MODERN BROADBAND NETWORKING ISSUES, QUALITY-OF-SERVICE, RESOURCE ALLOCATION, AND SERVICE DISCIPLINES

A Thesis in

Electrical Engineering

by

Samer Marwan Taha

© 2002 Samer Marwan Taha

Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

May 2002
We approve the thesis of Samer Marwan Taha.

Mohsen Kavehrad
Ameritech (W.L. Weiss) Professor of Electrical Engineering
Thesis Advisor
Chair of Committee

John J. Metzner
Professor of Electrical Engineering and Computer and Science Engineering

John F. Doherty
Associate Professor of Electrical Engineering

George Kesidis
Associate Professor of Electrical Engineering and Computer Science and Engineering

Natarajan Gautam
Assistant Professor of Industrial Engineering

W. Kenneth Jenkins
Professor of Electrical Engineering
Head of the Department of Electrical Engineering
ABSTRACT

This thesis focuses on the analysis, performance evaluation, and design of several aspects of the global data network beyond year 2000. The main issue in this study is the provisioning of Quality-of-Service (QoS) in an efficient way. The efficiency of some already-developed or proposed broadband networking techniques are investigated in terms of their ability and reliability in delivering QoS guaranteed services. Mainly, Multi-Protocol Over ATM (MPOA), Differentiated Services (DS), and Multi-Protocol Label Switching (MPLS) technologies are frequently discussed in this thesis. Also, other techniques are proposed, analyzed, and compared to some existing techniques.

The work in this thesis focuses mainly on second and third layer issues, with respect to the OSI seven layers model. This thesis contributes to a better understanding of the interaction and relations between the concepts of QoS, service disciplines, pricing, and revenues. Optimizing network systems operation is difficult without considering these concepts and the interaction between them.

Toward these objectives, the thesis starts with analyzing the performance of MPOA. The difference in the efficiency of this technique when implemented in Local Area Networks (LANs) or Wide Area Networks (WANs) is investigated. The caching operation and the cache table are recognized as the bottleneck in MPOA technique. Thus, a new cache table management system based on neural network technique is proposed and evaluated.
The thesis continues by investigating the usefulness of a Multi-Class Internet, specifically a bandwidth segregation approach is considered and analyzed. A special emphasis is exercised on the importance of connection-oriented networking techniques in enabling guaranteed QoS parameters. For this objective, a new technique for emulating connection-oriented services over IP layer is proposed and compared to the existing technique in an MPLS architecture.

Resource allocation may be considered as the most fundamental operation for the success of any multi-class network. Dynamic Bandwidth Allocation (DBA) operation is analyzed at the call level, controlling the call blocking rate at different Classes-of-Service (CoSs) is considered as the objective. A novel DBA algorithm called Virtual Demand Distribution (VDD) is proposed and evaluated. The VDD algorithm utilizes signaling messages in a connection-oriented multi-class network to achieve more accurate prediction of demands in a non-stationary environment.

An attempt is made to extend the multi-class concept to mobile wireless networks. Preliminary ideas are proposed to design fading-aware TDMA multi-class mobile wireless local area networks. However, a detailed Multiple Access Control mechanism is proposed to create a two-class Packet Reservation Multiple Access (PRMA) protocol. A revenue comparison framework is developed, and the efficiency of this new protocol is evaluated based on this framework. This framework clarifies the meaning and interaction between the concepts of QoS metric, QoS score, service disciplines, willingness-to-pay, and revenue.
The thesis ends by a brief study of a very practical form of dynamic pricing, namely; the call-setup dynamic pricing. The problem is modeled as a discounted Markov decision process and $\varepsilon$-optimal policies are found for several scenarios.
TABLE OF CONTENTS

LIST OF FIGURES ............................................................................................................. x

LIST OF TABLES .............................................................................................................. xiii

LIST OF SYMBOLS AND ACRONYMS ........................................................................ xiv

ACKNOWLEDGMENTS ..................................................................................................... xvii

Chapter 1 INTRODUCTION ........................................................................................ 1
  1.1 Motivation ...................................................................................................... 1
  1.2 Objective ....................................................................................................... 4
  1.3 Thesis Overview ........................................................................................... 10
  1.4 List of Contributions .................................................................................... 12

Chapter 2 BACKGROUND ........................................................................................... 14
  2.1 Asynchronous Transfer Mode (ATM) Architecture ....................................... 18
  2.2 Differentiated Services (DS) Architecture ...................................................... 20
  2.3 Multi-Protocol Label Switching (MPLS) Architecture .................................. 22
  2.4 Multi-Protocol Over ATM (MPOA) Architecture .......................................... 24

Chapter 3 MPOA FLOW CLASSIFICATION DESIGN AND ANALYSIS
    BASED ON NEURAL NETWORK TECHNIQUE ...................................................... 26
  3.1 Introduction .................................................................................................. 26
  3.2 Framework ................................................................................................... 29
    3.2.1 A Finite State Machine per Flow .......................................................... 30
    3.2.2 MPOA Resource Classification ............................................................. 33
3.2.3 Cost Function and Design Objective .............................................. 36
3.3 Flow Classification Design and The Self-Learning System .............. 38
  3.3.1 Static FC Algorithms ................................................................. 38
  3.3.2 Design Complexity Reduction .................................................... 40
  3.3.3 The Neural Network Design ....................................................... 46
    3.3.3.1 How Does The System Work? ............................................. 46
    3.3.3.2 The Neural Network Model and The Learning Strategy .......... 48
    3.3.3.3 The LRU-Based Adaptive CTM Algorithm ......................... 51
  3.3.4 Performance Analysis .............................................................. 53
    3.3.4.1 Simulation Results .............................................................. 53
    3.3.4.2 Complexity Analysis ............................................................ 57
3.4 Conclusion ..................................................................................... 58

Chapter 4 MULTI-CLASS INTERNET WITH CONNECTION-ORIENTED SERVICES, A NEW TECHNIQUE ................................................. 63
  4.1 Introduction ................................................................................... 63
  4.2 Background .................................................................................. 66
  4.3 The Multi-Class Internet ............................................................... 70
  4.4 CAC and Connection Establishment .............................................. 74
  4.5 Implementation Complexity of Proposed Multi-Class Internet and Protocols Emulating Connection-Oriented sessions .................... 81
  4.6 Conclusion ................................................................................... 84

Chapter 5 DYNAMIC BANDWIDTH ALLOCATION IN MULTI-CLASS CONNECTION-ORIENTED NETWORKS .............................................. 86
  5.1 Introduction ................................................................................... 86
  5.2 Single-Link Analysis ..................................................................... 90
    5.2.1 Call-Blocking Probability For One CoS of Stationary Traffic with Variable Size Bandwidth Requests .............................................. 91
    5.2.2 Controlling Call-Blocking Probabilities For Two CoSs of Non-Stationary Traffic with Variable Size Bandwidth Requests ............... 94
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3 Loss Network Analysis</td>
<td>100</td>
</tr>
<tr>
<td>5.4 The Virtual Demand Distribution (VDD) Algorithm</td>
<td>104</td>
</tr>
<tr>
<td>5.5 Performance Analysis</td>
<td>114</td>
</tr>
<tr>
<td>5.5.1 Simulation Settings</td>
<td>115</td>
</tr>
<tr>
<td>5.5.2 Simulation Results</td>
<td>116</td>
</tr>
<tr>
<td>5.5.3 Tradeoffs Regarding The VDD Algorithm</td>
<td>118</td>
</tr>
<tr>
<td>5.6 Conclusion</td>
<td>119</td>
</tr>
<tr>
<td>Chapter 6 MULTI-CLASS TIME DIVISION MULTIPLE ACCESS LOCAL AREA NETWORKS</td>
<td>129</td>
</tr>
<tr>
<td>6.1 Review Of MAC Layer in a TDMA Local Area Network (LAN)</td>
<td>129</td>
</tr>
<tr>
<td>6.2 Two Class Packet Reservation Multiple Access, Motivations and Design</td>
<td>134</td>
</tr>
<tr>
<td>6.2.1 Introduction</td>
<td>134</td>
</tr>
<tr>
<td>6.2.2 Revenue Capacity</td>
<td>136</td>
</tr>
<tr>
<td>6.2.3 Revenue Comparison Framework</td>
<td>144</td>
</tr>
<tr>
<td>6.2.4 Two Class PRMA Architecture</td>
<td>147</td>
</tr>
<tr>
<td>6.2.4.1 Background</td>
<td>147</td>
</tr>
<tr>
<td>6.2.4.2 System Model and Protocols for TCPRMA</td>
<td>148</td>
</tr>
<tr>
<td>6.2.4.2.1 Speech model</td>
<td>148</td>
</tr>
<tr>
<td>6.2.4.2.2 Packet loss</td>
<td>149</td>
</tr>
<tr>
<td>6.2.4.2.3 Multiple access and contention protocols</td>
<td>149</td>
</tr>
<tr>
<td>6.2.5 Performance Analysis of TCPRMA</td>
<td>151</td>
</tr>
<tr>
<td>6.3 Conclusion</td>
<td>159</td>
</tr>
<tr>
<td>Chapter 7 CALL-SETUP DYNAMIC PRICING FOR A LINK WITH VARIABLE-SIZE ARRIVALS, OPTIMAL POLICY</td>
<td>161</td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>161</td>
</tr>
<tr>
<td>7.2 System Description</td>
<td>164</td>
</tr>
<tr>
<td>7.3 Modeling The System As a Discounted Markov Decision Process</td>
<td>167</td>
</tr>
<tr>
<td>7.3.1 State Space</td>
<td>168</td>
</tr>
<tr>
<td>7.3.2 Decision Epochs</td>
<td>169</td>
</tr>
</tbody>
</table>
7.3.3 Actions........................................................................................................169
7.3.4 Reward Function ........................................................................................170
7.3.5 Transition Probabilities .............................................................................170
7.3.6 Optimality Equations ................................................................................173
7.4 Results And Analysis For Different Scenarios ..............................................173
7.5 Conclusion .....................................................................................................182

Chapter 8 SUMMARY AND CONCLUSIONS.........................................................184
8.1 Summary of Results........................................................................................184
8.2 Avenues For Further Research .......................................................................188
  8.2.1 Fading-Aware TDMA Multi-Class Wireless LAN, Preliminary Ideas ..........189
    8.2.1.1 Proposed Modifications On The MAC Layer Of A TDMA Wireless LAN System To Emulate A Multi-Class Network ..........192
    8.2.1.2 Proposed Modifications To The Network Layer Of A TDMA LAN System To Emulate A Multi-Class Network ............199
8.3 List of Publications and Reports ..................................................................201

BIBLIOGRAPHY.......................................................................................................203
LIST OF FIGURES

Figure 3–1: MPOA network and its shortcut setup protocols. ........................................31

Figure 3–2: Finite state machine for flow state in a MPC........................................32

Figure 3–3: Performance comparison of the X/Y/T algorithm with the X/Z/T algorithm.................................................................42

Figure 3–4: Sensitivity of G to Z, X, and T. Upper ones for lbl-trace, and lower ones for cisco-trace. (a), (d): X = 20, T = 30; (b), (e): Z = 15, T = 30; (c), (f): Z = 15, X = 20. .........................................................................................................43

Figure 3–5: Schematic view of switching gain G v.s. timeout T. ................................45

Figure 3–6: The SLCS architecture..............................................................................49

Figure 3–7: Neural Network architecture....................................................................50

Figure 3–8: Switching gain G for both the LRU-based algorithm and the SLCS......60

Figure 3–9: Switching gain G for the SLCS at β = 1. ................................................61

Figure 3–10: Switching gain G v.s. flow cache table size for fixwest-trace............62

Figure 4–1: End-to-end dropping probability ratio of two CoSs versus number of hops.................................................................................................................................68

Figure 4–2: Bandwidth segregation and sharing in the proposed architecture. ........72

Figure 4–3: Three overlapping networks sharing the same physical resources. ......74

Figure 4–4: Architecture of Multi-Class Internet and the mechanism of emulating connection-oriented services over an IP layer. .........................................................80

Figure 4–5: A schematic view for an example of a stack of output ports’ ID-numbers and the corresponding required number of bits for a possible 11-hop connection..................................................................................................................83

Figure 5–1: The configuration of the network used in simulating the VDD and other algorithms, (simulation tool: Bones Designer).........................................................120
Figure 5–2: A schematic view of a network that shows two trees only. ......................... 121
Figure 5–3: Blocking rates under the VDD at link 4-5. .................................................. 122
Figure 5–4: Blocking rates under the SD at link 4-5.......................................................... 123
Figure 5–5: Utilizations and allocations under the VDD algorithm at link 4-5................. 124
Figure 5–6: Utilizations and allocations under the VDD algorithm at link 5-7............... 125
Figure 5–7: Utilizations and allocations under the SD algorithm at link 5-7.................... 126
Figure 5–8: Performance of the VDD when link 5-6 was broken................................. 127
Figure 5–9: Performance of the SD algorithm when link 5-6 was broken..................... 128
Figure 6–1: A schematic view of TDMA frames used in wireless LANs....................... 131
Figure 6–2: A Markov model for voice activities at a voice terminal............................. 132
Figure 6–3: Demonstration of the relation between the $WTP$ function, the $QDCR$ function, and the $USP$ value for two applications and three users ....................... 141
Figure 6–4: QoS score, based on average of listeners ratings, versus $P_{loss}$ for voice-activated speech sessions .................................................................................. 155
Figure 6–5: Average $P_{loss}$ experienced by 2nd class users in the TCPRMA versus the number of 2nd class active sessions. Number of 1st class active sessions = 20. .......................................................... 156
Figure 6–6: Average $P_{loss}$ versus $P_p$ for the classical PRMA and the 2nd class of TCPRMA for a total number of users = 40. In TCPRMA, number of 1st class active sessions = 20. .......................................................... 157
Figure 6–7: Average $P_{loss}$ for 2nd class users in TCPRMA versus $P_p$ for different numbers of 1st class users while total number of users is fixed at 40. .............................. 158
Figure 6–8: $P_{loss}$ experienced by PRMA and 2nd class of TCPRMA versus total number of users, $P_p = 0.4$.................................................................................. 159
Figure 7–1: The three $f_a^k(.)$ functions used in this study; concave, linear, and convex .................................................................................................................. 167
Figure 7–2: Optimal policy assuming no class-2 or class-1 calls in the system, and convex $f_a^k(.)$ ........................................................................................................ 175
Figure 7–3: Optimal policy assuming no class-2 or class-1 calls in the system, and linear \( f^k_a(\cdot) \). .................................................................................................................................176

Figure 7–4: Optimal policy assuming no class-2 or class-1 calls in the system, and concave \( f^k_a(\cdot) \). ........................................................................................................................................177

Figure 7–5: Optimal CSC for class-3 arrivals versus class-3 utilization, for different values of \( \bar{\sigma} \). No class-1 or class-2 calls are in the system......................180

Figure 7–6: Optimal CSC for class-3 arrivals versus class-3 utilization, for different values of \( c_p \). No class-1 or class-2 calls are in the system. \( c_o = (1/6)(10^{-7}) \) $/sec/bps.........................................................................................................................181

Figure 7–7: Optimal CSC for class-1 arrivals when the total utilization is constant at 8Mbps, but the combination of class-1 and class-2 calls in the system varies. The policy is presented for different values of \( \mu_1 \). No class-3 calls are in the system. .........................................................................................182
Table 3-1: Average flow Statistics of both LAN and WAN traces..........................57
LIST OF SYMBOLS AND ACRONYMS

ABR  Available Bit Rate
ACK  Acknowledgment
ATM  Asynchronous Transfer Mode
BPS  Bit Per Second
CAC  Call Admission Control
CBR  Constant Bit Rate
CTM  Cache Table Management
CoS  Class-of-Service
CP   Complete Partitioning
CS   Complete Sharing
CSC  Call-Setup Charge
DBA  Dynamic Bandwidth Allocation
DoS  Denial-of-Service
DPRMA Dynamic Packet Reservation Multiple Access
DS   Differentiated Services
ED   Edge Device
E-LAN Emulated Local Area Network
FC   Flow Classification
FDD  Frequency Division Duplex
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEC</td>
<td>Forwarding Equivalence Class</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standardization Organization</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LRU</td>
<td>Least Recently Used</td>
</tr>
<tr>
<td>LSP</td>
<td>Label-Switched Path</td>
</tr>
<tr>
<td>MAC</td>
<td>Multiple Access Control</td>
</tr>
<tr>
<td>MCI</td>
<td>Multi-Class Internet</td>
</tr>
<tr>
<td>MDP</td>
<td>Markov Decision Process</td>
</tr>
<tr>
<td>MPC</td>
<td>Multi-Protocol Client</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multi-Protocol Label Switching</td>
</tr>
<tr>
<td>MPOA</td>
<td>Multi-Protocol Over ATM</td>
</tr>
<tr>
<td>MPS</td>
<td>Multi-Protocol Server</td>
</tr>
<tr>
<td>NACK</td>
<td>Negative Acknowledgment</td>
</tr>
<tr>
<td>NHRP</td>
<td>Next Hop Resolution Protocol</td>
</tr>
<tr>
<td>PACK</td>
<td>Positive Acknowledgment</td>
</tr>
<tr>
<td>PHB</td>
<td>Per-hop Behavior</td>
</tr>
<tr>
<td>PRMA</td>
<td>Packet reservation Multiple Access</td>
</tr>
<tr>
<td>QDCR</td>
<td>Quality-of-Service-Dependent Charging Rate</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality-of-Service</td>
</tr>
<tr>
<td>QS</td>
<td>Quality-of-Service Sensitivity</td>
</tr>
</tbody>
</table>
RC  Revenue Capacity
SD  Simple Distributed
SLCS  Self-Learning Cache table management System
SNR  Signal-to-Noise Ratio
TCP  Transport Control Protocol
TCPRMA  Two-Class Packet Reservation Multiple Access
TDD  Time Division Duplex
TDMA  Time Division Multiple Access
VBR  Variable Bit Rate
VCC  Virtual Channel Connection
VCI  Virtual Channel Identifier
VDD  Virtual Demand Distribution
VP  Virtual Path
VPCBR  Virtual Percentage Change in Blocking Rate
VPI  Virtual Path Identifier
VPT  Virtual Partitioning
WAN  Wide Area Network
WTP  Willingness-To-Pay
I would like to thank my thesis advisor, Dr. Mohsen Kavehrad, for his guidance and encouragement. His continuous scientific and moral support helped me a lot to have a clear vision and better understanding of this research. I would also like to thank the other members of my Ph.D. committee, Dr. John Doherty, Dr. Natarajan Gautam, Dr. George Kisidis, and Dr. John Metzner for their helpful comments and the constructive conversations we had, and Dr. Hao Che, a former member of my committee, who helped me in choosing and starting this research.

