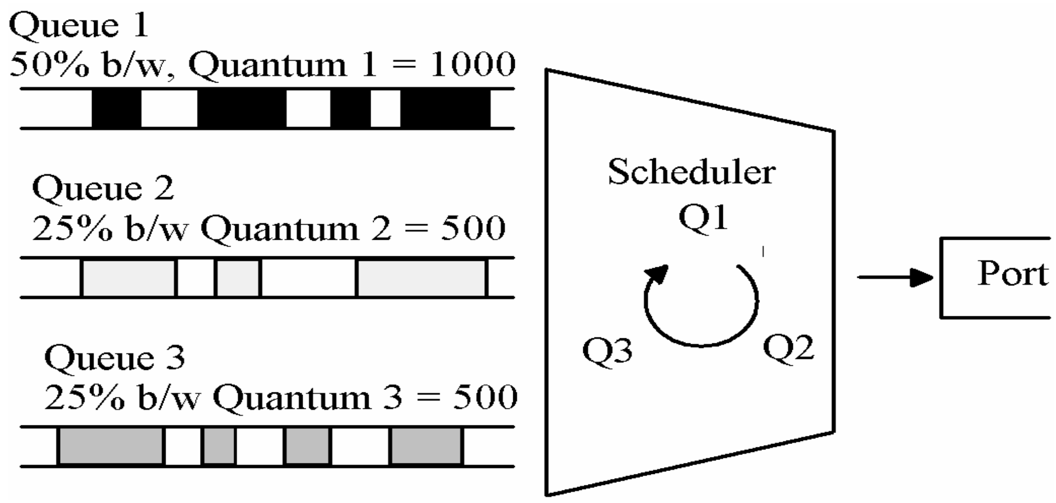


FIGURE 6 DWRR Queuing

The main attraction of DRR is its easy implementation since it requires only $O(1)$ processing work per packet to guarantee bandwidth. However, like WRR, it is unfair at time scales smaller than a round time, and the end-to-end delay bound of DRR significantly increases as the number of connections sharing the link increases.

2.3.9 Uniform Round Robin (URR)

To solve the drawback of WRR, i.e., that the end-to-end delay bound increases with the number of connections sharing the link, [50] proposed two new variants of WRR, *Uniform Round Robin* (URR) and *Idling Uniform Round Robin* (I-URR), for ATM networks, both of which provide end-to-end delay bounds which are independent of the number of connections. In ATM networks, all packets, called *cells*, have a small fixed length of 53 bytes. Below I describe the two variants briefly. The sequence of cells transmitted in a certain connection is referred to as a *flow*. Each flow is stored in a distinct queue, so that each flow is served separately. The WRR server cyclically serves the flows regardless of the pattern of cell arrivals. The cycle is referred to as a *round*, where the flows are served in a fixed sequence that is computed in advance. The round is divided into a fixed number of sub-intervals, called *slots*, each of which corresponds to one cell transmission time L_c / C, where L_c and C denote the size of a cell and the capacity of the server, respectively. The number of slots in a round is referred to as a *round size* denoted by R. Let slot j $(0 \le j \le R-1)$ denote the jth slot in a round, which is assumed to start from slot 0. Each slot is assigned to a flow that has priority to be served at the slot. Let flow f^j denote the flow to which slot j is assigned, then f^j represents the service sequence in a round. The WRR discipline is fully characterized by R and f^j. Let N, ρ_i and w_i be the number of flows sharing the link, the service rate allocated to flow i, and the number of slots assigned to flow i in a round, respectively, where $\sum_{i=1}^{N} w_i = R$ holds. The URR discipline operates as follows. Suppose that a *virtual finishing time* (VFT) kR/w_i is attached to each of w_i cells of flow i, where $k = 1,2,...,w_i$. Then all cells of all flows are sorted in increasing order of their VFT, where the tie breaking rule is used to select a cell belonging to the flow having the smallest index, under the condition that a cell with VFT of kR/w_i must be located after at least $\lceil (k-1)R/w_i \rceil$ cells. Thus, the slot assignment is determined by the resulting sequence of cells, namely, if the jth cell belongs to flow i, slot j is assigned to flow i, where $j = 0,1,..., R-1$. A slot at which the prior flow has no backlog is referred to as an *idle slot*. If an URR server encounters an idle slot, the server immediately skips over the slot. The behavior of an I-URR server is the same as that of URR when there is no idle slot in a round. However when the I-URR server encounters an idle slot, the server does not skip over it but simply becomes idle at the slot. The complexity of URR slightly increases with *N*, while I-URR has *O*(1) complexity with the end-to-end delay bound comparable to URR. Although both algorithms have sufficient fairness properties, their end-to-end delay bounds are rather large compared with WFQ or PGPS.

2.3.10 Priority Queuing Weighted Round Robin (PQWRR)

PQWRR scheduling algorithm for supporting Differentiated Services (DiffServ) [6] was proposed in [49].The IETF DiffServ working group has defined a set of PHBs that include Expedited Forwarding (EF) PHB [32], Assured Forwarding (AF) PHB [21] and Best Effort (BE) PHB. EF PHB defines a virtual leased line service, and requires low delay, low jitter, and assured bandwidth. It is proposed to support voice traffic in a voice data converged network. AF PHB provides a service that guarantees a minimum rate and requires low loss of packets. There are four independent AF classes: AF1, AF2, AF3, and AF4, which are meant to support business oriented data traffic. BE PHB defines the services with no particular requirements. PHBs are implemented at DiffServ network nodes using some scheduling and queuing mechanisms. The Priority Queuing (PQ) and Weighted Round Robin (WRR) scheduling schemes have been evaluated in support of EF PHB [25]. PQWRR is a hybrid scheme that combines the PQ and WRR scheduling mechanism. It assigns EF traffic higher priority over AF and BE traffic, and therefore guarantees the packets from EF traffic queue always get served first and reduces EF traffic's delay and jitter. It uses a WRR scheme among AF traffic and BE traffic queues with queues' weights based on their allocated bandwidth. In case the network is not congested, the unused bandwidth will be used to service oversubscribed queues. Note that PQWRR has also the disadvantage of the PQ scheme, i.e., if the EF traffic queue is always full, the lower-priority queues (AF queues and BE queue) are never serviced.

2.3.11 Weighted Round Robin using Relative Service Share (WRR-RSS)

A new WRR variant (WRR-RSS) was proposed in [38] for fixed length packets. It does not emulate the WF2Q algorithm, which uses a timestamp mechanism to determine the packet service sequence. Assuming that there are N flows in a router sharing the same outgoing link, ρ_i indicates the allocated bandwidth for flow i, $\Re_i$ means the relative service share for flow i, Ω_i is the accumulated relative service share for flow i, R is the service rate at the shared outgoing link and L_c indicates the constant packet length. The WRR-RSS mechanism works as follows.

Step1. Flow numbers (i) are assigned from the flow requiring maximum bandwidth to the flow requiring minimum bandwidth, i.e., flow 1 is for the flow with the maximum assigned bandwidth. The relative service share is defined as: $\Re_i = \rho_i / \sum_{j=i+1}^{N} \rho_j$, $i = 1,2,...,N-1$. The accumulated relative service share is a set of variables, which is initially set: $\Omega_i = \Re_i$, $i = 1,2,...,N-1$.

In step 2, Ω_i is updated to represent the accumulated relative service share depending on whether the slot is assigned to flow i.

Step2. The following algorithm assigns each slot:

```
(line number)
1    do {
2        for ($i$=1; $i$<$N$; $i$++) {
3            if ($\Omega_i$>1) {
4                $\Omega_i = \Omega_i$-1;
5                assign this slot to flow $i$;
6                go to line 11;}
7            else {
8                $\Omega_i = \Omega_i + \Re_i$; }
9        }
10       if ($i$=$N$) assign this slot to flow $N$;
11   } while ($\Omega_i \neq \Re_i$, for some $i = 1,2,...,N-1$)
```

A round is defined by the period in which the packets are transmitted by the service sequence determined by the above algorithm. In each round, this service sequence will be repeated. If Ω_i is restored to $\Re_i$ for flows 1 to N-1, there is a service sequence that will be repeated (in this case, the service sequence is stored to avoid calculating the sequence again). Otherwise, the service sequence in which the packets are transmitted by using step 2 will not be repeated (i.e., one round is infinite). As the timestamp mechanism is not used, WRR-RSS can be implemented with a low complexity and low processing delay for high-speed networks. It solves the disadvantage of WF2Q and URR, i.e., in case that the packets are continuously backlogged for each flow, the delay provided by the WF2Q and URR could be worse than that provided by the GPS, whereas the

WFQ or PGPS would provide a delay less than or equal to that offered by the GPS.

2.3.12 Variably Weighted Round Robin (VWRR)

A new lightweight QoS/CoS (Class of Service) control method called VWRR for use on high-speed backbone networks was proposed a [42]. Since core routers with many interfaces in the backbone network have to process some tens or hundreds of millions of packets per second, they cannot easily provide QoS or CoS using sophisticated scheduling algorithms such as WFQ, which imposes heavy processing loads on the routers because of bit-by-bit packet processing. This makes it preferable to use a simple scheduling algorithm like Weighted Round Robin, which considers only the number of packets. However, it is difficult for such a simple scheduling algorithm to control the exact bandwidth over IP networks because traditional WRR is only suitable for fixed length packets. The proposed VWRR utilizes three modules, a WRR router, packet length observers and a weight calculator. The packet classifiers in the WRR router classify each incoming packet into a flow according to its service class and put it into the corresponding queue. The scheduler in the WRR router accesses each queue in proportion to the weight to allocate requested bandwidth. Each packet length observer periodically measures the average packet length for each flow on its input line, and then reports the average length to the weight calculator. The weight calculator calculates the weight for each queue from the average packet length and the amount of the requested bandwidth, and notifies the WRR router of each weight. Thus, in VWRR, the weight of the WRR router varies adaptively depending on the average packet length, therefore the router can adaptively control bandwidth allocation without drastically increasing the processing load. VWRR can trade the attained fairness for the processing load imposed. In an ideal case where no control delay exists, fairness of VWRR varies from that of WRR to that of DRR [64]. On the other hand, in an actual case where control exists, fairness of VWRR becomes better than WRR if the measurement interval is appropriately determined.

2.3.13 Benefits and Limitations of Fair Queuing algorithms

The primary benefit of FQ is that an extremely bursty or misbehaving flow does not degrade the quality of service delivered to other flows, because each flow is isolated into its own queue. If a flow attempts to consume more than its fair share of bandwidth, then only its queue is affected, so there is no impact on the performance of the other queues on the shared output port. FQ also involves several limitations. Vendor implementations of FQ are implemented in software, not in the hardware. This limits the application of FQ to low-speed interfaces at the edges of the network.

The objective of FQ is to allocate the same amount of bandwidth to each flow over time. FQ is not designed to support a number of flows with different bandwidth requirements. FQ provides equal amounts of bandwidth to each flow only if all of the packets in all of the queues are of the same size. Flows containing mostly large packets get a larger share of output port bandwidth than flows containing predominantly small packets. FQ is sensitive to the order of packet arrivals. If a packet arrives in an empty queue immediately after it was visited by the round-robin scheduler, the packet has to wait in the queue until all of the other queues have been serviced before it can be transmitted.

2.3.14 Class Based Queuing (CBQ)

Figure 7 presents the main blocks for the CBQ. The classifier works in a way similar to packet filtering. It extracts flow information (for an IP flow IPsrc, IPdest, PROTO, PORTsrc, PORTdest) from a packet, then puts the packet into the corresponding class. General Scheduler is a scheduler mechanism that aims to share the bandwidth when all classes are backlogged. It guarantees the right quantity of service to each leaf classes, distributing the bandwidth according to their allocations, using their assigned weights. Link-Sharing Scheduler is a mechanism that aims to distribute the excess bandwidth according to the link sharing structure. Estimator measures the inter-packet time for each class, and estimates whether the class is under limit or over limit. It is the "feedback block" in the system.

CBQ bases its behavior on the interaction between the general scheduler, the link sharing scheduler and the estimator. The general scheduler is rather simple and it can be a generic scheduler mechanism. Current implementations use either the simple Packet Round Robin or the more sophisticated Weighted Round Robin. However the CBQ mechanism does not exclude the use of a more sophisticated general scheduler like WFQ.

Link sharing between agencies that are using the same physical link is provided by Class Based Queuing [15]. In this method, the use of dedicate pipes for each agency can be improved. Each agency can assign its own bandwidth to different kinds of traffic with the CBQ's hierarchical link sharing capabilities. In that case, the unused bandwidth is distributed first to its leaf classes instead of being shared with other agencies.

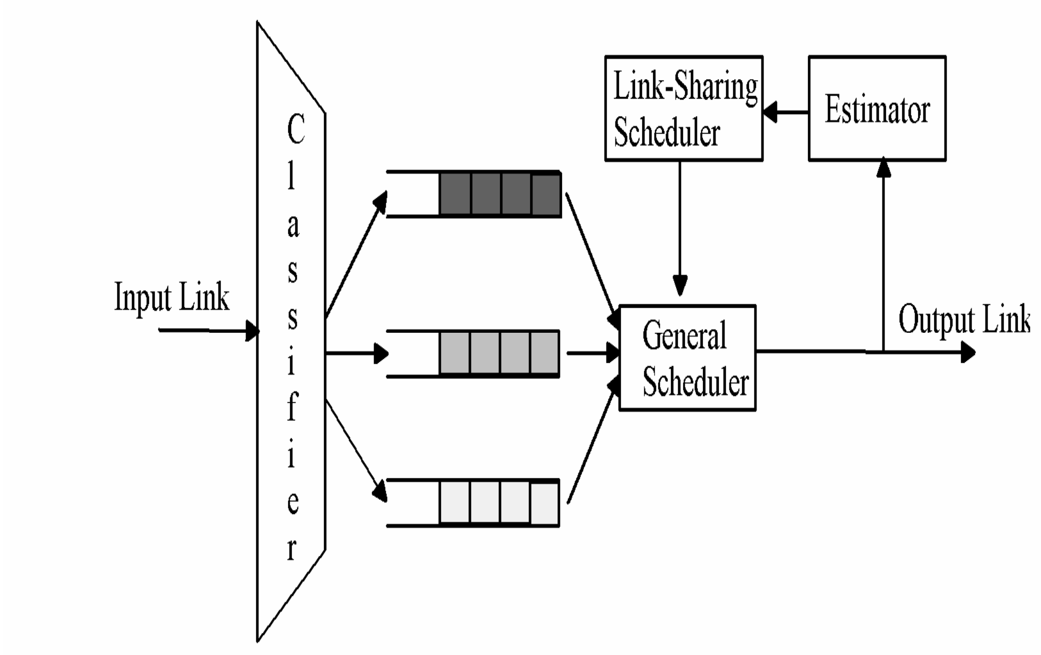


FIGURE 7 CBQ building blocks

3 THEORY AND ANALYSIS OF THE DYNAMIC RESOURSE ALGORITHMS

In this chapter we propose a scheduling model that optimizes the network service provider's revenue, not just in the worst case as in [48], but in a general case. The proposed algorithm ensures more bandwidth for the users paying more for the connection (i.e. higher service class) than those paying less. This work extends pricing and QoS research made in [29], to take into account scheduling issues by introducing fair bandwidth sharing mechanism. Here we propose that a good rate allocation mechanism should not only be fair, but should also allocate the available bandwidth in such a way that the overall utility of the users is maximized. We describe a scheduling mechanism that achieves these goals without requiring knowledge of the users' utility functions and without requiring any explicit feedback from the network. Our model allocates bandwidth by optimizing revenue as a target function.

3.1 Adaptive Tuning of Scheduling Parameters

WFQ based scheduling algorithm is presented in the simplified form. Let d_0 be the minimum processing time of the classifier for transmitting data from one queue to the output in Figure 8. The data packets may have different sizes.

