JYVÄSKYLÄ STUDIES IN COMPUTING
65

Ari Viinikainen

Quality of Service and Pricing in Future Multiple Service Class Networks

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 65

Ari Viinikainen

Quality of Service and Pricing in Future Multiple Service Class Networks

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Villa Ranan Blomstedtin salissa
joulukuun 1. päivänä 2006 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in the Building Villa Rana, Blomstedt Hall, on December 1, 2006 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

Quality of Service and Pricing in Future Multiple Service Class Networks

JYVÄSKYLÄ STUDIES IN COMPUTING 65

Ari Viinikainen

Quality of Service and Pricing in Future Multiple Service Class Networks

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

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

URN:ISBN:951392694X
ISBN 951-39-2694-X (PDF)

ISBN 951-39-2627-3 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2006

ABSTRACT

Viinikainen, Ari
Quality of Service and Pricing in Future Multiple Service Class Networks
Jyväskylä: University of Jyväskylä, 2006, 196 p.
(Jyväskylä Studies in Computing
ISSN 1456-5390; 65)
ISBN 951-39-2694-X
Finnish summary
Diss.

Nowadays, the pricing of the Internet is typically based on a flat rate and the users can use the network's resources as much as their connection allows. This leads to congestion of the network: for example, no guarantee can be given for example for real time applications to have the network's resources they need. Pricing that is related to the resource allocation of the network can be used to efficiently control the network's resources and combat congestion. In today's networks the mobile devices must also be taken into consideration, complicating the scheme of providing adequate Quality of Service (QoS) to all customers. Especially, the movement of mobile devices from one networks attachment point to another causes unwanted delays that should be minimized. In this thesis pricing and packet scheduling is combined to maximize the operator's revenue and to guarantee appropriate service for different service classes. Also a new method for minimizing the handover delay in Mobile IPv6 handover situation is proposed. The method uses pricing and service classes to determine the connections that are privileged to make use of the proposed fast handover method.

Keywords: Pricing, revenue optimization, delay minimization, bandwidth allocation, Quality of Service (QoS), fast handover, Mobile IPv6.

Author	Ari Viinikainen Department of Mathematical Information Technology University of Jyväskylä Finland
Supervisor	Professor Jyrki Joutsensalo Department of Mathematical Information Technology University of Jyväskylä Finland
Supervisor	Professor Timo Hämaläinen Department of Mathematical Information Technology University of Jyväskylä Finland
Reviewers	Professor Tu Guofang School of Information Science and Engineering Graduate School of the Chinese Academy of Sciences China
	Professor János Sztrik Department of Informatics Systems and Networks Faculty of Informatics University of Debrecen Hungary
Opponent	Professor Pertti Raatikainen VTT Telecommunications Networks Finland

ACKNOWLEDGEMENTS

I would like to thank my supervisors Prof. Timo Hämaläinen and Prof. Jyrki Joutsensalo for their valuable guidance and support during the entire course of this thesis. Also, I would like to thank all my colleagues at the Department of Mathematical Information Technology in the University of Jyväskylä and others that have contributed to my publications for their support.

For financial support I am grateful to the Department of Mathematical Information Technology at the University of Jyväskylä and the Ellen and Artturi Nyyssönen Foundation.

Finally, many thanks to my wife Heta and daughters Emmi and Sanni for their never-ending support and love.

Jyväskylä, November 16, 2006.

Ari Viinikainen

LIST OF PUBLICATIONS AND CONTRIBUTIONS OF THE AUTHOR

In the co-authored publications, all the authors have contributed in the writing of the articles. The contribution of the author of this thesis and a review of the articles is presented in this section.

Delay was studied as a QoS parameter in **PI** when the weights of a scheduler were derived using network provider's revenue as a target function. The pricing of the different service classes was based on linear pricing functions and a single node was considered. A Weighted Fair Queueing based scheduling algorithm is proposed in **PII** which uses a usage-based revenue criterion to allow the allocation of free resources between different service classes. In publications **PI** and **PII** the author of this thesis has participated in the analysis of the simulation results and on the discussion of the future research ideas.

Flat pricing functions were used for providing delay guarantees to four different service classes in **PIII**. The revenue of the service provider is maximized with the optimal weights. The gain factors of the flat pricing functions (i.e. the actual price) can be used to adjust the relative delays of the different service classes or absolute guarantees can be provided with a Call Admission Control (CAC) mechanism. In publication **PIII** the author has modified the original algorithm proposed by the second author to take into account variable average packet lengths in different traffic classes.

Bandwidth allocation and throughput guarantee was studied in **PVIII**. The weights of a packet scheduler are updated by a formula that is derived from a revenue-based optimization problem. The updating of the weights is fast and independent from any assumptions about the statistical behavior of the connections. In publication **PVIII** the algorithm were mainly developed by the first and second authors and the author of this thesis solely derived the addition of the bandwidth guarantee to the algorithm's Call Admission Control (CAC) mechanism. In publications **PIII** and **PVIII** experiments and their analysis was mainly done by the author of this thesis. The simulations in publications **PIII** and **PVIII** were implemented and ran in Matlab mainly by the author of this thesis.

In publication **PIV** the algorithm of **PVIII** is implemented in a network simulator environment ns-2 by the first and second authors. The simulations with UDP traffic were run by the first author and the simulation results were analyzed mainly by the first and second authors. The network simulator and simulations were further extended to study TCP based traffic in publication **PVI**, where the contributions of the authors is similar to publication **PIV**.

The algorithms are extended to multiple nodes in publications **PV** and **PVII**. In publication **PV** we presented a pricing and scheduling algorithm that minimizes the weighted mean delay in a multiple node network while maximizing the service provider's revenue. In publication **PVII** we presented a bandwidth allocation method for multiple nodes that offers better bandwidth for customers that pay more for the connection and on the same time the revenue is maximized. The simulations were done in Matlab environment by the author of this thesis and most of the result analyzing was done by the three first authors. The algorithms of **PV** and **PVII** were developed by the first author with the help from the other authors. The implementation of the algorithm in **PVII** to Matlab environment was made by the second and third authors. The simulations were run by the third author and the results were analyzed by the three first authors.

In publication **PIX** the author of this thesis has independently proposed a new pricing and scheduling method for minimizing delay and maximizing revenue. In this case the pricing of a connection is based on *the delay that it inflicts to other connections* contrary to previous publications where pricing of a connection was based on *the delay or bandwidth that it experiences by itself*.

In today's networks the number of mobile devices is very high which complicates the provision of Quality of Service (QoS) to customers. The movement of mobile devices, particularly, causes unwanted delays that should be minimized. The rest of the publications (**PX** - **PXVI**) are focused on minimizing delay in handover situations when using the Mobile IPv6 protocol.

A new method to decrease the handover delay in Mobile IPv6 networks is proposed in publication **PX**. The idea was introduced in a Master's thesis written by the first author and supervised by the second author. In publication **PXI** the theoretical analysis of the method was compared to simulation studies using constant bit rate (CBR) traffic. The simulation setup was made by the first author and the results analyzed mainly by the first two authors. The proposed method was then implemented in a real network environment using Mobile IPv6 for Linux (MIPL) and the results were presented in publication **PXII**. The implementation was mainly done by the second and third authors, with help from the first and fourth authors and the results were analyzed mainly by the first four authors.

In publication **PXIII** we summarized our experiments and compared several handover methods in theory, simulation and real network environment. The author of this thesis presented an improvement to the proposed method to minimize also the *upstream traffic* as the original idea was only applicable to *downstream traffic*. The upstream extension was simulated in ns-2 environment and was shown to outperform other methods in publication **PXIV**, where the idea of the author of this thesis was defined in detail with the help of other authors. The implementation and simulations were done by the second author and the results were mainly analyzed by the first and third authors. In publication **PXV** the proposed fast handover method with the fast upstream addition is compared against other methods in theory and simulations. Also, security issues of the method are considered and solutions proposed. The addition of fast upstream to the method was done by the author of this thesis, and all the authors took part in analyzing all the results.

PI T. Hämäläinen, K. Kaario, J. Joutsensalo and A. Viinikainen, "Traffic Optimization with Iterative Weighted Scheduling Method", WSEAS Transactions on Communications, Issue 4, Volume 2, October 2003, ISSN 1109-2742, pp. 404-409.

PII T. Hämäläinen, J. Siltanen, A. Viinikainen and J. Joutsensalo, "Adaptive Tuning of Scheduling Parameters", WSEAS Transactions on Computers, Issue 1, Volume 3, January 2004, ISSN 1109-2750, pp. 75-78.

PIII A. Viinikainen, J. Joutsensalo, M. Pääkkönen and T. Hämäläinen, "Packet Scheduling Algorithm with Weight Optimization", Proceedings of the International Conference on E-business and Telecommunication Networks (ICETE'04), Setùbal, Portugal, August 24-28, 2004, pp. 127-133.

PIV L. Kannisto, A. Viinikainen, J. Joutsensalo and T. Hämäläinen, "Optimized pricing and closed form algorithm for WFQ scheduling", Proceedings of the IEEE TENCON 2005, Melbourne, Australia, Nov. 21 - 24, 2005, pp. 1290 - 1295.

PV J. Joutsensalo, A. Viinikainen, L. Kannisto and T. Hämäläinen, "Packet Scheduling with Revenue Optimization and Weighted Delay Minimization", Proceedings of the IEEE GLOBECOM 2005, St. Louis, USA, Nov. 28 - Dec. 2, 2005, Volume 1, pp. 513-517.

PVI L. Kannisto, A. Viinikainen, J. Joutsensalo and T. Hämäläinen, "Adaptive Algorithm for Revenue Maximization in WFQ Scheduler", Proceedings of the IEEE 20th International Conference on Advanced Information Networking and Applications (AINA2006), Vienna, Austria, April 18 - 20, 2006, Volume 1, pp. 339 - 346.

PVII J. Joutsensalo, A. Viinikainen, M. Wikström and T. Hämäläinen, "Bandwidth allocation and pricing in multimode network", Proceedings of the IEEE 20th International Conference on Advanced Information Networking and Applications (AINA2006), Vienna, Austria, April 18 - 20, 2006, Volume 1, pp. 573-578.

PVIII J. Joutsensalo, A. Viinikainen, T. Hämäläinen and M. Wikström , "Pricing Based Adaptive Scheduling Method for Bandwidth Allocation", ***in press*** International Journal of Electronics and Communications, *to appear*.

PIX A. Viinikainen, "Packet Scheduling and Pricing Based on Inflicted Delay", Proceedings of the International Conference on e-Business and Telecommunication Networks (ICETE'06), Setùbal, Portugal, August 7 - 10, 2006.

PX M. Sulander, T. Hämäläinen, A. Viinikainen and J. Puttonen, "Flow-Based Fast Handover Method for Mobile IPv6 Network", Proceedings of the IEEE 59th Vehicular Technology Conference Spring 2004 (VTC'S04), Milan, Italy, May 17-19, 2004, Volume 5, pp. 2447 - 2451.

PXI J. Puttonen, A. Viinikainen, M. Sulander and T. Hämäläinen, "Performance Evaluation of the Flow-based Fast Handover Method for Mobile IPv6 Network", Proceedings of the IEEE 60th Vehicular Technology Conference Fall 2004 (VTC'F04), Los Angeles, USA, September 26-29, 2004, Volume 6, pp. 4437 - 4441.

PXII J. Puttonen, H. Suutarinen, T. Ylönen, A. Viinikainen, M. Sulander and T. Hämäläinen, "Flow-based Fast Handover Performance Analysis in Mobile IPv6 for Linux Environment", Proceedings of the IEEE 61st Semiannual Vehicular Technology Conference Spring 2005 (VTC'S05), Stockholm, Sweden, May 30 - June 1, 2005, Volume 4, pp. 2575 - 2579.

PXIII J. Puttonen, A. Viinikainen, M. Sulander and T. Hämäläinen, "A Fast Handover Method for Mobile IPv6 Networks", Journal of Internet Technology, Special Issue on Heterogeneous IP Networks, No.3, Volume 6, July 2005, ISSN 1607-9264, pp. 305 - 312.

PXIV A. Viinikainen, S. Kaščák, J. Puttonen and M. Sulander, "Fast Handover for Upstream Traffic in Mobile IPv6", Proceedings of the IEEE 62nd Semiannual Vehicular Technology Conference Fall 2005 (VTC'F05), Dallas, Texas, USA, September 25-28, 2005, Volume 2, pp. 802 - 806.

PXV A. Viinikainen, J. Puttonen, M. Sulander, T. Hämäläinen, T. Ylönen and H. Suutarinen, "Flow-based Fast Handover Method for Mobile IPv6 Environment - Implementation and Analysis", Computer Communications, Volume 29, Issue 16, More Than a Protocol for Next Generation Internet, 12 October 2006, Pages 3051-3065.

ABBREVIATIONS

Aliquem	Active List Queue Method
ATM	Asynchronous Transfer Mode
AR	Access Router
BR	bit-by-bit round-robin (a scheduling algorithm)
BSFQ	Bin Sorted Fair Queueing
BU	Binding Update
DRR	Deficit Round Robin
CAC	Call Admission Control
CBR	Constant Bit Rate
CN	Corresponding Node
CR	Crossover Router
CoA	Care-of-Address
CSFQ	Core-Stateless Fair Queueing
ERR	Elastic Round Robin
FFQ	Frame-based Fair Queueing
FIFO	First-In-First-Out
FMIPv6	Fast Handovers for Mobile IPv6
FQS	Fair Queueing based on Start-time
FRR	Fair Round Robin
FTP	File Transfer Protocol
GPS	Generalized Processor Sharing
HA	Home Agent
HMIPv6	Hierarchical Mobile IPv6
HoA	Home Address
HofA	Hand-of-Address
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

LFVC	Leap Forward Virtual Clock
MAC	Media Access Control
MAP	Mobility Anchor Point
MIPL	Mobile IPv6 for Linux
MIPv6	Mobility support in IPv6
MN	Mobile Node
PDRR	Pre-order Deficit Round Robin
PGPS	Packet by packet Generalized Processor Sharing
QoS	Quality of Service
RA	Router Advertisement
RR	Return Routability
RS	Router Solicitation
S-MIP	Seamless handoff architecture
SCFQ	Self-clocked Fair Queueing
SFQ	Start-time Fair Queueing
SPFQ	Starting Potential-based Fair Queueing
SRR	Smooth Round Robin
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VC	Virtual Clock
VoIP	Voice over Internet Protocol
WFQ	Weighted Fair Queueing
WF^{2}Q	Worst-case Fair Weighted Fair Queueing
WF^{2}Q+	A low complexity modification of WF^{2}Q
WSS	Weight Spread Sequence

LIST OF SYMBOLS

b_{ij} Packet size of connection j in class i

d_i Overall delay in class i

d_{ij} Delay of connection j in class i

δt_{ij} The time which passes when data is transferred through the queue j to the output in the node i

f A flow, which is a stream of packets using the same route from source to destination

g_i Minimum service rate of a Generalized Processor Sharing server

i_f^j Index of a bin where an arriving packet is stored using Bin Sorted Fair Queueing scheduler

l_f^j The length of p_f^j

l_f^{max} The length of the largest packet of flow f

p_f^j The j^{th} packet of flow f

ϕ_i Reserved service rate (weight) of a session or flow i

r_i Price that is paid in class i

t Time

$v(t)$ Virtual time in Weighted Fair Queueing and its modifications

v_{ts}^{WFQ} Virtual time stamp for sorting the packets e.g. in WFQ scheduler

w_i Weight of class i

w_{ij} Weight of node i and class j in a multiple node case

$A(p_f^j)$ The arrival time of packet p_f^j

$A_f(r)$ The amount of bytes allowed by flow f in round r of Elastic Round Robin scheduler

B_{ij} Bit rate of connection j in class i

B_{min} The smallest unit of bandwidth that can be allocated to a flow

$B(t)$ A set of backlogged flows at time t in a bit-by-bit round-robin server

C Capacity of a server or link

DC_f Deficit Counter of flow f in Deficit Round Robin scheduler

Δ Length of a bin in Bin Sorted Fair Queueing scheduler

$E(b_i)$ Average packet size of the connections in the class i

F_k Flow class for stratifying the flows in Stratified Round Robin scheduler.