Special thanks go to my parents; my mother, Samia, who no words will ever express my appreciation for her infinite and endless love and support; and my father, Marwan, who I wouldn’t be able to continue my graduate study without his support and who was and still my model in success. A special thank you, also, goes to my dear wife, Muna, for her patience, love, and continuous support during all the days and nights dedicated to this research.
Chapter 1
INTRODUCTION

1.1 Motivation

As the Quality-of-Service (QoS) demanding applications are emerging, guaranteed QoS delivery of data must be supported by the Internet, in order for the Internet to continue to be the main global medium for information exchange and to achieve data/voice convergence. There is always a competition over available resources as the demand is increasing rapidly with time, and there are always different types of applications/end-users who require different levels of QoS or have different spending powers. A direct benefit of the existence of flows belonging to different Classes of Service (CoSs) in a network is the relaxation of the problem of QoS guarantee. This can be easily realized by noticing that during congestion (in wired networks) or bad channel characteristics (in wireless networks) periods, the available resources/intelligence can be first utilized to guarantee the QoS parameters of the highest Class of Service (CoS), and if some sessions must suffer, then let them be those belonging to the lower CoSs. While without introducing different CoSs, resources/intelligence must be used to guarantee the QoS for all the active sessions, which will result in a lower number of subscribers or a higher degree of complexity and cost. Thus, introducing different CoSs to the global data network is a reflection of a natural social phenomenon, a useful trick to relax the problem
of QoS guarantee, and could be an economical optimization of the usage of the available resources, which may lead to maximizing revenues of Internet Service Providers (ISPs) [McKnight 00, Odlyzko 00].

The fundamental issue here is to understand how QoS, service disciplines, pricing, and revenue interact with each other. In other words, how can we tell which is the most efficient multi-class or single-class design out of a group of suggested/proposed designs for the same given resources? These questions motivated a major part of this thesis.

If multi-class Internet is to be designed, the first concern will be how to allocate resources to different CoSs. Dynamic Bandwidth Allocation (DBA) operations have been studied and analyzed within different contexts and with different objectives. These objectives can be, in general, classified into objectives at packets level and objectives at calls level. It is very clear that if guaranteed QoS parameters are to be supported by the future Internet, where resources are going to be dedicated for high quality sessions, DBA with objectives at the calls level becomes more critical. Naturally, call-blocking rates at different CoSs is very essential QoS metric to be controlled, when designing multi-class networks. Also, DBA operations can be classified into three possible types; Complete Sharing (CS), Complete Partitioning (CP), and Virtual Partitioning (VPT). CS works well under light load. However, a well-designed network must not be lightly loaded, all the time. CP works very well under heavy loads, if and only if demands can be predicted, accurately. CP also provides designers with the ability to control call-blocking rates of different CoSs, which cannot be done under CS. VPT combines characteristics of both,
CS and CP. But, still, no strict control over the call-blocking rates of different CoSs can be exercised under VPT. However, VPT scheme avoids wastage of resources, which is a problem in CP if demands cannot be accurately predicted. Several questions force themselves when the issue of multi-class networks is considered. How can we allocate resources to different CoSs, and based on what objective? Should we segregate resources between different CoSs or not? How can we extend the CoSs concept to wireless networks, especially since broadband wireless access will become a major access technology for the Internet in the future?

Asynchronous Transfer Mode (ATM) networks already defined different CoSs, like Constant Bit Rate (CBR), Variable Bit Rate (VBR), and Available Bit Rate (ABR). However, these CoSs are meaningless when we consider the global Internet, because of the existence of the Internet Protocol (IP) Layer, which does not understand these ATM CoSs. Thus, there is a need for a multi-class provisioning in the IP-Layer, great efforts have been undertaken toward this objective within the Differentiated Services (DS) architecture [Nichols 98, Nichols 99]. However, there are many questions that still deserve an answer. Probably, the most important question is: will the DS architecture be able to guarantee absolute quantitative QoS parameters?

Multi-Protocol Over ATM (MPOA), [ATM Technical Committee 97], technique was developed to be a bridge that can help both the IP layer and the ATM layer to cooperate efficiently. This technique involves complicated networking operations, and little work has been done on evaluating its performance. Does MPOA really achieve a
significant gain over traditional IP routing? Will MPOA technique perform well if extended to the backbone?

The pricing problem forces itself approximately in any modern networking design. One way to classify dynamic pricing policies is to divide them into two categories, usage-based dynamic pricing and call-setup dynamic pricing. Since, major networking companies are competing nowadays on providing simpler pricing plans, usage-based dynamic pricing sounds far from reality. However, call-setup dynamic pricing may still have a chance to be implemented. Understanding the behavior of some dynamic pricing policies motivated the last part of this thesis, where call-setup dynamic pricing problem was modeled and the corresponding optimal policies were analyzed.

1.2 Objective

The objective of this Ph.D. dissertation is to analyze and evaluate the performance of modern broadband networking techniques. Also, new protocols and designs are proposed whenever improvement over current techniques seems possible.

This thesis focuses on 2\textsuperscript{nd} and 3\textsuperscript{rd} layers issues, with respect to the seven layers model standardized by International Standardization Organization (ISO). 1\textsuperscript{st} layer (physical) issues are not considered in this thesis.

Since the Internet became a very attractive investment environment, many companies and standardization bodies rushed into proposing new techniques and systems without careful investigation and performance evaluation. For example, MPOA, DS, and Multi-Protocol Label Switching (MPLS), [Rosen 01-a], were standardized very quickly
and approximately at the same time. Only MPOA was practically implemented. While, despite the fact that most new routers, like those designed by Cisco and Lucent companies, are compatible with DS and MPLS techniques, these techniques have not been put into operation till now.

After more than two years of effort by the Multi-Protocol Over ATM (MPOA) Working Group at the ATM Forum, MPOA was ratified as an industry standard in July 1997. MPOA is an inter Emulated-LAN communication technology, which supports the transport of connectionless traffic via both layer-3 hop-by-hop forwarding through intermediate routers and layer-2 shortcut switching through ATM network [Young 99]. The objective is to alleviate the burden of rapidly increasing inter subnet traffic on routers, which have increasingly become the bottleneck in traditional routed network. To the best of our knowledge, the effectiveness of MPOA technique has not been well studied. Major concerns have been raised with regard to its performance, mainly because of the complex signaling structure and large processing/propagation delays involved in each shortcut setup. In other words, the network resources required by shortcut setup may substantially undermine its claimed throughput performance.

In this thesis, we develop a framework to characterize individual MPOA resources and propose a unified measure to quantitatively capture the overall resource requirement. The MPOA performance can therefore be studied on the basis of real Internet/Intranet trace simulations. Note that, MPOA is data-driven, i.e., a shortcut is triggered by the first few packets of a flow, and thus traffic flow dynamics have substantial impact on MPOA performance.
Also, the thesis will consider proposing some techniques and algorithms necessary for better operation of a multi-class network. For example, previously proposed mechanisms for emulating connection-oriented sessions above the IP layer, specifically the MPLS technique, is investigated and a new idea is presented. Also, accurate predictions of demands in a global multi-class connection-oriented network will be analyzed and algorithms will be developed, in order to enhance the efficiency of DBA algorithms.

DBA for networks carrying multi-class traffic has been investigated within different contexts. We are concerned with architectures that work on the call level. Most of the work on this level was directed to dynamically allocate bandwidth to different Virtual Paths (VPs). The virtual path concept is a basic characteristic of ATM and ISDN networks, which provide connection-oriented services. VPs are usually designed based on topological optimization. Also, bandwidth can be segregated between different CoSs with different QoS requirements, as in [Chan 94].

Bandwidth segregation is an effective way of managing resources to serve different CoSs of traffic with different QoS requirements. However, as was shown in [Maunder 98], allocating bandwidth to VPs may lead to a lower efficiency (throughput) if bandwidth is not carefully (optimally) allocated. We point out that the basic reason for non-optimal allocation of bandwidth is the difficulty and inaccuracy in predicting demands on different CoSs. One of the essential reasons for this inaccuracy is the nature of most DBA algorithms, which are absolutely distributed.
Much work has been done recently on traffic engineering over the MPLS architecture [Rosen 01-a]. However, most of the efforts were directed to dynamic load balancing on alternative paths in order to efficiently utilize resources while achieving the best possible (QoS), like the work in [Kar 00, Dinan 00]. Although CoSs were also supported in these studies, little efforts were directed to optimize the allocation of bandwidth among CoSs sharing a common link.

We believe DBA, traffic engineering, and routing are related tasks and cannot optimally perform without cooperation with each other. Routing and load distribution/balancing on alternative paths -Label Switched Paths (LSPs) in MPLS context- can be viewed as a necessary process on small time scales to efficiently utilize scattered resources. This process takes place at the edge of an MPLS network (ingress nodes). But at the same time, there can be a competition on bandwidth between different LSPs, which may represent different CoSs, at each link in the network, both at the core and at the edge of an MPLS network. As the aggregated demand on different CoSs/LSPs varies slowly over time, a DBA process is necessary to optimally distribute bandwidth between competing CoSs/LSPs based on a specific objective function. This objective function usually takes into consideration user-utilities and revenues which are functions of QoS, pricing model, and regulations governing different CoSs. In this thesis, we develop an algorithm to dynamically allocate bandwidth to different CoSs/LSPs sharing a common link in a connection-oriented multi-class network with a bandwidth segregation approach under non-stationary traffic.
One of the main objectives of this thesis, also, is to motivate and highlight the importance of multi-class networks in the next generation Internet, both in the wired and wireless parts of the network. Toward that objective, the interaction and tradeoffs between QoS, service disciplines, pricing, and revenue are investigated. This interaction will be clarified with a proposed example of a simple multi-class wireless LAN.

Understanding the relation and interaction between QoS, pricing, revenue, and service disciplines is not a new effort. Several initial and successful attempts were done. One can notice, by reviewing previous works in the literature, that different researchers approached the problem from different perspectives. The two major approaches are based on two major objectives. Some researchers consider their objective to be maximizing revenue (private-good network) [Fishburn 98, Honig 94], others, consider their objective to be maximizing the total users’ utility (public-good network) [Cocchi 93, Orda 97, Sairamesh 95, Wang 01]. In both approaches, differential pricing in multi-class networks was found to be very essential.

For example, in [Cocchi 93], the network was assumed to be a public-good. Based on that assumption, authors found that it is always true that a multi-class architecture can be designed to achieve a higher total users’ utility than a single-class architecture, for the same system capacity. While, for example, in [Honig 94], the network was assumed to be a private-good and authors showed that revenues can be maximized with differential pricing.

Most of the work available in the literature, as discussed above, dealt with optimizing pricing vectors for a specific multi-class architecture, in order to maximize
users’ utilities or network revenues. Little work has been done to compare different multi-class designs in terms of network revenue. For example, in [Fishburn 98], authors compared between three designs for two types of demands, delay-sensitive and delay-insensitive. The three designs are: two separated networks for the two demand types, one network with one high quality grade of service, and one network that admits the two types of demand but without resource-sharing. This analysis didn’t clearly address the issue of different possible levels of differentiation in the provisioned QoS between the different classes, and the effect of these different levels on network revenue. On the other hand, in [Edell 99] a similar comparison study was conducted through the INDEX project, but the differentiation was done at the level of access speeds, only. In this thesis, we are interested in comparing different multi-class architectures with different provisioned QoS parameters under different classes, even if all the classes have the same access speed.

To better understand the role of pricing in modern networking systems, this thesis will consider a practical form of dynamic pricing, namely, call-setup dynamic pricing with more analysis. Specifically, the behavior of optimal pricing policies will be studied under different settings for a single-link with dedicated resources.

In general, dynamic pricing can be applied to the usage-based charging rate or to the call-setup charge, or to both of them. In [Breker 96], for example, the call-setup charge was fixed, while the usage-based charging rate was designed to be adaptive. In [Roberts 98], on the other hand, the usage-based charging rate was fixed and the call-setup charge was designed to be adaptive. However, in [Roberts 98], an optimal policy
for controlling the value of the call-setup charge was not found. In this thesis, we will use a dynamic programming technique to analyze the behavior of an optimal policy for controlling the value of the call-setup charge.

1.3 Thesis Overview

This thesis is divided into eight chapters, including this introduction. Chapter 2 discusses background information needed for this research. This includes the basic principles of ATM, MPOA, DS, and MPLS techniques.

Chapter 3 investigates, in depth, the performance gain of MPOA technique. Real Internet traces from LANs and WANs are used in this study. Also, an intelligent cache table management based on neural network technique is designed for two objectives; first, to investigate the performance of MPOA; second, to provide a practical and optimal solution for the control of the time-out value of cache-tables in Multi-Protocol Clients (MPCs) of an Emulated LAN. Toward designing this system, previously proposed techniques by standardization bodies are analyzed using sensitivity analysis approach.

Chapter 4 initiates the discussion and study of multi-class networks by proposing a Multi-Class Internet (MCI) based on a bandwidth-segregation with controlled-sharing approach. This MCI supports both connection-oriented and connectionless services. A new technique for emulating connection-oriented services above IP layer is proposed and compared with the existing technique within the MPLS framework, in terms of overhead on both packet’s header and core routers.
Chapter 5 addresses the issue of DBA in a multi-class connection-oriented backbone network. First, previous work on predicting the call-blocking probability for a link with Poisson arrivals, exponential holding time, and variable-size requests is reviewed. Based on a previously developed recursive algorithm [Kaufman 81], a new recursive algorithm is developed to control the ratio of call-blocking probabilities of two CoSs sharing a common link. Also, loss network analysis is presented to support the importance of the proposed Virtual Demand Distribution (VDD) algorithm, which achieves much higher accuracies in predicting demands on different CoSs in a non-stationary environment. Simulation results using a sophisticated simulation tool (Bones Designer) are presented.

Chapter 6 extends the concept of multi-class networks to Time Division Multiple Access (TDMA) LANs. Preliminary ideas are presented to design a fading-aware multi-class TDMA LAN. However, a detailed description of a two-class Packet Reservation Multiple Access (PRMA) protocol is presented, assuming error-free wireless channel. Also, a revenue comparison framework is developed, which is used to compare the performance of the proposed Two-Class PRMA (TCPRMA) to the classical PRMA [Goodman 89]. This revenue comparison framework also clarifies the meaning of and interaction between QoS metric, QoS score, service disciplines, willingness-to-pay, and revenue. Both protocols, PRMA and TCPRMA are simulated and their performances are compared based on the revenue comparison framework.

Chapter 7 considers the problem of call-setup dynamic pricing for a link with variable-size arrivals. An exact Markov model is developed for this problem, which is
modeled as a discounted Markov Decision Process (MDP). \( \varepsilon \)-optimal policies for different scenarios are found using the Gauss Seidel value iteration algorithm. The behavior of these policies is investigated.