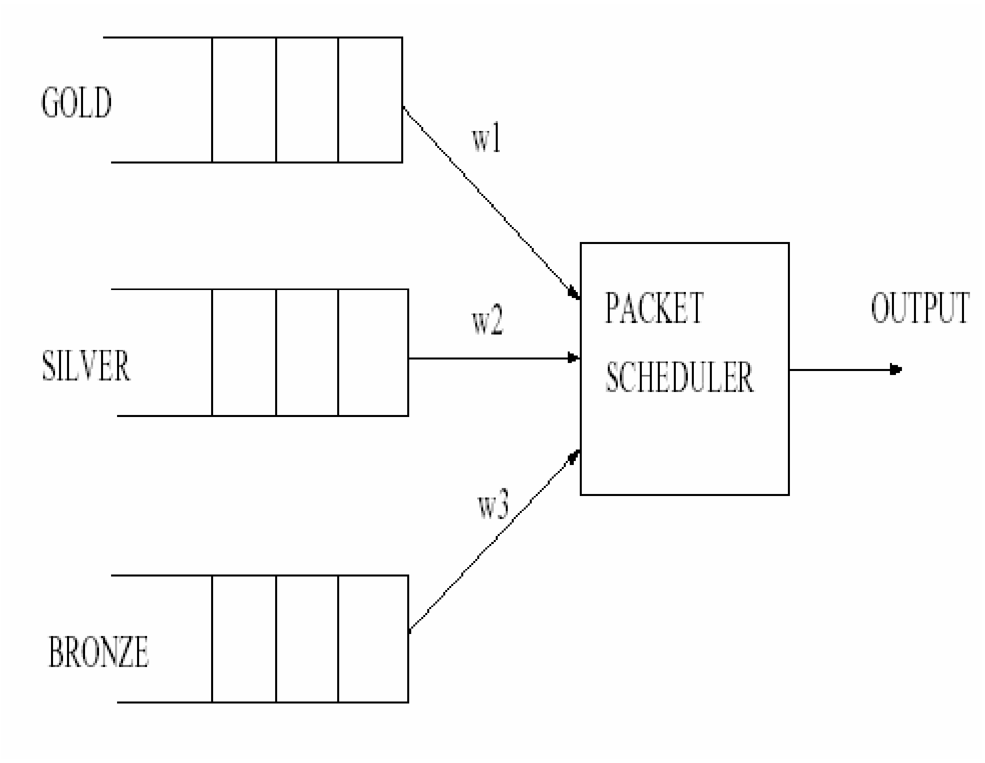


FIGURE 8 Traffic classification at the output buffers.

The number of service classes is denoted by m. In WFQ, the real processing time (delay) is

$$d = N_i E(b_i) d_0 / w_i \tag{3}$$

where $w_i(t) = w_i, i = 1, \ldots, m$, are weights allotted for each class, $N_i(t) = N_i$ is a number of customers in the i th queue, and $E(b_i)$ is the average packet length in the queue i . Here time index t has been dropped for convenience. The constraint for the weights are

$$w_i > 0 \tag{4}$$

$$\text{and } \sum_{i=1}^{m} w_i = 1. \tag{5}$$

If some weight is $w_i = 1$, then the other weights are $w_j = 0, j \neq i,$ and class i is served by time $E(b_i)d_0$, if $N_i = 1$ For each service class, a revenue or pricing function

$$r_i(d) = r_i(N_i E(b_i) d_0 / w_i + c_i) \tag{6}$$

(euros/minute) is decreasing with respect to the delay d. Here $c_i(t)=c_i$ includes insertion delay, transmission delay etc., and here it is assumed to be constant. A goal is to maximize revenue criterion

$$F(w_1,\ldots,w_m)=\sum_{i=1}^{m} N_i r_i (N_i E(b_i) d_0 / w_i + c_i) \tag{7}$$

under the weight constraint (4) and (5). As a special case, consider linear revenue model.
Definition: The function

$$r_i(t)=-r_i t+k_i, i=1,\ldots,m, \tag{8}$$

$$r_i>0, \tag{9}$$

$$k_i>0, \tag{10}$$

is called linear pricing function.

Theorem 1: Consider the linear pricing function (8) and the corresponding revenue function

$$F=F(w_1,\ldots,w_m,N_1,\ldots,N_m)=\sum_{i=1}^{m} N_i(-r_i \frac{N_i E(b_i)}{w_i}+k_i), \tag{11}$$

where $d_0=1$ and $c_i=0$ for convenience. Then upper bounds for buffer sizes are

$$q_i=\left\lfloor \frac{1}{2}\frac{k_i}{r_i E(b_i)} \right\rfloor, i=1,\ldots,m, \tag{12}$$

where $y=\lfloor x \rfloor$ denotes maximum integer y satisfying $y \le x$.
Proof: The optimal number of users for fixed weights is obtained as follows:

$$\frac{\partial F}{\partial N_l}=-2\frac{r_l}{w_l}N_l E(b_l)+k_l=0. \tag{13}$$

Therefore:

$$N_l=\frac{1}{2}\frac{w_l k_l}{r_l E(b_l)}, l=1,\ldots,m. \tag{14}$$

The second derivate is

$$\frac{\partial^2 F}{\partial N_l^2}=-2\frac{r_l E(b_l)}{w_l}<0, \tag{15}$$

because $r_l > 0$ and $w_l \geq 0$. Therefore F is strictly concave with respect to N_i, $i = 1, \ldots, m$, having one and only one global maximum, which is satisfied by Eq. (14). Because $w_i \leq 1$, $i = 1, \ldots, m$, then

$$N_l \leq \frac{1}{2} \frac{k_l}{r_l E(b_l)}, \tag{16}$$

for which Eq. (12) follows. This completes proof Q.E.D.
The solution (14) is plausible and easy to interpret:

- When w_l is large, then it gives large weights to those buffers, where the number of customers is large to prevent too large delay to those numbers.
- Positive k_l increases revenue. It is simply positive constant vertical shift. Thus, the larger k_l is, the larger the number of customers bringing large revenue.
- Negative $-r_i$ in Eq. (11) has an opposite effect from k_i. Thus, the number of customers is inversely proportional to r_i. The coefficient is a kind of a penalty term.
- When the average packet size $E(b_i)$ is large, the number of packets should be small.

Upper bound for revenue is stated as follows:
Theorem 2: In the case of linear pricing model (8), the upper bound for revenue is

$$F \leq \frac{1}{4} \sum_{i=1}^{m} \frac{k_i^2}{r_i E(b_i)}. \tag{17}$$

Proof: Select optimal value for N_i in Eq. (14), and substitute it in Eq. (11). Then

$$F = \sum_{i=1}^{m} \frac{1}{2} \frac{w_i k_i}{r_i E(b_i)} \left(-r_i \frac{1}{2} \frac{w_i k_i E(b_i)}{r_i w_i E(b_i)} + k_i \right) = \frac{1}{4} \sum \frac{w_i k_i^2}{r_i E(b_i)}. \tag{18}$$

Due to the condition $w_i \leq 1$, Eq. (17) follows. Q.E.D.
Interpretation of (17) is quite obvious: k_i increases upper limit, while r_i decreases it. As a special case, when all buffers are full according to the rule (12), we get the following result:
Theorem 3: When

$$N_i = \frac{1}{2} \frac{k_i^2}{r_i E(b_i)}, \tag{19}$$

revenue is

$$F = \frac{1}{2}\sum_{i=1}^{m} \frac{k_i}{r_i E(b_i)}\left(1 - \frac{m}{2}\right) \tag{20}$$

The proof is omitted. It is clear that in practice the buffer sizes must be selected smaller than in Eq. (12). As a special case, when there is only one class, i.e. $m = 1$, the upper bound (17) can be achieved, but not for the other values of m, if the buffers are full. The following theorem gives a sufficient condition for achieving non-negative revenue as well as an other upper bound for revenue:
Theorem 4: if weights are selected by

$$w_i = \frac{N_i E(b_i) r_i / k_i}{\sum_{l=1}^{m} N_l E(b_l) r_l / k_l}, \tag{21}$$

and a constraint

$$\sum_{i=1}^{m} \frac{N_i E(b_i) r_i}{k_i} < 1, \tag{22}$$

is used in the call admission control mechanism, then

$$0 \le F \le \sum_{i=1}^{m} N_i k_i \ . \tag{23}$$

Proof: Define

$$a = \sum_{l=1}^{m} N_l E(b_l) r_l / k_l. \tag{24}$$

Revenue is

$$\begin{aligned} F &= \sum_{i=1}^{m}\left(- r_i N_i^2 E(b_i) \frac{k_i a}{N_i E(b_i) r_i} + N_i k_i \right) \\ &= \sum_{i=1}^{m} \left(- N_i k_i a + N_i k_i \right) \\ &= \sum_{i=1}^{m} N_i k_i (1 - a) \\ &= \sum_{i=1}^{m} N_i k_i \left(1 - \sum_{l=1}^{m} \frac{N_l E(b_l) r_l}{k_l} \right) \ge 0, \end{aligned} \tag{25}$$

when constraint (22) is satisfied. Because $N_i \ge 0, r_i > 0, k_i > 0,$ then $0 \le a < 1$. Then it follows that

$$F = \sum_{i=1}^{m} N_i k_i (1-a) < \sum_{i=1}^{m} N_i k_i . \qquad (26)$$

This completes the proof Q.E.D.

Call Admission Control (CAC) mechanism can be made by simple hypothesis testing without assumption about call or dropping rates. Let the state (number of packets) at the moment t be $N_i(t),\ t = 1, \ldots, m$. Let the new hypothetical state at the moment $t+1$ be $N_i(t+1),\ i = 1, \ldots, m,$ when one or several calls appear. In hypothesis testing, hypothetical revenues $F(t)$ and $\tilde{F}(t)$ are calculated. If $F(t) > \tilde{F}(t),$ then call is rejected, otherwise it is accepted.

SIMULATIONS AND OBTAINED RESULTS

In the experiment, calls and durations are Poisson and exponentially distributed, respectively. In addition, the number of classes is $m = 3$. Data packets have lengths 1, 2, and 5 kbytes with equal probability. Call rates per unit time for gold, silver, and bronze classes are $\alpha_1 = 0.1, \alpha_2 = 0.2,$ and $\alpha_3 = 0.3,$ respectively. Duration parameters (decay rates) are $\beta_1 = 0.010, \beta_2 = 0.007,$ and $\beta_3 = 0.003,$ where probability density functions for durations are

$$f_i(t) = \beta_i e^{-\beta_i t}, \quad i = 1,2,3, \quad t \ge 0 . \qquad (27)$$

The number of unit times in the experiment was $T = 3000.$ Three service classes have the pricing functions

$$r_i(t) = -5t + 200 \qquad (28)$$

for gold class,

$$r_2(t) = -2t + 100 \qquad (29)$$

for silver class, and

$$r_3(t) = -0.5t + 50 \qquad (30)$$

for bronze class. Figures 9, 10, and 11 show the simulation results. In Figure 9, three delay profiles are represented, while in Figure 10, the number of users is shown. Most importantly, Figure 11 shows that the revenue is always clearly positive, justifying Theorem 4.

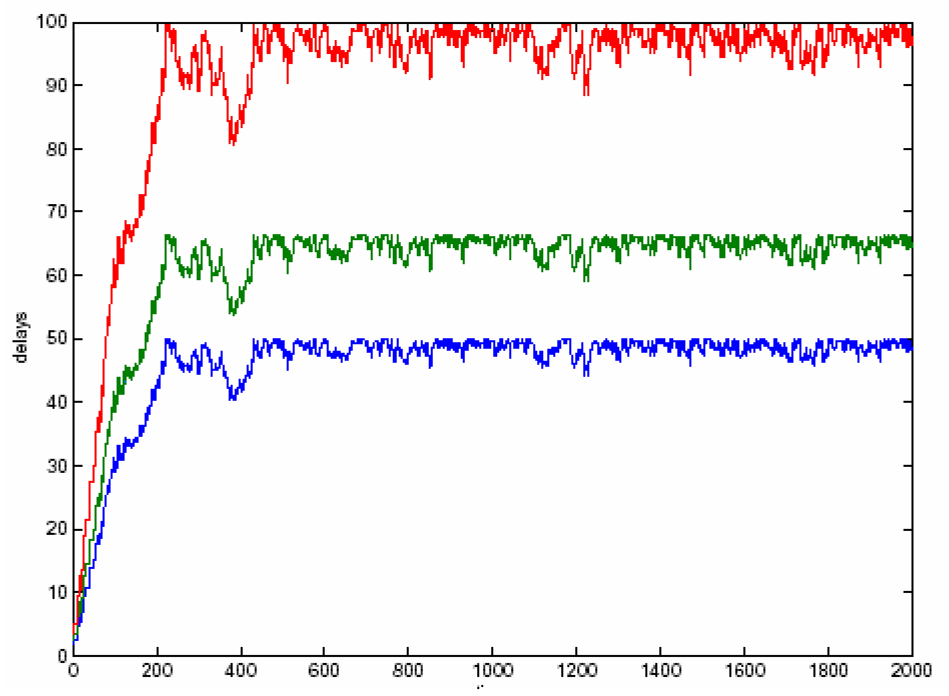


FIGURE 9 Delay as a function of time.

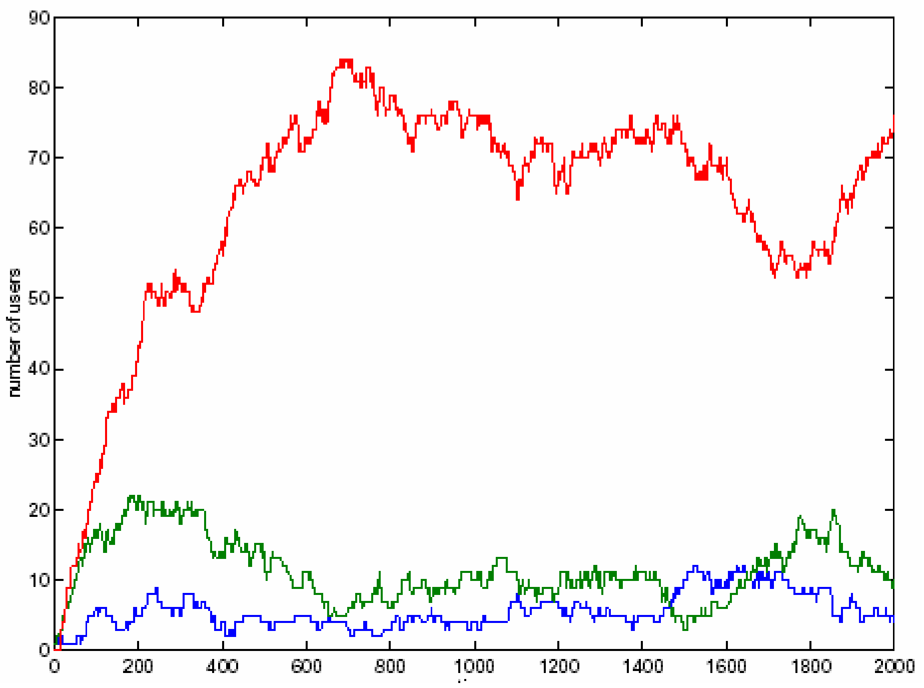


FIGURE 10 Number of the users as a function of time.