$F(p_f^j)$ Finish tag of packet p_f^j in priority based schedulers

$H(f, m)$ Fairness measure of flows f and m

$L(p_f^j)$ Departure time of packet p_f^j

N Number of flows

N_i Number of connections in the queue of class i

N_{ij} Number of connections in the node i and queue j in a multiple node case

$N_j^{i_1 \to i_2}$ The number of connections in the j^{th} service class which transfer data packets from node i_1 to i_2 in a multiple node case

P_i A pricing function for class i

Pq_k The kth priority queue used for pre-ordering in Pre-order Deficit Round Robin scheduler

Q_f Quantum of flow f in Deficit Round Robin scheduler

R Total revenue

$R(t)$ Number of rounds made in the round-robin service discipline up to time t

S The set of flows served by a server

$S_i(t_1, t_2)$ Quantity of session (flow) i traffic served in the time interval $[t_1, t_2]$

$S(p_f^j)$ Start tag of packet p_f^j in priority based schedulers

$SC_f(r)$ Surplus Count of flow f in round r of Elastic Round Robin scheduler

$SC_{max}(r)$ Maximum Surplus Count of all flows in round r of Elastic Round Robin scheduler

$Sent_f(r)$ The amount of bytes send by flow f in round r of Elastic Round Robin scheduler

Θ_{WFQ} Delay bound of e.g. WFQ scheduler

$W_{f,s}(t_1, t_2)$ The amount of data of flow f sent during time period $[t_1, t_2]$ by a scheduler s.

Z Number of priority queues used for pre-ordering in Pre-order Deficit Round Robin scheduler

CONTENTS

1 INTRODUCTION

The future of the Internet is endangered by the fact that there are no proper models of pricing that can also be used to control the use of the network's resources. Nowadays, pricing is typically based on a flat rate and the excess load of congested periods is billed by the usage. These kinds of coarse pricing schemes do not fully respond to the quickly changing user demands. Also, the users have only little motivation to use emerging differentiated classes of service. The distributed nature of the Internet provides a great challenge for researchers to develop a pricing mechanism that guarantees user and operator satisfaction by allocating the network's resources fairly.

Today the trend is towards connections being wireless and users moving around with their mobile devices. For delay critical applications mobility is a great issue. Specially the movement of mobile devices from one networks attachment point to another causes unwanted delays that should be minimized. In the third generation (3G) systems and beyond, Internet Protocol (IP) technology will play a key role in providing seamless services to users at all times and in all places. In future, IP version 6 (IPv6) will replace version 4 (IPv4) because of its better support for mobility.

In wireless networks users move and mobility must be supported. If the customers are paying for their service that depends on some QoS parameters, then this QoS should be maintained also in handover situations.

In this thesis several pricing and packet scheduling schemes are presented and a new fast handover method for Mobile IPv6 suggested. Also It is proposed that handover decisions in Mobile IPv6 should be prioritized according to service class, so that in situations where all handovers cannot be handled customers of higher service classes get the service.

1.1 Structure of this Thesis

This thesis concentrates on combining pricing, resource allocation and revenue optimization while providing adequate Quality of Service in future wired and wireless networks.

Different approaches to pricing of network services is reviewed in the first section of Chapter 2. The next three sections in Chapter 2 present the most common priority and frame-based scheduling methods, which do not consider pricing, for efficient scheduling of packets. In the last section of Chapter 2 a review of the research of publications **PI-PIX** is presented. In these publications, where pricing and packet scheduling has been combined to provide service according to customers service class and at the same time maximizing the service provider's revenue.

In Chapter 3 handover delay in future Mobile IPv6 networks is addressed. First the handover procedure of Mobile IPv6 is presented and its disadvantages pointed out. Then two most common proposals for decreasing the handover delay of MIPv6 are briefly presented and a review of numerous other proposals is given. In the last section a review of the results obtained in publications **PX-PXV** is done and the proposed fast handover method with its fast upstream enhancement is presented.

Finally, Chapter 4 summarizes the most important research results and discusses some ideas for future work.

2 SCHEDULING ALGORITHMS AND PRICING

2.1 Pricing in multiple service class networks

Today, telecommunication networks offer heterogeneous services but the pricing of the services is homogenous. There have been many proposals on how to price multiple service class networks. Here I review some important pricing models for multiple service class networks which have been proposed by various researchers.

Nowadays the commonly used method for pricing is flat pricing, where users pay a flat fee to gain unlimited access for some fixed period of time. However, flat pricing is not optimal although it is simple to implement. Flat pricing leads to users consuming as much bandwidth as possible and thus the network will get congested. Also, with flat pricing there is no possibility for different priority applications and the quality of service is just on the level of best effort. To overcome these problems, several pricing models have been studied. The auction approach requires users to attach bids to each packet. These packets are transmitted if the bid is above the market clearing price, which is defined from the congestion constraints of the network. With static and dynamic priority pricing models, the users have a variety of applications with different priority that are priced accordingly. The Paris Metro Pricing approach divides the network into subsections and charges different prices for these although the subsections themselves have same resources.

The interaction of pricing policies with quality of service in multiple service class networks was studied in [8], where it was argued that in order to have benefit on maximizing network performance with any multiple class service discipline, some form of pricing, related to the service classes, is required. The authors showed that customers are more satisfied when the pricing combines cost and network's performance objectives, than when there is the same pricing for all services.

An interesting pricing proposal for best-effort traffic is proposed in [43] where it is argued that congestion of the Internet will need to be controlled with user-based pricing and where the authors propose a usage-sensitive price to be charged when the network is congested. The user is capable of defining, how much s/he is willing to pay for the delivery of the packet. During the times of congestion the network will drop packets with low willingness (i.e. price) so that the congestion situation can be handled. This scheme guarantees priority, but it is only relative and no actual guarantee can be offered for e.g. delay. An important model based on the integrated services approach is formulated in [65]. The model includes both soft and hard guaranteed services for the network model.

An optimal control model to derive the pricing policy is formulated in [78] for ATM-integrated-services networks. In the model the demand elasticity for the service, as well as the required performance objectives and traffic pattern of each service, determine the optimal price of each service. The developed pricing schedule is also time-varying as the demand of network services usually changes during the day. The use of cost functions as a basis for defining scheduling and dropping algorithms is defined in [61] where different applications are considered. For packet switched networks these algorithms effectively support traffic with diverse performance objectives. A different approach is taken in [66] where the authors critique academic literature for having largely focused on researching optimal pricing policies and propose that research should concentrate more on structural and architectural issues.

How the user could dynamically control the price in a congestion based pricing network is discussed in [10]. The authors propose a specific controller to be installed in user terminals, which can be used to trade quality of service for price. However, this seems unpractical as these devices should be installed into each terminal. Priority-based pricing and congestion-based pricing is integrated in [21], where the services are divided into different priority classes. The price of a packet depends on the priority level of the packet and on the current network load. Congestion dependent pricing is studied in [59, 58, 57], where an optimal pricing strategy for maximizing the service provider's revenue or social welfare is determined. The authors found that suitably chosen static (congestion-independent) pricing can come very close to optimality, so that time-of-day pricing policies might be sufficient in most cases.

In [36, 37] the problem of sharing the available bandwidth between users sending bursty traffic is considered. The pricing is based on the effective bandwidth which is a function of the traffic profile of a source. The price increases with flow rate or with the share of the network consumed by a traffic stream. Another scheme for packet-based pricing for providing more efficient flow control is proposed in [16].

The idea of partitioning the network into several logically separate channels with no formal guarantee on quality of service is proposed in [54]. The channels would treat packets equally on a best effort basis and differ only in the prices paid for using them. Competition between two network service providers is analyzed in [15]. It is found that the competition leads to a situation where both service providers offer only one service class. But with more service providers this might not hold.

Another approach is a game theoretic pricing mechanism proposed in [68] that provides QoS differentiation in addition to congestion control and statistically guarantees service in packet switched networks. In [38] the optimal pricing problem is formulated as a nonlinear revenue optimization problem. The optimal prices for multiple services with guaranteed quality of service are solved with an auction based algorithm. In [44] a static pricing scheme for priority services is analyzed. It is shown that neither network revenue nor bandwidth allocation is influenced by the pricing scheme. Instead, minimal packet loss as a QoS parameter can be guaranteed by using pricing.

2.2 Packet scheduling

In the recent years packet scheduling has been extensively studied to attain good approximation of an ideal scheduler. The ideal packet scheduler would have low complexity, a bounded (small) delay for every packet and would provide every flow its fair share of the bandwidth. For fairness there exists several definitions which are shortly presented. For a more detailed discussion refer to [45].

A definition for *proportional fairness* is presented in [17]. If $W_{f,s}(t_1, t_2)$ is the amount of data of flow f sent during time period $[t_1, t_2]$ by scheduler s and ϕ_f the rate of flow f, then an allocation is fair if, for all intervals $[t_1, t_2]$ in which both flows f and m are backlogged

$$\frac{W_{f,s}(t_1,t_2)}{\phi_f} - \frac{W_{m,s}(t_1,t_2)}{\phi_m} = 0. \tag{1}$$

This assumes that flows can be served in infinitesimally divisible units, so it is an idealized definition of fairness. In [17] it is shown that if a packet scheduling algorithm guarantees that

$$\left| \frac{W_{f,s}(t_1,t_2)}{\phi_f} - \frac{W_{m,s}(t_1,t_2)}{\phi_m} \right| \leq H(f,m) \tag{2}$$

for all intervals $[t_1, t_2]$, then

$$H(f,m) \geq \frac{1}{2}\left(\frac{l_f^{max}}{\phi_f} - \frac{l_m^{max}}{\phi_m} \right), \tag{3}$$

where $H(f,m)$ is a function of the properties of flows f and m. The maximum lengths of packets of flows f and m are l_f^{max} and l_m^{max}, respectively. The function $H(f,m)$ is referred to as fairness measure.

A definition for *worst-case fairness* is defined in [2]. A scheduler s is worst-case fair to flow f if and only if the delay of a packet arriving at time t is bounded by

$$\frac{Q_{f,s}(t)}{\phi_f} + C_{f,s}, \tag{4}$$

where $Q_{f,s}(t)$ is the queue size of flow f at time t, ϕ_f is the guaranteed rate of flow f, and $C_{f,s}$ is a constant that is independent of the other flows' queues. A scheduler is said to be worst-case fair if it is worst-case fair to all flows in the system. In the case a scheduler s is worst-case fair, the fairness of the scheduler is measured by the *normalized worst-case fair index*

$$c_s = \max_f \left(\frac{\phi_f C_{f,s}}{C} \right), \tag{5}$$

where C is the total link bandwidth. Next, various scheduling algorithms presented in the academic literature are roughly categorized.

Scheduling algorithms can be basically classified into two categories: sorted priority (or time stamp) and frame-based, depending on the need for sorting the packets in the scheduler. The schedulers which are based on packet sorting (i.e. sorted priority), need to maintain a "virtual time" that is used to calculate the order of the packets. These kinds of schedulers include e.g. Weighted Fair Queueing (WFQ) [12, 56], Virtual Clock [81], Fair Queueing based on Start-time (FQS) [20], Self-clocked Fair Queueing (SCFQ) [17], Worst-case Fair Weighted Queueing (WF^{2}Q) [2], Start-time Fair Queueing (SFQ) [19], Frame-Based Fair Queueing (FFQ), and Starting Potential-based Fair Queueing (SPFQ) [70]. Generally these provide good fairness and low latency but due to the complexity in computation of the parameters used for sorting, they lack efficiency.

In the frame-based category the schedulers handle packets in a round robin manner (i.e. with low complexity), like Deficit Round Robin (DRR) [67], but they are not as good in fairness and latency. Several algorithms have been proposed for improving the latency and fairness in a round robin based schemes. These include Active List Queue Method (Aliquem) [41], Smoothed Round Robin (SRR) [22, 23], Pre-order deficit round robin [75], Nested DRR [32], Bin Sort Fair Queueing (BSFQ) [7], Elastic Round Robin (ERR) [34], Stratified Round Robin [63], and Fair Round Robin (FRR) [79]. Improvement is typically accomplished by evolving a round-robin scheme like DRR and incorporating some elements of a sorted priority scheduler.

Sorted priority based approaches have good bounded delay and fairness properties, but at the cost of relatively high complexity, $O(\log N)$, where N is the number of flows in the system. Due to the logarithmic complexity their implementation in high speed networks is problematic. On the other hand, Frame based approaches have an $O(1)$ complexity, but in general do not have good bounded delay and fairness properties. The rest of this chapter presents different scheduling algorithms discussing their advantages and disadvantages. First, the priority based scheduling schemes are discussed in 2.3 and then the frame-based scheduling schemes are discussed in 2.4.

2.3 Priority based scheduling schemes

The Generalized Processor Sharing (GPS) [56] service method is an ideal scheduling discipline that fairly allocates a predefined share of service capacity to each flow in a network. The GPS system assumes that the packets can be divided into infinitely small pieces and traffic can simultaneously be transmitted through the outgoing link at different rates by multiple sessions. Many proposals have been presented that emulate the GPS server as closely as possible in order to achieve a practically implementable algorithm.

2.3.1 Generalized Processor Sharing (GPS)

A Generalized Processor Sharing (GPS) [56] server is characterized by N positive real numbers (i.e. weights), $\phi_1, \phi_2, \ldots, \phi_N$ and it is defined to be work conserving and operating at a fixed rate C. A server is work conserving when it is busy at all times when there are packets waiting in the system. Let $S_i(t_1, t_2)$ be the quantity of session i traffic served at a time interval $[t_1, t_2]$. Now, a GPS server is defined as one for which

$$\frac{S_i(t_1,t_2)}{S_j(t_1,t_2)} \geq \frac{\phi_i}{\phi_j}, \quad j = 1, \ldots, N, \tag{6}$$

for any session i that is continuously backlogged during the time interval $[t_1, t_2]$.

By summing over all sessions j

$$S_i(t_1,t_2)\sum_{j=1}^{N}\phi_j \geq (t_2 - t_1)C\phi_i, \tag{7}$$

then as a consequence of this definition, it is seen that for every session a minimum service rate can be guaranteed to be

$$g_i = \frac{\phi_i}{\sum_{j=1}^{N}\phi_j}C. \tag{8}$$

GPS is an idealized server and does not send packets as entities as it assumes the traffic is infinitively divisible so that all the backlogged sessions can be served simultaneously. In a real packet based system only one packet can acquire service at one time instant and the whole packet must be served before any other packet can have service. There have been many proposals on how to emulate an idealized GPS server in a packet based system (e.g. WFQ, FQS, SCFQ).

2.3.2 Weighted Fair Queueing (WFQ)

The idea of using fair queueing to reduce congestion in packet networks was first proposed in [51], where Nagle suggests that each source could have its own queue in the switch and the queues would be serviced in a round-robin manner, thus providing an equal amount of service to all currently transmitting sources.

Thus, a source that is sending packets "too quickly", a so-called ill-behaving source, cannot increase its share of the bandwidth, it can only increase the length of its own queue. The drawback in Nagle's idea is that it does not consider different packet lengths and thus does not guarantee a fair allocation of bandwidth. This disadvantage in Nagle's idea was solved with an ideal algorithm, *bit-by-bit round-robin* (BR) [12].

Since sending packets in a bit-by-bit round-robin manner is not implementable in practice [12], a suggestion was made to emulate it by a practical packet-by-packet transmission scheme, Weighted Fair Queueing (also known as Packet GPS) [56]. The idea here is to simulate a GPS system in parallel with the packet-based system in order to identify the set of sessions that are backlogged at each instant. For each arriving packet, a "time stamp" is calculated indicating the time at which it would leave the system under GPS. Packets are then transmitted in increasing order of their time stamps.