Chapter 8 summarizes the results and contributions of this dissertation work. Suggestions for future research are also discussed.

### 1.4 List of Contributions

Following is a list of submitted or published papers related to the work in this thesis.

**Journal papers:**


**Conference Papers:**


**Reports:**

Chapter 2

BACKGROUND

Data networks can be divided into three types, based on their geographical size [Stallings 97]. These three types are: Local Area Network (LAN), Metropolitan Area Network (MAN), and Wide Area Network (WAN). Deciding to which type a network belongs is a subjective issue. One may consider a network connecting hosts in a specific building as a LAN, while a network covering a city, where that building exists, can be considered as a WAN. On the other hand, one may consider a network covering a city as a LAN, while a network covering a country, where that city exists, can be considered as a WAN.

In our research area, where we are interested in studying and evaluating the capabilities of different networking technologies to support QoS requirements, we propose to divide data networks based on a different basis. We consider a network as a LAN if the communication in that network has a broadcasting nature. On the other hand, we consider a network as a WAN if the communication in that network has a non-broadcasting nature. This division is fundamental, since networking techniques in broadcasting communications differ significantly from non-broadcasting communications. In this proposed division, a MAN does not have a significant meaning, but we tend to consider it as a LAN that has a large geographical size, as is the case in a big optical ring [Stallings 97].
Internet has become the basic architecture for global data networking. In their early deployment, Internet applications were limited to exchanging files and emails, and text applications. At that time, the QoS provided by the Internet was acceptable. As more interactive and multimedia applications are emerging, and as the size of the Internet community is dramatically increasing, much better QoS provisioning is required now.

The development of data networking technology has been fast in both LANs and WANs. Wired LANs have reached an acceptable level of development, nowadays Ethernet with 100Mbps access speed is common, and 10Mbps is even more common and affordable. These speeds are enough to play a full-screen movie on a screen of a PC connected to a network. On the other hand, despite all the development in WAN technologies, guaranteed QoS parameters are still not provided in general. This is why a person with 100Mbps access speed may not be able to view a small-screen movie played on some Internet sites.

Despite the introduction of ATM technique, which theoretically could solve most of the problems in the Internet architecture, it failed to force itself as the final architecture of the global data network. There are several reasons for this failure, which will be discussed later. Thus, it seems that the IP protocol will continue to be the basic networking protocol for the coming decade.

However, ATM technique is heavily deployed in the backbone (WAN) of the Internet as a supporting layer to the Internet layer. The concept of layers should be clear if the reader understands the seven-layer model (Open System Interconnection - OSI-
model) developed by ISO. We can also see ATM components in LANs, as is the case in an Emulated Local Area Network (E-LAN) [Stallings 97].

So, we can say that the current global network can be described as a hybrid of ATM and IP components that build the WANs. While LANs can be constructed using different technologies, Ethernet is the most commonly used, especially in institutions and large buildings. There are several access technologies to provide connections for residents at homes, like dialup modems, DSL, xDSL, and cable modems; these access technologies are well described in [Samueli 00]. It is important to note the difference between Ethernet as an access technology and other access technologies for residents at homes. In the earlier case, all users in an Ethernet can communicate directly via Ethernet protocols. In the latter case, users accessing the Internet, via any of the previously mentioned access technologies, can only communicate via Internet protocols, and not directly.

The bottleneck in providing guaranteed QoS is not due to limitations in the access technologies. Since, as we have mentioned, very high access speeds are available nowadays, but yet there are no QoS guarantee in the global Internet. Also, improving the access speed is not a networking issue, but rather it is a physical communication issue that involves modulation/demodulation and digital signal processing techniques, which are outside our research interests.

Before we continue our discussion, we have to mention the potential of optical communication and networking in shaping the next generation global Internet. Optical communication via optical fibers has provided us with tremendous communication
speeds, especially in point-to-point links. So, optical fibers are most commonly used in the backbone and within ATM networks. Also, optical LANs have been deployed with very high data transfer rates. But, even if every link in the backbone is replaced by an optical fiber, still we wouldn’t consider this an all-optical network. Because routing and switching functions are done electronically, preventing us from enjoying the benefits of the high-speed communications provided by optical fibers. Only when switching and routing can be done at the optical speed, we can say that we have achieved the promised all-optical network, at which a tremendous amount of bandwidth will be available. But, this may not happen soon, at least not in the coming decade.

Let us now consider wireless communication, which is a potential access technology, both in fixed and mobile scenarios. Unlike wired LANs, wireless LANs still suffer from a non-reliable communication. Although new emerging wireless technologies have proven their abilities to support high-speed connections, still this speed cannot be guaranteed for the whole life of a session, especially in mobile wireless networks. This is because of the unpredictable characteristics of wireless channels, which lead to severe degradations in QoS. This unpredictability is due to multi-path fading, atmospheric attenuations, and signal interference [Proakis 95]. So, unlike what we have said for wired LANs, wireless LANs need more research on the networking level, in order to improve the perceived QoS by subscribers, as we discuss in later chapters.

Let us now review the main attempts that have been made or being proposed, on the networking level, to support and enhance QoS provisioning. We start with briefly describing the ATM technique, which as we mentioned previously has failed to force
itself as the final solution for the next generation global data network. Then, we discuss three techniques that have been proposed and standardized: the DS architecture [Nichols 98, Nichols 99], the MPLS architecture [Rosen 01-a], and the MPOA architecture [ATM Technical Committee 97].

2.1 Asynchronous Transfer Mode (ATM) Architecture

ATM protocol was originally developed to correct the lack of networking capabilities in ISDN [Ayanoglu 96]. The basic idea was to combine good features of circuit switching and packet switching approaches to come up with an optimum switching scheme. The packet size was considered to be fixed and somewhat arbitrarily chosen to be 53 bytes. The basic idea was to choose the cell size not to be too large so that the switching can be done in real time and not too small so that the cell redundancy would not diminish the advantages of statistical multiplexing, also compatibility with sampling rate of voice sources was taken into consideration. Each fixed size packet (ATM cell) consists of a header and a payload. A five-byte header contains the information for routing the cell in the network and the rest of the bits in a cell, i.e. payload bits, are the desired information to be sent. To minimize the processing overhead to an infinitesimal level, error correction capabilities were only implemented in the header. Before the transmission begins, network establishes a virtual path between the transmitter and the receiver via a certain number of ATM switches. Every cell will go through this predefined path. In this regard, ATM is very similar to a circuit switching methodology, however, the virtual path is not dedicated to a specific pair of users and all
the ATM switches will attempt to get the best use of the channel resources by doing statistical multiplexing.

One of the most important aspects of the ATM protocol is its guaranteed QoS, once the connection between the two communicating ends has been established. Before establishing the final path, network will find the best possible paths based on the available network resources and the requested QoS by the user. After a possible negotiation period, the connection will be established and then the network has the duty of maintaining the QoS to the level it has promised to the user. The cell loss ratio and the cell transmission delay are the two most important QoS metrics.

The basic driving force behind ATM is the recognition that the traditional narrowband circuit switched connections (as deployed worldwide in support of voice services) are inadequate to meet the needs of multimedia traffic [Acompora 96]. Because of its ability to handle both synchronous and asynchronous traffic with a wide range of delays, bit rates and QoS values with negligible overhead, ATM is considered to be the best transmission protocol for wideband networks on a highly reliable wired channel. ATM can support all types of services, from delay-sensitive voice communications and desktop multimedia conferencing to bursty transaction processing and LAN traffic.

However, despite all the attractive features of the ATM technique, it has not been able to force itself as a final solution for the next generation global Internet. We can see this fact in both the access and the backbone regions. ATM was not deployed efficiently as an access technology because of the signaling overhead with respect to the low aggregation levels and high cost [Niehaus 97]. At the same time, due to the continuous
existence of the Internet Protocol (layer 3 routing), ATM was not successful to force itself as the only technology in the backbone, although it is being heavily deployed there. The current backbone is a hybrid of both IP routers (layer 3 routing) and ATM switches (layer 2 switching) [Trecordi 00]. The IP (layer 3) routing is going to be a basic feature of the next generation Internet at least for the coming decade [Rutkowski 99].

The coexistence of IP routers and ATM switches triggered many research efforts in different directions, both in the access and the backbone networks, in order to utilize the intelligence available in these technologies. MPLS [Rosen 01-a] and MPOA [ATM Technical Committee 97] are two examples from these efforts that were standardized. Thus, the current trend is to face the fact that there should be a reasonable solution that lives with the IP layer.

2.2 Differentiated Services (DS) Architecture

DS was proposed and standardized by the IETF [Nichols 98, Nichols 99], current routers also support the DS technique in addition to MPLS for possible future implementation. The DS architecture is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different behavior aggregates. Each behavior aggregate is identified by a single DS code-point. Within the core of the network, packets are forwarded according to the per-hop behavior associated with the DS code-point [Nichols 98].

Traditionally, network service providers (both enterprise and traditional ISPs) provide all customers with the same level of performance (best-effort service). Most
service differentiation has been in the pricing structure (individual vs. business rates) or the connectivity type (dial-up access vs. leased line, etc.). However, in recent years, increased usage of the Internet has resulted in scarcity of network capacity, compromising performance of traditional, mission critical applications. At the same time, new applications have emerged which demand much improved service quality. As a result, service providers are finding it necessary to offer their customers alternative levels of service. As well as meeting new customer expectations, this allows service providers to improve their revenues through premium pricing and competitive differentiation of service offerings, which in turn can fund the necessary expansion of the network.

The DS architecture offers a framework within which service providers can offer each customer a range of network services, which are differentiated on the basis of performance in addition to pricing tiers used in the past. Customers request a specific performance level on a packet by packet basis, by marking the DS field of each packet with a specific value. This value specifies the Per-hop Behavior (PHB) to be allotted to the packet within the provider's network. Typically, the customer and provider negotiate a profile (policing profile) describing the rate at which traffic can be submitted at each service level. Packets submitted in excess of this profile may not be allotted the service level requested. A salient feature of DS is its scalability, which allows it to be deployed in very large networks. This scalability is achieved by forcing as much complexity out of the core of the network into edge devices which process lower volumes of traffic and lesser numbers of flows, and offering services for aggregated traffic rather than on a per-micro-flow basis [Nichols 98].
Thus, we can see that one of the characteristics of the DS architecture is its inability to guarantee absolute QoS parameters. The differentiation between different CoSs is expected to be qualitatively rather than to be based on absolute and specific numerical characteristics of each CoS [Borgonova 98]. This feature is keeping researchers looking for other solutions that can provide QoS guarantee despite all the advantages of the DS architecture, which are mainly scalability and simplicity in the core of the network.

2.3 Multi-Protocol Label Switching (MPLS) Architecture

MPLS was proposed and standardized by the IETF [Rosen 01-a]. Current routers support this technique for possible future implementation of this technology. In connectionless network layer protocols, as a packet travels from one router hop to the next, an independent forwarding decision is made at each hop. Each router runs a network layer routing algorithm. As a packet travels through the network, each router analyzes the packet header. The choice of next hop for a packet is based on the header analysis and the result of running the routing algorithm.

Packet headers contain considerably more information than is needed simply to choose the next hop. Choosing the next hop can therefore be thought of as the composition of two functions. The first function partitions the entire set of possible packets into a set of "Forwarding Equivalence Classes (FECs)". The second maps each FEC to a next hop. Insofar as the forwarding decision is concerned, different packets,
which get mapped into the same FEC, are indistinguishable. All packets, which belong to a particular FEC and travel from a particular node, will follow the same path.

In MPLS, the assignment of a particular packet to a particular FEC is done just once, as the packet enters the network. The FEC to which the packet is assigned is encoded with a short fixed length value known as a "label", [Rosen 01-b]. When a packet is forwarded to its next hop, the label is sent along with it; that is, the packets are "labeled".

At subsequent hops, there is no further analysis of the packet's network layer header. Rather, the label is used as an index into a table, which specifies the next hop, and a new label. The old label is replaced with the new label, and the packet is forwarded to its next hop. If assignment to a FEC is based on a "longest match", this eliminates the need to perform a longest match computation for each packet at each hop; the computation can be performed just once.

Some routers analyze a packet's network layer header not merely to choose the packet's next hop, but also to determine a packet's CoS in order to apply different discard thresholds or scheduling disciplines to different packets. MPLS allows the CoS to be inferred from the label, so that no further header analysis is needed; in some cases MPLS provides a way to explicitly encode a CoS in the "label header".

The fact that a packet is assigned to a FEC just once, rather than at every hop, allows use of sophisticated forwarding paradigms. A packet that enters the network at a particular router can be labeled differently than the same packet entering the network at a different router, and as a result, forwarding decisions that depend on the ingress point
("policy routing") can be easily made. In fact, the policy used to assign a packet to a FEC need not have only the network layer header as input; it may use arbitrary information about the packet, and/or arbitrary policy information as input. Since this decouples forwarding from routing, it allows one to use MPLS to support a large variety of routing policies that are difficult or impossible to support with just conventional network layer forwarding.

Although MPLS technique sounds capable of guaranteeing QoS parameters for specific CoSs, but there is an important big question about its scalability that is still not clearly answered. Despite this, current research is trying to combine both MPLS and DS techniques, where DS protocols are deployed in the sub-networks which are connected to the edge devices, and MPLS protocols are deployed in the backbone starting from the edge devices [Faucheur 00]. But, scalability was not the only obstacle in the way of implementing MPLS technique. Signaling overhead, which is represented by the labeling bytes in each packet, is still another obstacle.

In industry, people are more attracted to MPLS, because it is the closest to ATM technique and the most capable for guaranteeing QoS parameters. With the increasing speed of switching and routing of the new devices, the labeling overhead may become acceptable.

2.4 Multi-Protocol Over ATM (MPOA) Architecture

MPOA technique was proposed and developed by the MPOA working group at the ATM Forum, MPOA was ratified as an industry standard in 1997 [ATM Technical
Committee 97]. However, there is a major difference between the MPOA architecture and the DS and MPLS architectures. MPOA is an inter E-LAN communication technology and it is not proposed as a potential solution for the problem of QoS over the global network. Actually, we have shown that MPOA technique is not efficient in the backbone [Taha 01]. A complete description and analysis of MPOA technique will be presented in chapter 3.
Chapter 3
MPOA FLOW CLASSIFICATION DESIGN AND ANALYSIS BASED ON NEURAL NETWORK TECHNIQUE

3.1 Introduction

After more than two years of effort by the Multi-Protocol Over ATM (MPOA) Working Group at the ATM Forum, MPOA was ratified as an industry standard in July 1997. MPOA is an inter E-LAN communication technology, which supports the transport of connectionless traffic via both layer-3 hop-by-hop forwarding through intermediate routers and layer-2 shortcut switching through ATM network [Young 99]. The objective is to alleviate the burden of rapidly increasing inter subnet traffic on routers, which have increasingly become the bottleneck in traditional routed network. To the best of our knowledge, the effectiveness of MPOA technique has not been well studied. Major concerns have been raised with regard to its performance, mainly because of the complex signaling structure and large processing/propagation delays involved in each shortcut setup. In other words, the network resources required by shortcut setup may substantially undermine its claimed throughput performance.

In this chapter, we develop a framework to characterize individual MPOA resources and propose a unified measure to quantitatively capture the overall resource requirement. The MPOA performance can therefore be studied on the basis of real
Internet/Intranet trace simulations. Note that, MPOA is data-driven, i.e., a shortcut is triggered by the first few packets of a flow, and thus traffic flow dynamics have substantial impact on MPOA performance.

Flow classification (FC) and cache table management (CTM) are the two critical MPOA design issues, which have not been carefully examined. The work in [Widjaja 99] demonstrated the significant effects that (CTM) has on the performance of MPOA technique. In an MPOA network, each multi-protocol client (MPC) runs a FC algorithm to identify long-lived flows for shortcut setup; the state information for each shortcut connection is maintained by a CTM scheme. Hence, the MPOA performance is largely dependent on the design of the FC algorithm and CTM scheme. The default FC algorithm proposed by the MPOA standard [ATM Technical Committee 97] is found inefficient in our study. The CTM in [ATM Technical Committee 97] requires a specified holding time to be assigned to each cached flow entry with possible extension of an extra holding time upon request. Yet, no scheme is proposed to effectively manage such a flow cache table for the overall network performance improvement.