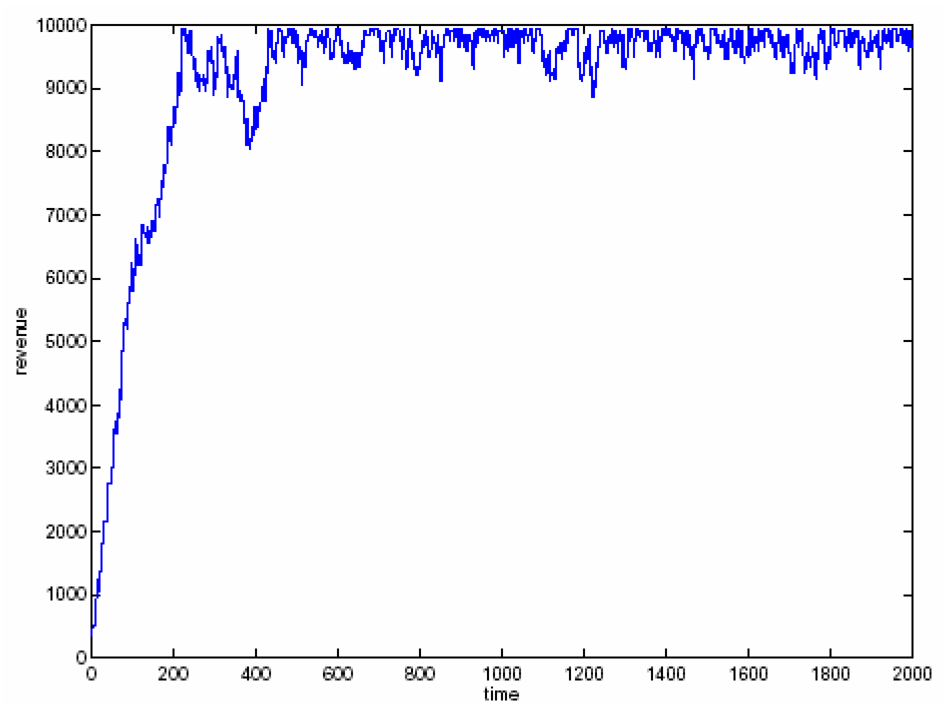


FIGURE 11 Obtained revenue as a function of time.

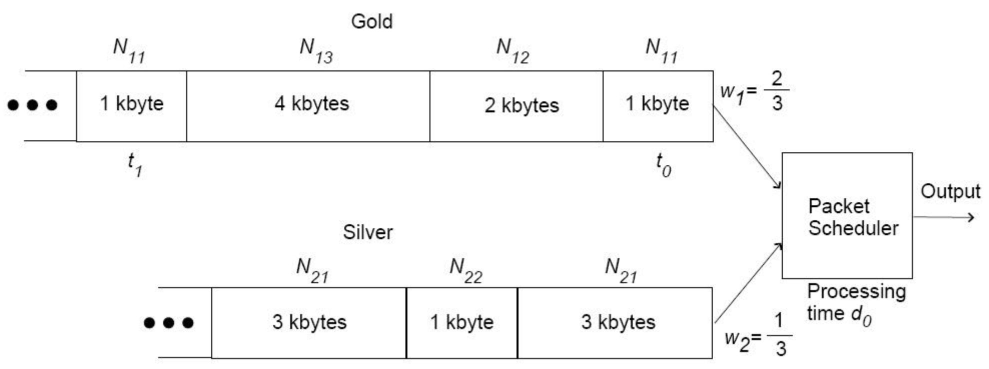
In this case, we introduced an adaptive WFQ algorithm, which dynamically adjusts weights in such a manner that the QoS requirements of the different traffic classes can be met and the network operator's revenue can be kept as high as possible. The experiments demonstrated this properly, while still allocating delays in a fair way. Here we investigated a linear pricing scenario, also the piecewise linear model, e.g. flat pricing model can be investigated.

In the near future, new connection admission control and queuing techniques (i.e. dynamic dropping) issues will be integrated to our model. Our goal is also to study the possibility of implementation of the proposed model under the switch (e.g. linux based).

3.2 Fast Close Form Approximation for Dynamic Network Resource Allocation

In this chapter we extend our research by taking into account queuing scheduling issues and introducing dynamic weight tracking algorithm in the scheduler. The QoS and revenue aware scheduling algorithm is investigated. It is derived from optimization problem, that resembles Lagrangian constrained approach, and an approximate optimal closed form solution is presented when QoS parameters are delay and bandwidth.

Next we formulate expressions for delays (seconds) and bandwidth (bit rate) of the data traffic. Consider the packet scheduler for two service classes. There are now two service classes. Gold class customers pay most of money and get the best service while silver class customers pay less money. Bronze class customers pay the least and get the worst service.



Parameter Δt_i denotes time which passes when data is transferred through the queue i to the output in the switch, when $w_i = 1$. If the queue is almost empty, the delay is small, and when the buffer is full, it is large. Variable w_i is the weight allocated for class i. The constraint for weights w_i is

$$\sum_{i=1}^{m} w_i = 1, \quad w_i > 0. \tag{31}$$

Variables w_i give weights according to how long time queues i are served per total time. Therefore, delay d_i in the queue i is actually

$$d_i = \frac{\Delta t_i}{w_i}. \tag{32}$$

Without loss of generality, only non-empty queues are considered, and therefore

$$w_i \neq 0, \quad i = 1, \ldots, m, \tag{33}$$

where m is the number of service classes. When one queue becomes empty, $m -> m - 1$.

Bandwidth or bit rate is formulated as follows. Let the processing time of the data be T [seconds/bit] in the packet scheduler. There are N_i connections or packets in the class i. Let us denote the packet size b_{ij} [bits] or [kbytes] in the class i = 1, . . . ,m and the connection j = 1, . . . ,N_i. It is easy to see that bandwidth of the packet (i, j) is

- linearly proportional to the packet size b_{ij} ,
- linearly proportional to the weight w_i,
- inversely proportional to the processing time T, and
- inversely proportional to the total sum of the packet lengths b_{ij} , $j = 1, \ldots ,N_i$, because other packets occupy the same band in a time-divided manner.

Therefore, the expression for the bandwidth is

$$B_{ij}[bits/s] = \frac{b_{ij} w_i}{N_i E(b_i) T} = \frac{b_{ij} w_i}{N_i E(b_i)} = \frac{b_{ij} w_i}{\sum_{l=1}^{N_i} b_{il}} \quad , \tag{34}$$

where the processing time T can be scaled $T = 1$, without loss of generality.

Here

$$E(b_i) = \frac{1}{N_i}\sum_{j=1}^{N_i} b_{ij} \tag{35}$$

is mean packet length in the class *i*.

3.3 Pricing Models and Revenue Maximization

We concentrate on the pricing and fair resource allocation from the point of view of the customers. On the other hand, we try to maximize revenue from the point of view of the service provider. First, we introduce the concept of *pricing functions*. Within the scope of our study, there are two QoS parameters, namely delay and bandwidth. Therefore, two separate pricing functions are defined.

A. General pricing function

Let the general pricing function be $f = f(w_1, \ldots, w_m)$. That means, f depends on the QoS parameters - in our study, delay and bandwidth - while QoS parameters depend on the weights w_i of the scheduler. Let the constraint for the weights be

$$\sum_{i=1}^{m} w_i = 1, \quad w_i > 0. \tag{36}$$

Revenue has the Lagrangian form

$$R = f + \lambda\left(1 - \sum_i w_i\right). \tag{37}$$

Derivative with respect to the weights is

$$\frac{\partial R}{\partial w_i} = \frac{\partial f}{\partial w_i} - \lambda = 0. \tag{38}$$

Then, - because $\sum_i w_i = 1$:

$$\lambda = \lambda\sum_i w_i = \sum_i \lambda w_i = \sum_i \frac{\partial f}{\partial w_i} w_i. \tag{39}$$

Thus derivative of the revenue is

$$\frac{\partial R}{\partial w_i} = \frac{\partial f}{\partial w_i} - \sum_i \frac{\partial f}{\partial w_i} w_i. \tag{40}$$

Let us check the correctness of the derivative by direct substitution. Substitute Eq. (39) into Eq. (37).

Then we obtain

$$R = f + \sum_k \frac{\partial f}{\partial w_k} w_k \left(1 - \sum_l w_l\right), \tag{41}$$

and remember the constraint (36). Then we obtain

$$\frac{\partial R}{\partial w_i} = \frac{\partial f}{\partial w_i} + \sum_k \frac{\partial^2 f}{\partial w_i \partial w_k} w_k + \frac{\partial f}{\partial w_i}$$

$$-\sum_k \frac{\partial^2 f}{\partial w_i \partial w_k} w_k \sum_l w_l - \frac{\partial f}{\partial w_i} \sum_l w_l - \sum_k \frac{\partial f}{\partial w_k} w_k$$
$$= \frac{\partial f}{\partial w_i} - \sum_k \frac{\partial f}{\partial w_k} w_k \, . \tag{42}$$

But this is the same as (40) in the space $\{\sum_i w_i = 1\}$.

B. Pricing models for delay and bandwidth

The drawback in the previous scenario is that the algorithm developed from the gradient of the revenue is too complicated for fast implementation. Here we present a modified revenue criterion. Let the revenue be presented in the constrained form

$$R = f + \sum_k w_k \left(1 - \sum_l w_l\right) . \tag{43}$$

Then

$$\frac{\partial R}{\partial w_i} = \frac{\partial f}{\partial w_i} + 1 - \sum_l w_l - \sum_k w_k = \frac{\partial f}{\partial w_i} - 1 = 0 \tag{44}$$

under the constraint $\sum_i w_i = 1$. It is seen from the criterion that the sum of the weights acts as the penalty term. If we consider a linear pricing scenario, where revenue is decreasing as a function of the delay, and increasing as a function of the bandwidth, we get the pricing functions

$$f_i(d) = -\sum_i r_i d_i , \tag{45}$$

where r_i is the penalty factor for the delay in the class i in the pricing function, and d is the delay for the scheduler. On the other hand,

$$f_2(B) = \sum_i e_i B \tag{46}$$

is the pricing function for bandwidth B, where e_i is the pricing factor for the service class i. Total revenue can be expressed in the form

$$R = f_i + f_2 + \sum_k w_k \left(1 - \sum_l w_l\right) \tag{47}$$

For all the connections, the pricing function can be presented in the form

$$f = \sum_{i=1}^{m} \frac{N_i r_i \Delta t_i}{w_i} + \sum_{i=1}^{m} e_i w_i \,, \tag{48}$$

$$\frac{\partial f}{\partial w_i} - 1 = \frac{N_i r_i \Delta t_i}{w_i^2} + e_i - 1 = 0\,. \tag{49}$$

Then the closed form approximation for the weights is as follows:

$$w_i = \frac{\sqrt{\dfrac{N_i r_i \Delta t_i}{1-e_i}}}{\sum_k \sqrt{\dfrac{N_k r_k \Delta t_k}{1-e_k}}}\,. \tag{50}$$

Discussion

Here we discuss the results and conclude the work:

- We considered delay guarantee and bandwidth allocation of communications network.
- Pricing scheme was linear for both QoS parameters.
- We developed a novel constrained optimization approach which resembles Lagrangian approach.
- The approach yielded fast closed form approximation for optimizing the revenue of the service provider.
- The algorithm gives a fair resource allocation.

Our general conclusion is that this approach makes it possible for everyone, including the people of modest means, to use communication services due to facilitation by different pricing classes

3.4 Bandwidth Allocation, Bit Rate and Pricing

Publication [PV] studies a closed form formula for updating the adaptive weights of a packet scheduler that is derived from a revenue-based optimization problem. The weight updating procedure is fast and independent of the assumption of the connections' statistical behavior. The features of the algorithm are simulated and analyzed with a Call Admission Control mechanism. We also show, in context of the CAC procedure a mechanism for guaranteeing a specified mean bandwidth for different service classes.

The following conclusions are drawn from theory and experiments:

- In the polynomial pricing scenario, we have derived analytic forms of the revenue and the weights that allocate traffic to the connections of different service classes.
- The updating procedure is deterministic and nonparametric i.e. it does not make any assumptions of the statistical behavior of the traffic and connections. Thus it is robust against the errors that may occur from erroneous models.

- The algorithm is unique and optimal, which has been theoretically proved by the Lagrangian optimization method.
- A closed form solution makes algorithm quite simple.
- Minimum bandwidth can be guaranteed in call admission mechanism by adding a specific constraint to the updating rule. In the experiments, bandwidths always stay above the minimum limits.
- When the gain pricing factors *ri* are high, the corresponding connections obtain more bandwidth.
- Because all gain factors *ri* are positive, all classes obtain service in fair way.
- The adaptive weight algorithm outperforms the fixed weight algorithm in the sense that it gives larger revenue.

3.5 Delay Guarantee and Bandwidth Allocation for Network Service

Publication [PVII] explores the efficiency of the gradient and fixed-point type algorithms for updating the weights of a packet scheduler derived from a revenue-based optimization problem. In the linear pricing scenario, algorithms are simple to implement. We compared algorithms with optimal brute-force method. Especially the fixed-point algorithm converges very fast to the optimal solution, typically in one iteration and in about 40 operations, when the number of classes is three. The weight updating procedures are independent of the assumption of the connections' statistical behavior, and therefore they are robust against erroneous estimates of statistics. Also, a Call Admission Control is implemented in the context of our scenario.

We can draw the following conclusions, confirmed by our algorithms and experiments:

- In the pricing scenario, we have derived an analytic form to the revenue and gradient as well as fixed-point algorithms for updating the weights *wi*, which allocate data traffic to the connections of different service classes.
- The updating procedure is deterministic and nonparametric, i.e., it does not make any assumptions of the statistical behavior of the traffic and connections. Thus it is robust against the errors that may occur due to wrong models.

3.6 Bandwidth Broker and Pricing in Multinode Network

In this chapter, we present the main result of the publication [PIX]. We used bandwidth broker algorithm for maximizing the revenue in the multinode network scheduler. The proposed algorithm ensures more bandwidth for the users paying more for the connection (i.e. higher service class) than for those paying less. The algorithm is fast to realize, and thus practical to implement. It converges typically in a few iterations for a given number of connections, and it is robust against erroneous estimates of customers' behavior.

We draw the following conclusions from our algorithms and experiments: In the pricing scenario, we have derived an analytic form to the revenue and gradient as well as fixed-point algorithms for updating the weights wi, which allocate data traffic to the connections of different service classes. The updating procedure is deterministic and nonparametric, i.e., it does not make any assumptions of the statistical behavior of the traffic and connections. Thus it is robust against errors that may derive from the wrong models. The algorithms are unique, which have been proved by the Lagrangian optimization method. Because all the gain factors are positive, all the classes obtain service in a fair way.

3.7 Adaptive Scheduling Method for Maximizing Revenue in Flat Pricing Scenario

Publication [PVI] presents an adaptive scheduling algorithm for traffic allocation. We use a flat pricing scenario in our model, and the weights of the queues are updated using revenue as a target function. Due to the closed form nature of the algorithm, it can operate in non-stationary environments. In addition, it is nonparametric and deterministic in the sense that no assumptions about connection density functions or duration distributions are made.

Next, we present a summary of our approach as well as of our experiments.

- The proposed weight-updating algorithm is computationally inexpensive within the scope of our study, when weights are updated and CAC is performed in the connection level.

- Experiments clearly justify the performance of the algorithm. For example, revenue curves are positive, and the maximum delays are guaranteed using the weight constraint.

- Some of the statistical and deterministic algorithms presented in the literature assume quite strict *a priori* information about parameters or statistical behavior such as call densities, duration or distributions. However, such methods usually are - in addition to being computationally complex - not robust against erroneous assumptions or estimates. Our algorithm, on the other hand, is deterministic and nonparametric, i.e., it uses only the information about the number of connections. We believe that in practical environments it is a competitive candidate due to the robustness.

Our general conclusion is that the flat pricing scenario is reasonably simple, in a way tempting and practical as well.

4 PRACTICAL EXPERIMENTS WITH DYNAMIC RECOURCE ALGORITHMS

This chapter presents our adaptive scheduling method and some simulation scenarios. The issues presented here have been published, in more detail, in the papers: “An Adaptive Approach to WFQ with the Revenue Criterion” [60], “The simulation and analysis of the revenue criterion based adaptive WFQ” [61], “On providing bandwidth and delay guarantees using the revenue criterion based adaptive WFQ” [62], and “Adaptive Tuning of Scheduling Algorithms" [24].

In these papers we extend our previous pricing and QoS research, in which the optimal link allocation between traffic classes using different pricing scenarios [23], [27] and the revenue maximization model [26] have been discussed. The possibility of using revenue as the criterion for updating weights in the WFQ service discipline for the single-node case was theoretically considered in [60], and the simulation for more complex network environment was presented in [61]. However, the only QoS parameter taken into account was the throughput. Since many network services are critical to such QoS parameters as queuing delay and end-to-end delay, it became obvious that the extension of the proposed model is necessary.