To calculate this order of departure, WFQ associates a start tag and a finish tag with every packet of a flow. Specifically, let p_f^j and l_f^j denote the j^{th} packet of flow f and its length, respectively. If $A(p_f^j)$ denotes the arrival time of packet p_f^j at the server, then start tag $S(p_f^j)$ and finish tag $F(p_f^j)$ of packet p_f^j are defined as

$$S(p_f^j) = \max(v(A(p_f^j)), F(p_f^{j-1})), \quad j \geq 1, \tag{9}$$

$$F(p_f^j) = S(p_f^j) + \frac{l_f^j}{\phi_f}, \quad j \geq 1, \tag{10}$$

where $F(p_f^0) = 0$ and the virtual time $v(t)$ is defined as

$$\frac{\mathrm{d}v(t)}{\mathrm{d}t} = \frac{C}{\sum_{j \in B(t)} \phi_j}. \tag{11}$$

Here, C is the capacity of the server and $B(t)$ is the set of backlogged flows at time t in the bit-by-bit round-robin server.

WFQ has bandwidth guarantee

$$B_{min} = C \frac{\phi_i}{\sum_{j=1}^{N} \phi_j} \tag{12}$$

and a guarantee on the delay bound

$$\Theta_{WFQ} = \frac{b_i}{B_{min}}, \tag{13}$$

where b_i is the number of packets in a burst.

Unfortunately, the implementation of WFQ is not computationally efficient as it requires that $v(t)$ of (11), which in turn requires simulation of bit-by-bit round-robin server, must be computed in real time. This needs processing of $O(N)$ events in a single packet transmission time, where N is the number of flows served. The computational complexity is a serious drawback of the algorithm and many modifications have been proposed to improve WFQ.

Stochastic Fairness Queueing was suggested in [46] to reduce the computational complexity of WFQ by reducing the number of queues well below the number of possible flows. The flows are mapped to the queues by hashing. Those flows that are hashed into the same queue are treated equivalently, but unfairly with respect to the queues occupied by only one flow. Thus the fairness guarantee is probabilistic and it will be small only if the number of queues is sufficiently larger than the number of *active* flows. Although the number of queues can decrease from the number of all possible flows to a small multiple of the number of active flows, it is still unacceptably large for obtaining fair service.

2.3.3 Virtual Clock (VC)

One of the earliest methods proposed to isolate network flows is the Virtual Clock (VC) method [80], where each flow has an independent clock associated with it (i.e., each flow might run at different rates). When a packet of flow f arrives it is stamped by an algorithm that is independent of the arrivals of any other flow. The stamped packets are placed in a single queue shared by all the flows, and are served in order of stamped value.

If p_f^j is the j^{th} packet of flow f, then its virtual time stamp is assigned as

$$v_{ts}^{VC}(p_f^j) = \max(A(p_f^j), v_{ts}^{VC}(p_f^{j-1})) + \frac{l_f^j}{\phi_f}, \text{ for } j > 0, \tag{14}$$

where $A(p_f^j)$ is the arrival time of packet p_f^j and l_f^j is the length of p_f^j. The virtual time stamp $v_{ts}^{VC}(p_f^0)$ is set to zero.

If the sum of the rate of all flows that are sharing a single link does not exceed the link capacity, then the departure time $L(p_f^j)$ of packet p_f^j is bounded by

$$L(p_f^j) \leq v_{ts}^{VC}(p_f^j) + \frac{l_f^{max}}{C}, \tag{15}$$

where l_f^{max} is the length of the largest packet of flow f and C is the data rate of the output link.

The Virtual Clock scheduling algorithm provides a delay guarantee to flows (which is the same as for WFQ), but there is a price to be paid in terms of fairness. Packets from a flow that has been idle for some time will be assigned smaller virtual time stamp values and thus receive a larger than reserved share of service.

The fairness problem in VC is solved by the Leap Forward Virtual Clock (LFVC) [72] method, where the oversubscribed flows are temporarily moved to a holding area with low priority (i.e. they get no service). A flow f is defined as oversubscribed when the difference between the virtual time stamps of the current packet of f and the system time exceeds a certain threshold. If all flows become oversubscribed, then the system clock is advanced, which allows some of the flows to be moved to the high priority area. The LFVC has almost identical delay bound and throughput fairness to that of WFQ, but has the complexity of $O(\log\log N)$, where N is the number of flows.

2.3.4 Fair Queueing based on Start-time (FQS)

In [20] Fair Queueing based on Start-time (FQS) was proposed where the packets are scheduled based on the starting times, instead of the finishing times. The calculation of these values is done exactly as in WFQ by (9) and (10). So the virtual "time stamps" for packet p_f^j can be defined as

$$v_{ts}^{WFQ}(p_f^j) = S(p_f^j) + \frac{l_f^j}{\phi_f}, \quad j \geq 1, \tag{16}$$

for WFQ and

$$v_{ts}^{FQS}(p_f^j) = \max(v(A(p_f^j)), F(p_f^{j-1})), \quad j \geq 1, \tag{17}$$

for FQS. Since the virtual time of FQS is defined by (11) as in WFQ, FQS has disadvantages similar to that of WFQ. In fact, FQS is not known to have any advantage over WFQ for scheduling packets in a network, although it has advantages for processor scheduling.

2.3.5 Self-clocked Fair Queueing (SCFQ)

To reduce the complexity of WFQ, [17] analyzed a method called Self-Clocked Fair Queueing (SCFQ) which was previously proposed in [11]. SCFQ reduces the complexity of WFQ by using the time stamp of the packet currently in service to compute the time stamp of an arriving packet.

Let the virtual time stamp of a packet in service at time t be defined as in (11). Then packet p_f^j belonging to the j^{th} of flow f in a SCFQ system will receive a virtual time stamp

$$v_{ts}^{SCFQ}(p_f^j) = \max(v(A(p_f^j)), v_{ts}^{SCFQ}(p_f^{j-1})) + \frac{l_f^j}{\phi_f}, \text{ for } j > 0. \tag{18}$$

The value of $v_{ts}^{SCFQ}(p_f^0)$ is set to zero and the ascending order of the virtual time stamp values is used for transmitting the packets. SCFQ provides a fairness guarantee [17] as well as an end-to-end delay guarantee [19] to analogous flows. The value of fairness measure for SCFQ is

$$H(f,m) = \left(\frac{l_f^{max}}{\phi_f} + \frac{l_m^{max}}{\phi_m}\right), \tag{19}$$

which is away from the lower bound [17] only by a factor of two. The SCFQ has one main limitation, which is that the maximum delay endured by the packets is significantly increased. Let S be the set of flows served by a server and C its capacity, then packets of flow f may undergo

$$\frac{1}{C} \sum_{n \in S \wedge n \neq f} l_n^{max} \tag{20}$$

more delay in SCFQ than in WFQ [18]. This is unacceptably large in many cases.

2.3.6 Worst-case Fair Weighted Fair Queueing algorithm (WF^{2}Q)

In [2], Worst-case-fair weighted Fair Queueing (WF^{2}Q) was proposed to improve the emulation of BR server of WFQ by limiting the number of packets considered for service. WF^{2}Q defines the virtual time, start and finish tags as in WFQ by (11), (9) and (10), respectively. In WF^{2}Q for a packet to be eligible for service at time t, its start tag should not be greater than the virtual time

$$S(p_f^j) \leq v(t). \tag{21}$$

In WFQ all the packets are considered eligible. These eligible packets are scheduled in the increasing order of the finish tags. WF^{2}Q emulates the BR server well and has a fairness measure

$$H(f, m) = \left(\frac{l_f^{max}}{\phi_f} + \frac{l_m^{max}}{\phi_m} \right). \tag{22}$$

However, WF^{2}Q is computationally inefficient and unsuitable for achieving fairness over variable rate servers as it utilizes $v(t)$ as defined in (11).

A low complexity version of WF^{2}Q was proposed in [3], in which it was referred to as WF^{2}Q+. It modifies the definition of the start tag of packet p_f^j to be the finish tag of previous packet p_f^{j-1}, i.e.,

$$S(p_f^j) = F(p_f^{j-1}), \tag{23}$$

if flow f is backlogged on arrival of p_f^j and otherwise

$$S(p_f^j) = \max(v(A(p_f^j)), F(p_f^{j-1})). \tag{24}$$

The finish tag of a packet and the set of eligible packets are defined as in WF^{2}Q but $v(t)$ is defined as

$$v(t) = \max(v(\tau) + t - \tau, \min_{n \in B(t)} S(p_n^k)), \tag{25}$$

where $\tau < t$ is the time when the last packet finished service, p_n^k is the next packet of flow n to receive service, and $B(t)$ is the set of backlogged flows at time t. Just as in WF^{2}Q, the eligible packets of WF^{2}Q+ are scheduled in increasing order of finish tags. The fairness measure of WF^{2}Q+ is not derived in [3], although worst-case fairness of WF^{2}Q+ is derived.

2.3.7 Start-time Fair Queueing (SFQ)

Start-time Fair Queueing (SFQ) [19] uses the concept of SCFQ to approximate the virtual time but schedules the packets in increasing order of start tags as in FQS. However, SFQ outperforms the SCFQ in delay guarantee and FQS in fairness over variable rate servers and implementation complexity. For SFQ $v(t)$ is defined as the start tag of the packet in service at time t. A packet p_f^j is stamped with start tag $S(p_f^j)$, computed as

$$S(p_f^j) = \max(v(A(p_f^j)), F(p_f^{j-1})), \quad j \geq 1, \tag{26}$$

where $F(p_f^j)$, the finish tag of packet p_f^j, is defined as

$$F(p_f^j) = S(p_f^j) + \frac{l_f^j}{\phi_f}, \quad j \geq 1. \tag{27}$$

Here $F(p_f^0) = 0$ and ϕ_f is the weight of flow f. In SFQ the computation of $v(t)$ is inexpensive since it only involves examining the start tag of the packet in service. The computational complexity of SFQ is $O(\log N)$ per packet, where N is the number of flows at the server.

2.4 Frame-based scheduling schemes

Priority based schedulers such as WFQ, FQS, SCFQ, SFQ, WF^{2}Q, WF^{2}Q+ and VC discussed in Section 2.3 sort and schedule packets based on virtual time which is used to emulate the ideal GPS server. Hence, per-packet computational complexity is unacceptably large for these methods. Deficit Round Robin (DRR) is proposed in [66] to reduce this per-packet computational complexity. It is basically a derivative of the weighted round-robin algorithm [35] designed to accommodate variable length packets of a flow. Although the per-packet computational complexity of DRR is low, its fairness measure might deviate arbitrarily from the lower bound. Also, the maximum delay incurred by packets can be significantly higher than in WFQ. Many algorithms have been proposed to do away with the disadvantages of DRR.

2.4.1 Deficit Round-Robin (DRR)

Deficit Round-Robin (DRR) [67] is a simple modified weighted round robin scheduling algorithm, where the server rotationally selects packets to send out from all flows with packets in the queue. In order to avoid examining empty queues a service list containing the flow sequence being served in a round is maintained.

In a DRR system two variables *Quantum* (Q_f) and *DeficitCounter* (DC_f) are maintained for each flow f. *Quantum* is proportional to the weight of the flow and is the amount of credits in byte allocated to a flow within the period of one round. The quantum of flow f is proportional to the weight of the flow and is derived as

$$Q_f = \frac{\phi_f}{C} \times \sum_{f}^{N} Q_f, \tag{28}$$

where ϕ_f is the rate allocated to flow f, C the link service rate, and $\sum_f^N Q_f$ represents the frame size (i.e. summation of Quantums for all flows). DC_f^{j-1} accumulates the amount of Q_f of flow f which could not be used in the $(j-1)^{th}$ round. In the j^{th} round that flow f is served, it is allowed to send out additional DC_f^{j-1} bytes of data. After flow f has been served the server updates DC_f^j as

$$DC_f^j = DC_f^{j-1} + Q_f, \tag{29}$$

and it verifies the size of the packet of flow f waiting to receive service. If the size is smaller than DC_f^j, DC_f^j is decreased by this packet size and the packet is sent out. This operation is repeated as long as there are credits or there are no packets left. The Quantum for a flow should be larger than the maximum packet size within the flow so that in each round at least one packet per backlogged flow can be served. DRR is simple to implement and has $O(1)$ time complexity, but it is not fair in short term and has undesirable output burstiness.

The latency bound of DRR can be improved by Nested Deficit Round Robin (Nested-DRR) [32]. In Nested-DRR each DRR round is split into one or more smaller rounds. On these smaller rounds a modified version of the DRR scheduling discipline is used. Nested-DRR has per-packet work complexity of $O(1)$ and the same relative fairness bound as DRR.

2.4.2 Smoothed Round Robin (SRR)

Smoothed Round Robin (SRR) scheme [22, 23] was designed to improve the short term fairness of DRR. SRR has $O(1)$ time complexity and a certain scheduling delay bound. In SRR the weights of the flows are coded into binary vectors to form a Weight Matrix. A Weight Spread Sequence (WSS) is used to scan the Weight Matrix in order to distribute the output traffic of each flow equivalently. While Smoothed Round Robin does improve the average packet delay over DRR, the scheduling delay bound is proportional to the number of flows in the system and is bounded by

$$\Theta_{SRR} < \frac{2L_{max}}{\phi_i} + (N-1)\frac{2L_{max}}{C}, \tag{30}$$

where L_{max} is the maximum packet length, N is the number of flows, ϕ_i is the weight of flow i and C is the bandwidth of the output link. The fairness measure of Smoothed Round Robin is

$$\left| \frac{S_{i,SRR}(0,t)}{\phi_i} - \frac{S_{j,SRR}(0,t)}{\phi_j} \right| \leq \frac{(k+2)L_{max}}{2\min(\phi_i, \phi_j)},$$

where k is the order of the current WSS and $S_{i,SRR}(0,t)$ is the service rate of flow i from time 0 to t.

2.4.3 Pre-order deficit round robin (PDRR)

Pre-order Deficit Round Robin (PDRR) is suggested in [75] to overcome the problems of DRR, by approximating packet by packet generalized processor sharing (PGPS), by ordering the packets before the actual DRR process. Let the number of priority queues assigned to pre-ordering be Z and $Pq_1, Pq_2, \ldots, Pq_Z$ denote the priority queues. Then a packet is transmitted from Pq_k with the smallest k among all nonempty priority queues. The decision about the priority queue where an arriving packet should be placed is made by a classifier sub-module before the DRR scheduling discipline. Assuming that the flows are in a heavy backlog, the virtual finishing time stamp of a packet p_f^j in PDRR is computed as

$$v_{ts}^{PDRR}(p_f^j) = v_{ts}^{PDRR}(p_f^{j-1}) + \frac{l_f^j}{\phi_f}, \tag{31}$$

where $v_{ts}^{PDRR}(p_f^0)$ is set to zero at time t, ϕ_f denotes the allocated rate of flow f, and l_f^j is the size of the j^{th} packet of flow f after time t. PDRR achieves a per-packet time complexity of $O(1)$ in most cases, but in some specific cases the complexity is dependent on the number of priority queues i.e. $O(\log Z)$.

2.4.4 Active List Queue Method (Aliquem)

Active List Queue Method (Aliquem) [41, 42] is an evolution of DRR that allows to scale down the quantum assigned to a flow in each round. This improves the delay and burstiness properties of DRR. However, since the quantum may be less than the maximum packet size, a flow may not be able to transmit any data in a round, which leads to a mechanism that keeps track of the round in which a flow has accumulated enough credits to transmit a packet. Also, a flow can send several packets during its service quantum, thus creating output burstiness which worsens e.g. average delay. A modified version of Aliquem (called Smooth Aliquem) where a flow may transmit only one packet at a time is proposed in [42].

2.4.5 Bin Sort Fair Queueing (BSFQ)

Bin Sort Fair Queuing (BSFQ) [7] is a frame-based method that uses an approximate bin sort mechanism to schedule packets. BSFQ uses equally divided slices of the "virtual time" of priority based schemes called *bins*. Each packet is assigned a time stamp representing the finishing time of its service. The packets are stored into bins according to their time stamps. Inside a bin the packets are queued in FIFO order i.e. no sorting of packets is performed within the bin.