In this chapter we propose a FC algorithm, which substantially reduces the implementation complexity while achieving the same level of performance as compared to the default one in [ATM Technical Committee 97]. A simple timeout mechanism is also introduced in the CTM for significant performance improvement. The first major contribution of this work is the design of a self-learning CTM system (SLCS) based on the NN technique. Using a periodic off-line supervised learning and on-line prediction, the SLCS can adapt to traffic level/pattern changes for any given flow cache table size at
time scales ranging from seconds, minutes, hours to days. Our real trace simulation has shown that the SLCS achieves near-optimal operation in terms of overall performance gain. Designing adaptive CTM algorithms such as the SLCS is important not only because the traffic characteristics as a function of time is unknown a priori, but also the overall resource availability varies from one switching system to another. A switching system design varies depending on time, vendors, technologies and applications. Especially in the product design stage, it is difficult to predict future applications and their traffic characteristics. As a consequence, the utilization of major resources at a switching system is likely to be highly unbalanced and time varying, and any static control based on a fixed parameter setting will result in a very inefficient resource utilization.

The second major contribution of this work is to show that MPOA can indeed provide significant performance gain over the traditional routed network in an inter ELAN environment. With the SLCS, we were able to show that even at high cost with large round-trip delay for each shortcut setup, MPOA's performance gain can reach more than ten fold provided that a sufficiently large cache table size is available to accommodate most of the long-lived flows and an efficient flow cache table management is used. In contrast, the performance gain in an Internet backbone environment is found to be much less significant, mainly due to the short-lived flow nature of dns and http traffic.

A few works are available on the performance analysis and FC algorithm design for the flow-based IP/ATM hybrid switching systems, which are mainly based on the Ipsilon IP switching architecture [Newman 98, Lin 97]. However, there are two basic
distinctions between MPOA and Ipsilon IP switching, which makes the separate study on MPOA necessary. First, the cost and delay for shortcut setup in MPOA are generally much greater than those for cut-through connection setup in Ipsilon IP switching. This is because the MPOA’s complex signaling procedure for shortcut setup and its associated two end-to-end round-trip delays. Second, per connection maintenance cost in MPOA is generally much less than that in Ipsilon IP switching. This is because MPOA maintains a "hard state" for each shortcut connection, whereas the Ipsilon IP switching maintains a "soft state" for each cut-through connection. The paper [Newman 98] placed the emphasis on per connection maintenance cost under the assumptions of small processing cost and zero delay on cut-through setup. The study in [Lin 97] examined the delay effect while neglected both setup and maintenance costs for cut-through connections. In contrast, the MPOA analysis requires consideration of both long delay and high cost involved in shortcut setup while neglecting per connection maintenance cost.

3.2 Framework

This section is composed of three subsections. Subsection 3.2.1 formulates the process of each individual flow by a finite state machine. Subsection 3.2.2 identifies the requirement of MPOA resources and their associated costs for the process of aggregate flow dynamics. Subsection 3.2.3 then proposes a design objective to minimize the overall cost in the FC algorithm design and CTM.
3.2.1 A Finite State Machine per Flow

As shown in Figure 3-1, an MPOA network is logically divided into ELANs [ATM Forum 97]. Each ELAN is composed of multi-protocol servers (MPSs), edge devices (EDs) and directly attached hosts. Each ED is further attached by one or more legacy shared-media LANs. Unlike the traditional inter-networking where EDs and MPSs are regular routers, MPOA employs the client-server model where EDs are clients and MPSs are servers. An MPS performs internetworking route calculation for all the associated EDs and directly attached hosts. Thus the number of devices participating in route calculation is substantially reduced, which increases the routing scalability. An important component in each ED is the so called multiprotocol client (MPC) that performs the MPOA client functions including FC, CTM and shortcut setup.
To explain how a flow is shortcut switched in Figure 3-1, let us assume that an end-user A1 on an Ethernet behind MPC1 of ELAN1 wants to send a data flow to another end-user B3 on another Ethernet attached to MPC3 of ELAN3. In this case, MPC1 is the ingress MPC and MPC3 is the egress MPC. Such a flow is first to be identified by the FC algorithm in MPC1. During the identification period, all its input packets will be hop-by-hop forwarded through intermediate MPSs to B3. Once it is identified as a long-lived flow, MPC1 will generate a shortcut setup request to MPC3. There are two steps for a successful shortcut setup. The first step is to resolve the MPC3’s ATM address through the next hop resolution protocol (NHRP) via MPSs, i.e., the hop-by-hop forwarding path. The binding information from the layer-3 address of B3 to the MPC3’s ATM address will
then be cached in the MPC1's flow cache table. The second step is to establish a virtual channel connection (VCC) through the standard ATM signaling protocol via ATM network. This step may be bypassed if there already exists a VCC from MPC1 to MPC3. Notice that, the management of all the VCCs at MPC1 requires a VPI/VCI table, which is separate from the flow cache table. In the worst case, a shortcut setup process requires two round trip delays, during which all the input packets of the flow are hop-by-hop forwarded.

![Finite state machine for flow state in a MPC](image)

*Figure 3–2: Finite state machine for flow state in a MPC.*

One can describe the process of each flow by a finite state machine in *Figure 3–2.* State *ID* represents the idle period before the first packet arrival of the flow. State *S* is
the flow identification period before making a request for shortcut setup. State $SN$ is the time period required to resolve the ATM address of the egress MPC. State $SCS$ is the time period for ATM signaling to establish a VCC. While in states $S$, $SN$ and $SCS$, all input packets of the flow are software forwarded. State $CT$ is the only state during which all input packets are shortcut switched. Note that, a flow can terminate at any time as described by the transition from any other state to state $ID$ in Figure 3–2. Further, the direct transition from state $SN$ to state $CT$ represents the situation in which the ATM signaling is bypassed. A transition probability $(1-\beta)$ is assigned to describe the possibility for a flow to find an existing VCC to its egress MPC.

The actual value of $\beta$ depends on VPI/VCI table management. The VPI/VCI table management again largely depends on the spatial distribution of the traffic flows. Multiple flows between each pair of ingress MPC and egress MPC may share a single VCC. Since the spatial distribution of the traffic flows cannot be extracted from the collected traces due to the address filtering for security consideration, our design will be solely based on a single node simulation without taking into account the VPI/VCI management. Instead, we shall examine the sensitivity of MPOA performance to $\beta$ by simply considering two extreme values, namely, $\beta = 1$ for the worst-case scenario and $\beta = 0$ for the best-case scenario.

### 3.2.2 MPOA Resource Classification

For each flow, one can use the finite state machine to characterize its present demand on individual MPOA resources according to its current state or transition. For
aggregate flows, the demand on individual resources can therefore be measured by the number of flows staying in each individual state or taking each individual transition at a given time interval $\Delta t$. Defined below are the demands of aggregate flows on five major MPOA resources at the $n$th time interval:

- **$f(n)$**: software-forwarding demand, measured by the number of packets generated per second by flows in states $S$, $SN$ and $SCS$;
- **$r(n)$**: address-resolution demand, measured by the number of flows taking transition from state $S$ to state $SN$;
- **$a(n)$**: active-address-binding demand, measured by the number of flows in states $SCS$ and $CT$;
- **$s(n)$**: ATM-signaling demand, measured by the number of flows taking transition from state $SN$ to state $SCS$;
- **$c(n)$**: active-shortcut demand, measured by the number of VCCs in state $CT$.

Note that $f(n)$, $r(n)$ and $s(n)$ are the rate measurement defined by the number of packets or flows per second, which are more related to the demands on CPU processing powers. $a(n)$ and $c(n)$ are the cumulative measurement defined by the number of flows, which are related to the memory requirement for table management. The capacities of the individual resources available in a given system are further represented by $F_{\text{max}}$, $R_{\text{max}}$, $A_{\text{max}}$, $S_{\text{max}}$ and $C_{\text{max}}$, respectively. For simplicity, we assume these capacities are fixed, although our work generally applies to variable capacities.
Accordingly, \( F_{\text{max}} \) and \( R_{\text{max}} \) refer to the packet forwarding and NHRP capacities of the MPCs and MPSs along the hop-by-hop software forwarding path. \( S_{\text{max}} \) represents the signaling capacity of the MPCs and ATM switches along the shortcut switched path. These are nonlocal resources. Both \( A_{\text{max}} \) and \( C_{\text{max}} \) are referred to the flow cache table size and the VPI/VCI table size at the local ingress MPC. With these definitions, we implicitly assumed that the demands \( a(n) \) and \( c(n) \) at the ingress MPC are solely constrained by the corresponding local resources \( A_{\text{max}} \) and \( C_{\text{max}} \), respectively. In practice, however, the state information of each flow needs to be maintained at both ingress and egress MPCs, and a shortcut connection consumes the VPI/VCI table resource in each intermediate ATM switch along the path. Based on our single node simulation, such nonlocal resource constraints will not be considered.

Since each VCC between a given pair of ingress/egress MPCs can be shared by multiple flows, the demand on \( C_{\text{max}} \) is expected to be significantly less than that on \( A_{\text{max}} \). We then assume that resource \( C_{\text{max}} \) is unconstrained. We further assume that all the three nonlocal resources, \( F_{\text{max}}, R_{\text{max}} \) and \( S_{\text{max}} \), are unconstrained. Hence, \( A_{\text{max}} \) becomes the only constrained resource. That is, a long-lived flow will be blocked from shortcut switching if and only if \( A_{\text{max}} \) is fully occupied by the existing flows, and a blocked flow due to \( A_{\text{max}} \) overflow can still be software forwarded without loss. Note that, the unconstrained nonlocal resources are not for free. On the contrary, the objective of our adaptive FC algorithm design is to minimize the nonlocal resources utilization subject to the local resource constraint.
3.2.3 Cost Function and Design Objective

Let us first consider the cost for maintaining and updating the two local resources at the ingress MPC. Define the cost per unit time per VPI/VCI entry in the active-shortcut demand by $C_c$ and the cost per unit time per cached flow entry in the active-address-binding demand by $C_a$. The aggregate cost of the two demands at the $n$th $\Delta t$ interval can then be represented by $a(n)C_a \Delta t + c(n)C_c \Delta t$. For the three nonlocal resources, we define $C_f$ as the cost per packet in the software-forwarding demand, $C_r$ as the cost per flow in the address-resolution demand, and $C_s$ as the cost per flow in the ATM-signaling demand.

The total cost on all the resources at the $n$th $\Delta t$ interval can therefore be described by

$$C(n) = a(n)C_a \Delta t + c(n)C_c \Delta t + f(n)C_f + r(n)C_r + s(n)C_s.$$ (3–1)

Also, we need to develop a unified cost measure among the five different cost factors. For simplicity, we take $C_f = 1$, which is the cost per software-forwarding packet. That is, all the other cost factors will be measured in the units of software-forwarding packet. For instance, taking $C_r = 20$ ($C_s = 20$) means that the cost of address resolution (ATM signaling) to set up a shortcut connection per flow is equivalent to the cost of software-forwarding 20 packets. Since MPOA maintains a "hard state" for each shortcut connection, the cost of maintaining and updating both VCC and flow cache tables are comparatively negligible, i.e., $C_a \approx 0$ and $C_c \approx 0$. We then have

$$C(n) \approx f(n) + r(n)C_r + s(n)C_s.$$ (3–2)

where $f(n)$ is the number of software-forwarding packets and $r(n)C_r + s(n)C_s$ is the equivalent switching overhead.
Given this cost structure, we can introduce a performance measure, called \textit{switching gain}, \( G \), which is defined as the ratio of the total number of software-forwarded and hardware switched data packets to the total number of software-forwarded data packets plus the switching overhead. Accordingly,

\[
G = \frac{\sum_{n} (d(n) + f(n))}{\sum_{n} (f(n) + r(n)C_r + s(n)C_s)}
\]  \hspace{1cm} (3–3)

where \( d(n) \) represents the number of hardware-switched packets at the \( nth \) \( \Delta t \) interval.

Notice that, for pure layer-3 forwarding, we get \( d(n) = r(n) = s(n) = 0 \) and so \( G = 1 \). If \( G < 1 \), a negative performance gain is achieved by the hybrid switching as compared to the pure software forwarding. Otherwise, a positive gain is achieved and the value of \( G \) provides us a quantitative measure on how many times the processing power is saved by the hybrid switching over the pure software forwarding. Our objective is to maximize \( G \), constrained by the local flow cache table size, through the design of an adaptive FC algorithm. For the adaptive FC, the instantaneous demand on each individual resource is controllable through the adaptive assignment of a control vector \( \bar{x}_n \) based on the present condition of flow dynamics and resources utilization at time \( n \). Denote the control sequence up to time \( n \) by \( \mathcal{X}_n = \{ \bar{x}_m \}_{0}^{n} \). We have

\[
\max_{\{\mathcal{X}_n\}} G. \hspace{1cm} (3–4)
\]

Indeed, when such a control objective is achieved at every ingress MPC, the overall usage on the MPOA network processing resources is expected to be minimized. Note that, the adaptive control is important to achieve near-optimal performance,
especially given the time varying behavior of the traffic and the dynamic sharing nature of the resources.

Three Internet/intranet traces are used in the simulation study, which are referred to as cisco-trace, lbl-trace and fixwest-trace, respectively. The cisco-trace is a 20-minute trace collected from a 100-BT campus network at Cisco Systems Inc. on March 4, 1997. The lbl-trace is a 16-minute trace collected from a 100-BT at Lawrence Berkeley Laboratory (LBL) on July 14, 1997. The fixwest-trace is a 20-minute trace collected from the FDDI Internet backbone at FIXWEST on Oct. 21, 1996. The utilization at the time of data collections are 5.5%, 4.0%, and 27.3%, respectively.

3.3 Flow Classification Design and The Self-Learning System

3.3.1 Static FC Algorithms

A flow is identified as a sequence of packets that are treated identically by routing function. The flow identifier can be defined at various granularity levels. Examples are host+port granularity and host granularity. The former is described by a pair of \{source address, source port\} and \{destination address, destination port\}. The latter is by a pair of \{source address\} and \{destination address\}. In this work, we only consider flows at host granularity.

Three static FC algorithms were proposed, which are called the application-based, X/Y/T, and X/Z/T algorithms, respectively [ATM Technical Committee 97, Che
The application-based algorithm is to classify application flows based on the measured average flow duration and average number of packets per flow for each application. The algorithm was found ineffective as compared to the other two algorithms [Che 98, Lin 97] and thus will not be considered. Both the X/Y/T and X/Z/T algorithms use a timeout, T, for the flow cache entry management, i.e., once the idle time of any cache entry reaches T it has to be removed from the cache table. These entries hold the binding information between the layer-3 address of the destination host and its MPC's ATM address. The X/Y/T algorithm is to request a shortcut setup if and only if X packets with the same flow identifier have been forwarded within Y seconds, which is the same as the default FC algorithm proposed in the MPOA specification [ATM Technical Committee 97]. The X/Z/T algorithm is to request a shortcut setup if and only if X packets with the same flow identifier are software forwarded, where the inter-arrival time between any two successive packets are smaller than Z. The X/Y/T algorithm keeps a cumulative counter of the forwarded packets and the timestamps in a fixed time window size Y per flow. Upon each packet arrival, the timestamps are updated by deleting the old ones while adding the new one, in addition to the counter update. In contrast, the X/Z/T algorithm maintains a cumulative counter of the forwarded packets and a single timestamp for the last forwarded packet. Upon each packet arrival, both timestamp and counter are updated. Obviously, the X/Z/T algorithm has the time/space complexity much smaller than that of the X/Y/T algorithm, which is $O(1)$ versus $O(X)$ per flow.

Let us compare the performance of the two static algorithms. For simplicity, our simulation study assumes $C_r = C_s$. Further, the time required to resolve a destination
ATM address is assumed identical to the time for ATM signaling to establish a VCC, which is denoted by $T_d$. Taking $C_r = C_s = 20$, $\beta = 1$, and $T_d = 200$ ms while fixing $Y = Z$ at 15 sec and $T$ at 30 sec, depicted in Figure 3-3 are the results of switching gain $G$ and the average number of flows $E[a(n)]$ as a function of $X$ for $lbl$-trace, with respect to the two static algorithms. As one can see, the two algorithms achieve similar performance. In conclusion, the $X/Z/T$ algorithm should be adopted since it achieves similar performance as the $X/Y/T$ algorithm with much less complexity.

The SLCS will be built upon the $X/Z/T$ algorithm through the adaptation of the control vector $\bar{x}_n = \{X_n, Z_n, T_n\}$ at each time interval $n$.

3.3.2 Design Complexity Reduction

The proposed adaptive control problem falls into the category of nonlinear stochastic control, which is generally difficult to tackle in terms of optimal control design.

To explore the possible design simplification, we first examine the sensitivity of $G$ to individual control parameters $Z$, $X$, and $T$ in the static sense. Setting $C_r = C_s = 20$ and $T_d = 600$ ms, we performed multiple simulations on the basis of $lbl$-trace and $cisco$-trace, where one control parameter is selected to change at a time in a wide range. The results are summarized in Figure 3-4.