Each service can be characterized with certain requirements for packet transmission in one direction across a set of one or more paths within a network [6]. These characteristics may be specified in quantitative or statistical terms of throughput, delay, jitter, and/or loss. On the one hand, services are differentiated to accommodate heterogeneous application requirements and user expectations.

On the other hand, it is always possible to find a set of Internet services that have common characteristics. As a result, services with the same set of QoS parameters can be grouped into a service class. In this framework, a provider has to share limited resources between service classes to provide the required QoS for end-users. Thus, each class should be assigned with a minimum percentage of the link bandwidth while allowing ″unused″ bandwidth to be available at other times. This sharing model must be able to provide performance guarantees for various data and especially for real-time sessions such as voice and video. At the same time, a provider is interested in maximizing the revenue while serving users. Since each class has its own QoS requirements the pricing of service classes can be differentiated as well. It is natural that a QoS class with higher requirements would cost more than lower QoS classes. Thus, the problem is how to distribute a set of limited resources between service classes with different prices in the most efficient way.

The resource allocation can be handled by adjusting parameters of a service policy implemented by the router. Unfortunately, there is a problem of adapting parameters of a service policy to a varying network environment because it is not obvious which criterion should be chosen. The problem of the dynamic adaptation has been targeted in later research and different criteria have been proposed. For instance, the dynamic adaptation of weights in the Weighted Round Robin policy according to the mean packet size has been proposed by [25]. Other investigations have considered the possibility of using the state of queues [22], [72]. But neither of these works has investigated the possibility of using the pricing to adjust the parameters of a service policy. Pricing research in networks has been quite intensive during the last years. Pricing and link allocation issues for real-time services that require strict QoS guarantees were studied in [56], [57]. The possibility of using the revenue as the criterion for updating weights was considered by [60]. That work proposes a resources sharing model that extends our previous pricing and QoS research, in which the optimal link allocation between traffic classes using different pricing scenarios [23], [27] and the revenue maximization model based on the WFQ policy have been discussed [26]. The proposed model relies upon the WFQ policy [13] because it is one of the policies that are capable of providing various QoS [65]. The model acts as a superstructure over the WFQ policy, weights of which are updated in accordance with the amount of requested bandwidth and pricing of each service class. It has been shown in [11], [36] that it is a fair way of pricing Internet services with QoS requirements in terms of their effective bandwidth. Furthermore, charges are predictable by the end users and no additional mechanism is needed to communicate tariffs to users. Besides, the proposed model does not make any assumptions about user behavior and traffc patterns. Thus, it can be applied to various network environments. The aim is to provide a simulation of the proposed resources sharing model, for which only theoretical evaluation in the single-node case was given. This work considers a more realistic network scenario and compares the adaptive and non-adaptive approaches in terms of obtained revenue and state of queues at intermediate nodes. To make the problem tractable a simple network topology, which consists of several switching nodes with the proposed model, is considered. These nodes serve simultaneously several groups of clients with different QoS requirements.

Network applications such as streaming media and content distribution generate real-time flows that have tight bandwidth and delay requirements. Guaranteeing performance involves provisioning of resources and enforcing their usage during run-time. A straightforward approach to this problem is to allocate sufficient resources so that flows never lack them. However, such an approach has several serious drawbacks. It does not take into account the fact that the amount of active flows varies all the time consuming different portions of resources. Therefore, it is possible to increase the link utilization by adjusting parameters of a service policy and dynamically allocating dynamically resources at a router. The problem of adapting parameters of different queuing disciplines has received considerable attention lately. For instance, [25] has proposes the use the average packet length to adapt weights in the Weighted Round Robin policy. The adaptive WFQ algorithm, in which the state of queues is used to adapt weights, has been proposed by [22]. The problem of adjusting weights in the WFQ and WRR policies according to the dynamics of the average queue size has been considered in [72]. But neither of these works has investigated the use of the revenue criterion.

4.1 An Adaptive Approach to WFQ with the Revenue Criterion

In publication I we propose a model for serving multiple service classes and show how the revenue criterion can be used to optimize the way a provider shares resources [60]. Queuing policies are the basic principle for allocating resources between customers and service classes. There are a number of queuing policies that have been proposed and investigated in the context of networks [13], [39], 54]. But these works do not consider the way parameters of queuing models can be adapted to varying network environment and user requirements. Later researches have targeted the problem of dynamic adaptation of queuing models. For instance, [25] has proposed the use of an average packet length to adapt weights in the Weighted Round Robin policy. The adaptive Weighted Fair Queue algorithm, in which the state of queues is used to adapt weights, has been proposed by [22]. The problem of adjusting weights in WFQ and WRR policies according to the dynamic of the average queue size has been considered in [72]. But neither of these works has investigated the use of revenue to adapt weights. This work extends the previous pricing and QoS research [26], [31], [30], and considers the WFQ policy, in which weights are updated using the revenue criterion. This publication provides a simulation of a single node that serves several service classes with different QoS requirements.

Queuing policy is the basic principle for allocating resources. The choice of an appropriate service discipline at nodes of a network is the key for providing an effective flow control. A good scheme should allow the treatment of service classes differently in accordance with their desired QoS. The most popular queuing policies are First-In-First-Out, Priority Queue [39], Weighted Round Robin [20] and Weighted Fair Queue [54]. The FIFO determines a packet's service order strictly based on its arrival order with respect to other packets. Therefore, this policy cannot perform necessary bandwidth allocation and provide desirable QoS. The PQ policy prefers classes with higher priority and therefore packets from a higher priority queue are always served first. Thus, if a higher priority queue is always full, then lower priority queues are never served. This problem can be eliminated using the WRR queuing policy, in which each service class queue is assigned a weight and serviced in a Round Robin (RR) fashion. The number of packets transmitted from each queue during a cycle is proportional to the weight. However, if one queue has a longer average packet size than another one then the first service class receives more bandwidth. This disadvantage was overcome with the Bit-Round Fair Queue technique proposed in [13]. However, it is not able to allocate different portions of capacity to support QoS requirements. An enhanced BRFQ queuing policy, which supports differential allocation capability, has been considered in [54]. This technique, also known as WFQ, schedules packets according to their arrival time, size and associated weight. Weights for different classes can be assigned in such a way that the performance of high priority classes is guaranteed and no starvation of low-priority classes occurs. Thus, the WFQ technique is proposed here since it is one of the queuing policies that can be effectively used for providing various QoS [65].

To realize a WFQ delivery mechanism a network router has to support QoS networking in accordance with the service model. The common structure of a QoS router is depicted in Figure 12. In a QoS-enable router, a multi-queues architecture exists that classifies and serves packets according to the defined QoS parameters. The packet classifier puts received packets into appropriate queues, each of which corresponds to a service class. The packet scheduler is responsible for taking data from input queues and transmitting it to the output according to weights associated with each queue.

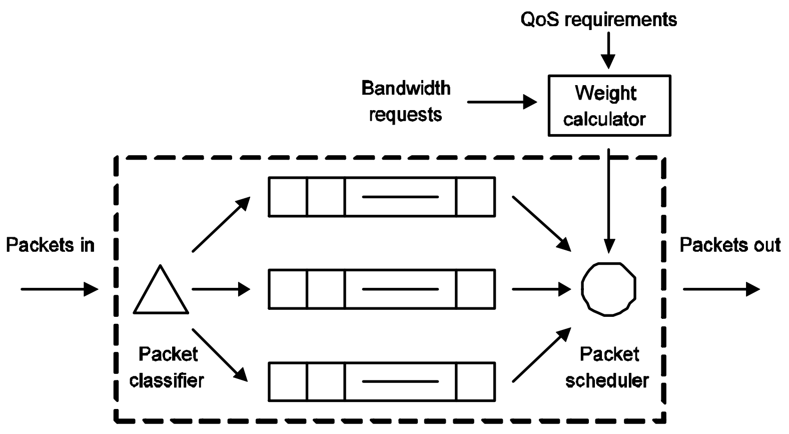


FIGURE 12 Structure of a QoS router

The weights of service classes should conform to the following constraint:

$$\sum_{i=1}^{m} w_i = 1, 0 < w_i \leq 1, \qquad (51)$$

where w_i is the weight of the *i*th class and *m* denotes the total number of service classes, and, as a result, input queues. Each weight w_i indicates the portion of data in the *i*th queue that will be sent out during the next service cycle. For example, all packets of a queue will be scheduled to deliver if the weight is assigned to one.

The WFQ policy does not specify any mechanism for changing weights. It is common that they are set to certain predefined values and are not changed over the course of time. Since there is a need to adapt to varying environment an additional component should be introduced that would be responsible for calculating new values of weights. Such a component might be referred to as the *Weight calculator*. It may use external information in conjunction with sophisticated criteria for updating weights. External information may include QoS requirements for every service class and bandwidth requests. One of the criteria that can be effectively used for calculating weights is pricing. Because there are several charging models, it is necessary to consider them.

Charging Models

It has been demonstrated that pricing is an effective method for achieving fair allocations of resources between service classes [71]. Nowadays several charging models are used in pricing Internet services [58]. Among these, flat charging and *usage charging* are the most popular. The flat pricing implies that a customer pays the joining fee only and has an unlimited access to network resources, regardless of the connection time or amount of data transferred. But it is usually the case that this pricing strategy does not guarantee any QoS as it fails to take into account the amount of data transferred or the time a service is provided. On the other hand, the usage pricing is based on the amount of resources used or reserved by a customer. Experiments show that it is a fair way to charge customer and to allocate network resources [3]. Traditionally, in Internet the volume-based charging is used because it suits Internet resources and access speeds. The price can remain fixed or change over the course of time. It can depend on such parameters as the time of day, congestion level [16], and bandwidth provided [17]. This research implies that the price remains fixed for relatively long periods compared to the period of time when weights do not change their values. Hence, the usage price for the *i* th class will be referred to as C_i.

To charge customers a provider uses the *pricing function*. As mentioned earlier, each service class obtains its portion of processing resources according to the assigned weight. Suppose, the d_0 is the minimum processing time it takes the scheduler to transmit one byte of data from one of the input queues to the output. Since there are several service classes and each of them has its own weight then the *i*th class on average requires the following minimum amount of processing time to transfer one byte of data:

$$P_i = \frac{d_0}{w_i} (seconds). \tag{52}$$

As a result, $1/P_i$ bytes of data can be transferred per second from the i th queue. If all customers within a service class are charged equally and a provider is paid for data transferred then the pricing function for the i th class can be chosen of the following form:

$$R_i = \frac{C_i}{P} = \frac{C_i w_i}{d_0} (monetary\ units/second). \tag{53}$$

It should be noticed that the proposed pricing function does not depend on the numbers of customers. Indeed, if a provider has switching equipment with a certain capacity then it does not matter how many users there are. The more users are served within a service class the less bandwidth each user has. But the total amount of data capable of being transferred over a period of time remains the same. This is true even for the case when all the users work continuously and use all resources.

Since a provider has not only to share resources but also to provide the required QoS it is necessary to introduce constraints that will reflect QoS parameters for every service class.

QoS Constraints

Fundamental goals of QoS include guaranteed bandwidth, jitter and latency that are usually required by real-time and interactive applications. One of the QoS constraints that can be chosen for specifying customers' demands is the bandwidth. Indeed, nowadays customers require a minimal guaranteed bandwidth to network resources. Taking into account the number of customers in a service class it is possible to propose the function of the following form:

$$B_i^{user} = \frac{1}{P_i N_i} = \frac{w_i}{N_i d_0} (bytes/second), \tag{54}$$

where B_i^{user} specifies the bandwidth allocated for each customer, and N_i is the number of customers in the i th service class.

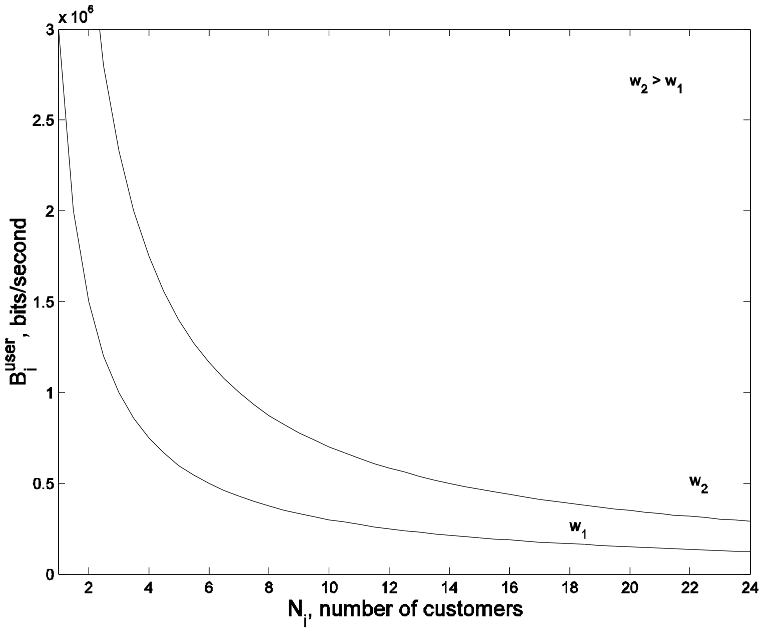


FIGURE13 QoS bandwidth threshold

This function provides a mean for setting up the minimal bandwidth that is guaranteed for every customer. As can be seen from Figure 13, this function decreases with respect to the number of customers or the minimum processing time. At the same time, it increases with respect to the assigned weight. It is obvious that for the given d_0 and w_i only the certain maximum number of customer N_i can be served. Therefore, it is easy to setup QoS limits using the value of the B_i^{user}. If it is necessary to increase the guaranteed bandwidth then either the weight w_i must be increased or the minimum processing time d_0 must be decreased. As an example, Figure 13 illustrates that for a service class with the weight w_2 the QoS of 1 Mbits/second can be provided if there are fewer than 7 active users approximately.

The bandwidth constraint can be optional since a certain service class may have no bandwidth requirements associated with it. Mail and news can belong to such services because in most cases they are not critical at all. These services might always be provided with the available capacity only. Furthermore, certain classes may serve users that share simultaneously available resources. In other words, the specified bandwidth could be guaranteed not for every customer but for the whole service class. In this case the simplified expression (54) can be used

$$B_i = \frac{w_i}{d_0}, \tag{55}$$

where B_i is the bandwidth allocated for a whole service class.

The General Model

The general model for sharing resources comprises of a pricing function and a set of constraints. As considered earlier, the R_i specifies the revenue from the i th service class. Thus, the overall revenue can be expressed with the following function:

$$F(w_1, w_2, ..., w_m) = \sum_{i=1}^{m} \frac{C_i w_i}{d_0} \quad . \qquad (56)$$

This function should be maximized since the goal of a service provider is to increase the revenue. The weights are the only parameters that can be transparently manipulated over the course of time. Therefore, a provider's task is to choose optimal values for weights. Thus, the equation (56) must be solved with the constraints (51), (54) and (55) kept in mind. The general model is as follows:

$$\max \left\{ \sum_{i=1}^{m} \frac{C_i w_i}{d_0} \right\} \text{ subject to: } \quad \sum_{i=1}^{m} w_i = 1,\ 0 < w_i \le 1,\ \ w_i \ge N_i d_0 B_i^{user},\ w_i \ge d_0 B_i .$$

It is the linear optimization problem, which can be solved using a set of methods. It is possible to obtain the optimal values of the w_i coefficients using the linear programming method and the *simplex* algorithm [69]. After new values are calculated, a provider can immediately use them or smoothly change the existing ones.

The proposed model can be implemented within switching routers that are used to provide the QoS for transmitted packets. In this case, two problems must be solved: first, how to differentiate service classes from each other and, second how to keep track of currently active users. The Internet architecture has two major models for providing QoS. The Integrated Services [7] was one of the first solutions. It implies the presence of network elements along the data path, which guarantees the QoS for packets, and a protocol to convey QoS management information between them [8]. The process of reserving resources is initiated by a sending application and the actual reservation is done by a receiving application. In turn, each intermediate node reserves appropriate resources for further usage. Thus, each node keeps track of reservations and may use this information for determining the amount of active users. Since incoming packets are passed through the packet classifier the QoS class, for which a packet belongs to, can be determined.