In BSFQ the virtual system clock $v(t)$ is defined as the starting time of the current bin and the bin is labeled as $[v(t), v(t) + \Delta]$, where Δ is the length of the bin (i.e. a slice of the "virtual time"). The virtual time clock is incremented by Δ when all the packets of the current bin are transmitted.

The j^{th} packet p_f^j of flow f is assigned with the virtual time stamp

$$v_{ts}^{BSFQ}(p_f^j) = \max(v(A(p_f^j)), v_{ts}^{BSFQ}(p_f^{j-1})) + \frac{l_f^j}{\phi_f}, \quad j \geq 1, \tag{32}$$

where $v(A(p_f^j))$ is the system virtual time at time $t = A(p_f^j)$ and $v_{ts}^{BSFQ}(p_f^0) = 0$. Arriving packets are stored in their corresponding bins in the FIFO order. The packet p_f^j is stored in a bin indexed by

$$i_f^j = \left\lfloor \frac{v_{ts}^{BSFQ}(p_f^j) - v(A(p_f^j))}{\Delta} \right\rfloor, \quad j \geq 1. \tag{33}$$

In the case $i_f^j = 0$, then p_f^j is stored in the current bin, if i_f^j is smaller than the number of bins it is stored in the i_f^j-th bin following the current bin. Otherwise, the packet is discarded.

2.4.6 Elastic Round Robin (ERR)

A frame-based scheduling discipline Elastic Round Robin (ERR) is proposed in [34] for best-effort traffic. ERR maintains a linked list of the active flows and empty queues are added to the end of the list on packet arrival. The flow at the head of the list is served first by ERR and if the queue has not emptied then it is added to the end of the list. The amount of bytes a flow can transmit is determined in each round and it is denoted by $A_f(r)$ for flow f in round r. However, this is not a strict limit and a flow can send a value of $Sent_f(r)$ in a round if its packet is larger than its allowance for that round. This unfairness is recorded by the scheduler by a Surplus Count (SC) as

$$SC_f(r) = Sent_f(r) - A_f(r). \tag{34}$$

The allowance for the next round is calculated by

$$A_f(r+1) = 1 + SC_{max}(r) - SC_f(r), \tag{35}$$

where $SC_{max}(r)$ is the largest surplus count of all flows that received service during round r. The addition of 1 in (35) ensures that the flow with the largest surplus count will get at least one packet transmitted in the next round. In [33] it is shown that the latency bound of ERR can be adapted for guaranteed rate of service and that it has a low latency bound.

2.4.7 Stratified Round Robin

Stratified Round Robin [63] assigns slots to flows, in a fashion similar to round-robin schedulers. A flow can send a certain number of its packets when it gets a slot. To decide which of the flows should get the next slot, Stratified Round Robin aggregates flows into flow classes based on their weights w_i. A flow class F_k is defined as

$$F_k = \left\{ f_i : \frac{1}{2^k} \leq w_i \leq \frac{1}{2^{k-1}} \right\}, \tag{36}$$

for $k \geq 1$ so F_k groups together all flows whose weights are approximately 2^{-k}. Let B_{min} denote the smallest unit of bandwidth that can be allocated to a flow. If a flow has a reserved bandwidth of B_{min}, then it belongs to flow class F_n, where $\frac{1}{2^n} \leq B_{min} < \frac{1}{2^{n-1}}$. Therefore, the number of flow classes required is $n = \log_2 \frac{C}{B_{min}}$, where C is the output link bandwidth of the scheduler. Stratified Round Robin has a single packet delay bound that is related to the guaranteed rate of the flow and is independent of the number of flows in the system.

2.4.8 Core-Stateless Fair Queueing (CSFQ) and Reduced State Fair Queuing

In [71] the complexity is localized in the edge routers, where the per-flow information is computed, while the core routers just use first-in-first-out (FIFO) queueing and keep no per-flow state. Thus the overall complexity of the system is decreased and the bandwidth can be allocated in an approximately fair way. Unfortunately, Core Stateless requires header changes that can be difficult to deploy.

In [39] the problem of state reduction is addressed by designing a throughput-fair scheduler that reduces by an order of magnitude the amount of state required by comparable throughput-fair schedulers. The algorithm provides a good traffic isolation and does not require global changes, only uncoordinated local implementation changes in each router are required.

2.4.9 Fair Round Robin (FRR)

Fair Round Robin (FRR) proposed in [79] is a worst case fair round robin scheduler that groups the flows in to classes just as in the Stratified Round Robin scheme [63]. The classes contain flows with similar weights and for $k \geq 1$, class F_k is defined as

$$F_k = \left\{ f_i : \frac{1}{\gamma^k} \leq w_i < \frac{1}{\gamma^{k-1}} \right\}, \tag{37}$$

where γ is a constant. The number of classes in the system is $n = \lceil \log_\gamma (\frac{C}{\phi}) \rceil$, where C is the link capacity and ϕ the rate of a single flow. Stratified Round Robin might be considered as a special case of FRR with $\gamma = 2$. Like the Stratified Round Robin scheme [63] FRR has a worst-case single packet delay bound that is only related to the requested rate of the flow. It is also independent of the number of flows in the system. The delay of a packet at the head of flow $f_i \in F_k$ at time t is bounded by

$$\Theta_{FRR} < c_1 \times \frac{L_{max}}{\phi_i}, \tag{38}$$

where c_1 is a constant, ϕ_i is the minimum guaranteed rate of flow i and L_{max} is the maximum packet size. The normalized worst-case fair index for FRR is

$$c_{FRR} = \max_i \left(\frac{\phi_i \times d \frac{L_{max}}{\phi_i}}{C} \right) = d \times \frac{L_{max}}{C}, \tag{39}$$

where $d = 7\gamma + n - 1$ and so the normalized worst-case fair index for FRR is constant. The authors of [79] state that FRR is the first $O(1)$ complexity scheduler scheme that has a constant worst-case fairness index. They also define the proportional fairness of FRR by showing that there exists two constants c_1 and c_2 such that

$$\left| \frac{S_{i,FRR}(t_1, t_2)}{\phi_i} - \frac{S_{j,FRR}(t_1, t_2)}{\phi_j} \right| \leq \frac{c_1 \times L_m}{\phi_i} + \frac{c_2 \times L_m}{\phi_j}. \tag{40}$$

2.5 Methods for combining pricing and scheduling

By using differentiated service classes the networks' resources can be divided between customers so that those paying more will get better service. In my research I have studied variable pricing and scheduling algorithms that optimize the revenue of the service provider. Next, I show how the delay and bit rate of a packet scheduler are defined and then the algorithms that have been published in **PI-PIX**.

Let the processing time of 1 kbyte of data be T [seconds/kbyte] in the packet scheduler of Figure 1. Thus, the overall delay [seconds] in the i^{th} class is

$$d_i = \frac{N_i E(b_i) T}{w_i}, \tag{41}$$

where N_i is the number of connections in the queue of class i , and $E(b_i)$ is the average packet size of the connections in the i^{th} class. The delay of the class is also

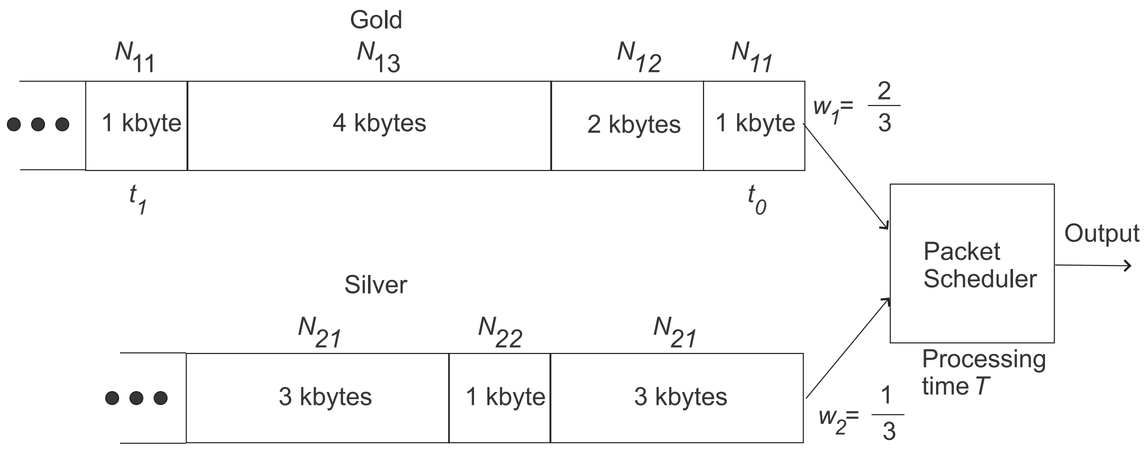


FIGURE 1 An example of a packet scheduler with two classes

affected by the connections in the other service classes so weight w_i also affects the delay. Because sizes of the different packets vary, the delay of connection j in class i, d_{ij} scaled by the specific packet of size b_{ij}, is

$$d_{ij} \ \ \text{[seconds/kB]} = \frac{N_i E(b_i) T}{b_{ij} w_i}. \tag{42}$$

The bit rate B_{ij} (kbytes/second) of class i is inversely proportional to the delay/packet of class i

$$B_{ij}[kB/s] = \frac{1}{d_{ij}} = \frac{b_{ij} w_i}{N_i E(b_i) T} = \frac{b_{ij} w_i}{N_i E(b_i)} = \frac{b_{ij} w_i}{\sum_{l=1}^{N_i} b_{il}}, \tag{43}$$

where $\sum_{l=1}^{N_i} b_{il}$ is the sum of all packet sizes in class i, $E(b_i) = \frac{1}{N_i} \sum_{j=1}^{N_i} b_{ij}$ is the mean packet size in class i and the processing time T can be scaled $T = 1$ [s/kB], without loss of generality.

The number of customers in the network in the following pricing and scheduling schemes can be arbitrary large, which will lead to decrease in Quality of Service. By implementing a Call Admission Control (CAC) mechanism the number of customers can be limited. The CAC will reject new customers if the revenue of the operator would decrease due to increase in congestion (i.e. decrease in Quality of Service). Also, QoS guarantees can be implemented with the CAC mechanism to guarantee e.g. a certain maximum delay to customers in a specific service class. A single customer that would use up all resources from one class would decrease QoS of all other customers in all classes and lead to decrease in total revenue, thus such a customer would be rejected by the CAC mechanism.

First, delay was studied as a QoS parameter. A linear pricing function was considered in publication **PI**

$$P_i(t) = r_i t + k_i, \quad r_i > 0, k_i > 0, \tag{44}$$

where r_i is the price paid in class i and t is time. This leads to optimal weights that maximize revenue

$$w_i = \sqrt[4]{\frac{r_i N_i^2 (1 - \sqrt{w_i}) E(b_i)}{\sum_{k \neq i}^{m} r_k N_k^2 E(b_k) / \sqrt{w_k}}}. \tag{45}$$

Here the weights need to be iteratively calculated. Similar results are obtained by choosing the weights with linear pricing function, as was shown in publication **PII**:

$$w_i = \frac{N_i E(b_i) r_i / k_i}{\sum_{k=1}^{m} N_k E(b_k) r_k / k_k} \tag{46}$$

with a constrained

$$\sum_{i=1}^{m} \frac{N_i E(b_i) r_i}{k_i} < 1 \tag{47}$$

used in call admission control mechanism. These weights do not require iterative calculation, but still they provide near optimal revenue.

In publication **PIII** a flat pricing scenario was used and the pricing function is defined via maximum delay for each class i

$$P_i(d_i) = r_i \tag{48}$$

under the constrained

$$\frac{N_i E(b_i)}{w_i} \leq d_{i,max}, \quad i = 1, \ldots, m, \tag{49}$$

where $d_{i,max}$ are preselected maximum delays to be guaranteed. Optimal weights are then

$$w_i = \frac{r_i N_i}{\sum_k r_k N_k}. \tag{50}$$

When bandwidth is the most important QoS parameter, a polynomial pricing function is chosen as in publication **PVIII** where simulations were ran in Matlab

$$P_i(B_{ij}) = r_i B_{ij}^p, \quad r_i > 0, \quad p > 0. \tag{51}$$

In the function, $r_i > 0$ is a factor that depends on the money paid for the class (bandwidth of the connection). The optimal weights to maximize the revenue are

$$w_i = \frac{(\sum_{j=1}^{N_i} b_{ij})^{\frac{p}{p-1}}}{r_i^{\frac{1}{p-1}} \left(\sum_{j=1}^{N_i} b_{ij}^p\right)^{\frac{1}{p-1}} \sum_{k=1}^{m} \frac{(\sum_{l=1}^{N_k} b_{kl})^{\frac{p}{p-1}}}{r_k^{\frac{1}{p-1}} \left(\sum_{l=1}^{N_k} b_{kl}^p\right)^{\frac{1}{p-1}}}}, \tag{52}$$

when $0 < p < 1$.

The same algorithm was implemented in a Network Simulator 2 (ns-2) [53] environment in publication **PIV** and experiments were done with constant bit rate UDP traffic. In publication **PVI** the experiments were broadened to TCP traffic.

The scenario was extended to multiple nodes in publication **PV** where an algorithm to maximize revenue and minimize the weighted mean delay in a multiple node system was presented.

The algorithm is described by a four-node system with schedulers as illustrated in Figure 2. Data are transmitted from ingress nodes to either egress nodes. Parameter Δt_{ij} denotes the time which passes when data is transferred through the queue j to the output in the node i, when $w_{ij} = 1$. Variable w_{ij} is the weight allocated for node i and class j. The constraints for weights w_{ij} are

$$\sum_{j=1}^{m} w_{ij} = 1, \quad w_{ij} > 0, \tag{53}$$

where m is the number of service classes. Variables w_{ij} weight the service times of the queues. Therefore, the delay d_{ij} in the queue (i, j) is actually

$$d_{ij} = \frac{\Delta t_{ij}}{w_{ij}}. \tag{54}$$

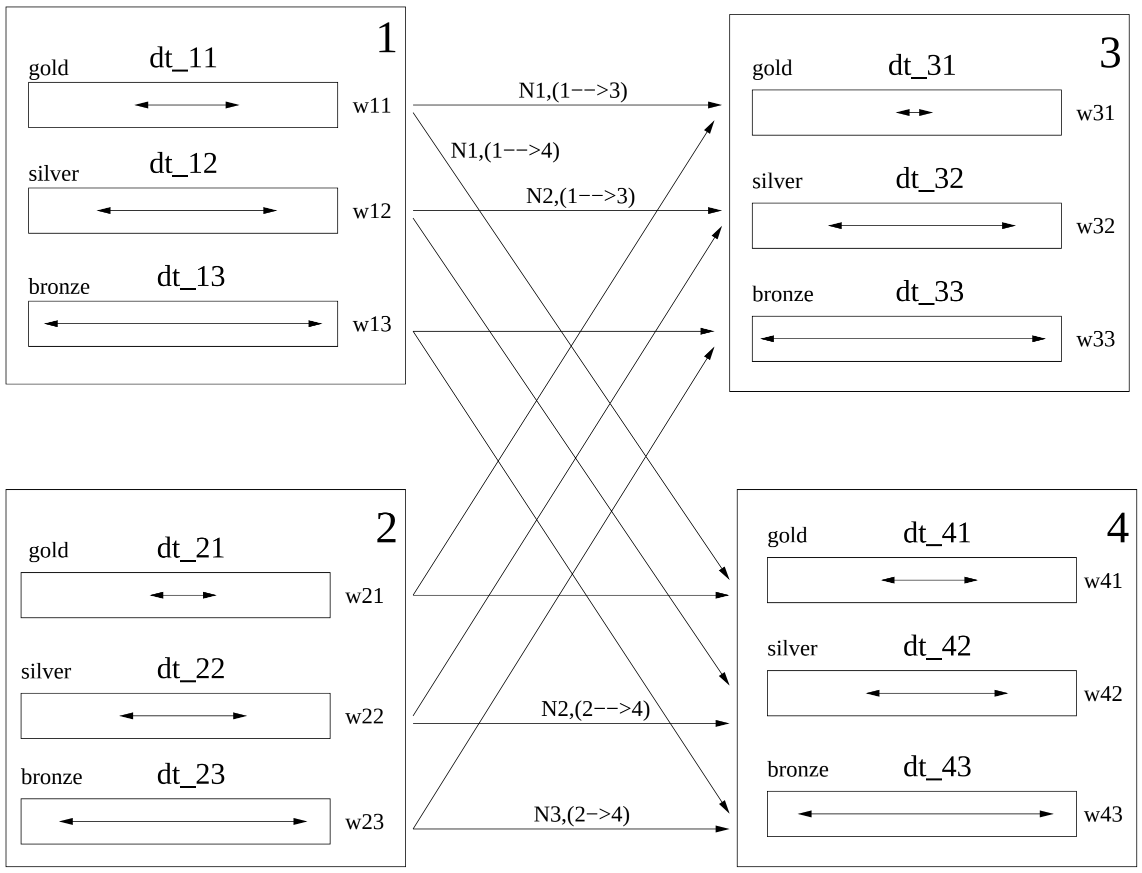


FIGURE 2 Four edge node system. In the middle of the system there is a core network, which is not shown in the figure.