From Figure 3–4 a, d, $G$ is found insensitive to $Z$ when $Z$ is greater than 5 sec. It means that most end-to-end traffic streams have their packet inter-arrival time less than 5 sec, and hence further increasing $Z$ will no longer significantly change the flow
dynamics. Similarly from Figure 3–4 b, e, G is found insensitive to X once X > 5, especially when β is large. Notice that, increasing X has the combination effect of reducing the number of shortcut setups and increasing the number of software forwarding packets. When X is large, the gain through the reduction of shortcut setups is virtually cancelled out by the increase of software forwarding packets. When X becomes small, the significant increase of G with X in Figure 3–4 b, e suggest that a large portion of flows are actually short-lived with only a few packets in each flow. From Figure 3–4 c, f, G is found sensitive to T and monotonically increases with T. The selection of T will be largely restricted by the flow cache table size.

The static sensitivity analysis suggests that it suffices to fix Z and X at some reasonably large values, e.g., Zn = 15 sec and Xn = 20, ∀n . Hence, Tn becomes the only adaptive control parameter in the adaptive FC design.

Now the problem is how to find the optimum sequence Tn that maximizes the overall switching gain. In general, the optimal Tn sequence is a complicated function of flow statistics and flow cache table size. Figure 3–5 depicts a schematic diagram of G as a function of statically assigned T value with given flow statistics and flow cache table size. When T is small, G increases monotonically with T until it reaches a peak value at T = T0. G starts decreasing when T exceeds T0. T0 is the optimal value and a moving target that the adaptive algorithms should be able to track. There are two key factors that determine T0. First, the flow cache table utilization is a direct indicator for how many more flows the flow cache table can take before frequently deleting existing entries immaturely. The decrease of G when T > T0 is a consequence of heavy utilization of the
flow cache table due to overly large T value. Most of the new flows requesting shortcut setup are likely to be blocked when the cache table is heavily utilized, resulting in high cost due to shortcut setup failure and increased packet software forwarding. Second, the application flow composition in the traffic has great impact on the switching gain, since different applications have different flow characteristics. Ideally, each application flow should have its own flow cache timeout value. However, the formidable high implementation complexity renders this kind of CTM algorithms impractical. While using a single timeout value to keep the complexity low, we believe that a good design of CTM algorithms should have a built-in mechanism which recognizes the relative importance of different application flows to the optimal control of T. Also, in [Veeraraghavan 99], the authors pointed out that MPOA technique may not perform very well because of its dependency on the flow composition/characteristics and the difficulty to predict and understand this dependency. However, we believe this difficulty can be overcome to some extent by introducing intelligent systems as we show in this chapter.

Figure 3–3: Performance comparison of the X/Y/T algorithm with the X/Z/T algorithm.
The above analysis implies that the control system to be designed should involve multi-dimensional variables associated with various application flows as well as cache table utilization. Here, we propose to use the NN technique for the design of the control system. The NN is powerful when dealing with complex systems with a large number of variables. With a large number of input variables associated with a pre-selected set of applications and/or application groups, the self-learning/training process of the NN can automatically update the relative weights (or relative importance) of the input variables.

Figure 3–4: Sensitivity of G to Z, X, and T. Upper ones for lbl-trace, and lower ones for cisco-trace. (a), (d): X = 20, T = 30; (b), (e): Z = 15, T = 30; (c), (f): Z = 15, X = 20.
The weights thus obtained are then used on-line for the prediction of the future T value. The NN periodically update T which adaptively tracks the optimal $T_0$ for CTM. So our problem is categorized as a dynamic control problem. In this case, there are two choices for learning/training the NN: on-line and off-line.

In on-line learning the weights are dynamically adjusted. The current state of the system and the input to the system form the input to the NN at time $n$. The output of the NN is used to control the system for the next time interval $(n+1)$. The output of the system at time $(n+1)$ is also used to evaluate the decision made at time $n$. On the basis of this evaluation, the weights are then updated according to a pre-defined algorithm. The on-line learning is not suitable for cache table management because the output of our CTM system is the switching gain, which is not an immediate response to the input T. A much larger time interval is needed to compute the switching gain with high credibility. So, the NN will not be able to track the optimal $T_0$ under this kind of learning algorithms.
In off-line learning, the NN is trained using many input patterns, collected from the real-time application, and their corresponding optimum decisions. The learning process is performed outside the real system. After the learning process is completed, the NN is used to control the real system. During the controlling period, the weights remain fixed. Of course, this method is much simpler than the previous one.

The off-line learning in its simple form is not suitable for cache table management, simply because the traffic level/pattern changes over time. So, we propose a learning method that is a combination of both off-line learning and on-line prediction. The off-line learning is performed periodically at a large fixed time interval, say, one

*Figure 3–5: Schematic view of switching gain G v.s. timeout T.*
hour. The purpose of the off-line learning is to update the weights to track hourly traffic variations. In each off-line learning period, several traffic segments are collected and used to run an off-line learning algorithm. The NN will continue to perform CTM using the old weights until the off-line learning process is finished, when the old weights are replaced by the new ones. Hence, by using this method there is virtually no time limitation on the learning phase. During the on-line prediction period, the weights are fixed and $T$ is updated every $L$ time unit, say, $L = 20 \text{ sec}$. Hence, the purpose of the on-line prediction is to track traffic variations at small time scales. Since the learning process is performed off-line, the complexity for the implementation of the NN control system is low.

### 3.3.3 The Neural Network Design

#### 3.3.3.1 How Does The System Work?

The proposed system is shown in Figure 3–6. The traffic is segmented into $L$-second segments. Each segment is passed through the features extraction process. In this process the following parameters are computed for every segment:

1. Number of packets software forwarded that belong to each application or application group during the current segment. During the current segment, the cache table is controlled using the $T$ value that was predicted by the NN in the previous segment.
2. Number of packets shortcut switched that belong to each application or application group during the current segment. During the current segment, the cache table is controlled using the T value that was predicted by the NN in the previous segment.

3. The total number of active entries currently in the cache table.

In this work, we only consider \{source-address, destination-address\} pair flow granularity, regardless of the applications. However, our algorithm does check to which application or application group each incoming packet belongs to. For each application or application group, the parameters correspond to the total numbers of packets for both software forwarding and shortcut switching within each L-second segment are collected. As shown in Figure 6, the total number of parameters collected is \(m + 2\), which corresponds to \(m\) different applications/application-groups in addition to the number of active entries currently in the cache table and the previous T value. These parameters were selected as the input to the NN. We believe that they contain most of the required information that is needed to specify the optimal timeout \(T_0\) for a given cache size.

In every K minutes, the off-line learning algorithm is activated. First, a brute-force search of the optimal T using the current segment is performed. Then the parameters that were collected for the previous segment and the T value used in the previous segment are taken as the input vector to the NN. The optimal T calculated for the current segment is used as the target value for the learning process. The input vector and the target value are used to form an \(\text{(input, output)}\) pair. Then this process is repeated \(J\) times to form \(J\) \(\text{(input, output)}\) pairs, which are then used for the supervised learning. This is a learning process in the sense that the NN is being trained by adjusting the
weight vector $\vec{\omega}$ to better predict the optimal value of $T$. After this learning process, the resulting $\vec{\omega}$ is then used by the NN for the CTM control in the forthcoming segments. Between two successive off-line learning periods, the NN predicts the future value of $T$ in an attempt to maximize $G$ for an extended period of time $K$.

3.3.3.2 The Neural Network Model and The Learning Strategy

As shown in Figure 3–6 the NN is composed of four layers. The first layer is the input layer. The next two layers called hidden layers. The fourth layer is the output layer with 1 node. We used 130 nodes for each hidden layer to increase the capability of the Neural Network to learn under highly fluctuating traffic.

We identified eleven major application flows. They are http, dns, ftp-data, x-win, ip-in-ip, sunrpc, rsv, smtp, telnet, cmd and kerberos. All the rest of the applications are grouped into one application group. Therefore, we have a total of 12 applications and application group. As explained earlier, the input vector to the NN contains the following parameters:

1. The numbers of packets software forwarded and shortcut switched for each segment of length $L$ for each application/application group, which amounts to 24 parameters.
2. The number of active entries in the cache table.
3. The value of $T$ that was used in computing the 24 parameters.
So, we have 26 nodes in total in the first layer. In our real-trace simulation, we fixed the application decomposition for all the traces we studied. However, one can easily extend our algorithm to allow dynamic classification of applications to reflect the traffic pattern changes. For example, those applications which are found to have very small contributions, can be regrouped under the application group category, whereas those applications initially in the application group category can be re-classified as separate

**Figure 3–6**: The SLCS architecture.
applications once they emerge to be major applications in the traffic flow. Note that, the previous T value is used as part of the input. The reason for that is that the future T value should be highly correlated with the T value used in the past.

Figure 3–7: Neural Network architecture.
In Figure 3–7, $\rho_1$-http represents the number of http packets software forwarded and $\rho_2$-http stands for the number of http packets shortcut switched. The same notation applies for the rest of applications.

The system described above is an example of the well-known Multi-Layer Feed Forward NN [Mehra 92]. We decided to use the Feed Forward NN because it is well known and has been well studied. The object of this work is to find a workable solution, rather than a best solution among various self-learning techniques. We shall show that the NN technique is indeed a viable solution, which results in a near-optimal CTM control.

3.3.3.3 The LRU-Based Adaptive CTM Algorithm

Here we propose another CTM algorithm which serves as a benchmark for the performance analysis of the SLCS.

An algorithm which achieves near-optimal CTM control is the so called Least Recently Used (LRU) algorithm, which ensures the maximization of $T_n$ at any time $n$ without cache table overflow. The LRU algorithm [Galvin 94] was originally proposed for the page-replacement for computer virtual memory scheduling. Consider the cache table management by the LRU algorithm, Upon each shortcut setup request, a free flow entry will be allocated if it is available, otherwise the flow entry which has been idle for the longest period of time will be re-allocated to the new flow. Note that, the LRU algorithm does not always give near-optimal solution because of its nonblocking nature, which may cause the so called thrashing effect. That is, when the cache table size is too small compared with the number of shortcut setup requests, the flow entries can be
frequently added and deleted, leading to frequent shortcut setups and so the substantial reduction of \( G \). However, since up to \( X \) packets in a flow will be software forwarded each time the flow is blocked for shortcut setup, the thrashing effect is much less significant for flow cache management than that for virtual memory management. Hence, the LRU algorithm can be expected to provide near-optimal solutions, except for very small cache table size.

However, the LRU-based algorithm is not very useful in practice due to its high computational complexity. For example, a widely used technique for implementing the LRU algorithm is to keep a stack of doubly linked list of data structures, each of which is further doubly linked to the corresponding flow entry in the flow cache table. The data structure which corresponds to the least recently used flow entry is placed at the bottom of the stack, whereas the data structure corresponding to the most recently used flow entry is placed at the top of the stack. Upon each packet arrival at a given flow entry, the corresponding data structure in the stack will be moved to the top of the stack with the exchange of six pointers. Since the six pointer exchange operations need to be done for each and every packet, which finds an entry in the flow cache table, the computational complexity grows linearly with the increase of traffic volume, which is intolerable for any high-speed switch connections.

Despite its high computational complexity, it is an ideal algorithm for the purpose of performance comparison because it offers near-optimal solution when the cache table size is not too small. As we mentioned at the beginning of this subsection, we use this algorithm just as a benchmark for the performance analysis of the SLCS.
3.3.4 Performance Analysis

3.3.4.1 Simulation Results

Suggested values for the variables that are presented in section 2.3.3.A are \( L = 20 \) seconds, \( K = 60 \) minutes, and \( J = 10 \). Also, we suggest to use an initial training period for the initial operation when the SLCS is just turned on.

However, since the two LAN traces, *lbl-trace* and *cisco-trace* are only 16 minutes and 20 minutes long, respectively, in our simulation we use \( L = 20 \) seconds, \( K = 4 \) minutes, and \( J = 4 \) to mimic a real system. We did not perform initial training due to the shortness of the trace. In spite of the disadvantages of this simulation setting, we shall show that the SLCS can still achieve a near-optimal operation.

First, we compare the performance between the SLCS and LRU-based algorithm for the best-case scenario, i.e., \( \beta = 0 \). The other parameters are set at: \( \Delta t = 2 \) sec, \( C_r = C_s = 20 \), and \( T_d = 200 \) ms. The results are shown in Figure 3-8 for both traces.

It is clear that the SLCS performance is close to that of the LRU-based algorithm. Although the LRU-based algorithm out-performs the SLCS for most of the cache table sizes, the difference between the two is small. It implies that the SLCS also offers a near-optimal performance. When the cache table size becomes extremely small, however, the LRU-based algorithm led to negative performance gain (\( G < 1 \)) because of the thrashing effect, whereas the SLCS out-performs the LRU-based algorithm. When the cache table size is sufficiently large (about 500 entries), however, \( G \) saturates for the SLCS whereas
the LRU-based algorithm achieves higher G values before it saturates at a larger cache table size. The reason is that when the flow cache table size is sufficiently large, it can accommodate most of the incoming flows requesting flow caching, and the LRU-based algorithm can better explore the available flow cache entries without immaturely deleting the existing flow entries, whereas the SLCS uses a fixed T value throughout each $L = 20$ sec interval, which does not fully explore the available flow cache resource within each of these intervals. However, since we are more interested in the situation where the flow cache resource is constrained, we can expect that the SLCS can offer a near-optimal performance as shown in Figure 3–8.

Now, we examine the performance of the SLCS for both traces for the worst-case scenario, i.e., $\beta = 1$. The rest of the parameters remain unchanged. The simulation results are shown in Figure 3–9. One can see that G can reach 16 and 43 for lbl-trace and cisco-trace, respectively. It indicates that MPOA can achieve significant switching gain in an inter ELAN environment, and that the SLCS is a viable CTM control system to attain high performance gain.

In the previous simulations, we set X at 20. To study the sensitivity of the performance to X, we further used the LRU-based algorithm to find an optimal X for the maximum switching gain using lbl-trace, at each given table size of 200, 300 and 400, respectively. Our comparison study indicates that such maximum performance achieved by the LRU-based algorithm using the optimal X is within the 5% difference from the performance achieved by the SLCS using the fixed X=20 at each given table size. In
other words, not only the performance is insensitive to \( X \) given the range of \( X \) is properly selected, but also the SLCS indeed provides the near-optimal performance.

So far, the studies are based on the intranet traces. We now examine the performance of the SLCS using a backbone Internet trace, \( \text{fixwest-trace} \). Fix \((C_c, T_d, \beta)\) at \((20, 0.2, 0.5)\) with \( C_s = C_r \) while the rest parameters remain unchanged. As one can see in Figure 3–10, \( G \) saturates at about 3.8 even as the cache table size approaches 40,000. Obviously, MPOA achieves much less switching gain in the backbone Internet environment as compared to that in the campus intranet environment. Given the 27.3% utilization of \( \text{fixwest-trace} \) on a 100 Mbps FDDI ring, its cache table demand is also significant. For instance, the corresponding cache table demand on an OC-3 link in backbone networks may well exceed 100,000.

Clearly, it must be the difference between the flow dynamics of campus intranet and backbone Internet that has the major impact on the MPOA performance. Here, we further decompose the flow dynamics into different applications at the \( \text{host+port} \) granularity. Listed in Table 3–1 are the detailed average statistics of such flow dynamics for the three traces, given the flow timeout value \( T = 128 \text{ sec} \). For clarity, only the statistics of major applications are collected. For \( \text{cisco-trace} \), the major applications are represented by those whose flow ratio is greater than 1% or packet ratio is greater than 8%. For the other two traces, they are represented by those whose flow ratio is greater than 5% or packet ratio is greater than 4%. For the two campus traces, let us first focus on the unknown applications. Although these applications comprise only a small fraction of the total flows, they actually represent the major portion of the total packets. In other
words, the flows of these applications consist of a large number of packets, which are best suited for shortcut switching. They are the major contributors to the large switching gain achieved by MPOA for the campus traces. Another major contributor is the well-known tcp X-window application.

Notice that, both http and dns applications appeared in lbl-trace and fixwest-trace, but not in cisco-trace. It is because the former two traces were collected from educational institutes while the latter one is from a vendor. In contrast, http, dns and ipip in fixwest-trace are the three applications which contribute to the major portion of the total packets. While ipip is suited for shortcut switching, most flows in http and dns consists of a small number of packets. Yet, both http and dns contribute 74% of the total flows. This is why only a marginal switching gain is achievable by MPOA for fixwest-trace. Similarly, the reason for lbl-trace to achieve significantly less switching gain than that of cisco-trace is the non-negligible impact of http and dns applications.