Another application area of the proposed model is the Differentiated Services technology [6]. It specifies several Per-Hop-Behaviors, each of which describes the externally observable forwarding behavior of routed packets. Each PHB can be treated as a separate service class with certain QoS requirements. All packets are marked using the DiffServ field value that can be used by the classifier in a DiffServ router. Unlike the IntServ, the DiffServ technology avoids per-customer states that are required to keep track of currently active users. One of the possible solutions for this problem is CIM (Common Information Model) proposed in the context of the policy-based management [51]. It can be used to supply DiffServ routers with configuration and management information.

Simulation Model and Results

Simulation is used to evaluate the proposed model. As mentioned earlier, the service provider's aim is to maximize revenue by manipulating weights for each service class. Recalculation of weights can be done upon events that affect the parameters of the model. These events might include the introduction of a new service class with certain QoS parameters or upgrade of the equipment. However, it is obvious that changes in parameters of service classes and equipment do not occur often considering the number of users. It is often the case that customers subscribe to certain services, change subscription parameters or refuse from certain services at all. Therefore, it is necessary to consider the way the proposed model behaves in such situations.

The simulation considers a single switching node that serves three service classes of customers. A bandwidth of 1600 kb/second is allocated to each customer in the first class. The second class provides a bandwidth of 800 kb/second for every customer. The third class guarantees no QoS characteristics to its customers and has a total bandwidth of 1 MB/second. The number of customers in the first class is limited to 20. The second and the third service classes are not allowed to have more than 50 customers. The mean time it takes the node to transfer one byte of data is 0.1 ms. All customers are charged using the byte-wise usage scheme. The prices for 1 kB of the first, second and third class data are 0.02, 0.01 and 0.005 monetary units, respectively. During the simulation, it is assumed that every customer generates uniform constant traffic and uses all the resources provided by the correspondent service class.

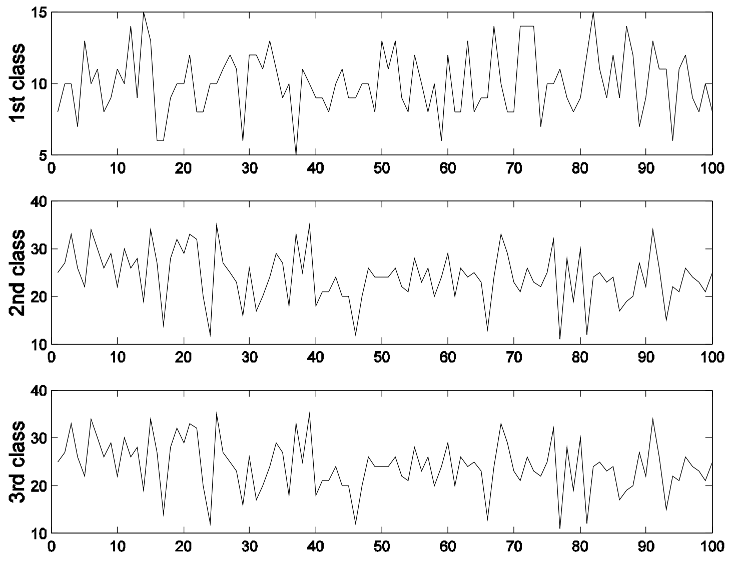


FIGURE 14 Simulated number of customers

The simulation corresponds to a situation where customers begin to use resources at various times, use them continuously and then inform the switching node that resources are not needed anymore. In this case, the total numbers of active customers within each service class may significantly vary. The random number of customers, show in the Figure 14, is obtained using the normal distribution. Figure 14 shows the generated number of customers for the appropriate service class.

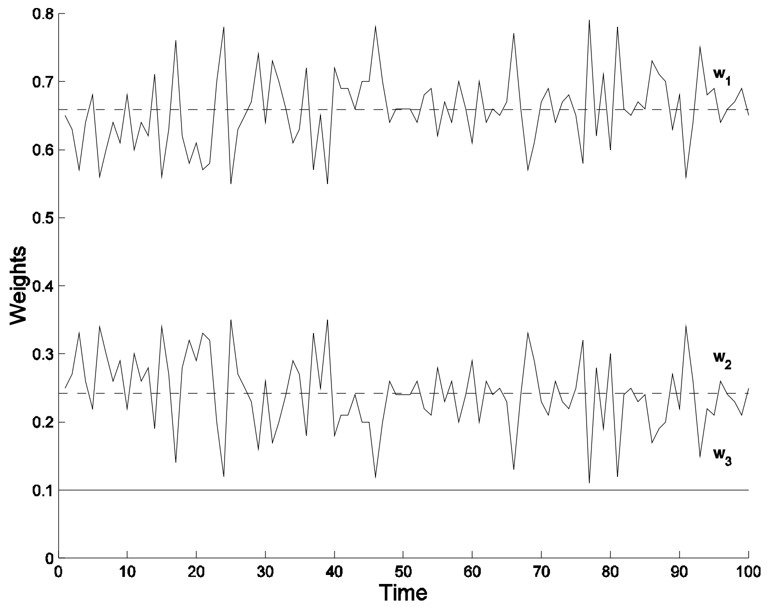


FIGURE 15 Progress of the weights

As the number of customers within service classes changes each of them requires different portion of resources. As it can be seen from Figure 15, weights are recalculated in response to changes in the number of the users. This figure shows shows the mean values of the weights as well. It can be noticed that on average the proposed model gives higher values for those weights that correspond to more expensive service classes. It should also be noticed that the weight of the third class was not changed at all since this class does not guarantee any QoS and provides the fixed bandwidth for the whole service class.

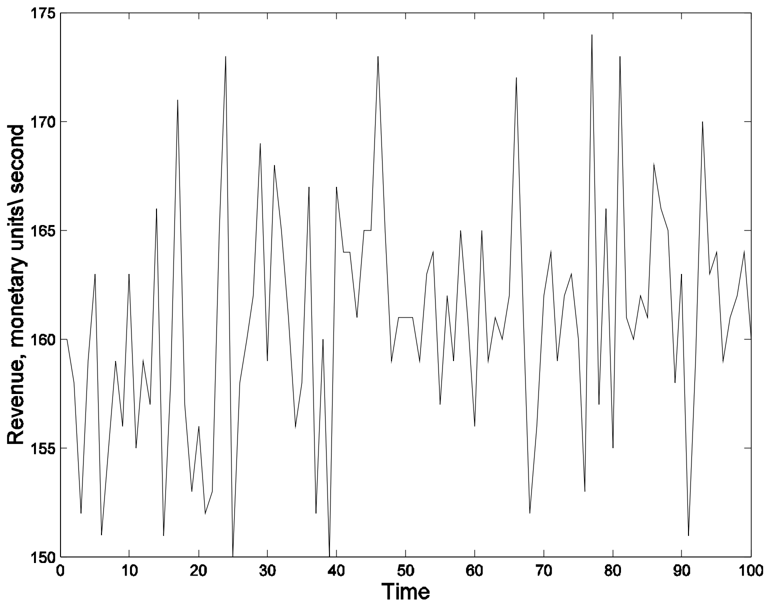


FIGURE 16 Progress of the revenue

Figure 16 shows the revenue in monetary units/second received during the simulation. The peaks of the revenue correspond to the moments of time when the weight of the first service class has a peak value. That is because the first class is charged two times more than the second one. When comparing the peaks of the revenue and the number of customers in the second class, it is possible to come to a conclusion that in most cases the number of users is small. Thus, more resources are allocated to first-class users, who are charged differently. Contrary to this, lower revenue is received when there are many users in the second class and fewer resources are allocated for the first-class customers. It can also be seen that the numbers of customers within the third class does not influence the revenue since the appropriate weight has a constant value.

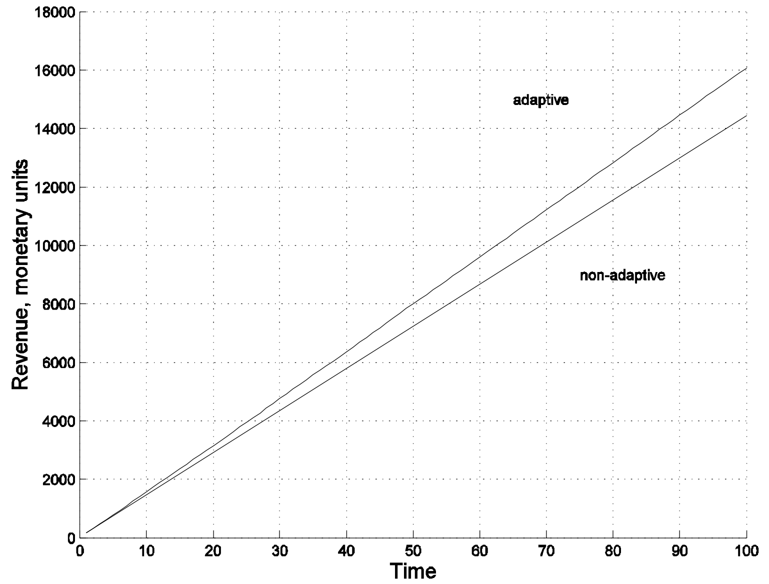


FIGURE 17 Total revenue in the case of adaptive and non-adaptive schemes

Figure 17 compares the revenue obtained by a provider when the adaptive and non-adaptive WFQ algorithms are used. The non-adaptive approach has fixed weights that are not updated over the course of time. Although they satisfy all QoS requirements resources are not shared optimally. The adaptive approach, on the contrary updates weights and, as a result, increases the total revenue over a period of time when compared to the non-adaptive method. Thus, it is possible to come to the conclusion that the proposed model tries to maximize the total revenue whenever it is possible to do so. It guarantees the required QoS for each service class and at the same time tries to allocate as many resources as possible resources to those service classes that are charged higher.

4.2 Simulation and analysis of the revenue criterion based adaptive WFQ

This chapter presents a more deep analysis of the developed revenue criterion based adaptive WFQ [61]. Queuing policy is the basic principle for allocating processing resources. The choice of an appropriate service discipline at nodes in the network is the key for providing an effective flow control. A good scheme should allow the treatment of service classes differently in accordance with the desired QoS. Among existent service disciplines the Generalized Processor Sharing (GPS) policy [39], which is based on the fluid traffic model, is capable of providing the delay and buffer occupancy bounds for the desired QoS. The throughput and delay bounds of the GPS discipline are closely approximated by the WFQ policy [54]. This policy guarantees a minimum fraction of the available bandwidth irrespective of the behavior of incoming packet streams. If one of the service class queues is empty then its processing share will be redistributed among the active ones, in proportion to their own share. However, reallocation of the unused bandwidth is static and can result in ineffective resource distribution. Thus, there is a need for a real-time adjustment of weights according to varying conditions.

To realize the WFQ delivery mechanism a network router has to support QoS networking in accordance with the service model. In a QoS-enabled router, a multi-queue architecture exists that classifies and serves packets according to the defined QoS parameters. The packet classifier puts received packets into appropriate queues, each of which corresponds to a service class. The packet scheduler is responsible for taking data from input queues and transmitting it to the output according to weights associated with each queue. Since there is a need to adapt to the varying network environment, an additional component should be present. Its responsibility is to update values of weights using certain criteria.

We decided to use the model that is considered and precisely described in [60]. This model uses the revenue criterion for updating weights of the WFQ discipline. It has been demonstrated that pricing is an effective method for achieving fair allocation of resources between service classes [71]. If a provider uses the usage-based charging then the *pricing function* can be expressed as follows:

$$F(w_1, w_2, ..., w_m) = \sum_{i=1}^{m} C_i w_i B \; [monetary\ units / \sec ond], \qquad (57)$$

where w_i is the weight of the i th class, B is the total output of a router, C_i is the price for one byte of data, and m is the number of service classes. The weights are the only parameters that can be transparently manipulated over the course of time. So, parameters can over course a provider's task is to obtain, for weights, obtain values, which maximize the revenue and ensure the required QoS. The model, which enables the provider to do thit, can be represented as follows:

$$\max\left\{\sum_{i=1}^{m} C_i w_i B\right\} \text{ subject to: } \sum_{i=1}^{m} w_i = 1,\ 0 < w_i \le 1,\ \ w_i \ge N_i \frac{B\frac{flow}{B}}{B} \quad ,\ w_i \ge \frac{B_i}{B}.$$

Here, N_i is the number of active data flows within the i th class, B_i^{flow} is the bandwidth guaranteed for each flow, and B_i is the bandwidth guaranteed for the whole service class. Depending on the resource allocation scheme a provider chooses the necessary constraint. In other words, certain bandwidth can be guaranteed either for each data stream or for the whole service class. In the latter case, all flows share the same bandwidth. On the other hand, both constraints can be used simultaneously if a provider wants to allocate a certain minimum portion of processing resources for a whole service class and, at the same time, guarantee QoS on the per-flow basis. The general model is linear optimization problem that can be solved using a set of methods. One of the methods, which can be used to obtain the optimal values of the w_i coefficients, is the *simplex* algorithm [69].

Simulation environment and results

The aim of the simulation is to check the proposed model and the way it allocates limited resources between service classes. To analyze its behavior the following metrics are used: the total revenue, state of queues, and the number of dropped packets. The total revenue shows whether the proposed model enables a network provider's to functioning to improve. To evaluate the efficiency of the model the revenue obtained in the case of the adaptive WFQ is compared with the revenue obtained with the non-adaptive WFQ. Another important issue to consider is the size of queues at switching nodes when the proposed model is in effect. The reason for this is that such important metrics as end-to-end delay and queuing delay depend significantly on the mean queue size. The simulation is done in the NS-2 simulator [70] using the implementation of the WFQ policy made by [45]. The proposed model is implemented in C++ and the appropriate NS-2 interface is created so that it can be used from a simulation script. Besides, certain enhancements are added to the original implementation of the WFQ policy to enable dynamic manipulation of weights. The simulation scenario does not consider any specific signaling protocol that clients, intermediate nodes and a destination node can use to inform each other about required resources. This issue is left open and it is a subject for future research. Instead, the inner functions of the simulation environment are used to keep track of the number of active flows at intermediate nodes. Though that environment does not correspond to a real-life scenario, the amount of additional signaling information is not great and the time taken for this is not significant. In the future Resource ReServation Protocol (RSVP) [8] can be used for these purposes.

Simulation setup

The simulation environment is depicted in Figure 18. It consists of a set of clients, three intermediate nodes and a destination node. Clients are divided into two groups that will be later referred to as Group A and Group B. Each group is served by the correspondent intermediate node, which implements the proposed model and performs resource allocation between active clients using the adaptive WFQ policy (shown as A–WFQ on the picture). Hence, these nodes will be referred to as dispatching nodes. During the simulation each client generates exactly one flow of data. This is sent to the correspondent dispatching node over a link, the bandwidth and propagation delay of which are set to 500 kbps and 2ms respectively. Dispatching nodes A and B are connected to the intermediate node C. No classification of the input traffic is performed and forwards all data is forwarded to the destination node using the FIFO queuing policy. Furthermore, the dispatching nodes perform classification of data that comes from clients only. All responses from the destination node to clients are forwarded using the simple FIFO queuing policy. They are neither categorized nor prioritized in any manner. All clients in each group are divided into three service classes, which, hence, will be referred to as *Gold*, *Silver* and *Bronze*. Table 1 shows the bandwidth provided for each flow, the maximum number of active flows within each service class, the price for 1Mb of data, and parameters of the ON/OFF model that control the activity of clients. The ON/OFF intervals are used to generate random numbers and to simulate the variation in the number of clients. Since the proposed model does not make any assumption about the behavior of clients, the uniform distribution is chosen.