The total number of connections in the node i and queue j is denoted by N_{ij}, and is obtained by summing all the connections which pass through the node i on their way to the n different egress nodes accessible from node i

$$N_{ij} = \sum_{p=1}^{n} N_j^{i_1 \to i_2}. \tag{55}$$

Here $N_j^{i_1 \to i_2}$ denotes the number of such connections in the j^{th} service class which transfer data packets from node i_1 to i_2.

In this case pricing depends on the end-to-end delay of the data and a linear pricing function is used

$$r_j(d) = -r_j d + k_j, \quad j = 1, \dots, m, \quad r_j > 0, \quad k_j > 0, \tag{56}$$

where r_j is a "penalty" factor, and k_j is a "shifting" factor. The optimal weights that maximize the revenue are then

$$w_{ij} = \frac{\sqrt{N_{ij} r_j \Delta t_{ij}}}{\sum_{l=1}^{m} \sqrt{N_{il} r_l \Delta t_{il}}}, \quad i = 1, \dots, n, \quad j = 1 \dots, m, \tag{57}$$

where n is the number of nodes and m is the number of service classes.

Next, let us consider the bandwidth in the node i in the four-node system as is done in publication **PVII**. There are N_{ij} connections or packets in the class j. The packet size of the connection $k = 1, \ldots, N_{ij}$ in the node i, $i = 1, \ldots, n$, and in class $j = 1, \ldots, m$ is denoted by b_{ijk} [bits] or [kbytes]. Variable w_{ij} is the weight allocated for class j in the node i. The constrained for weights w_{ij} are as in (53). Now, the expression for the bandwidth of the packet (i, j, k) is

$$B_{ijk}[bits/s] = \frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}}. \tag{58}$$

The polynomial pricing function in this case is

$$r_j(B) = r_j B^p, \quad r_j > 0, \quad p \in (0, 1), \tag{59}$$

where B is the bandwidth.

When connection (i, j, k) in the node i in class j obtains the bandwidth B_{ijk}, the price is

$$r_j(B_{ijk}) = r_j \left(\frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \right)^p \tag{60}$$

units of money per second to the service provider. The closed form solution to the weights are

$$w_{ij} = \frac{r_j^{-\frac{1}{p-1}} \left[\sum_{k=1}^{N_{ij}} \left(\frac{b_{ijk}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \right)^p \right]^{-\frac{1}{p-1}}}{\sum_{q=1}^{m} r_q^{-\frac{1}{p-1}} \left[\sum_{s=1}^{N_{iq}} \left(\frac{b_{iqs}}{\sum_{h=1}^{N_{iq}} b_{iqh}} \right)^p \right]^{-\frac{1}{p-1}}}. \tag{61}$$

A new approach to combine pricing and packet scheduling was proposed by the author of this thesis in publication **PIX** where pricing of the connection is based on the delay it inflicts on others. The derivation of the algorithm is presented next.

Let us consider a packet scheduler which receives packets to be delivered from m different queues (i.e. classes). Now, let T be the processing time of the classifier for transmitting a data packet from one queue to the output of a packet scheduler. For any connection k there are $N_{ik}(t)$ packets in the queue of class i at some time t. The total number of packets in the queue i at the time t is therefore

$$N_i(t) = \sum_{k=1}^{K_i(t)} N_{ik}(t), \tag{62}$$

where $K_i(t)$ is the total number of connections in the class i at time t.

Now, every packet in the queue increases delay for the other connections in the queue so a new connection k appearing at time t in class i will increase the delay to the other connections by a value proportional to $N_{ik}(t)$. Then

$$d_{ik}(t) = \frac{N_{ik}(t) T}{w_i(t)} = \frac{N_{ik}(t)}{w_i(t)}, \tag{63}$$

where T can be scaled to $T = 1$ without loss of generality and $w_i(t) = w_i, i = 1, \dots, m$ are weights allocated for each class as a new connection in any class will inflict delay on the other classes depending on the amount of processing time of the scheduler allotted to each class. The overall delay in class i is then

$$D_i(t) = \sum_{k=1}^{K_i} d_{ik} = \sum_{k=1}^{K_i} \frac{N_{ik}}{w_i}, \tag{64}$$

where the time index t has been dropped for convenience until otherwise stated. The natural constraints for the weights are

$$\sum_{i=1}^{m} w_i = 1, \quad w_i > 0. \tag{65}$$

For every connection the pricing is based on the absolute number of packets of the connection that are queued at the scheduler i.e. the increase in delay that it produces. Every connection induces delay and those connections that induce more delay than the connections on average will pay more for the connection.

Definition 1. For each service class the function

$$P_i = \sum_{k=1}^{K_i} K_i r_i + s_i d_{ik}^p, \quad r_i > 0, \quad s_i > 0, \quad p > 0, \tag{66}$$

is called a polynomial pricing function.

In the polynomial pricing function r_i is a base price of a connection in the class i, s_i is a penalty factor that increases the price for those connections which induce more delay and p assures that the price does not grow too rapidly with the delay. The total revenue of the operator is

$$R = \sum_{i=1}^{m} P_i = \sum_{i=1}^{m} \sum_{k=1}^{K_i} p_{ik} = \sum_{i=1}^{m} \sum_{k=1}^{K_i} K_i r_i + s_i d_{ik}^p. \tag{67}$$

Theorem 1. *Optimal weights for the revenue R under the polynomial pricing model are*

$$w_i = \frac{s_i^{\frac{1}{p-1}} N_i^{\frac{p}{p-1}}}{\sum_{j=1}^{m} s_j^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}}, \tag{68}$$

when $0 < p < 1$.

Proof. By using Lagrangian approach, the revenue can be presented in the form

$$R = \sum_{i=1}^{m} \sum_{k=1}^{K_i} K_i r_i + s_i \frac{N_{ik}^p}{w_i^p} + \lambda(1 - \sum_{i=1}^{m} w_i). \tag{69}$$

Optimal weights are obtained from the first derivative

$$\frac{\partial R}{\partial w_i} = \sum_{k=1}^{K_i} -p s_i \frac{N_{ik}^p}{w_i^{p-1}} - \lambda = 0. \tag{70}$$

Because Lagrangian solution

$$\frac{\partial F}{\partial \lambda} = 0 \tag{71}$$

yields $\sum_{j=1}^{m} w_j = 1$, then

$$\begin{aligned} w_i &= \left(\frac{-ps_iN_i^p}{\lambda}\right)^{\frac{1}{p-1}} = \frac{(-ps_i)^{\frac{1}{p-1}} N_i^{\frac{p}{p-1}}}{\lambda^{\frac{1}{p-1}} \sum_{j=1}^{m} w_j} \\ &= \frac{(-ps_i)^{\frac{1}{p-1}} N_i^{\frac{p}{p-1}}}{\lambda^{\frac{1}{p-1}} \sum_{j=1}^{m} \frac{(-ps_j)^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}}{\lambda^{\frac{1}{p-1}}}} = \frac{s_i^{\frac{1}{p-1}} N_i^{\frac{p}{p-1}}}{\sum_{j=1}^{m} s_j^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}}. \end{aligned} \tag{72}$$

From the expression (72) it is seen that λ has the form

$$\lambda^{\frac{1}{p-1}} = \sum_{j=1}^{m} s_j^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}, \tag{73}$$

or

$$\lambda = \sum_{j=1}^{m} s_j N_j^p. \tag{74}$$

Thus, the first order derivative is

$$\frac{\partial R}{\partial w_i} = \sum_{k=1}^{K_i} \frac{-ps_iN_{ik}^p}{w_i^{p-1}} - \sum_{j=1}^{m} s_j N_j^p, \tag{75}$$

and so the second order derivative is

$$\frac{\partial^2 R}{\partial w_i^2} = \sum_{k=1}^{K_i} -(p-1)\frac{-ps_iN_{ik}^p}{w_i^{p-2}} < 0, \tag{76}$$

when $0 < p < 1$. In that condition, R is concave with respect to the weights w_i, and the optimal solution is unique. $\square$

By using optimal weights (68), revenue R can be expressed as follows:

$$\begin{aligned} R &= \sum_{i=1}^{m} \sum_{k=1}^{K_i} K_i r_i + \frac{s_i N_i}{w_i} = \sum_{i=1}^{m} K_i r_i + \frac{s_i N_i}{\frac{s_i^{\frac{1}{p-1}} N_i^{\frac{p}{p-1}}}{\sum_{j=1}^{m} s_j^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}}} \\ &= \sum_{i=1}^{m} K_i r_i + s_i^{\frac{p-2}{p-1}} N_i^{\frac{-1}{p-1}} \times \sum_{j=1}^{m} s_j^{\frac{1}{p-1}} N_j^{\frac{p}{p-1}}. \end{aligned} \tag{77}$$

The method presented above is verified by simulations in **PIX**. For the other methods reviewed in this section, proofs are presented and verified by simulation results in **PI - PVIII**.

3 FAST HANDOVERS FOR MOBILE IPV6

3.1 Introduction

Mobility support in IPv6 (MIPv6) [28] enables transparent routing of packets to the Mobile Node (MN). In a home network, MN is assigned a permanent home address (HoA), which is used in every application layer (Layer 7) connection. When MN changes its IP layer (Layer 3) attachment in the network, it acquires a new local address from the foreign network to be accessible again. This address, called the *Care-of-Address* (CoA), is registered through a *binding update* (BU) process to a special router in the home network called the *Home Agent* (HA). The HA maintains a binding cache, in which the HoA-CoA bindings are stored, and employs tunneling to redirect the flows to the current CoA of the MN.

The most significant enhancement in MIPv6, compared to the mobility support in IPv4, is *route optimization* which means that the MN can also send BUs to the nodes it is communicating with. These corresponding nodes (CNs) also maintain a binding cache. Thus CNs can send packets directly to the MN without triangular routing via HA as in IPv4, therefore improving the End-to-End communication delay. This happens at the expense of the handover delay, which is increased by about two round-trip times due to sending the BUs to CNs and the Return Routability (RR) procedure.

Although MIPv6 enables mobility at the IP layer, the processes related to MIPv6 mobility management result in a short period of time when the MN cannot receive or send any packets. This period of time, known as the *handover delay*, degrades the performance of real time applications such as multimedia or Voice over Internet Protocol (VoIP).

The MIPv6 handover delay can be reduced in three different places: IP layer movement detection, CoA acquiring/forming and CoA registration delay to the HA and CN(s). After the MN changes its point of attachment to another network, it receives a *Router Advertisement* (RA) from the new access router (nAR). Using the stateless or the stateful address auto-configuration, MN forms or receives a new reachable CoA. If the address is acquired with stateless address auto-configuration, it needs to be verified for its uniqueness with Duplicate Address Detection (DAD) process. Next, the new CoA needs to be registered to MN's HA and CNs. This requires a two-way BU process to all parties and the RR procedure.

The reduction of the MIPv6 handover delay can be divided into two methods: L2 Layer 2 (L2) dependant and L2 independent. The basic MIPv6 was developed to be independent from the underlying L2 technology. This enables the mobility protocol to work without modifications with any link technology, such as a wireless local area, cellular network or bluetooth network. On the other hand L2 triggers could speed up the L3 handover procedure, but only at the expense of L2 dependency.

There have been numerous different proposals for minimizing the L3 handover delay in Mobile IPv6 networks during the past few years. The most promising of these are *Hierarchical MIPv6* (HMIPv6) [69] and *Fast Handovers for MIPv6* (FMIPv6) [40]. HMIPv6 proposes a mobility anchor point (MAP) to act as a local HA in the foreign network. FMIPv6 uses link layer triggers to speed up the CoA configuring time and employs tunneling from the old Access Router (AR) to the new AR during the BU process to minimize packet loss. Next I shall briefly present the functionality of HMIPv6 and FMIPv6 and then review other proposals for decreasing the handover delay of MIPv6.

3.1.1 Hierarchical Mobile IPv6

Hierarchical Mobile IPv6 (HMIPv6) [69] introduces a new node, *Mobility Anchor Point* (MAP), to reduce the registration time of the new CoA and signaling load. The MAP separates the mobility into local and global mobility (also known as micro and macro mobility). The MAP manages the mobility inside the local domain and the HA is responsible for the management of mobility between separate MAP domains. MAP functions basically as a local HA in the foreign network, tunneling flows to the current location of the MN.

The implementation of local mobility is done with the usage of two CoAs, *regional CoA* (RCoA) and *on-link CoA* (LCoA). Every time when MN moves to an entirely new MAP domain, it receives or forms a new RCoA, which is registered to the MAP, HA and CNs. When the attachment point of the MN changes inside the MAP domain, only the MAP needs to be informed about the new on-link CoA, which matches to the current subnet prefix. The packets heading to the old LCoA are intercepted by the MAP and tunneled to the new LCoA similarly to the way it is done in the HA.

The HMIPv6 decreases the amount of signaling to CNs and to the HA. However, HMIPv6 might have scalability problems if MAP has to handle too many MNs [48]. The use of MAP increases the complexity of the system and is a possible point of failure in the network. Also, the MAP is likely to be at the ingress/egress point of the (sub)network which means that the registration delay might still be too large for delay critical applications.

3.1.2 Fast Handovers for Mobile IPv6

Fast Handovers for Mobile IPv6 (FMIPv6) [40] decreases the delay of the CoA acquiring phase and employs tunneling to reduce the packet loss during the handover.

The fast handover procedure is started when MN receives information about the next point of attachment and sends a *Router Solicitation for Proxy* (RtSolPr) message to the old AR (oAR). MN formulates a prospective new CoA (nCoA) with the information the AR provides in the PrRtAdv message and sends a Fast Binding Update (FBU) message that authorizes oAR to bind previous CoA to nCoA, so that arriving packets can be tunneled to the new location of the MN. Two different modes of operation, *predictive* and *reactive*, are described by FMIPv6. In the predictive mode MN sends FBU and receives Fast Binding Acknowledgement (FBACK) in the oAR link, contrary to the reactive mode in which they are send on the nAR link.

The performance of FMIPv6 relies greatly on L2 information for predicting the upcoming handover. FMIPv6 loses its effectiveness with unreliable L2 information, such that would be caused by rapid signal strength fluctuations occurring in multipath fading environments. Also, as basic MIPv6 is developed to be independent of the underlying L2 technology, for FMIPv6 the dependence on L2 technology is an unwanted feature.

In [31, 30] a combination of the HMIPv6 and FMIPv6 named F-HMIPv6 is proposed. In that proposal the entity performing the functionality of FMIPv6 is the MAP instead of the oAR. Although F-HMIPv6 has been found to be very effective [62], it still shares the disadvantages of HMIPv6 and FMIPv6. The methods were simulated with Networks Simulator 2 (ns-2) [53] and during handover situations the effect of the number of MNs and different applications were studied.