In summary, both http and dns applications in the backbone Internet environment are the major factors to the ineffectiveness of the data-driven MPOA technique. In contrast, since data file transfers are the major applications in the campus intranet environment, a significant performance gain can be achieved by the MPOA technique.
### 3.3.4.2 Complexity Analysis

For the SLCS the update interval $\Delta t$ is fixed at 2 sec. After each update, every active flow entry is checked to see if it should be deleted by comparing its idle time with the present $T_n$. It requires one subtraction and one comparison operation per active flow.

**Table 3–1: Average flow Statistics of both LAN and WAN traces.**

<table>
<thead>
<tr>
<th>trace</th>
<th>(port, port)</th>
<th>flow%</th>
<th>pkt%</th>
<th>byte%</th>
<th>pktpf</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>cisco</strong></td>
<td>(tcp, 4240 = unknown)</td>
<td>0.0</td>
<td>36.1</td>
<td>10.4</td>
<td>556773.0</td>
</tr>
<tr>
<td></td>
<td>(tcp, 6000-6063 = Xwin)</td>
<td>0.3</td>
<td>8.8</td>
<td>3.6</td>
<td>666.6</td>
</tr>
<tr>
<td></td>
<td>(udp, 111 = sunrpc)</td>
<td>2.3</td>
<td>0.1</td>
<td>0.0</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>(udp, 1020 = unknown)</td>
<td>0.4</td>
<td>8.7</td>
<td>9.6</td>
<td>542.7</td>
</tr>
<tr>
<td></td>
<td>(udp, 8224 = unknown)</td>
<td>2.1</td>
<td>0.4</td>
<td>1.9</td>
<td>4.5</td>
</tr>
<tr>
<td><strong>lbl</strong></td>
<td>(tcp, 80 = http)</td>
<td>5.5</td>
<td>3.0</td>
<td>2.7</td>
<td>11.2</td>
</tr>
<tr>
<td></td>
<td>(tcp, 514 = cmd)</td>
<td>0.0</td>
<td>4.2</td>
<td>8.6</td>
<td>2392.7</td>
</tr>
<tr>
<td></td>
<td>(tcp, 1023 = rsv)</td>
<td>0.0</td>
<td>4.8</td>
<td>3.5</td>
<td>43673.0</td>
</tr>
<tr>
<td></td>
<td>(tcp, 6000-6063 = Xwin)</td>
<td>0.1</td>
<td>4.3</td>
<td>5.1</td>
<td>978.5</td>
</tr>
<tr>
<td></td>
<td>(udp, 53 = dns)</td>
<td>5.3</td>
<td>1.7</td>
<td>0.6</td>
<td>6.3</td>
</tr>
<tr>
<td></td>
<td>(udp, 1021 = unknown)</td>
<td>0.2</td>
<td>41.6</td>
<td>27.8</td>
<td>5270.5</td>
</tr>
<tr>
<td><strong>fixwest</strong></td>
<td>(tcp, 20 = ftp-data)</td>
<td>0.5</td>
<td>7.7</td>
<td>1.0</td>
<td>213.0</td>
</tr>
<tr>
<td></td>
<td>(udp, 80 = http)</td>
<td>33.6</td>
<td>36.2</td>
<td>43.2</td>
<td>13.5</td>
</tr>
<tr>
<td></td>
<td>(udp, 53 = dns)</td>
<td>40.1</td>
<td>15.3</td>
<td>7.0</td>
<td>4.8</td>
</tr>
<tr>
<td></td>
<td>(ipip or MBONE)</td>
<td>2.2</td>
<td>18.1</td>
<td>18.0</td>
<td>102.4</td>
</tr>
</tbody>
</table>
entry. Unlike the LRU-based algorithm, no separate data structure needs to be built. For both $lbl$-trace and $cisco$-trace, the maximum number of concurrently active flows is about 500, see Figure 7 and Figure 8. So up to 500 flow entries need to be checked every $\Delta t$ interval, i.e., up to 250 flow entry checks per second. Thus the computational complexity related to the cache table of the SLCS is insignificant.

For the learning process, it is performed using independent processing components. So we are not concerned about the computational complexity that is involved in the learning process. Usually, the convergence time is a critical issue in the systems that use the NN for their operations.

However, the convergence time is not a problem for the SLCS for two reasons. First of all, since the off-line learning process is performed on an hourly basis, training patterns are not accumulated throughout days or weeks. Thus, our NN convergence time is in general very short. Second, it is not required that the learning process be completed immediately after the $J$ vectors are collected. The NN can use the old weights in its decision making while the new set of weights is being processed.

### 3.4 Conclusion

We developed a framework for the flow classification design and performance analysis of MPOA network. A timeout mechanism was introduced to flow cache table management for performance improvement with reduced complexity. An effective self-learning system based on the NN technique was proposed to achieve a near-optimal control for the maximum performance gain. Our study indicates that MPOA can offer
significant performance gain over the traditional routed network in an inter ELAN communication environment. As we have seen, application composition in traffic has great impact on MPOA performance. As more and more multimedia services, such as MBONE, being deployed and more features being added in http application, such as JPEG pictures and MPEG videos, one can expect that the MPOA performance will be improved in an Internet backbone environment.
Figure 3–8: Switching gain G for both the LRU-based algorithm and the SLCS.

(a): lbl-trace and (b): cisci-trace.
Figure 3–9: Switching gain $G$ for the SLCS at $\beta = 1$. 
Figure 3–10: Switching gain $G$ v.s. flow cache table size for fixswest-trace.
Chapter 4

MULTI-CLASS INTERNET WITH CONNECTION-ORIENTED SERVICES, A NEW TECHNIQUE

4.1 Introduction

The issue of QoS provisioning has become an essential requirement in any Next Generation Internet architecture. Many researchers are trying to develop new protocols and switching architectures in an attempt to achieve some degrees of QoS guarantee end-to-end. Basically, we can divide the proposed solutions into two categories. The first category contains all the proposals that suggest introducing different CoSs for the Internet. The second category suggests for expanding network’s bandwidth to a degree at which no congestion can occur. An all-optical network is the main objective of the latter group.

We believe all-optical network is certainly coming, but this may take 10 years or more. So, it is necessary to provide an intermediate solution that can maximize both revenues and users' utilities by introducing a multi-class network [Cocchi 93, Honig 95]. There are many users who are willing to pay more than the usual flat monthly charge [Shenker 96], if they can get better QoS. Of course, if different CoSs are going to be introduced, then an advanced pricing system should also be introduced [Orda 97].
As the Internet offering quickly finds wide-spread market, there has been much recent discussion about the role of pricing in computer networks. Pricing is being proposed not only to support multi-class networks, but also as a congestion control mechanism [Peha 97]. There is a debate between two groups of researchers. One group supports a so called "flat pricing"; the other supports a usage-based pricing. The first group believes usage-based pricing is not necessary and may cause negative effects on the Internet [Odlyzko 98]. The second group believes usage-based pricing is necessary to maximize revenues and improve the Internet efficiency [Breker 96, Edell 95, Paschalidis 98]. In [Shenker 96], there is a nice treatment of this debate.

The issue of providing and guaranteeing QoS is most important when considering any proposed multi-class network. Currently, some proposed Dynamic Bandwidth Allocation and pricing systems attempt to respond to every QoS request made by participating applications/end-users [Fulp 98]. In our opinion, these proposals represent the other end of spectrum, since the current Internet provides one CoS (Best Effort), with flat pricing. The question that forces itself is that, is it necessary to respond to every QoS request made by every application/end-user? Or, is it enough to design a network such that it can provide different CoSs with different well-defined QoS and associated prices?

We think the latter approach is more practical. Many examples in our lives support this philosophy, in the airline industry, usually there are three classes, first, business, and economy. The customer is responsible to select the class he/she wants or can afford. A similar situation can be found in many of the utility services being provided, such as telephone and cable TV services. The customer pays for the CoS he/she
wants, it is then his/her responsibility to efficiently utilize the available resources in this CoS. When one leases a T1 or T3 line, one pays the flat price regardless of the utilization of the line.

Although the idea of different levels of service has been introduced under the DS [Nichols 99, Nichols 01] and MPLS architectures [Rosen 01-a], the idea of segregating the bandwidth of the network into several CoSs with well-defined and declared QoS parameters was not given enough attention.

The main contribution in this chapter is that we formulated a framework for a multi-class network and proposed techniques to provide guaranteed QoS in an attempt to overcome the problems from which the DS architecture suffers. DS architecture fails to guarantee absolute QoS end-to-end [Feng 98, Ibanez 98, Yeom 99]. But we still consider our framework as a new version of the DS architecture and we don't classify it as Integrated Services architecture. We don't require per-micro-flow nor per-aggregate-of-flows state information at the core of the network, unlike MPLS. This is done by the novel approach we propose to emulate connection-oriented services over an IP layer. In this Multi-Class Internet, we propose three CoSs with well-defined QoS parameters. The three CoSs are Premium, Assured, and Best Effort, similar to the terminology used in the DS architecture, but with different QoS characteristics. We propose usage-based pricing for the first two CoSs and flat pricing for Best Effort CoS.

The rest of the chapter is organized as follows: in section 4.2, we give background and motivations. In section 4.3, we present the framework of proposed Multi-Class Internet. In section 4.4, we explain the mechanisms for connection establishments and
Call Admission Control (CAC). In section 4.5, we discuss the implementation complexity of our proposal. We wrap up by the conclusions in section 4.6.

4.2 Background

In this section, we discuss the issues of QoS guarantee over a global network, service differentiation, and Data/Voice convergence. First, we discuss some of the proposed solutions. Second, we present our vision regarding this important issue. Before that, we like to point out that QoS routing is a very essential issue when designing multi-class networks, however, this issue is outside the scope of this work.

Some proposals attempt to provide service differentiation by modifying the TCP protocol [Gevros 99], but moving in that direction disallows defining an absolute QoS. QoS guarantee is one of the most important requirements to achieve Data/Voice convergence and multimedia integration. However, these proposals can be very useful in optimizing the performance of the Assured CoS [He 00].

The DS architecture [Nichols 99], which seems to be attractive nowadays because it avoids complexity in the core of the network and pushes it to the edge devices, has failed to prove absolute QoS can be guaranteed through the global network. In fact the work in [Feng 98, Ibanez 98, Yeom 99] showed that declared QoS cannot be always guaranteed by simply assuming enough resources in the core of the network.

Actually, we can also prove that the DS architecture cannot guarantee the declared form of differentiation end-to-end, all the time. A very important QoS metric here, is the end-to-end dropping probability of packets. In the DS architecture,
differentiation between services is proposed to be done hop-by-hop by implementing per-hop behaviors. It is important to realize that differentiating between two CoSs for a specific hop does not mean that the same level of differentiation will be achieved end-to-end, as we explain in the following paragraph.

Here, we consider two classes only, to simplify the discussion. Allocating different amounts of resources (buffer, bandwidth, or priority) for two CoSs based on per-hop behaviors will result in differentiating the dropping probability of packets at each hop, if any congestion occurs. Let us assume that our objective is to make the dropping probability of class-2 twice that of class-1 per hop, and that dropping events at all hops are independent. Then the ratio between the dropping probabilities of the two classes end-to-end can be calculated as shown in (4–1). Let \( \gamma_1(i) \), \( \gamma_2(i) \) be the dropping probability of class-1 and class-2 at the \( i \)th hop, respectively. Let \( \theta_21 \) be the end-to-end ratio of class-2 to class-1 dropping probability. Let \( M \) be the number of hops end-to-end. Then:

\[
\theta_{21} = \frac{1 - \prod_{i=1}^{M} (1 - \gamma_2(i))}{1 - \prod_{i=1}^{M} (1 - \gamma_1(i))} \quad (4–1)
\]

Figure 4–1 shows the end-to-end dropping probability ratio for a special case when \( \gamma_1(i) = 0.1, \gamma_2(i) = 0.2; i = 1, 2, ..., 30 \). Usually a 30-hop connection is assumed to be the maximum possible length for any Internet connection [Stallings 97]. Normally, regular Internet connections to major web sites have around 10 hops. On the other hand, in-campus Internet connections may have less than 5 hops. The reader is invited to try the MS-DOS command ‘trcaert’ to find out the number of hops associated with an Internet connection.
connection to any valid IP address. Note that, the values we assumed for $\gamma_1(i)$ and $\gamma_2(i)$ are not frequently encountered in a real backbone network. Much smaller values are usually experienced in a well-designed network. At lower dropping probabilities, (4-1) becomes more flat, approaching a horizontal line. However, one is interested in enforcing the differentiation end-to-end specifically during highly congested periods of time, in order for the differentiation in the QoS between different CoSs to be meaningful. So, the DS architecture is not only questionable in guaranteeing the declared QoS parameters, but also the differentiation between different CoSs can not be predicted end-to-end in a congested or highly utilized network.

Figure 4–1: End-to-end dropping probability ratio of two CoSs versus number of hops.
It should be clear that we are not attacking the DS architecture, which has attracted a great amount of research. But, we are clarifying why the DS architecture in its original shape that was proposed by the IETF cannot fulfill the requirements of the Multi-Class Internet that is discussed in this chapter. In addition to that, ATM networks, which can deliver all the QoS requirements, is unable to force itself in the final picture of the global network. It is becoming clear nowadays that the IP layer is going to stay for many coming years [Rutkowski 99]. This creates the necessity for practical solutions that can be implemented on the current network technologies and protocols and can deliver at least some of the QoS requirements of the evolving advanced applications.

MPLS [Rosen 01-a] is another architecture that attempts to provide QoS guarantee to networks. However, due to its scalability problems, MPLS is being proposed to be implemented only at the backbone of a network, while the DS can be implemented at and before the edge routers. Under the MPLS architecture absolute QoS parameters can be guaranteed, this is because, in MPLS, end-to-end paths can be identified at the beginning of a session and can be fixed the whole life of that session. However, MPLS introduces extra overhead into both packet’s header and routers. It requires an MPLS header be added above the IP header and it also requires matching tables at all routers inside an MPLS domain. These tables are necessary to bind labels with the corresponding forwarding behaviors (switching and priority decisions). In this chapter, we discuss the tradeoffs between a proposed novel approach that depends on the concept of source routing, but with much less overhead, and the MPLS architecture, in terms of their ability to emulate connection-oriented communications and the introduced overhead.
Based on the current situation of the Internet, we believe, the practical solution is to introduce a multi-class network that can be friendly to the IP layer, while at the same time it avoids complexity in core of the network, and, most importantly can, to some degree, guarantee QoS parameters. In the following section, we describe our Multi-Class Internet and show how our proposal attempts to answer many of the concerns addressed previously.

4.3 The Multi-Class Internet

Different applications/end-users have different QoS requirements and for each application/end-user there is a tradeoff between QoS and cost. In [Fulp 98], there is a good analysis of this tradeoff and how it can be used to optimize the performance of a network. However, there is a large range of QoS requirements for existing and emerging applications. A quantization can be done on this range, resulting in well-defined CoSs that can help in developing a reasonable pricing system that will be, more likely, accepted by customers, since customers dislike any complicated pricing system as discussed in chapter 7. Well-defined CoSs can significantly improve performance of many architectures that have been implemented or proposed to integrate the IP layer with ATM layer (MPOA) or with optical layer. Thus, with well-defined CoSs, there is no need for complicated methods to identify or to evaluate every new call in an attempt to determine if it deserves an ATM/optical shortcut or not. [IEEE Comm. Mag. 99, Taha 01].

We define only three CoSs as follows:
1- **Premium CoS**: can guarantee the peak rate of an accepted call, emulating the performance of continuous transmission of information using a direct link, plus switching time. From a data network point-of-view, this means a free-of-queuing-delay delivery of packets.

2- **Assured CoS**: can guarantee a minimum average bit rate for an accepted call. This CoS will operate under a modified TCP protocol. Queuing delay is possible in this CoS, which may also produce delay jitters. Packets under this CoS will always have priority to be served before packets of Best Effort CoS. Several sub-classes can be defined under the Assured CoS, where each sub-class can guarantee specific delay jitter limits by enforcing specific scheduling and priority schemes.

3- **Best Effort CoS**: does not hold any privilege. Sessions under this CoS will have a service similar to what is available nowadays in the Internet world. Similarly, several CoSs can be defined within the Best Effort CoS. Both, Premium and Assured CoSs are connection-oriented services, while Best Effort CoS is a connectionless service.

In section 3.4, we explain how we can emulate connection-oriented services without any per-micro-flow or per-aggregate-of-flows state information in the network core.

With these three CoSs defined at each output port of the Multi-Class network, we will have bandwidth segregation as shown in *Figure 4–2*. Each CoS competes with other CoSs to consume as much bandwidth as possible to answer the demand and minimize the loss. This competition shall follow the following rules:
1- *Premium* CoS can consume at most a bandwidth of $Q_P$, also the Premium CAC unit considers an available physical bandwidth of $Q_P$.