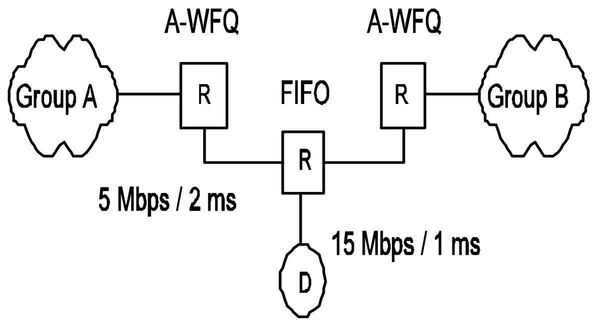
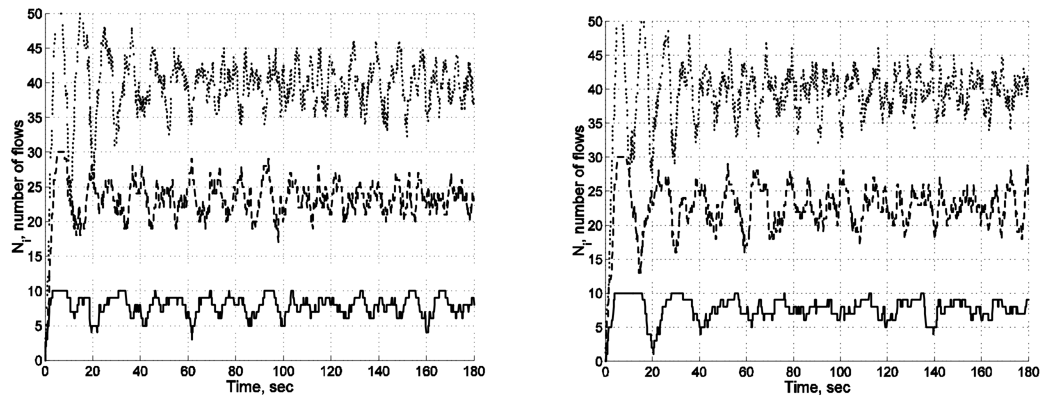


FIGURE 18 Simulation environment

To simulate bulk traffic flows from clients to the destination node, the FTP (File Transfer Protocol) application is placed on every client node. For these purposes TCP agents are created on each client node and TCP sink agents are placed on the destination node. The simulation uses the varying size of a TCP packet, which is uniformly distributed between 400 and 600 bytes for all service classes. The TCP window size is set to 2. To simulate overprovision of resources, the weight of the Silver class at node A is set to a fixed value. Each dispatching node has a buffer space to compensate traffic jitters. Since during the simulation the packet size does not significantly vary, the buffer space is measured in the maximum number of packets that can be stored in a queue at the dispatching node. For the Gold, Silver and Bronze class the buffer space is set to 10, 30 and 50 packets respectively. As might be noticed, the length of the queues is the same as the maximum number of the flows in the correspondent service class. Of course, such short queues are not capable of keeping the necessary amount of packets if the clients begin to transmit large amount of data. Such limits for maximum lengths of queues were chosen to demonstrate the way different values of weights impact the state of queues and the loss rate in each service class. The whole simulation lasts for 180 seconds. As will be shown later, such relatively short simulation time is quite sufficient to demonstrate the proposed model. During the simulation statistical data is gathered at intervals of 0.1 second. The gathered data is related to the number of active flows, current values of weights, state of queues, the number of dropped packets and the obtained revenue. Since it is a burden for the dispatching node to recalculate the weights every time the number of flows changes, they are updated at intervals of one second.

TABLE 1 Parameters of the service classes

Class	Bandwidth (Kb/sec	Max number of flows	Price for 1Mb of data	ON interval	OFF interval
Gold	100	10	2	8-20	3-5
Silver	50	30	1	6-15	2-4
Bronze	20	50	0,5	7-8	1-3



(a) Group A (b) Group B

FIGURE 19 Dynamics of the number of active flows

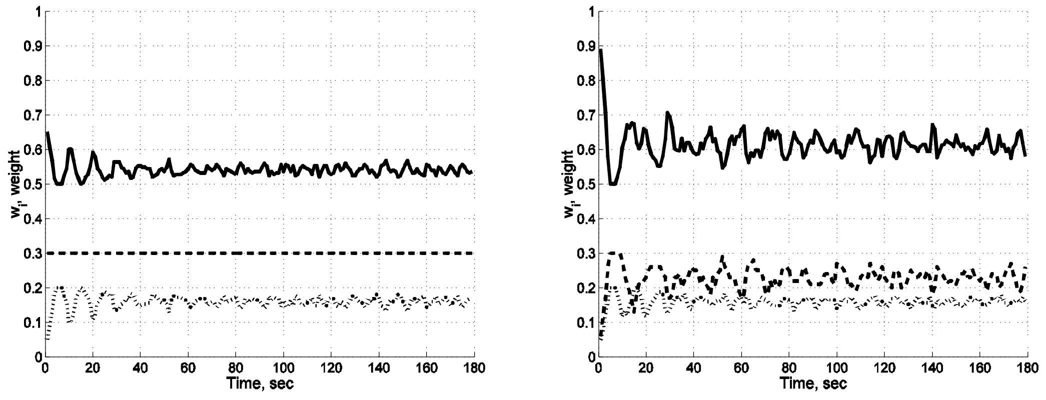
Results and analysis

Figure 19 shows the dynamics of active flows for each group of clients. The solid line corresponds to the Gold class, the dashed line represents the Silver class, while the dotted line indicates the Bronze class. As can be seen, the number of active flows fluctuates within certain margins but on average it is close to the upper limit. Thus, almost the maximum number of clients are active over the simulation period and the dispatching nodes have to share a limited set of resources effectively. Comparing Figure 19(a) and Figure 19(b) it is possible to notice that the patterns of user behavior in each group are very similar. In effect, dispatching nodes A and B have to deal with the same traffic patterns. Table 2 summarizes the basic information about the dispatching nodes that was obtained during the simulation. As can be noticed, most of the packets are transferred within the Gold service class and no packets are dropped. On the other hand, the Bronze class has the least number of transferred packets and the greatest number of dropped packets. Though a certain portion of the packets was dropped the loss ratio is not great and can be tolerated by many network services. It is to be expected that the more expensive a service class is, the fewer the packets should dropped be. Thus, the Gold service class guarantees a reliable data delivery with high QoS requirements. It may be used to provide services such as real-time audio and video. The Silver and Bronze classes might be used for ordinary services, like WWW or E-mail. Comparing the results one can notice that the total revenue at dispatching node A is less than at dispatching node B. This is because the Silver class has a fixed weight that provides this class with redundant processing resources. As a result, fewer resources are allocated for the Gold class, which has the highest price. On the other hand, more resources are allocated for the Silver and the Bronze classes the loss rate of which is less than that of dispatching node B.

TABLE 2 General statistics for the dispatching nodes

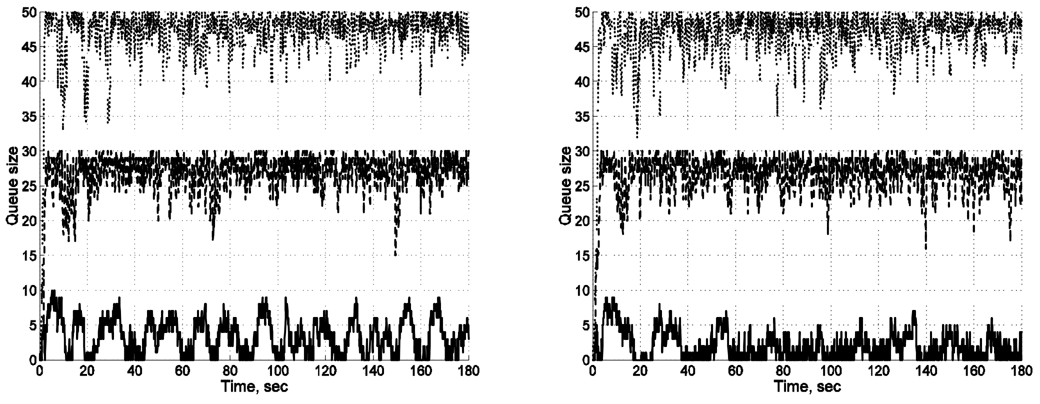
Class	Node A			Node B		
	Packets departed	Packets dropped	Loss ratio	Packets departed	Packets dropped	Loss ratio
Gold	87253	0	0 %	127361	0	0 %
Silver	65622	1204	2 %	45661	1215	3 %
Bronze	52799	2174	4 %	31861	2232	7 %
Total	205674	3378		204883	3447	
Revenue	1111			1321		

The allocation of resources between the service classes is done according to the weights, which are recalculated by the proposed model when the number of active flows changes. The dynamics of weights at the dispatching nodes is illustrated in Figure 20. The solid line represents the weight of the Gold class, the dashed line corresponds to the Silver class, and the dotted line shows the weight of the Bronze class. As might be expected, the model tries to allocate as many resources as possible to the more expensive service classes. As a result, on average the Gold class has the highest value of the weight and the Bronze class has the lowest value. Figure 20(a) shows that the value of the weight for the Silver class at dispatching node A is constant. This class has the reserved amount of the processing resources. Because of it, less weight is assigned for the Gold class. As can be seen in Figure 20(b), the Gold class has the biggest weight, more Gold class packets are forwarded, and the total revenue is bigger at dispatching node B than at node A.



(a) Node A (b) Node B

FIGURE 20 The dynamics of weights at the dispatching nodes



(a) Node A (b) Node B

FIGURE 21 State of queues at the dispatching nodes

Figure 21 shows the state of queues at the dispatching nodes. As it can be seen from this figure, the Gold class has the shortest queue size, which almost never approaches the upper limit. As expected, the Bronze class has the longest queue size. On one hand, the number of clients in each service class influences the mean queue size. On the other, if a weight is not big then it is quite likely that a queue will not be short. Since the Gold class is always assigned with a big weight, it has the shortest queue size. On the contrary, the Silver and Bronze classes have on average low weights. Thus, as might be noticed, the size of their queues approaches the limit. Comparing Figure 21(a) and Figure 21(b) it is possible to arrive at the conclusion that the state of queues at the dispatching nodes is almost the same. It is also possible to notice that the mean size of the Gold class queue at node A is bigger than at node B. The fixed portion of resources reserved for the Silver class explains it. Figure 22 compares the total revenue obtained during the simulation in the case when the adaptive and the non-adaptive approach is used. The solid line shows the adaptive revenue, and the dashed line shows the non-adaptive revenue. As can be seen, the adaptive WFQ increases the total revenue. The gap between the total revenue of the adaptive and non-adaptive approaches increases with time. In general, when the adaptive WFQ is used, the behavior of the revenue is very similar to theoretical evaluations made in [60]. The obtained revenue depends on the traffic type (CBR or VBR), overall activity, congestion methods, and such parameters as window size of the TCP protocol. It is also understandable that the more free the resources a router has the more of them can be allocated for expensive service classes. The fluctuations in revenue can be easily caused by the packet nature of the network and by the district interval, in which weights at the routers are updated.

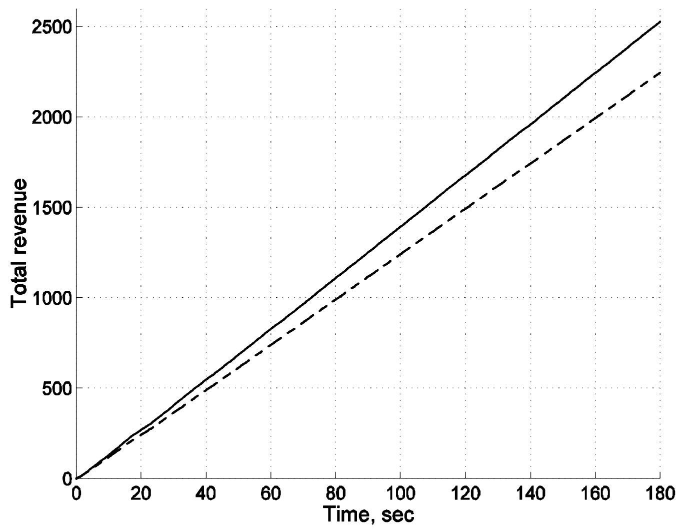


FIGURE 22 Total revenue

4.3 On providing bandwidth and delay guarantees using the revenue criterion based adaptive WFQ

This chapter presents a resource sharing model that provides throughput and delay QoS guarantees and is capable of maximizing a service provider's revenue [62]. The proposed model increases the total revenue by allocating unused resources to certain flows at the expense of reducing the amount of resources previously assigned to other flows. As in the previous works, the WFQ rate-based queuing policy is chosen because the relationship between QoS parameters and the amount of consumed resources can be explicitly specified. Furthermore, the WFQ is one of the queuing policies that are capable of providing QoS in various network environments [65]. The research considers a single node that implements the proposed resource sharing model and acts as a router for data flows with different QoS requirements.

The Model

Though WFQ is one of the queuing policies that are capable of providing various QoS it has one serious drawback that limits its usage in high-speed network environments. To perform fair resource allocation the WFQ algorithm has to keep track of all active sessions. If the number of data flows is great then it is a computational burden for a router. To cope with this problem it is proposed to group flows with similar QoS requirements in *service classes* and perform allocation of resources between them rather than between flows. On one hand, it is possible to provide sufficient resources for each service class when the maximum amount of flows is reached. On the other, it is much more efficient and resource conserving to perform dynamic allocation of resources according to the current number of active flows. Free resources can be allocated for those service classes that improve a service provider's functioning in certain manner. Since a service provider is often interested in maximizing the *revenue* it is possible to provide more resources for those service classes that increase it.

It is usually the case that network services can be described in terms of such performance characteristics as throughput, delay and packet loss. Thus, each service class, which groups network services with similar characteristics, may have associated QoS parameters that specify requirements for every flow that belongs to it. As considered in [73], the most important parameters are throughput and delay. Another important QoS parameter is packet loss, which can occur due to the buffer overflow or delay bound violation. Depending on the model type – *deterministic* or *statistical* - either zero or non-zero packet loss is guaranteed. The proposed model is considered to be a deterministic model that always guarantees zero probability loss. In other words, all service classes are always provided with enough buffer space to keep incoming packets, and the problem of managing the length of the queues is left open. Because resources are allocated on the per service class basis, there is a need to estimate bandwidth requirements and worst-case queuing delay of a flow that shares resources with other flows belonging to the same service class. Thus, two performance parameters, bandwidth and delay, will be considered in more detail.

A. Bandwidth

Suppose that B is the total throughput that a router with the WFQ policy has. If all service class sessions are active, then each service class receives a portion of the total bandwidth, which is determined by weight w_i and is equal to $w_i B$ [37]. Hence, to simplify expressions we imply that all w_i obey the expression $\sum w_i = 1$ *and* $0 < w_i \leq 1$. If there are N_i active flows within the i th service class then each flow has bandwidth that can be approximated as follows:

$$B_i^{flow} = \frac{w_i B}{N_i}. \tag{58}$$

The B_i^{flow} can be treated as one of the QoS parameters that specifies the required bandwidth for a flow in the service class. Thus, the minimum weight w_i^b, which is necessary to guarantee this QoS for every flow can be expressed in the following manner:

$$w_i^b = N_i \frac{B_i^{flow}}{B}. \tag{59}$$

It is clear that the more active the flows are and the bigger the required per-flow bandwidth is the bigger the portion of resources that is necessary for the whole service class if the total throughput is fixed.

B. Queuing delay

Due to the buffering, scheduling and transmitting packets the size of a scheduler's queues vary all the time. On the other hand, the length of a queue at a routing node impacts the queuing delay and overall end-to-end delay of packets as well. It can be shown that the worst-case queuing delay experienced by a packet belonging to a flow in the WFQ service policy is given by the following expression, where $L_{\max}$ denotes the maximum packet size:

$$D = \frac{\sigma + L_{\max}}{w_i \rho} + \frac{L_{\max}}{B} . \tag{60}$$

The previous equation assumes that each incoming flow is regulated by the Leaky Token Bucket scheme [12] with the bucket depth σ and the token rate ρ. The σ and ρ can be viewed as the maximum burst size and the long term bounding rate. Since ρ is a long term bounding rate it is possible to imply that it is equal to bandwidth allocated for the certain service class. As considered in the previous section, it is equal to $w_i B$. Another important assumption is the value of σ when there are several active flows. Suppose, if N flows begin to send data simultaneously, then the maximum burst size arriving to a router is equal to $N\ L_{\max}$. Thus, (60) can be presented in the following manner:

$$D_i = \frac{(N+1)L_{\max}}{w_i^{2B}} + \frac{L_{\max}}{B} . \tag{61}$$

where D_i is the worst-case delay of the i th class.