3.1.3 Other methods for decreasing the handover delay in MIPv6

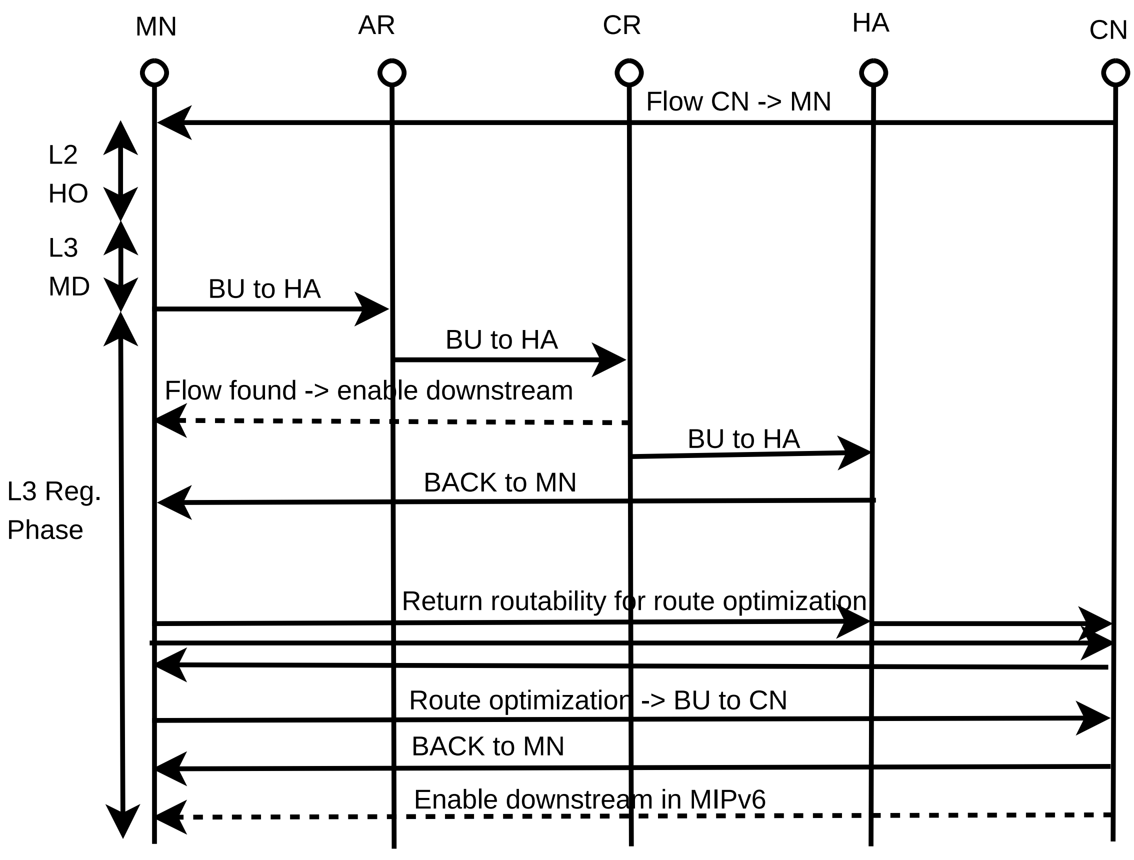
Numerous different proposals have been presented to minimize the handover delay in Mobile IPv6 networks. Unfortunately, many of these methods require modifications to the Access Routers (ARs). Multicast routing approach has been chosen in [4] and [14] to decrease the handover delay in MIPv6 networks. In [4] the CNs see the MN as a Multicast group. When the MN moves from one subnet to another it joins the Multicast group with its new CoA and if possible, remains connected to the Multicast Group with its previous CoA. In [14] all the CNs, in connection with a particular MN, subscribe to a multicast group. CNs are informed about the handovers of the MN via this multicast group. The maintenance of the multicast groups means additional tasks for the network elements.

A "virtual" HA located closer to the MN than the actual HA has been proposed, in different forms. A Routing Agent to be used between the HA and the Foreign Agent is presented in [6] and [77] to speed up the handover process. A Local Mobility Agent to act as a local HA is proposed in [73]. In [64] a specialized path setup scheme is proposed. In that proposal host-based forwarding entries are installed in specific routers so that the MN's network address remains the same within a domain.

While many proposals concentrate on modifying the ARs, another approach to diminish the handover delay is the modification of the MN. In [55] a mobile node buffering function is proposed to mitigate the effects of handover. The MN buffers the TCP ACKs just before the handover forcing the CN to stop transmitting. After the handover the buffered TCP ACKs are delivered to the CN and the transmission is resumed. In [60] a new handover mechanism and an enhancement to the Mobile IP registration is proposed. In the proposal the periodic Router Advertisements (RAs) are modified to be send only when a handover is necessary. The MN keeps a Router Advertisement (RA) cache to help on the decision where to make the handover. These methods require minor modifications to the network elements but require additional resources from the MN and detection, beforehand, of the upcoming handover.

Optimization of routing protocols has also been proposed in order to decrease the delay in handovers [26, 9, 76]. An improvement to HMIPv6 [69, 5] is presented in [26], in which a mobility profile (an average residence time in the current subnet) is maintained and, according to a defined threshold time, the MN can use a Local CoA instead of a Regional CoA to accomplish route optimization. Another improvement to HMIPv6 is presented in [76], where the reception of packets from the new AR is possible at the same time with the BU registration process. This requires for the MN to "see" the approaching of the handover, so that the old Mobile Anchor Point (MAP) can set up a multicast group to deliver packets destined to the MN. The possible new ARs are asked to join the multicast group and start buffering the packets destined to the MN. These HMIPv6 improvements require some additional resources from the network elements. In [9] a small part of the overall delay in MIPv6 handover is addressed. The delay of unicast Router Advertisement (RA) in response to Router Solicitation (RS) is removed by assigning one router to provide immediate unicast responses to RSs. This proposal could be used in conjunction with other methods to further decrease the overall handover delay.

In [25] *Seamless handoff architecture* (S-MIP) is presented and the effect of different handover methods to the performance of TCP-based applications is studied. S-MIP is based on movement tracking techniques, and it is found to provide handovers with almost no packet loss. In [27] the performance of different TCP variants is simulated with different handover methods. Performance degradation is found to be comparative to the round-trip time between MN and CN. Also the FMIPv6 and F-HMIPv6 do not necessarily improve the performance. In [49] the MIPv6 and both the tunnel-based and anticipated modes of FMIPv6 is studied with simulative environment. In general the FMIPv6 is found to have shorter disruption times, in particular with its the tunnel-based mode. In [24] the authors have concentrated on simulative scalability and robustness analysis of the methods. The combination is found to have the best handover performance and improve handover signaling overhead, but cannot remove tunneling overhead.



MN
AR
CR
HA
CN
Flow CN -> MN
L2
HO
L3
MD
BU to HA
BU to HA
Flow found -> enable downstream
BU to HA
BACK to MN
L3 Reg.
Phase
Return routability for route optimization
Route optimization -> BU to CN
BACK to MN
Enable downstream in MIPv6

FIGURE 3 FFHMIPv6 functionality

It was found to be also the most robust to failures. In [29] the authors study the MIPv6 and FMIPv6 method in both simulative and experimental environment, and, due its predictive nature, FMIPv6 is found to be more effective.

3.2 Flow-based Fast Handover for Mobile IPv6 (FFHMIPv6)

The need to minimize the handover delay is essential in order to provide adequate Quality of Service to customers with wireless access to the network. In the research a new method for decreasing the handover delay in MIPv6 based networks has been presented and analyzed. In this section I present the new method briefly and describe the research results that have been published in **PX-PXV**.

In publication **PX** a new method, *Flow-based Fast Handover for Mobile IPv6 Network* (FFHMIPv6), is proposed for faster L3 handover in Mobile IPv6 network, which addresses the CoA registration delay. The FFHMIPv6 method uses IPv6 flow information to redirect the flows to the new location *during* the handover, thus enabling the reception of flows simultaneously with the BU registration process.

The FFHMIPv6 functionality is shown in Figure 3. When the MN moves to a new logical subnet, it configures a new CoA and starts to register it to the HA and possibly to the CN(s) via BU process. In the FFHMIPv6 method a *Hop-by-Hop* header, including the old CoA and the addresses of the CNs, is added to the BU register message heading for the HA. The goal of this BU message is to redirect all the MN's flows to the new location.

Every IPv6 capable router maintains a flow cache of the active flows that it routes. In every router between the MN and the HA, the flow cache is compared with the flow information found from the Hop-by-Hop header. If the traffic flow is found, an IPv6 tunnel is established between the router and the new CoA of the MN, and the traffic flow is redirected to the established tunnel. This router is called the Crossover Router (CR) for the flow in question. Next, the CN's address in question is erased from the Hop-by-Hop frame, so that the FFHMIPv6 process is not performed again in another router for the same flow and the BU message is forwarded toward the HA. At the next hop the same procedure is repeated as long as there are flows to be redirected or the BU message reaches the HA. The FFHMIPv6 enables reception of the traffic flow simultaneously with the BU process through the tunnel, therefore minimizing *downstream* packet loss.

The FFHMIPv6 method is designed to be used as a micro mobility solution. Network topologies are often built hierarchically, so that all of the domain's ingress and egress traffic pass through the same router (Border Router). Given this assumption, the Crossover Router would very likely be found in most networks. Thus, with one BU message all the MN's flows could be redirected, whether they are tunneled flows from HA or direct flows using route optimization. If the flows are not found from the routers' flow cache or the routers do not support FFHMIPv6, normal MIPv6 BU process is applied.

In publication **PXI** *Flow-based Fast Handover for Mobile IPv6 Network* (FFHMIPv6) was simulated in Network Simulator 2 (ns-2) [53] environment with UDP and TPC traffic and compared to MIPv6 and HMIPv6. Using UDP traffic it was shown that the FFHMIPv6 method outperforms MIPv6 and HMIPv6 methods. When the CN distance increases, the delay and packet losses are similar and almost constant with FFHMIPv6 and HMIPv6, but with MIPv6 they increase almost linearly with the CN distance. As the MAP of HMIPv6 is usually located at the ingress/egress point of the network, the handover delay might be large in large subnets. This was shown by increasing the MAP distance, in which case FFHMPIv6 still had constant handover delay and packet loss while with HMIPv6 these increased with the MAP distance. With the FFHMIPv6 handover method, the packet loss of TCP traffic during handover was found to be significantly smaller than with MIPv6. The packet loss was 10% of that of MIPv6. The experiment results verified the conclusions drawn earlier from theory, namely that FFHMIPv6 out-performs the basic MIPv6 and its most common enhancements.

The FFHMIPv6 method was implemented in a real environment using Mobile IPv6 for Linux (MIPL) [47]. The handover effect on the VoIP traffic and the effect of the distance of the CNs is analyzed in publication **PXII**, which also discusses the scalability and processing load issues of the FFHMIPv6. The handover delay was found to be significantly shorter with FFHMIPv6 method than with the basic MIPv6 and, even better, independent of the distance of the CNs. With VoIP traffic a seamless handover was obtained with the FFHMIPv6. The Hop-by-Hop processing delay caused by the FFHMIPv6 was found to be almost negligible when compared with the benefits acquired with the flow redirection. The scalability issues which might arise with the per-flow information stored in the flow caches were insignificant, because all main manufacturers already implement the routing cache, which is used by the method.

In publication **PXIII** the experiments which compare several handover methods in theory, in simulation and in real network environment are summarized. There the author of this thesis presents an improvement to the proposed method to minimize the *upstream traffic*, as the original idea was only applicable to *downstream traffic*.

According to MIPv6 specification [28] the *upstream* traffic is possible only after a BACK message has been received from the HA as a response to a BU message. To overcome this limitation a method for decreasing the upstream handover delay was defined and preliminary simulations presented in publication **PXIV**. The simulation scenario was broadened and security issues for the FFHMIPv6 method with fast upstream were considered and solutions proposed in publication **PXV**. Next the fast upstream modification is presented in details.

I propose that a suitable amount of IP protocol version 6 (IPv6) subnet's addresses are reserved solely to be used in handover situations. In IPv6 the address can be presented on 128 bits, so there is practically no situation where a subnet would run out of addresses. The *handover address* would be used when MN arrives at a new subnet where its old CoA cannot be used because of ingress filtering [1], which is performed to ensure that it is not possible to send a malicious packet with an incorrect source address to perform a hostile action. The incoming traffic is checked at the router's *incoming interface* to verify that the sender address matches the network area defined at the same router interface.

Ingress filtering is always performed at the closest router (i.e. AR) interface on which the subnet for the MN (CoA) is defined. Additionally ingress filtering could be performed on the other input interfaces (upstream routers) to verify that the sender address matches the network routes pointing to the same direction. In the cases where tunneling is used, the tunnel is terminated usually to a specific *tunnel interface* on the router. This interface performs (in case filtering is needed) the ingress filtering for the incoming tunneled (decapsulated) packets. Ingress traffic filtering is currently supported by all network routers and is performed before any packet routing or processing is done. To enable fast upstream traffic the MN is given a temporary address that will pass ingress filtering.

The MN is assigned a specific handover address (called Hand-of-Address (HofA)) as the source address during a handover situation, when the MN cannot use its new CoA (which is not verified) or the old CoA (which leads to packet drops due to ingress filtering). The MN tunnels its packets in an IPv6-in-IPv6 tunnel where the destination is the AR's address and the source is the HofA. As the AR receives the packet, ingress filtering is done at the *incoming interface*. In case of handover the source is a HofA so the packet is not dropped. As the destination is the AR itself, it disassembles the packet and finds a packet with destination as the CN's address and a source as an old CoA from a different subnet. This packet is forwarded to the *tunnel interface*. Now ingress filtering at this interface is not used or, if used, then AR could keep track of oCoAs that are allowed to pass this ingress filtering.

There are basically three approaches on defining the HofA and they all have their advantages and disadvantages.

- Specify *one* global IPv6 address that is always used as a HofA.
- HofA is chosen from a global range of addresses reserved for handover purposes.
- Let the AR specify the HofA from its own (subnet's) address range.

Basically only one global address could be reserved for all handovers. There is two ways of "assigning" the HofA to the MNs. The HofA can be assigned to the MN by the new AR or the MNs can just use the HofA when needed. The latter approach is faster for the handover but is a severe security vulnerability. If the AR assigns the HofA then security issues can be better taken into consideration by using e.g. ingress filtering at the tunnel interface. Why not just use the old CoA instead of this HofA? This would lead to complications with ingress filtering at the incoming interface. When using a specified handover address, ingress filtering at the incoming interface needs to be modified so that the (global) HofA is always allowed access. Also, as current routers assume that one IP address corresponds to only one MAC address some modifications should be done to allow several MAC addresses to use the same global HofA. For this reason one global HofA cannot be used e.g. for speeding up the downstream traffic.

If the HofA is chosen from a global address range, then the option for the global HofAs to be used in downstream traffic is left open, but is not practically implementable as the (downstream) routing information of the HofAs should be maintained and updated globally.

By choosing the HofA from the new AR's subnet address range, the HofA could be extended for downstream traffic. Now the scheme becomes more complicated as ARs must inform all the MNs of the HofAs used so that the MN's new CoA cannot be from the HofA range. The HofA range could be static and predefined in all subnets or it could be flexible and modifiable by the network operator. When the HofA is from the AR's address range, ingress filtering at the incoming interface is not a problem.

As fast handover in one direction is only half of the handover delay problem, faster *upstream* handover need to be implemented within a fast *downstream* handover method. FFHMIPv6 is used as an example, but the proposal to use separate addresses for handovers could also be implemented with other methods.