2- *Assured* CoS can consume at most a bandwidth of $(W_t-Q_E)$, where $W_t$ is the total physical bandwidth of the output link. However, the Assured CAC considers an available physical bandwidth of $(W_t-Q_E-Q_P)$, only. This means, assured flows can expand and utilize any unused bandwidth in the Premium CoS.

3- *Best Effort* CoS can theoretically consume all available bandwidth $W_t$ in the absence of any flows in the other two CoSs. But, at any time, a minimum bandwidth of $Q_E$ is guaranteed for the Best Effort CoS to prevent starvation.

---

*Figure 4–2*: Bandwidth segregation and sharing in the proposed architecture.

We believe, these rules can be fulfilled by assigning an independent queue for each CoS, at every output port. Weighted fair queuing has proved its ability to distribute services on different queues with specific ratios. So, by adjusting the service rate according to the values of $Q_P$ and $Q_E$, we can guarantee that each CoS will get the necessary service rate
that can guarantee the promised QoS to all admitted calls under Premium and Assured CoSs. Note that, any dynamic queuing scheme should be work-conserving. So, when there are no packets to be served in the Premium queue, packets in the Assured queue will automatically receive an extra service power. Furthermore, at those moments where there are no packets to be served in the Premium and Assured queues, packets in the Best Effort queue will automatically receive all the serving power at these specific instances. With this sharing of serving power, we can guarantee a high utilization, as long as there is enough demand on one or more of the three CoSs. Thus, we overcome the drawbacks of resource reservations.

Since our objective in this research is to develop a framework for a multi-class network at the architecture level, we are not going to focus on the multiple-queuing system. From now on, we will assume the above three rules can be fulfilled, and that the resources required by calls of different CoSs is represented by a specific amount of requested bandwidth. With this segregation of bandwidth at each output port, we can imagine the global network as three overlapping networks sharing the same physical resources as shown in Figure 4–3.
4.4 CAC and Connection Establishment

Our work can be considered as an extension of the work done on scalable Resource Reservation Protocol (RSVP), as in [Almesberget 97, Yeh 01]. However, our work seeks to emulate connection-oriented sessions above the IP layer and introduces different CoSs into the design of the proposed next generation Internet. We borrowed the idea of cumulative resource reservation from [Almesberget 97], but our system does not estimate actual transmission rates of users to estimate reservations, it simply considers resources requested by users, as we explain later. However, our multi-class architecture allows resource sharing between different CoSs to avoid wastage of resources. In the following we explain, in details, the protocols involved in establishing a connection within the proposed multi-class Internet.
First, since there are three CoSs, every packet should have a field that identifies to which CoS the packet belongs. A major assumption here is that the classification process can be done at speed of a link. If classification of packets cannot be done at this speed, talking about multi-class networks and service differentiation will be obsolete. A three-bit field is necessary because we need to identify the following cases: first, packets belonging to Best Effort CoS; second, Premium call-setup packets; third, Assured call-setup packets; fourth, packets belonging to an established Premium connection; fifth, packets belonging to an established Assured connection; sixth, Premium release packets; and seventh, Assured release packets.

In the proposed Multi-Class Internet, Best Effort flows will follow classical hop-by-hop forwarding, and hence, no call-setup process is required for this CoS. However, both Premium and Assured CoSs are connection-oriented services that include a call-setup phase. Whenever an end-user (host-A), Figure 4–3, asks for a new connection under Premium or Assured CoSs to another end-user (host-B), the following protocols are followed:

1- Host-A sends a request packet to an edge device (ED1). This request packet contains the CoS and the bandwidth required. Once ED1 receives the request packet, it saves the state information of this new connection.

2- Then, ED1 sends a call-setup packet (scout or discovery packet) to the intended destination; this call-setup packet will be hop-by-hop forwarded based on QoS routing tables. At each output port, through this journey, CAC units will reject or accept this new call and will mark the call-setup packet by "rejected" or
"accepted". If a call is rejected at a specific output port, the call-setup packet will continue to be forwarded to the destination and CAC units will continue to decide regardless of the previous decisions. In addition, the ID-number of each output port corresponding to each input port in the forward path is written to the call-setup packet, in the same sequence that these output ports are being visited by. So, the ID-numbers of output ports that are written in the forward path are those that define the backward path. Also, the maximum number of ports in each visited router is written in the call-setup packet, in the same order that these routers are being visited by. For example, assume the call-setup packet will move through link \((N_x-N_y)\), Figure 4–3, on its journey from ED1 to ED2, then at node \(N_y\), the ID-number of the output port corresponding to link \((N_y-N_x)\) is written, and so on.

Two basic assumptions are being made here. First, a router can assign ID-numbers to its output ports and recognize them; second, all links between routers are bi-directional. Note that, so far, CAC units only write their decisions and no actual reservation of bandwidth occurs. Also note, QoS routing is defined on the call level and not per-packet, the philosophy we propose here is, route first, then switch. So, whenever a new call is received by a router, the routing algorithm will try to find the best path for this new call and that path will be fixed for the duration of this call.

3- Once the call-setup packet reaches ED2, ED2 checks if there are any "rejected" messages written. If there is one or more "rejected", ED2 sends a negative acknowledgment (NACK) to ED1 which then deletes the state information and
sends a notice to host-A. NACK or PACK (Positive Acknowledgment) packets will be classified as call-setup packets with an acknowledgment field activated. The reason for having a clear definition for the classes of the call-setup, release, NACK, and PACK packets is to guarantee quick signaling and connection establishment, since these packets will be served by Premium or Assured queue depending on the CoS of the new connection under establishment. On the other hand, if ED2 finds no "rejected" messages, it sends a request packet to host-B. If host-B accepts the call, it sends a notice to ED2, which saves the state information for this connection and reacts with one of two actions, depending on the type of connection. If the connection is unidirectional from A to B, ED2 sends PACK to ED1. PACK will use the list of output ports, available from the call-setup packet that came from ED1, to define the backward path from ED2 to ED1. While PACK is moving to ED1, it reserves the required bandwidth at each output port that forms the original forward path, and writes its ID-number. So, the ID-numbers of the output ports that are being written are those that form the original forward path from ED1 to ED2. If the connection is bi-directional, ED2 will send a call-setup packet asking for establishing a connection to ED1, in addition to sending a PACK to ED1. ED2 deletes state information once it delivers the PACK or NACK message.

4- Once PACK reaches ED1, ED1 saves the output ports list and binds it with the state information for this connection. Also, ED1 triggers an accounting module. Then, it sends a notice to host-A informing it that the connection is ready. When
host-A starts sending information, the list of output ports available at ED1 will be inserted in the header of each packet coming from host-A. By this, all packets belonging to the connection A-B will follow the same path, through which enough resources have been reserved to guarantee the declared QoS. All these packets will belong to an established Premium/Assured connection; no routing will be applied to these packets. Once one of these packets reaches a router, the router will switch this packet to the output port that has the ID-number appearing on top of the list. This ID-number will be deleted after that, and the next ID-number will be pulled up to the top of the list, and so on.

5- While a PACK is returning to ED1 and reserving bandwidth, it may happen that some output ports reject the connection due to fast changes in utilization. In this case, PACK will be changed to NACK at the router where this rejection occurs and the ID-number of this output port will be marked. Once the NACK reaches ED1, ED1 will recognize the NACK was not generated by ED2. Then, it sends a special type of release packet that will be activated only after the router where the unexpected rejection occurs. Note that, we use the words "request" and "notice" when we describe communications between a host and an ED; here we are not concerned about the exact protocols that will be used between a host and an ED. This depends on the technology implemented in the LAN containing the host.

6- When host-A ends its session, it sends a termination notice to ED1, ED1 sends a release packet to ED2, removing all the reserved bandwidth throughout the whole path. ED1 turns the accounting module off and deletes the state information and
the output ports list. When ED2 receives the release packet, it sends a notice to host-B. In addition to that, an ED can have a timeout value. So, if there is no traffic being transmitted from a specific host for a period more than the timeout value, that ED sends a release packet and removes all reservations that belong to that specific connection. By this mechanism, the network can be protected from misbehaving hosts.

CAC units are, simply, counters that count the amount of currently reserved bandwidth. Whenever a new call comes in, CAC units accept the new call if the following inequalities are true: 

\[ R_P + U_P < Q_P \] for Premium Calls, 

\[ R_A + U_A < (W_t - Q_P - Q_E) \] for assured calls. 

Where, \( R_P \) and \( R_A \) are required bandwidth by Premium and Assured calls, respectively. \( U_P \) and \( U_A \) are utilizations of both Premium CoS and Assured CoS, respectively. Again, this is not the measured utilization, this is the total bandwidth reserved. For example, the measured Assured utilization can exceed \( (W_t - Q_P - Q_E) \) while the total reserved bandwidth does not exceed this value and new calls can still be admitted. Figure 4–4 shows a schematic view of the proposed architecture and protocols.

Note, Premium sessions do not use TCP protocols, instead, any streaming protocol can be used. Also, since bandwidth is reserved for any Premium session, there is no need for a startup stage; transmission can be immediately adjusted to be at or below the peak reserved rate. For Assured sessions, a modified TCP protocol must be used, since there is no need to decrease the transmission under the minimum guaranteed average rate when congestion occurs. Modifying the TCP protocol for the Assured CoS is outside the scope of this work; however, the work that has been done in [He 00] shows
promising results in guaranteeing the minimum transmission rate under a TCP protocol with connection-oriented sessions.

Note also, the above protocols share some concepts and techniques that can be found in source routing algorithm [Estrin 96] and RSVP protocols [Braden 97]. The major difference is the usage of output ports ID-numbers and also in the compatibility with multiple CoSs and CAC algorithms.

Figure 4–4: Architecture of Multi-Class Internet and the mechanism of emulating connection-oriented services over an IP layer.

Note also, the above protocols share some concepts and techniques that can be found in source routing algorithm [Estrin 96] and RSVP protocols [Braden 97]. The major difference is the usage of output ports ID-numbers and also in the compatibility with multiple CoSs and CAC algorithms.
4.5 Implementation Complexity of Proposed Multi-Class Internet and Protocols Emulating Connection-Oriented sessions

This clear definition of three CoSs requires, as we explained earlier, three independent queues at each output port. Today's new routers support multiple queues at each output port, which makes implementation of the proposed Multi-Class network a practical idea. CAC units can be totally implemented by software algorithms; counter and comparison functions are basic building blocks of Premium and Assured CAC units. Also, today's routers can provide a classification speed up to the link's speed. On the other hand, due to the possibility of rejecting a session while it is physically reserving bandwidth in its way back to the source ED, bandwidth may be temporarily wasted until a release packet is initiated to remove the reserved bandwidth from, possibly, several nodes. Also, Denial of Service (DoS) attacks may cause a serious problem, as is the case with many other connection-oriented protocols. However, because of the sharing policies proposed in our architecture, Assured and Best Effort sessions can expand and utilize that unused bandwidth until the reservation is canceled by a release packet, this can significantly decrease the negative effect of this phenomenon.

The proposed idea of inserting a list of output ports numbers in the header of each packet is possible. Actually, the overhead introduced to the header of an IP packet using the list of ID-numbers of output ports is comparable to the overhead introduced in MPLS architecture. In MPLS, the MPLS header is 32 bits, out of which 20 bits form the MPLS label [Rosen 01-b]. In our architecture, we proposed to write the ID-number of the visited output port at each visited router. Also, the call-setup packet collects the maximum number of ports at each visited router. Thus, the number of bits required to represent an
ID-number of an output port depends on the maximum number of ports at the corresponding router. So, in the backbone we expect routers to have very small number of ports, as small as 8 ports, which require 3 bits only. As we move toward the edge routers, usually routers will start to have more and more ports. Within the last mile facilities some routers may have thousands of ports, which may require more than 10 bits. Thus, the exact overhead in our architecture is variable and depends on the length and path of a connection. Note also, while a packet is forwarded, once an ID-number is used it will be removed out of the stack, which will reduce the size of the overhead. In Figure 4–5 we show an example of a stack of ID-numbers that can be seen in the header of an IP packet implementing our proposed protocols for a session connecting an end-user behind a modem-pool, for example, to a major web site. Note that, each router knows its maximum number of ports, so it reads only the corresponding number of bits from the header, since all the ID-numbers are written in the correct order. Note that, in Figure 4–5, the header in the first hop will be 57 bits, while in the last hop it will be 10 bits only.
Consider that one bit traveling one hop is equivalent to a load of 1. For a scenario as in Figure 4–5, there are 11 hops, so MPLS will introduce a load of $32 \times 11 = 352$. On the other hand, our proposed protocol will introduce a load of 393. Thus, our protocol in this example introduces a little more load on the network than MPLS. However, for shorter connections, and specifically for backbone connections, where MPLS is proposed to be implemented [Rosen 01-a], our protocol definitely introduces lesser load on the network than MPLS. Even for long end-to-end connections, the load introduced by our protocol is not too high.

However, introduced overhead and load on the network are not the only comparison criteria. Our proposal removed overhead from routers and eliminated the

Figure 4–5: A schematic view for an example of a stack of output ports’ ID-numbers and the corresponding required number of bits for a possible 11-hop connection.
need for forwarding (switching) tables, i.e. non-scalable state information, which is required in MPLS. Our proposal added the forwarding/switching decisions’ overhead directly into the packet’s header. Our proposal is scalable, in the backbone, where the number of flows can be huge, routers have small number of ports, a 3-bit field may be enough to represent all the switching decisions at a specific router. So, even if there are hundred of thousands of flows they all need to use as small as 8 ID-numbers. In MPLS, as we move toward the backbone, the required number of labels become so large, if end-to-end connections are to be supported. This results in large tables, which slow down the packet processing speed.

All other proposed fields are only used in the call-setup and ACK packets, which form a negligible amount of total traffic. Some proposals in resource allocation for quality applications, [Fulp 97], suggest negotiation protocols that require much more signaling overhead than what we propose here. However, the large overhead introduced into call-setup and ACK packets, despite the fact that they form a tiny fraction of the total traffic, will consume a non-negligible ratio of the total bandwidth. We argue here, that the small fraction of resources utilized by control and signaling information will buy us guaranteed QoS parameters.

4.6 Conclusion

In this chapter, we formulated a framework for a Multi-Class Internet. Bandwidth segregation was proposed to support three CoSs, Premium, Assured, and Best Effort. Despite the segregation, the proposed framework forces specific policies that aim at
maximizing utilization by allowing controlled sharing of bandwidth between the three CoSs. Protocols were proposed to emulate connection-oriented services over IP layer, these protocols were designed to be scalable and remove overhead from routers and introduce it to packet’s header, unlike MPLS architecture where overhead is introduced to both, packets’ header and routers.
Chapter 5

DYNAMIC BANDWIDTH ALLOCATION IN MULTI-CLASS CONNECTION-ORIENTED NETWORKS

5.1 Introduction

Dynamic Bandwidth Allocation (DBA) for networks carrying multi-class traffic has been investigated within different contexts. We are concerned with architectures that work on the call level. Most of the work on this level was directed to dynamically allocate bandwidth to different Virtual Paths (VPs). The virtual path concept is a basic characteristic of ATM and ISDN networks, which provide connection-oriented services. VPs are usually designed based on topological optimization. Also, bandwidth can be segregated between different CoSs with different QoS requirements, as in [Chan 94].

Bandwidth segregation is an effective way of managing resources to serve different CoSs of traffic with different QoS requirements. However, as was shown in [Maunder 98], allocating bandwidth to VPs may lead to a lower efficiency (throughput) if bandwidth is not carefully (optimally) allocated. We point out that the basic reason for non-optimal allocation of bandwidth is the difficulty and inaccuracy in predicting demands on different CoSs. One of the essential reasons for this inaccuracy is the nature of most DBA algorithms, which are absolutely distributed.
Much work has been done recently on traffic engineering over the MPLS architecture [Rosen 01-a]. However, most of the efforts were directed to dynamic load balancing on alternative paths in order to efficiently utilize resources while achieving the best possible (QoS), like the work in [Kar 00, Dinan 00]. Although CoSs were also supported in these studies, little efforts were directed to optimize the allocation of bandwidth among CoSs sharing a common link.

We believe DBA, traffic engineering, and routing are related tasks and cannot optimally perform without cooperation with each other. Routing and load distribution/balancing on alternative paths -Label Switched Paths (LSPs) in MPLS context- can be viewed as a necessary process on small time scales to efficiently utilize scattered resources. This process takes place at the edge of an MPLS network (ingress nodes). But at the same time, there can be a competition on bandwidth between different LSPs, which may represent different CoSs, at each link in the network, both at the core and at the edge of an MPLS network. As the aggregated demand on different CoSs/LSPs varies slowly over time, a DBA process is necessary to optimally distribute bandwidth between competing CoSs/LSPs based on a specific objective function. This objective function usually takes into consideration user-utilities and revenues which are functions of QoS, pricing model, and regulations governing different CoSs. From here on, we will use the concept of CoSs to refer to both LSPs and classes of different levels of QoS in any connection-oriented multi-class network.