Since there is a need to know the value of w_i^d, under which required queuing delay can be guaranteed, it is possible to use (4) to obtain it:

$$w_i^d = \frac{(N+1)L_{\max}}{BD_i - L_{\max}} . \tag{62}$$

Here, w_i^d specifies the minimum value of w_i that is necessary to guarantee this QoS. It is clear that the more active the flows are and the less the required delay is the bigger portion of resources that must be allocated.

C. General model

It has been demonstrated that pricing is an effective method for achieving fair allocations of resources between service classes [71]. Nowadays several charging models are used in pricing Internet services [58]. Among these, *flat charging* and *usage charging* are the most popular. The flat pricing implies that a customer pays a joining fee only and has an unlimited access to network resources, regardless of the connection time or amount of data transferred. Usually the case is that this pricing strategy does not guarantee any QoS. Furthermore, flat charging is not suitable for maximizing a provider's revenue as it charges all customers equally and fails to take into account the amount of data transferred or the time a service is provided. Since every customer has a different impact on the network load, another charging approach should be chosen. For these purposes the usage pricing could be based on the amount of resources used or reserved by a customer. Experiments show that it is a fair way to charge customer and to allocate network resources [3]. Usually, in Internet services the volume-based rather than the time-based charging is used since it reflects better connection time and access speeds. Hence, it will be referred to as $C(\tau)$ measured in monetary units per bit of data. The charging can remain fixed or change over the course of time. It can depend on parameters such as the time of day, congestion level [16], and provided bandwidth [17].

Revenue obtained by a service provider can be expressed as follows:

$$R = \sum_{i=1}^{m} C_i(\tau) w_i B, \qquad (63)$$

where *m* denotes the number of service classes if the fluid model is taken into consideration and if all bandwidth resources are used.

Expression (63) should be maximized because a service provider is interested in obtaining the maximum revenue. At the same time the QoS parameters should be guaranteed as well. Thus, the general model can be represented as follows:

$$\max\left\{\sum_{i=1}^{m} C_i(\tau) w_i B\right\} \text{subject to: } \sum_{i=1}^{m} w_i = 1,\ 0 < w_i \le 1,\ w_i \ge \max\left(w_i^b, w_i^d\right).$$

It is a linear optimization problem, which can be solved by a set of methods. One of the methods that can be used to obtain the optimal values of the w_i coefficients is the *simplex* algorithm [66].

Simulation model and results

A. Simulation setup

In this section, we study the proposed model and compare it to the traditional WFQ policy in terms of such parameters as obtained revenue, provided bandwidth and queuing delay. The simulation is done in the NS-2 simulator [70] using the implementation of the WFQ policy proposed by [45]. The proposed model is implemented in C++ and the appropriate NS-2 interface is created so that it can be used from a simulation script. Moreover, because the original implementation of the WFQ policy is not designed for dynamic change of weights certain modifications are made. They concern an update of active sessions that enable proper allocation of resources.

The simulation environment is illustrated in Figure 23. It consists of a router, a destination point, and a set of clients, each of which generates exactly one flow of data. All clients are divided between three service groups, each of which has its own set of QoS parameters. For short, they will be later referred to as *Gold, Silver,* and *Bronze* classes. Each client is connected to the router node with a link, the bandwidth and delay of which are set to 1Mbps and 2ms appropriately. The bandwidth and delay of a link, which connects the router and the destination node, are set to 2.5Mbps and 2ms.

The details of each service class are summarized in Table 3. The details include such information as the traffic type, price for 1Mb of data, maximum amount of flows within each service class, guaranteed bandwidth (in kilobits per second), maximum queuing delay (in seconds), parameters of the on-off model (in seconds), and packet sizes (in bytes). The simulation uses the on-off model when making the random amount of active flows. The period of time when a flow is active is determined by the ON time, which represents a uniformly distributed random number taken from the appropriate interval.

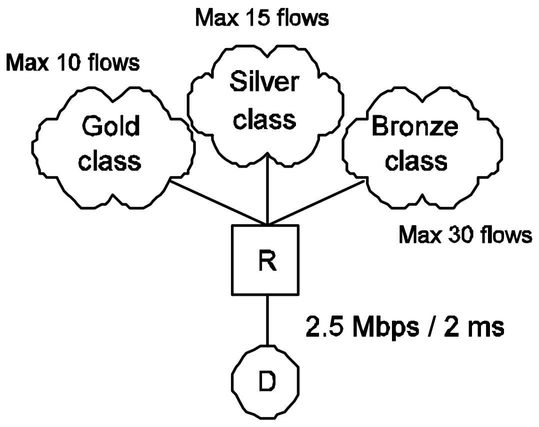


FIGURE 23 Simulation environment

The period of inactivity is determined by the OFF time that obeys exponential distribution with an appropriate mean value. Besides, different flow behaviors are used to check how the proposed model allocates resources. The simulation uses the FTP-like application source type to generate bulk data, which is transferred over the TCP protocol. To simulate audio- and video-like sources the Constant-Bit-Rate (CBR) traffic type is chosen that is transferred over the UDP protocol. It should be noticed that price for each service class is constant and remains the same during the whole simulation period. To analyze how the adaptive WFQ allocates resources within a service class a variable packet size is used. For these purposes, uniform random numbers are generated from the interval, which is specified in the table. The only class that uses the fixed packet size is the Silver class. Since data in Gold and Bronze classes is sent over the TCP protocol, the congestion control is effected by the destination node by sending back acknowledgement (ACK) packets. To make the simulation scenario more robust the window size of the TCP protocol is set to 1, thus enforcing each sent packet to be acknowledged and preventing a client from sending any data until an acknowledgment is received. Since the Silver class data is sent over the UDP protocol at the constant bit rate, source hosts them selves expert implicit congestion control.

The simulation scenario does not consider any specific signaling protocol in which source hosts, the router and the destination node can be used to inform each other about required resources. Instead, inner functionalities of the simulation environment are used to keep track of the number of active flows at the routing node. Though it does not correspond to a real-life scenario the amount of additional signaling information, which nodes would exchange if an appropriate protocol were present, is not great. Thus, it cannot influence significantly the results of simulation. And in the future, one of the protocols, such as RSVP [8], can be adapted for carrying appropriate signaling information.

TABLE 3 Source Models Parameters

Class	Type	Price for 1Mb	Max flows	Flows parameters				
				Rate	Delay	ON time	OFF time	Packet size
Gold	VBR (TCP)	2	10	100	0,05	5-10	8	300-500
Silver	CBR (UDP)	1	15	70	0,015	10-20	5	100
Bronze	VBR (TCP)	0,5	30	10	0,3	15-25	10	100-300

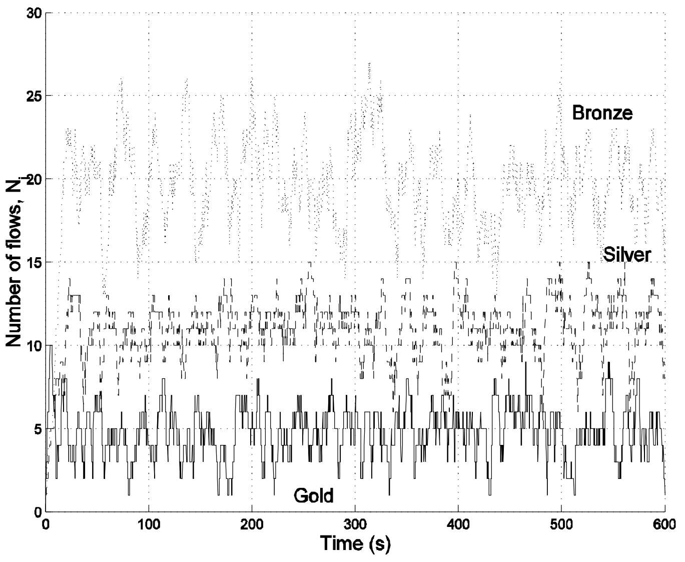


FIGURE 24 The dynamics of the amount of flows

The simulation is run twice: first time for the normal WFQ and the second time for the adaptive WFQ. Each lasts for 600 seconds. While running the simulations, the behavior of the flows is submitted to provide an accurate comparison of results. During the simulation, statistical data is gathered at intervals of 0.1 second. The gathered data concerns the amount of active flows, current values of weights, and state of the queues. Furthermore, the history of all packets is logged so that after the simulation the delay and bandwidth behavior can be analyzed on the per service class basis. Weights are also recalculated and updated at intervasl of 0.1 seconds.

D. Simulation result

As mentioned earlier, to generate a random number of flows within each service class the on-off model is used. Figure 24 shows the dynamics of the flows during the simulation. As it can be seen from the figure, the number of flows within each service class approaches the maximum limit. As a result, almost all the time a significant amount of resources is consumed and the router node has to share them appropriately. It should also be noticed that the same pattern of flows is submitted when the adaptive and the non-adaptive WFQ is used. Thus, the router node in both cases has to deal with the same input traffic that enables their fair comparison in terms of different statistical parameters. The allocation of resources between the service classes is done according to the weights associated with each service class. In the non-adaptive case they are fixed and have the values of 0.40, 0.45, 0.15 for the Gold, Silver and Bronze service classes respectively. Such values are chosen based on the maximum amount of flows, per-flow bandwidth, and delay requirements.

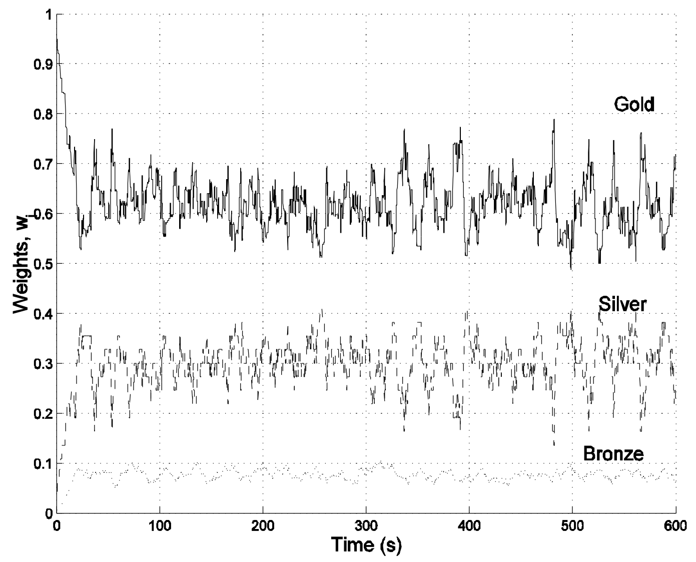


FIGURE 25 The dynamics of the weights

The adaptive WFQ recalculates and updates weights according to the current number of flows. Figure 25 shows the dynamics of weights during the simulation when the adaptive WFQ is used. As can be seen from this figure, the Gold class is given on average the highest value of weight. That is because it has the highest price and the proposed model always tries to allocate as many resources as possible to this service class. On the contrary, the Bronze service class has the lowest price and it gets the lowest value of the weight. Allocation of resources occurs mainly between the Gold and Silver service classes since they are the most expensive ones.

Table 4 compares the results of the simulation in terms of various statistical parameters in the case when the non-adaptive and the adaptive approach is used. Comparing the number of packets departed in both cases it is possible to notice that more Gold class packets and less Bronze class packets are departed when the adaptive WFQ is in effect. As a result, the total revenue in the adaptive WFQ is bigger. The relatively small difference in the total revenue between the adaptive and the non-adaptive approaches can be explained by the simulation period that lasts for 600 seconds only and the total throughput of the router. As these parameters increase, the gap between revenues increases as well. The amount of Silver class packets remains the same since in both cases source nodes generate traffic at the same rate. Considering the mean bandwidth within each service class,

TABLE 4 Statistic for Non-adaptive and Adaptive approaches

Quantity	WFQ			A-WFQ		
	Gold	Silver	Bronze	Gold	Silver	Bronze
Packets departed	187444	591002	255803	197437	591002	236886
Mean bandwidth (Kbps)	219,81	70,11	40,09	229,26	70,11	37,08
Bandwidth violations (%)	0	10,69	0	0	13,35	2,92
Mean delay (sec)	0,0005	0,0003	0,004	0,0005	0,0004	0,004
Peak delay (sec)	0,0139	0,0036	0,1016	0,01	0,009	0,55
Delay violations (%)	0	0	0	0	0	0,0827
Total revenue	1970			2021		

it is possible to arrive at the conclusion that it is not lower than the guaranteed threshold. Again, because of the adaptive scheme, the Gold class has on average the bigger bandwidth and the Bronze class has lesser bandwidth. Because performance bounds are very important for the guaranteed services Table 4 also includes the percentage of packets that have violated thresholds. The Bronze class has the non-zero percentage of bandwidth violations, it is rather small, and can be easily tolerated by many network applications. On the other hand, non-zero and relatively high percent of bandwidth violation in the Silver class can be explained by the packet-based nature of transmitted data. Because the

router outputs data in packets it is quite natural that CBR traffic fluctuates near its threshold (see Figure 26) but on average it remains at a constant level.
Table 4 also provides information on mean delay, peak delay and delay violations. As it can be seen, both the adaptive and the non-adaptive WFQ have almost the same mean delay for transmitted packets. However, the adaptive WFQ is likely to have a bigger peak delay for those service classes that do not have a high price. The only service class that violates the delay threshold is the Bronze class. But as can be seen, the number of packets that violate this threshold is not great and can be easily tolerated by network applications.

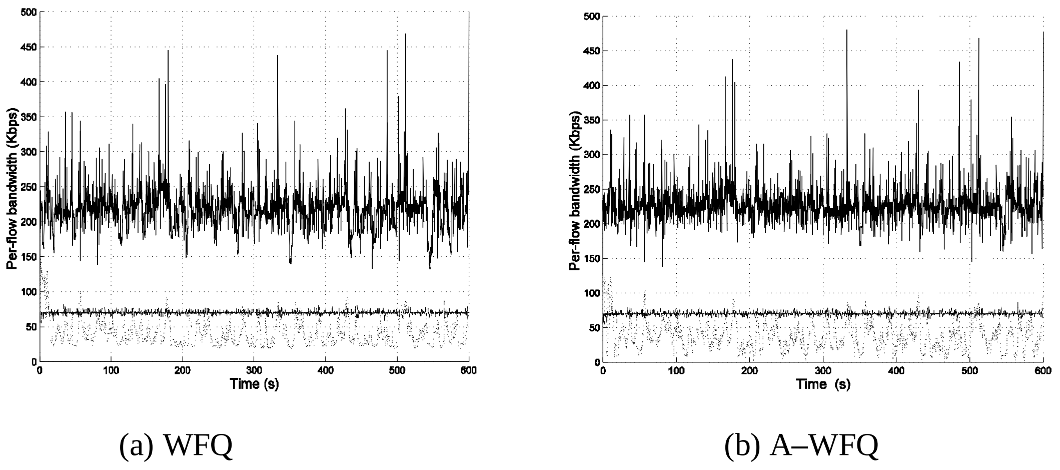
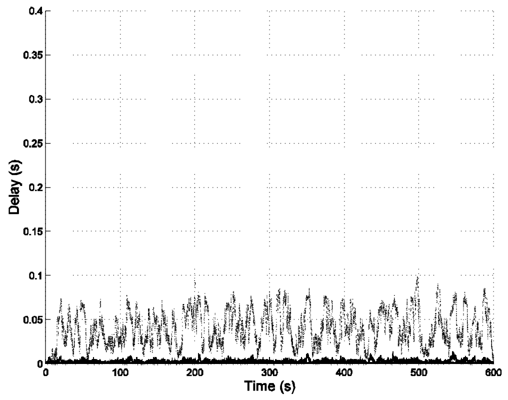


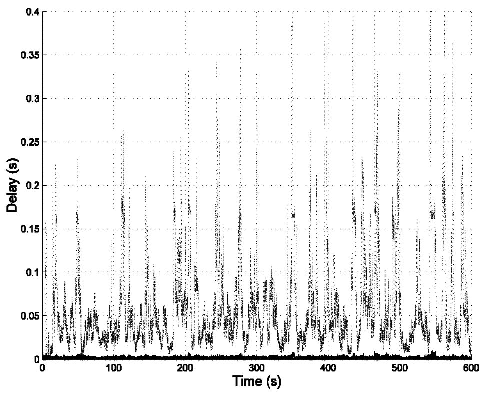
FIGURE 26 Provided bandwidth

Figure 26 provides a comparison between bandwidths when different approaches are used. It plots the mean bandwidth provided for one flow in each service class versus the simulation time. At the first glance, there is not much difference between these figures but it is possible to notice that the Gold class has a slightly bigger bandwidth when the adaptive WFQ is used and the Bronze class has bigger bandwidth when the non-adaptive WFQ is in effect. That is because the non-adaptive WFQ always allocates a fixed portion of resources for service classes. However, regardless of the approach, the bandwidth, which is provided for the Gold and the Bronze service classes, is bigger than their thresholds. The behavior of the Silver class is the same in both cases: because of its CBR nature, it fluctuates near the threshold.