The proposed method requires that every device in IPv6 sub-network knows the range of addresses (i.e. the HofA range) that cannot be used when forming CoAs with stateless address autoconfiguration [74]. Stateful address autoconfiguration using Dynamic Host Configuration Protocol version 6 (DHCPv6) [13]

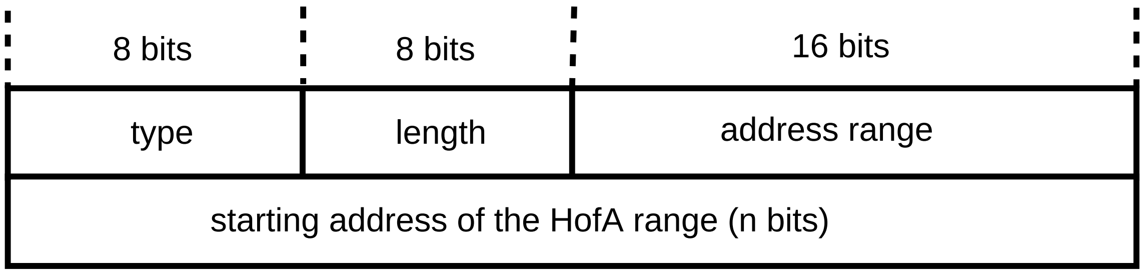


FIGURE 4 Example of RA message's new option type, with the HofA range

consumes too much time as the DHCPv6 server can be located anywhere within the boundary of the organization and the traffic delays behind the remote sites can be very high (typically up to 100 ms). Also, using DHCPv6 to distribute a HofA is too slow, because the protocol includes a negotiation phase which is too time consuming. Server software functions will also delay the process.

Each AR should have a pool of HofAs which cannot be assigned as CoAs. The ARs include the HofA address range in a new option type *HofA option* (Figure 4) in its Router Advertisement (RA) message [52]. The prefix part of the HofA is obtained from the *prefix information* option of the RA. This enables a flexible approach as there can be a different number of HofAs in different subnets. It could be possible to let the Mobile Node choose a handover address, by including

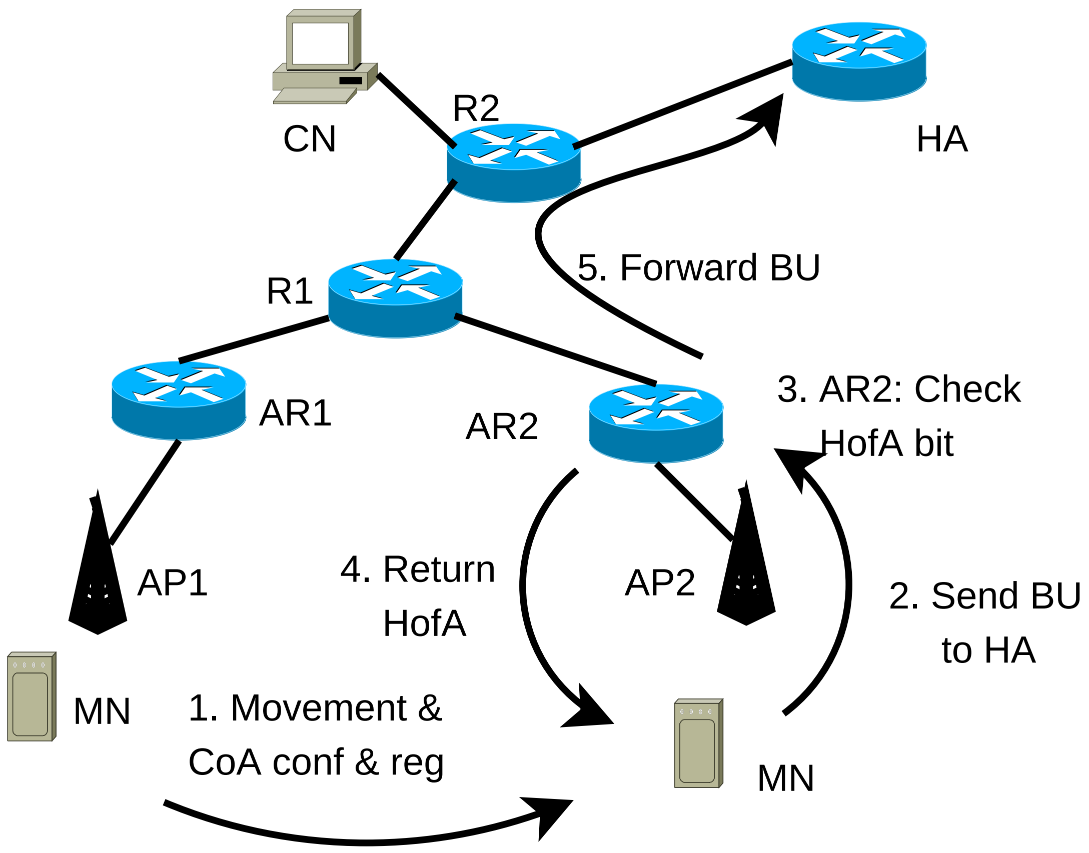


FIGURE 5 Assignment of the HofA

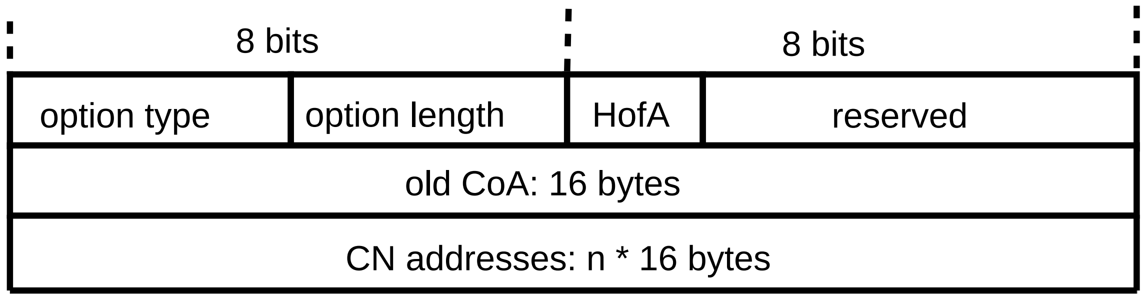


FIGURE 6 Example of the Hop-by-Hop header with a HofA Request bit

in the option type the number of handover addresses that are available. This could be implemented e.g. by dividing the 16 bit field *address range* to two fields *address range* and *available address range*. Now the Mobile Nodes choose the largest address from the available addresses. When MN's Binding Update message goes through the AR, the AR updates its RA message. The available address range is based on the smallest Handover address which is still in use.

In FFHMIPv6, when the MN moves to a new logical subnet, it receives a RA which includes the range for the HofAs. The MN forms a new CoA (that is outside the HofA range) (Figure 5) and checks with Duplicate Address Detection (DAD) that the new CoA is unique. The CoA registration process could be speeded up for example by using Optimistic Duplicate Address Detection (oDAD) [50], where the MN can send its BU message immediately after sending a Neighbor Solicitation to determine if its new address is already in use. This would be desirable as we can use the unique HofA, even if the DAD process fails.

In FFHMIPv6, the BU message contains a *Hop-by-Hop* frame, including the old CoA and the addresses of the CNs, and with that information, the *downstream* traffic flows from the CNs can be rerouted to the MN's new destination when a Crossover Router (the first router where the flow to the old CoA is found) is discovered **PX**. However, for the MN to send upstream data in FFHMIPv6 (and in MIPv6), the MN must wait for a BACK message from the HA (response to the BU message) to ensure that its new CoA is bound with its Home Address (HoA).

In the FFHMIPv6 method the MN must maintain the CoA from the last visited subnet, as it is used for tunneling the downstream traffic during the BU process. A *HofA request* bit is added to the BU message (*Hop-by-Hop* frame) (Figure 6), which is used to inform the new AR (AR2 in Figure 5) that the MN needs a HofA (e.g. the MN has delay critical upstream traffic). So the HofA is assigned only when requested, because in some cases, the MN making the handover might not have any delay critical upstream traffic to be delivered. The assigning of the HofA could be possible e.g. for certain Quality of Service (QoS) classes that are used to transfer time critical data traffic.

The functionality of the FFHMIPv6 method (with the fast upstream handover) and the sequence of packets during the BU process is shown in Figure 7. When the MN's BU message arrives at the new AR, the AR checks the *HofA request* field and if the value is 1, the AR sends one of the free HofAs in a BACK

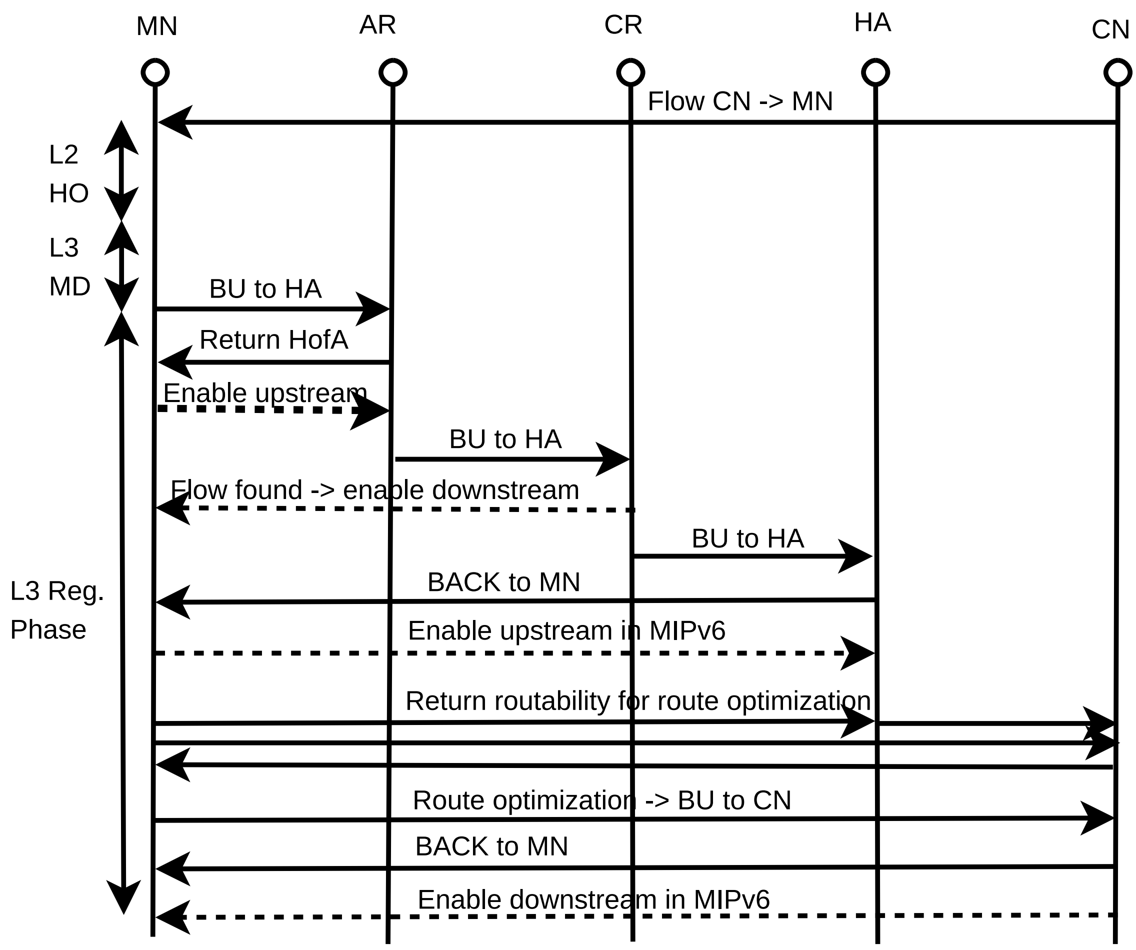


FIGURE 7 Flow chart of the FFHMIPv6 with Fast Upstream

message to the MN to be used for upstream traffic. The BACK message [28] should indicate (e.g. with the *status* field) that it contains a HofA in its *mobility options* field (Figure 8), and so the MN must wait a BACK message from the HA (with the same sequence number) before the MN can use the new CoA.

After obtaining the HofA, the MN can send upstream traffic flow via an IPv6-in-IPv6 tunnel. The MN encapsulates its upstream traffic flow (destination address: CN address, source address: old CoA) to a packet with destination as the new AR address and source as the HofA. When the "actual" BACK message

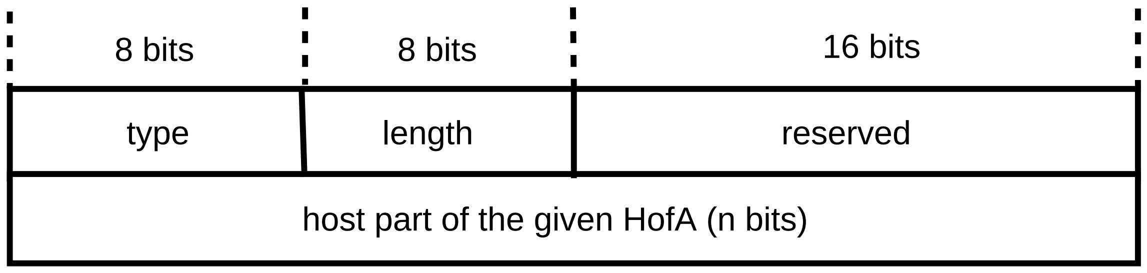


FIGURE 8 Example of Binding Acknowledgment's mobility option field

arrives from the HA, the MN starts to use the new CoA that is ensured to be unique by the DAD process and bound with the MN's HoA. The new AR releases the MN's HofA (places it in a pool of unused HofAs), after a specified timer (life time of the HofA) has expired.

The operation of the FFHMIPv6 method with the fast upstream addition was verified by simulation experiments in **PXIV** and **PXV** where it is compared to MIPv6 in Network Simulator 2 (ns2) [53]. The packet loss of CBR UDP traffic during L3 handover was studied for downstream and upstream traffic as a function of number of mobile nodes per base station, and FFHMIPv6 with fast upstream addition clearly outperforms MIPv6 in this respect. The obtained bandwidth was studied in case of FTP traffic using TCP protocol. In this case the proposed method again outperforms MIPv6, but the difference is small because TCP regulates the amount of bytes sent according to network capabilities. Also, as BU delay time is insignificant compared to the total simulation time, it plays only a small role in impact on FTP bandwidth. With VoIP traffic the ratio of received data to transmitted data was studied and again FFHMIPv6 with fast upstream addition outperformed MIPv6

3.3 Discussion of the results

From theoretical and experimental results it can be concluded that FFHMIPv6 performs very well compared to Mobile IPv6. The downstream redirection requires a hierarchically built network to be most effective, but that is the most popular network topology that exists today. Also, even if Crossover Router is not found, FFHMIPv6 performs as efficiently as MIPv6. The downstream handover performance difference compared to HMIPv6 is not that notable, because MAP is always located in the same domain as the MN, thus probably being more closer than the HA to a moving host. However, MAP could also be located quite far, which would linearly increase the overall handover delay. However, the FFHMIPv6 does not present a single point of failure, as does the MAP in HMIPv6, being more robust. FMIPv6 is a quite effective method, providing truly seamless handovers, but its performance is dependent on anticipation of the handovers, which is an unwanted feature and makes the system more complex. The fast upstream extension to the FFHMIPv6 shortens the upstream handover delay significantly. The MN does not have to wait for the BACK from the HA, as the round-trip time to AR is sufficient (depending of the HofA acquiring mode). A performance improvement to MIPv6 was significant and should be noticed also if the fast upstream is compared with other methods, such as HMIPv6 or FMIPv6.

4 CONCLUSIONS

Nowadays the pricing of Internet is homogenous for all kinds of users and with today's broadband connections the network is becoming more and more congested. By using different service classes for applications that require different quality and by pricing the usage accordingly Quality of Service requirements can be met. A node in a network commonly forms a bottleneck as it must deliver traffic from several sources to several destinations. Several packet scheduling schemes have been proposed in literature to schedule packets in multiple service class networks, but these schemes do not consider pricing. In this thesis several methods for packet scheduling are presented that guarantee delay or bandwidth for the customers based on some pricing function. The pricing function also gives optimal weights for a packet scheduler, and that maximizes the service providers revenue.

Guaranteeing Quality of Service to customers that are mobile and thus changing the point of attachment to the network is more complicated than in wire line networks. The future protocol Mobile IPv6 provides good support for mobility and independence of the technology underneath used for the wireless access. However, MIPv6 lags in efficiency in handover situations and several methods for decreasing the handover delay have been proposed in the literature. These, however, add complexity to the network, and often depend on lower layers i.e. the underlying technology. This thesis presents a method that is based on the flow information in routers, is independent of the underlying technology and needs only little modifications, while providing decreased handover delays for the downstream traffic. For minimizing the upstream traffic a new address type that is reserved for handover purposes is proposed. I also propose that when the network has multiple service classes and the packet scheduling is based on pricing, the fast handover methods can only be used in class(es) that are used for delay critical traffic. Other classes then use the basic MIPv6 method.

In future work, several Quality of Service parameters are to be combined with a suitable pricing function. Also, the simulation studies are to be extended to investigate packet scheduling and pricing in Mobile IPv6 environment with the proposed enhancements.

REFERENCES

[1] F. Baker and P. Savola, "Ingress Filtering for Multihomed Networks," IETF RFC 3704, Mar. 2004.

[2] J. C. R. Bennett and H. Zhang, "WF^{2}Q: worst-case fair weighted fair queueing," in *Proc. 15th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'96)*, vol. 1, 1996, pp. 120–128.

[3] J. C. R. Bennett and H. Zhang, "Hierarchical packet fair queueing algorithms," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, pp. 675–689, Oct. 1997.

[4] C. Castelluccia, "A Hierarchical Mobility Management Scheme for IPv6," in *Proc. Third IEEE Symposium on Computers and Communications 1998 (ISCC '98)*, 1998, pp. 305–309.

[5] C. Castelluccia, "HMIPv6: A Hierarchical Mobile IPv6 Proposal," *ACM Mobile Computing and Communications Review*, vol. 4, no. 1, pp. 48–59, Jan. 2000.

[6] W. Chen, E. Lin, and H. Wei, "Dynamic Location Control for Mobile Nodes," Department of Computer Science and Engineering, Southern Methodist University, Technical Report 97-CSE-10, 1997.

[7] S. Y. Cheung and C. S. Pencea, "BSFQ: bin sort fair queueing," in *Proc. 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'02)*, vol. 3, June 2002, pp. 1640–1649.

[8] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang, "Pricing in computer networks: Motivation, formulation and example," *IEEE/ACM Trans. Networking*, vol. 1, no. 6, pp. 614–627, Dec. 1993.

[9] G. Daley, B. Pentland, and R. Nelson, "Effects of Fast Router Advertisement on Mobile IPv6 Handovers," in *Proc. Eighth IEEE International Symposium on Computers and Communication 2003 (ISCC 2003)*, 2003, pp. 557–562.

[10] K. Danielsen and M. Weiss, "User control modes and IP allocation," *Internet Economics*, pp. 305–322, 1997.

[11] J. R. Davin and A. T. Heybey, "A simulation study of fair queueing and policy enforcement," *SIGCOMM Comput. Commun. Rev.*, vol. 20, no. 5, pp. 23–29, 1990.

[12] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queuing algorithm," *Journal of Internetworking Research and Experience*, vol. 1, no. 1, pp. 3–26, Sept. 1990.

[13] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," IETF RFC 3315, July 2003.

[14] T. Ernst, C. Castelluccia, and H.-Y. Lach, "Extending Mobile-IPv6 with Multicast to Support Mobile Networks in IPv6," in *Proc. 1st European Conference on Universal Multiservice Networks 2000 (ECUMN 2000)*, 2000, pp. 114–121.

[15] R. Gibbens, R. Mason, and R. Steinberg, "Internet service classes under competition," *IEEE J. Select. Areas Commun.*, vol. 18, no. 12, pp. 2490–2498, Dec. 2000.

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

[17] S. J. Golestani, "A self-clocked fair queueing scheme for broadband applications," in *Proc. 13th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'94)*, vol. 2, 1994, pp. 636–646.

[18] P. Goyal and H. M. Vin, "Generalized guaranteed rate scheduling algorithms: A framework," *IEEE/ACM Trans. Networking*, vol. 5, no. 4, pp. 561–571, Aug. 1997.

[19] P. Goyal, H. M. Vin, and H. Cheng, "Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, pp. 690–704, Oct. 1997.

[20] A. G. Greenberg and N. Madras, "How fair is fair queuing?" *J. ACM*, vol. 39, no. 3, pp. 568–598, July 1992.

[21] A. Gubta, D. O. Stahl, and A. B. Whinston, "A stochastic equilibrium model of internet pricing," *Journal of Economic Dynamics and Control*, vol. 21, no. 4–5, pp. 697–722, May 1997.

[22] C. Guo, "SRR: an $O(1)$ time complexity packet scheduler for flows in multiservice packet networks," in *Proc. ACM SIGCOMM'01*. ACM Press, 2001, pp. 211–222.

[23] C. Guo, "SRR: an $O(1)$ time-complexity packet scheduler for flows in multiservice packet networks," *IEEE/ACM Trans. Networking*, vol. 12, no. 6, pp. 1144–1155, Dec. 2004.

[24] Y. Gwon, J. Kempf, and A. Yegin, "Scalability and Robustness Analysis of Mobile IPv6, Fast Mobile IPv6, Hierarchical Mobile IPv6, and Hybrid IPv6 Mobility Protocols Using a Large-scale Simulation," in *Proceedings of IEEE International Conference on Communications*, vol. 7, June 2004, pp. 4087–4091.

[25] R. Hsieh and A. Seneviaratne, "A Comparison of Mechanisms for Improving Mobile IP Handoff Latency for End-to-End TCP," in *Proc. 9th Annual International Conference on Mobile Computing and Networking 2003 (Mobicom'03)*, 2003, pp. 29–41.

[26] S.-H. Hwang, B.-K. Lee, Y.-H. Han, and C.-S. Hwang, "An Adaptive Hierarchical Mobile IPv6 with Route Optimization," in *Proc. 57th IEEE Semiannual Vehicular Technology Conference 2003 (VTC'S03)*, vol. 3, 2003, pp. 1502–1506.

[27] S. Jaiswal and S. Nandi, "Simulation-based Performance Comparison of TCP-variants over Mobile IPv6-based Mobility Management Schemes," in *Proc. 29th Annual IEEE International Conference Local Computer Networks*, Nov. 2004, pp. 284–291.

[28] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," IETF RFC 3775, June 2004.

[29] N. Jordan and A. Poropatich, "Fast Mobile IPv6 Approach for Wireless LAN based Networks," in *Proc. 1st International Conference on E-business and TElecommunication Networks (ICETE2004)*, vol. 3, Aug. 2004, pp. 102–108.

[30] H. Jung, E. Kim, J. Yi, and H. Lee, "A Scheme for Supporting Fast Handover in Hierarchical Mobile IPv6 Networks," *ETRI Journal*, vol. 27, no. 6, pp. 798–801, Dec. 2005.

[31] H. Jung and S. Koh, "Fast handover support in hierarchical mobile IPv6," in *Proc. The 6th International Conference on Advanced Communication Technology (ICACT'04)*, vol. 2, June 2004, pp. 551–554.

[32] S. S. Kanhere and H. Sethu, "Fair, efficient and low-latency packet scheduling using nested deficit round robin," in *Proc. IEEE Workshop on High Performance Switching and Routing*, 2001, pp. 6–10.

[33] S. S. Kanhere and H. Sethu, "Low-latency guaranteed-rate scheduling using elastic round robin," *Computer Communications*, vol. 25, no. 14, pp. 1315–1322, Sept. 2002.

[34] S. S. Kanhere, H. Sethu, and A. B. Parekh, "Fair and efficient packet scheduling using elastic round robin," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 3, pp. 324–336, Mar. 2002.

[35] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE J. Select. Areas Commun.*, vol. 9, no. 8, pp. 1265–1279, Oct. 1991.

[36] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, Jan. 1997.

[37] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, Mar. 1998.

[38] N. J. Keon and G. Anandalingam, "Optimal pricing for multiple services in telecommunications networks offering quality-of-service guarantees," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 66–80, Feb. 2003.

[39] R. R. Kompella and G. Varghese, “Reduced state fair queuing for edge and core routers,” in *Proc. ACM NOSSDAV’04*, 2004, pp. 100–105.

[40] R. Koodli, “Fast Handovers for Mobile IPv6,” IETF RFC 4068, July 2005.

[41] L. Lenzini, E. Mingozzi, and G. Stea, “Aliquem: a novel DRR implementation to achieve better latency and fairness at O(1) complexity,” in *Proc. IEEE International Workshop on Quality of Service (IWQoS’02)*, 2002, pp. 77–86.

[42] L. Lenzini, E. Mingozzi, and G. Stea, “Tradeoffs between low complexity, low latency, and fairness with deficit round-robin schedulers,” *IEEE/ACM Trans. Networking*, vol. 12, no. 4, pp. 681–693, Aug. 2004.

[43] J. K. MacKie-Mason and H. R. Varian, *Public access to the Internet*. Cambridge, Mass.: MIT Press, 1995, ch. Pricing the Internet, pp. 269–314.

[44] P. Marbach, “Analysis of a static pricing scheme for priority services,” *IEEE/ACM Trans. Networking*, vol. 12, no. 2, pp. 312–325, Apr. 2004.

[45] L. Massoulié and J. Roberts, “Bandwidth sharing: objectives and algorithms,” *IEEE/ACM Trans. Networking*, vol. 10, no. 3, pp. 320–328, June 2002.

[46] P. E. McKenney, “Stochastic fairness queueing,” *Internetworking: Research and Experience*, vol. 2, pp. 113–131, Jan. 1991.

[47] “MIPL Mobile IPv6 for Linux,” http://www.mobile-ipv6.org, 2005.

[48] N. Montavont and T. Noel, “Handover Management for Mobile Nodes in IPv6 Networks,” *IEEE Communications Magazine*, vol. 40, pp. 38–43, Aug. 2002.

[49] N. Montavont and T. Noel, “Analysis and Evaluation of Mobile IPv6 Handovers over Wireless LAN,” *Mobile Networks and Applications: Journal on special issues on Mobility of Systems, Users, Data and Computing*, vol. 8, no. 6, pp. 643–653, Dec. 2003.

[50] N. Moore, “Optimistic Duplicate Address Detection for IPv6,” IETF RFC 4429, Apr. 2006.

[51] J. B. Nagle, “On packet switches with infinite storage,” *IEEE Trans. Commun.*, vol. 35, no. 4, pp. 435–438, Apr. 1987.

[52] T. Narten, E. Nordmark, and W. Simpson, “Neighbour Discovery for IP version 6 (IPv6),” IETF RFC 2461, Dec. 1998.

[53] “The UCB/LBNL/VINT Network Simulator – ns (version 2),” http://www.isi.edu/nsnam/ns/, 2005.

[54] A. M. Odlyzko, “Paris metro pricing for the internet,” in *ACM Conference on Electronic Commerce (EC’99)*, 1999, pp. 140–147.

[55] K. Omae, T. Ikeda, M. Inoue, I. Okajima, and N. Umeda, "Mobile Node Extension Employing Buffering Function to Improve Handoff Performance," in *Proc. The 5th International Symposium on Wireless Personal Multimedia Communications 2002*, vol. 1, 2002, pp. 62–66.

[56] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[57] I. C. Paschalidis and Y. Liu, "Distributed resource allocation in multiservice communication networks using pricing," in *Proc. 41st IEEE Conference on Decision and Control*, vol. 2, 2002, pp. 2023–2028.

[58] I. C. Paschalidis and Y. Liu, "Pricing in multiservice loss networks: Static pricing, asymptotic optimality, and demand substitution effects," *IEEE/ACM Trans. Networking*, vol. 10, no. 3, pp. 425–438, June 2002.

[59] I. C. Paschalidis and J. N. Tsitsiklis, "Congestion-dependent pricing of network services," *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 171–183, Apr. 2000.

[60] L. Patanapongpibul and G. Mapp, "A Client-Based Handoff Mechanism for Mobile IPv6 Wireless Networks," in *Proc. Eighth IEEE International Symposium on Computers and Communication 2003 (ISCC 2003)*, 2003, pp. 563–568.

[61] J. M. Peha and F. A. Tobagi, "Cost-based scheduling and dropping algorithms to support integrated services," *IEEE Trans. Commun.*, vol. 44, no. 2, pp. 192–202, Feb. 1996.

[62] X. Perez-Costa, M. Torrent-Moreno, and H. Hartenstein, "A Performance Comparison of Mobile IPv6, Hierarchical Mobile IPv6, Fast Handovers for Mobile IPv6 and Their Combination," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 4, pp. 5–19, Oct. 2003.

[63] S. Ramabhadran and J. Pasquale, "Stratified round robin: A low complexity packet scheduler with bandwidth fairness and bounded delay," in *Proc. ACM SIGCOMM'03*. ACM Press, 2003, pp. 239–250.

[64] R. Ramjee, K. Varadhan, L. Salgarelli, S. Thuel, S.-Y. Wang, and T. L. Porta, "HAWAII: a Domain-Based Approach for Supporting Mobility in Wide-area Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 10, pp. 396–410, June 2002.

[65] S. Shenker, "Fundamental design issues for the future internet," *IEEE J. Select. Areas Commun.*, vol. 13, no. 7, pp. 1176–1188, Sept. 1995.

[66] S. Shenker, D. Clark, D. Estrin, and S. Herzog, "Pricing in computer networks: reshaping the research agenda," *SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 2, pp. 19–43, Apr. 1996.

[67] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," *IEEE/ACM Trans. Networking*, vol. 4, no. 3, pp. 375–385, June 1996.

[68] J. Shu and P. Varaiya, "Pricing network services," in *Proc. 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03))*, vol. 2, 2003, pp. 1221–1230.

[69] H. Soliman, C. Castelluccia, K. El-Malki, and L. Bellier, "Hierarchical Mobile IPv6 Mobility Management (HMIPv6)," IETF RFC 4140, Aug. 2005.

[70] D. Stiliadis and A. Varma, "Efficient fair queueing algorithms for packet-switched networks," *IEEE/ACM Trans. Networking*, vol. 6, no. 2, pp. 175–185, Apr. 1998.

[71] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high-speed networks," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 33–46, Feb. 2003.

[72] S. Suri, G. Varghese, and G. Chandranmenon, "Leap forward virtual clock: a new fair queuing scheme with guaranteed delays and throughput fairness," in *Proc. 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'97))*, vol. 2, 1997, pp. 557–565.

[73] V. Thing, H. Lee, and Y. Xu, "Performance Evaluation of Hop-by-Hop Local Mobility Agents Probing for Mobile IPv6," in *Proc. Eighth IEEE International Symposium on Computers and Communication 2003 (ISCC 2003)*, 2003, pp. 576–581.

[74] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," IETF RFC 1971, Aug. 1996.

[75] S.-C. Tsao and Y.-D. Lin, "Pre-order deficit round robin: a new scheduling algorithm for packet-switched networks," *Comput. Networks*, vol. 35, no. 2-3, pp. 287–305, Feb. 2001.

[76] I. Vivaldi, B. Ali, H. Habaebi, V. Prakash, and A. Sali, "Routing Scheme for Macro Mobility Handover in Hierarchical Mobile IPv6 Network," in *Proc. 4th National Conference on Telecommunication Technology 2003 (NCTT 2003)*, 2003, pp. 88–92.

[77] F. Wang, M. Papaefthymiou, and M. Squillante, "Performance evaluation of gang scheduling for parallel and distributed multiprogramming," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson and L. Rudolph, Eds. Springer Verlag, 1997, pp. 277–298. [Online]. Available: citeseer.ist.psu.edu/article/wang97performance.html

[78] Q. Wang, J. M. Peha, and M. Sirbu, "The design of an optimal pricing scheme for ATM integrated service networks," *Journal of Electronic Publishing, Special Issue on Internet Economics*, vol. 2, no. 1, May 1996.

[79] X. Yuan and Z. Duan, "FRR: a proportional and worst-case fair round robin scheduler," in *Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, vol. 2, 2005, pp. 831–842.

[80] L. Zhang, "Virtual clock: a new traffic control algorithm for packet switching networks," in *Proc. ACM SIGCOMM'90*, 1990, pp. 19–29.

[81] L. Zhang, "Virtualclock: a new traffic control algorithm for packet-switched networks," *ACM Trans. Comput. Syst.*, vol. 9, no. 2, pp. 101–124, May 1991.