Figure 27 shows the queuing delay experienced by the packets during the simulation. It plots the packet delay versus a packet's arrival time. As can be noticed, the behaviors of the Gold and the Silver class are almost identical, all packets have a low queuing delay in both cases. On the contrary, the delay of packets that belong to the Bronze class differs. The adaptive WFQ is likely to provide a bigger delay compared to the non-adaptive approach. It results in peaks that sometimes violate the delay threshold. However, there are not many such peaks and they do not significantly exceed the delay threshold.



(a) WFQ



(b) A–WFQ

FIGURE 27 Queuing delay

5 CONCLUSIONS AND FUTURE STUDIES

This dissertation has presented a scheduling model that optimizes the revenue and bandwidth of the network. The proposed algorithms ensure more bandwidth for the users paying more for the connection (i.e. higher service class) than those paying less. A good rate allocation mechanism should not only be fair, but should also allocate the available bandwidth in such a way that the overall utility of the users is maximized.

The adaptive models presented here are capable of ensuring the QoS guarantees of several service classes with various requirements and pricing schemes. As considered theoretically in Chapter 3 and presented in the simulation results in Chapter 4, the models can be applied easily to the QoS frameworks.

An adaptive WFQ algorithm, dynamically adjusts weights in such a manner that the QoS requirements of the different traffic classes can be met and the network operator's revenue can be kept as high as possible. The experiments demonstrated this property, while still allocating delays in a fair way.

A designed QoS -aware scheduling and pricing model takes into account the user's satisfaction (price vs. received QoS) and the optimal use of the limited network resources. The presented solution gives the service provider and consumers a new way to use and obtain services from the networks.

It has been demonstrated that the presented models are capable of increasing a provider's revenue compared to the non-adaptive approach. The large sets of simulations have shown that the proposed models try to assign bigger weights for the expensive service classes while ensuring the QoS requirements. Attention was paid on how values of weights impact the state of queues at the dispatching nodes. The analysis of distribution of dropped packets showed that the proposed model fairly allocates limited resources between service classes and packet drops were caused by the queue size. The proposed models lack certain features are important for a sophisticated allocation of resources. Currently, it does not take into account the state of queues and is not capable of guaranteeing the minimum queuing delay, which is important for real-time audio and video services. It is the task for future research to extend these models - modify the target function and include additional constraints that can provide such characteristics. Besides, no mechanism that can limit weights and to prevent them from begin maximized were considered. It is also the subject for future research to choose the interval in which weights should be updated. The possibility of implementation of the proposed model under the Linux based switch is also under the consideration.

REFERENCES

[1] E. Altman, T. Basar, T. Jiminez, and N. Shimkin, "Competitive routing in networks with polynomial costs", IEEE Transactions on Automatic Control, Vol. 47 Issue: 1, pp. 92—96, January 2002.

[2] E. Altman, T. Basar and R. Srikant, "Nash equilibria for combined flow control and routing in networks: Asymptotic behavior for a large number of users", IEEE Transactions on Automatic Control, Vol. 47 Issue: 6, pp. 917-930, June 2002.

[3] J. Altmann, B. Rupp, and P. Varaiya, "Internet user reactions to usage-based pricing", In *Internet Economics Workshop (IEW'99)*, May 1999.

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

[5] J. C. R. Bennett and H. Zhang, "WF2Q: Worst-case Fair Weighted Fair Queueing", Proc. of INFOCOM 1996.

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

[7] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview", IETF RFC 1633, June 1994.

[8] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jasmin, "Resource reservation protocol (RSVP) – version 1 functional specification", IETF RFC 2205, September 1997.

[9] H. M. Chaskar and U. Madhow, "Fair Scheduling with Tunable Latency: A Round Robin Approach", Proc. of IEEE Globecom 1999, pp.1328-1333, 1999.

[10] Chipalkatti, J. Kurose, Townsley, "Scheduling policies for Real-Time and non Real-Time traffic in a statistical multiplexer", *Proceedings of IEEE INFOCOM*, pp 774-783, April 1989.

[11] C. Courcoubetis, F. Kelly, and R. Weber, "Measurement-based usage charges in communications networks", *Operations and Research*, 48(4):535–548, 2000.

[12] R. Cruz, "A calculus for network delay, Part I: Network elements in isolation", *IEEE Transaction on Information Theory*, vol. 37, no. 1, pp. 114–131, January 1991.

[13] A. Demers, S. Keshav, and S. Shenkar, ”Analysis and simulation of a fair queueing algorithm”, *Internetworking: Research and Experience*, pp. 3–26, September 1990.

[14] Extreme Networks, “Building Scalable Metro Ethernet Networks”, White Paper, 1184 0 2/2006.

[15] S. Floyd and V. Jacobson, ”Link-sharing and Resource Management Models for Packet Networks”, *IEEE/ACM Transactions on Networking*, 3(4):365–386, August 1995.

[16] E. Fulp, M. Ott, D. Reininger, and D. Reeves, “Paying for QoS: an optimal distributed algorithm for pricing network resources”, In *Sixth International Workshop on Quality of Service (IWQoS’98)*, pp. 75–84, 1998.

[17] E. Fulp and D. Reeves, “Optimal provisioning and pricing of differentiated services using QoS class promotion”, In *GI Jahrestagung (1)*, pp. 144–150, 2001.

[18] R. J. Gibbens and F. P. Kelly, "Resource pricing and the evolution of congestion control". Automatica, vol. 35, no. 12, pp. 1969–1985, 1999.

[19] S.J. Golestani, “A SelfClocked fair queueing scheme for broadband applications”, *Proceedings of IEEE INFOCOM '94*, pp 636-646, June 1994.

[20] E. Hahne, “Round Robin scheduling for fair flow control”, PhD thesis, MIT, Cambridge, December 1986.

[21] J. Heinänen, F. Baker, W. Weiss, J. Wroclawski, “Assured Forwarding PHB Group”, RFC 2597, June 1999.

[22] M-F. Horng et al, ”An adaptive approach to Weighted Fair Queue with QoS enhanced on IP network”, *IEEE Catalogue No. 01CH37239*, pp. 181–186, 2001.

[23] T. Hämäläinen and J. Joutsensalo, “Link allocation and revenue optimization for future networks”, In *IEEE Globecom 2002*, No. 1350, November 2002.

[24] T. Hämäläinen, J. Siltanen, A. Viinikainen, and J. Joutsensalo: ” Adaptive Tuning of Scheduling Parameters”, WSEAS Transactions on Communications, 2003,

[25] Y. Ito, S. Tasaka, and Y. Ishibashi, “Variably weighted round robin for core ip networks”, *Performance, Computing, and Communications Conference*, pp. 159–166, 2002.

[26] J. Joutsensalo and T. Hämäläinen, "Optimal link allocation and revenue maximization", *Journal of Communications and Networks*, 4(2):136–147, June 2002.

[27] J. Joutsensalo and T. Hämäläinen, "Qos-aware adaptive pricing for network services", In *IEEE Globecom 2001*, pp. 2455–2459, November 2001.

[28] J. Joutsensalo, T. Hämäläinen, M. Pääkkönen, and A. Sayenko, "Adaptive Weighted Fair Scheduling Method for Channel Allocation", Proc. of IEEE ICC 2003, Anchorage, Alaska, 2003.

[29] J. Joutsensalo, T. Hämäläinen, M. Pääkkönen, and A. Sayenko, "QoS and revenue aware adaptive scheduling algorithm", *Journal of Communications and Networks*, vol. 6, no. 1, pp. 68–77, March 2004.

[30] J. Joutsensalo, T. Hämäläinen, and J. Zhang, "Resource allocation for differentiated QoS by adaptive weighted fair queue technique", In *16th Nordic Teletraffic Seminar*, pp. 291–302, August 2002.

[31] J. Joutsensalo, T. Hämäläinen, and J. Zhang, "Revenue maximization-based adaptive WFQ", In *Proceedings of APOC 2002*, October 2002.

[32] V. Jacobson, K. Nichols, K. Poduri, "An Expedited Forwarding PHB," RFC 2598, June 1999.

[33] M. Katevenis, S. Sidoropoulos and C. Courcoubetis, "Weighted Round Robin Cell Multiplexing in a General Purpose ATM Switch Chip," IEEE Journal on Selected Areas in Communications, vol. 9, pp. 1265-1279, Oct 1991.

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

[35] 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.

[36] F. Kelly, "On tariffs, policing and admission control for multiservice networks", *Operations Research Letters 15*, pp. 1–9, 1994.

[37] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability". Oper. Res. Soc., vol. 49, pp. 237–252, 1998.

[38] Jong-Seon Kim; Lee, D.C., "Weighted round robin packet scheduler using relative service share", Proc. of IEEE MILCOM 2001, Vol.2, pp. 988-992, 2001.

[39] L. Kleinrock. *Queueing systems*. John Wiley & Sons, 1975.

[40] Y. Korilis, A. Lazar, and A. Orda, "Architecting noncooperative networks", IEEE Journal on Selected Areas in Communications, vol. 13, pp. 1241-1251, 1995.

[41] Y. A. Korilis, A. Lazar, and A. Orda, "Achieving network optima using Stackelberg routing strategies", IEEE/ACM Transactions on Networking, vol. 5, pp. 161-173, February 1997.

[42] S. Kunniyur and R. Srikant, "End-to-end congestion control: Utility functions, random losses and ECN marks", Proc. of IEEE INFOCOM 2000, Tel Aviv, Israel, pp. 1323–1332, 2000.

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

[44] R. J. La and V. Anantharam, "Optimal routing control: a game theoretic approach", Proc. of the 36th IEEE Conference on Decision and Control, San Diego, CA, December 1997.

[45] Paolo Losi, "Implementation of the WFQ discipline for the NS-2 simulator", http://www.pao.lombardiacom.it/wfq.html, 1997.

[46]S. H. Low, "Equilibrium bandwidth and buffer allocations for elastic traffics", IEEE/ACM Transactions Networking, vol. 8, pp. 373-383, June 2000.

[47] J. K. MacKie-Mason and H. R. Varian, "Pricing the Internet, in Public Access to the Internet", B. Kahin and J. Keller, Eds. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[48] E. Magaña, D. Morató, and P. Varaiya, "Router scheduling configuration based on the maximization of benefit and carried best effort traffic", *Telecommunication Systems*, vol. 23, no. 2-4, pp. 275–292, October-December 2003.

[49] J. M. Mao, W. M. Moh, and B. Wei, "PQWRR Scheduling Algorithm in Supporting of DiffServ", In Proc. of ICC2001, June 2001.

[50] N. Matsufuru and R. Aibara, "Efficient Fair Queuing for ATM Networks using Uniform Round Robin", Inforcom'99, pp. 389-397, 1999.

[51] B. Moore, E. Elleson, E. Strassner, and A. Westerinen, "Policy core information model – version 1 specification", IETF RFC 3060, February 2001.

[52] Nagle, "On Packet Switches with Infinite Storage", *IEEE Trans. on Communications*, 35(4), pp .435-438, April 1987.

[53] A. Orda, R. Rom, and N. Shimkin, "Competitive routing in multiuser communication networks", IEEE/ACM Transactions on Networking, vol. 1(5), pp. 510-521, October 1993.

[54] A.K. Parekh and R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case", *IEEE/ACM Transactions on Networking,* 1(3):344–357, June 1993.

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

[56] I. Ch. Paschalidis and Y. Liu, "Pricing in multiservice loss networks: static pricing, asymptotic optimality and demand substitution effects", *IEEE/ACM Transactions on Networking,* 10(3):425–438, June 2002.

[57] I. Ch. Paschalidis and John N. Tsitsiklis, "Congestion-dependent pricing of network services", *IEEE/ACM Transactions on Networking,* 8(2):171–184, April 2000.

[58] P. Reichl, S. Leinen, and B. Stiller, "A practical review of pricing and cost recovery for internet services", In *Internet Economics Workshop (IEW'99),* May 1999.

[59] F. Risso, "Quality of Service on Packet Switched Networks", pp. 14–22, January2000.

[60] A. Sayenko, J. Joutsensalo, T. Hämäläinen, and J. Siltanen: "An adaptive Approach to WFQ with the Revenue Criterion", Proceedings of Eighth IEEE International Symposium on Computers and Communication, (ISCC 2003), Turkey, pp. 181 –186, June 2003.

[61] A. Sayenko, T. Hämäläinen, J. Joutsensalo, and J. Siltanen: "The simulation and analysis of the revenue criterion based adaptive WFQ". Proceedings of 18th IEEE International Teletraffic Conference (ITC'18), Germany, August 2003.

[62] A. Sayenko, T. Hämäläinen, J. Joutsensalo, and J. Siltanen, "On providing bandwidth and delay guarantees using the revenue criterion based adaptive WFQ", Proceedings of the 9th IEEE Asia-Pacific Conference on Communications (APCC), Penang, Malaysia, September 2003.

[63] C. Semeria, "Supporting Differentiated Service Classes: Queue Scheduling Disciplines", White Paper 200020-001, pp. 5–18, December 2001.

[64] M. Shreedhar and G. Varghese, "Efficient Fair Queuing Using Deficit Round Robin", Proc. ACM SIGCOMM '95, pp. 231-242, September 1995.

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

[66] D. Stidialis and A. Varma, "Efficient Fair Queuing Algorithms for Packet-Switched Networks", IEEE/ACM Transactions on Networking, Vol. 6, No. 2, pp. 175-185, 1998.

[67] D. Stiliadis and A. Varma, "Latency-rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms", Proc. IEEE Infocom'96, pp. 111-119, May 1996.

[68] D. Stiliadis and A. Varma, "Rate-proportional servers: A design methodology for fair queuing algorithms", IEEE/ACM Transactions on Networking, Vol.6, No. 2, pp. 164-174, 1998.

[69] P. Thie, "An Introduction to Linear Programming and Game Theory", John Wiley & Sons, New York, second edition, 1988.

[70]UCB/LBNL/VINT. Network simulator ns-2, http://www.isi.edu/nsnam/ns, 1997.

[71] X. Wang and H. Schulzrinne, "Pricing network resources for adaptive applications in a differentiated services network", In *IEEE INFOCOM*, pp. 943–952, 2001.

[72] H. Wang, C. Shen, and Shin. K. G. "Adaptive-weighted packet scheduling for premium service", In *IEEE International Conference on Communications (ICC 2001)*, volume 6, pp. 1846–1850, 2001.

[73] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks", *Proceeding of IEEE*, vol. 83, no. 10, pp. 1374–1396, October 1995.