In the context of connection-oriented services, usually objective functions can be designed at higher levels, like controlling the call blocking rates of calls. We can assume
that a lower level control will take the responsibility of computing the required bandwidth by each call, based on the CoS of that call, the QoS specifications of that CoS, and the optimization criteria defined for that CoS. This computed bandwidth is usually called “effective bandwidth” [Cue’rin 91]. So, these low level controls may for example minimize the average delay of sessions within a specific CoS or minimize the average packet loss rate for sessions within another CoS, and so on. On the other hand, a higher level control is needed to allocate (segregate) bandwidth between the CoSs in an optimal way. We recommend that these two levels of control be carried at different layers, this would achieve two objectives: first, routing stability; since this dynamic segregation of bandwidth would usually be a slow process and thus QoS routing algorithms can have valid and meaningful information about the status of the network. Second, controlling call blocking rates of different CoSs would become a possible task. Controlling call blocking rates is expected to be an important issue, especially when advanced pricing systems will be introduced [Paschalis 00].

When we consider different DBA algorithms, we can classify them as centralized and decentralized algorithms. Centralized algorithms provide a single point of failure and require huge amounts of information, which usually is broadcasted using flooding protocols. Also, their ability to provide a solution for a large-scale network with varying demands in an appropriate time is questionable. On the other hand, decentralized algorithms are implementable, but cannot provide an optimal solution because they don't have a global vision of the network.
The major contribution in this chapter is that we propose and evaluate a novel algorithm that performs a slow DBA to allocate bandwidth globally over all backbone links of a multi-class connection-oriented network. We call this proposed algorithm Virtual Demand Distribution (VDD) algorithm. The VDD algorithm is a distributed algorithm that can approach global optimality. The VDD collects more information than classical distributed algorithms, but the amount of collected information is still small, compared with the amount of information that any centralized algorithm needs in order to achieve global optimality. This extra information is obtained by utilizing the signaling messages used for establishing new connections in any multi-class network providing connection-oriented services. Also, to provide stability for routing algorithms we avoid using rapid DBA (on a per-session level) control. Instead, our slow DBA system deals with aggregates of traffic and responds to variations in demands on a relatively slow basis. The word "relatively" here means with respect to the speed of updating routing tables.

To evaluate the performance of proposed VDD algorithm we used Bones Designer, a sophisticated simulation tool, and built a multi-node network with a configuration shown in Figure 5–1. We built this simulation to emulate as real a network as possible, with 48 hours of non-stationary varying traffic generated at each node. A detailed description of this simulation is provided in section 5.5. In addition to VDD algorithm, we simulated performance of an equivalent Simple Distributed (SD) algorithm.
5.2 Single-Link Analysis

When different traffic categories or classes share a common link, three possible resource-sharing techniques can be considered. These are Complete Sharing (CS), Virtual Partitioning (VPT), and Complete Partitioning (CP). In CS, no control over the performance of different CoSs can be exercised. However, it is the simplest possible technique and works well under light loads. But, a well-designed network should not be lightly loaded all the time. In CP, a strict control over the performance of any CoS can be exercised, but wastage of resources is an issue to worry about. Under high load conditions, CP works very well if and only if demands on different CoSs can be estimated accurately [Ross 95]. VPT can be considered as a mixture of both CS and CP techniques. Resources are not wasted under VPT technique, but because of the possible sharing of resources by different CoSs and because performance depends on allocated capacities to different CoSs, no strict control over the blocking rates of different CoSs can be achieved, especially when the traffic is non-stationary.

In this chapter, we are interested in a highly-utilized backbone network supporting different CoSs which are sensitive to call blocking rates. Because of the differences in these CoSs due to differentiation in processing sessions of these CoSs at the packet level, or because some CoSs may represent virtual paths with different geographical significance, a differentiation between the blocking rates of these CoSs is desired. Thus, we will adopt a CP scheme, where capacities will be dynamically allocated in a non-stationary traffic environment.
5.2.1 Call-Blocking Probability For One CoS of Stationary Traffic with Variable Size Bandwidth Requests

We begin by reviewing previous work on predicting call blocking rates experienced by calls with stationary Poisson arrival process, stationary exponential holding time, and variable-size requests with known distribution. Such systems are known as stochastic Knapsack systems.

Let the requested bandwidth be a positive real value \( b \) with a density function \( f_b(b) \), \( B_p \): blocking probability, \( \lambda \): mean arrival rate, \( \mu \): mean service rate, and \( W_t \): total capacity. A closed form formula for \( B_p \) was found when \( f_b(b) \) is uniform and \( 0 \leq b \leq W_t \), [Robert 91]. The probability of blocking a request of size \( b \) is:

\[
B_p(b) = 1 - \frac{J_0\left(2\sqrt{\frac{\rho(W_t-b)}{W_t}}\right)}{J_0(2\sqrt{\rho})},
\]

(5–1)

where \( \rho = \frac{\lambda}{\mu} \), \( J_0(.) \) is the modified Bessel function of order 0. Thus, we can write:

\[
B_p = \frac{1}{W_t} \int_0^{W_t} B_p(b) db.
\]

(5–2)

What renders (5–1) impractical is the assumption that \( f_b(b) \) is uniform over \( 0 \leq b \leq W_t \). In any real network, usually the maximum possible value for \( b \) is a small fraction of \( W_t \). In addition, the distribution of requested bandwidth is far from uniform.

The other alternative is to consider a limited number of bandwidth sizes, such that a size-\( k \) session requests \( b_k \) units of bandwidth with a probability of \( P_b(b_k) \). In the context of our analysis, we have \( \lambda_k = \lambda P_b(b_k) \) and \( \mu_k = \mu, \forall k \). Let \( \bar{b} = [b_1 b_2 ... b_M] \),
\[ \bar{n} = [n_1, n_2, ..., n_M], \] where \( n_i \) is the number of size-\( i \) active calls in the link. An exact expression was found [Kaufman 81, Roberts 81], not closed form though, for the probability of blocking a size-\( k \) call (\( B_p(b_k) \)):

\[
B_p(b_k) = 1 - \frac{\sum_{n \in S_k} \prod_{j=1}^{M} \rho_j^{n_j} / n_j!}{\sum_{n \in S} \prod_{j=1}^{M} \rho_j^{n_j} / n_j!}, \tag{5–3}
\]

where \( S_k = \{ \bar{n} \in S : \bar{b} \bar{n} \leq W_t - b_k \} \) and \( S = \{ \bar{n} \in I^M : \bar{b} \bar{n} \leq W_t \} \). \( I \) is the set of non-negative integers. So, we can write:

\[
B_p = \sum_{k=1}^{M} P(b_k) B_p(b_k). \tag{5–4}
\]

The expression in (5–3) cannot be computed easily for reasonable values of \( M \) and \( W_t \). A recursive algorithm exists to compute the expression in (5–3), [Kaufman 81]. It was shown in [Kaufman 81] that this recursive algorithm works well for large values of \( M \) and \( W_t \). This is important, because in a real backbone we expect large values of \( W_t \), for sure.

One more technique that needs to be explored is the asymptotic approximation explained in [Reiman 91]. This approximation becomes more accurate as \( W_t \) increases, which is a desired behavior in our analysis. However, unfortunately, it seems no closed form formula can be derived from this approximation. The following equations constitute the approximation:

\[
B_p(b_k) = b_k \frac{\delta}{W_t}, \tag{5–5}
\]
\[ \delta = \frac{\left(\frac{\alpha}{\sigma}\right)}{\sigma \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} dx}, \quad (5-6) \]

\[ \sigma^2 = \frac{\sum_{k=1}^{M} b_k^2 \rho_k}{W_t}, \quad (5-7) \]

\[ \alpha = \sqrt{W_t} - \sqrt{\frac{\sum_{k=1}^{M} b_k \rho_k}{W_t}}, \quad (5-8) \]

Another problem in this approximation is that it gives accurate results as long as \(|\alpha| \leq 1.0\).

If this approximation is to be used in a practical situation, \(|\alpha|\) can reach high values, especially when the demand is higher than total serving rate. Our simulations showed that the approximation was inaccurate, at all, for large values of \(|\alpha|\).

As we can see, a closed-form formula that can provide us with the expected blocking rate in the case of variable size bandwidth requests is not available, neither in an exact nor in an approximate form. Thus, based on the above review of previous work, it is clear that the recursive algorithm presented in [Kaufman 81] is the best approach to build upon, in order to find an approach for bandwidth allocation between CoSs with the objective of forcing specific differentiation ratios between the call blocking rates of these different CoSs. In the following subsection, we propose a recursive algorithm to achieve this objective. For simplicity, and without loss of generality, we consider two CoSs.
5.2.2 Controlling Call-Blocking Probabilities For Two CoSs of Non-Stationary Traffic with Variable Size Bandwidth Requests

Our objective is to dynamically allocate bandwidth between two CoSs, such that the ratio of class-2 blocking rate ($B_2^p$) to class-1 blocking rate ($B_1^p$) is $\beta$, over a decision interval $t_d$. A class-1 call requests a bandwidth $b_m^1$ with probability $P_h^1(b_m^1)$, a class-2 call requests a bandwidth $b_n^2$ with probability $P_h^2(b_n^2)$. Class-1 mean arrival rate is $A_1$ calls/sec. and class-2 mean arrival rate is $A_2$ calls/sec. Class-1 mean holding time is $S_1^{-1}$ sec., class-2 mean holding time is $S_2^{-1}$. We also define $\bar{b}_1$: the average bandwidth size of class-1 calls, $\bar{b}_2$: the average bandwidth size of class-2 calls, $\lambda_m^1 = P_h^1(b_m^1)A_1$ calls/sec., $\lambda_n^2 = P_h^2(b_n^2)A_2$ calls/sec., $\mu_m^1 = S_1, \forall m$, and $\mu_n^2 = S_2, \forall n$. $m = 1, 2, \ldots, M_1$ and $n = 1, 2, \ldots, M_2$, where $M_1$ is the number of possible bandwidth sizes at class-1 and $M_2$ is the number of possible bandwidth sizes at class-2. In general, we assume non-stationary stochastic process where the mean arrival rate of class-$k$ is $A_k(t)$, $t$ is time. However, the holding time is assumed to be a stationary process with constant mean value $(1/S_k)$ sec. We assume that

$$\frac{\partial A_k(t)}{\partial t} \approx 0.0 \text{ for } t_o \leq t \leq t_o + t_d,$$

thus the system is assumed to be under stationary traffic throughout the decision interval $t_d$.

An important phenomenon arises here, at each decision instance, a situation where the allocated capacity is less than the utilization of class-$k$ may occur for a duration of time equals to $\zeta_k$ sec., where $\zeta_k$ is the time for class-$k$ utilization to drop below the new
allocated capacity. This is because we assume that the system should not drop any active session, in order to maintain QoS parameters promised to users. During this time interval \((\zeta_k)\) all class-\(k\) arriving calls will be blocked. Expected value of \(\zeta_k\) (\(\overline{\zeta}_k\)) can be exactly found if all class-\(k\) calls require the same amount of bandwidth. In this case, the problem becomes finding the expected time for the number of class-\(k\) active calls to drop, say from \(\omega_k\) to \(\omega'_k\), while no new calls are accepted. This is a death process, thus:

\[
\overline{\zeta}_k = \sum_{m=\omega_k}^{\omega_k} \frac{1}{mS_k} \text{ sec.},
\]

where the expression in (5–10) can be approximated as follows:

\[
\overline{\zeta}_k \approx \frac{\omega_k - \omega'_k}{S_k (\omega_k + \omega'_k)/2} \text{ sec.}
\]

The approximation in (5–11) is accurate as long as \(\frac{\omega_k - \omega'_k}{\omega_k} \ll 1\), which is the case in a realistic backbone network, especially as long as the assumption in (5–9) is true. Note, \(\omega'_k\) is determined by the capacity allocation decision for class-\(k\).

To find an exact expression for \(\overline{\zeta}_k\) in the case of variable size bandwidth requests, the system must keep the exact number of calls of each bandwidth size. In addition, only an exact expression, not a closed form solution, can be determined. Based on the available information to the system, \(\omega_k\) and \(\omega'_k\) can be approximated as the utilization and the allocated capacity, respectively, divided by the average size of the bandwidth requests. Let \(Q'_1\) be the allocated capacity (bps) to class-\(1\), \((W_i - Q'_1)\) is the allocated capacity (bps) to class-2. Let \(u_k\) be the utilization (bps) of class-\(k\) just before calculating a
new bandwidth allocation decision. We can extend the approximation in (5-11) to arrive at the following approximations for the length of the transient periods:

\[
\zeta_1 \approx \frac{(u_i/b^1 - Q_i/b^1)}{(S_i)(u_i/b^1 + Q_i/b^1)}/2 U(u_i - Q_i) \text{ sec.}, \tag{5–12}
\]

\[
\zeta_2 \approx \frac{(u_2/b^2 - [W_i - Q_i]/b^2)}{(S_2)(u_2/b^2 + [W_i - Q_i]/b^2)}/2 U(u_2 - [W_i - Q_i]) \text{ sec.} \tag{5–13}
\]

Note, \(U(.)\) is the unit step function. We can proceed now to develop a procedure for bandwidth allocation between the two CoSs. Again, based on the assumption in (5–9), and assuming that \(A_1\) and \(A_2\) can be accurately estimated every \(t_d\) sec., the objective of the following procedure can be stated as follows:

- Every \(t_d\) sec., find \(Q_1\) such that:

\[
\frac{\text{Number of class } 2 \text{ blocked calls throughout } t_d \text{ sec.}}{\text{Number of class } 1 \text{ blocked calls throughout } t_d \text{ sec.}} = \beta. \tag{5–14}
\]

The following algorithm is built upon the algorithm explained in [Kaufman 81]. Note, in the following algorithm, all bandwidth quantities must be integer numbers, the allocation decision is updated every \(t_d\) sec., \(Q_i(n)\) is the \(n^{th}\) allocation decision.

**Algorithm 1:**

At the \(n^{th}\) decision instance:

Estimate \(A_1\) and \(A_2\).

Measure \(u_1\) and \(u_2\).

Set \(Q_1' = Q_i(n-1)\)

While \((|\delta| > \varepsilon)\)

{
Set \( g_1(0) = 1, g_2(0) = 1, g_1(j) = 0 \) for \( j < 0 \), \( g_2(j) = 0 \) for \( j < 0 \).

For \( h = 1, 2, \ldots, Q_1' \)

\[
g_1(h) = \left( \frac{1}{h} \sum_{k=1}^{M_2} \left( b_k \right) \right) \left( \frac{P_k^2 (b_k^2) A_k}{S_1} \right) g_1(h - b_k^2). 
\] (5–15)

For \( h = 1, 2, \ldots, [W_i - Q_1'] \)

\[
g_2(h) = \left( \frac{1}{h} \sum_{k=1}^{M_2} \left( b_k \right) \right) \left( \frac{P_k^2 (b_k^2) A_k}{S_2} \right) g_2(h - b_k^2). 
\] (5–16)

Set

\[
G_1 = \sum_{h=0}^{Q_1'} g_1(h). 
\] (5–17)

Set

\[
G_2 = \sum_{h=0}^{[W_i - Q_1']} g_2(h). 
\] (5–18)

For \( h = 0, 1, \ldots, Q_1' \)

\{
\[ q_1(h) = \frac{g_1(h)}{G_1}. \] (5–19)

For \( h = 0, 1, \ldots, [W_t - Q'_1] \)

\[ q_2(h) = \frac{g_2(h)}{G_2}. \] (5–20)

\[
B^1_p = \sum_{k=1}^{M_1} P^1_b(b^i_k) \sum_{h=Q_k - b^i_k + 1}^{Q_k} q_1(h).
\] (5–21)

\[
B^2_p = \sum_{k=1}^{M_2} P^2_b(b^2_k) \sum_{h=Q_k - b^2_k + 1}^{Q'_k} q_2(h).
\] (5–22)

\[
\phi = \frac{(A_2 B^2_p(t_d - \tilde{\zeta}_2) + A_2 \tilde{\zeta}_2)/A_2 t_d}{A_1 B^1_p(t_d - \tilde{\zeta}_1) + A_1 \tilde{\zeta}_1/A_1 t_d}, \quad (\tilde{\zeta}_1 \text{ from (5–12) and } \tilde{\zeta}_2 \text{ from (5–13))})
\] (5–23)

If \( \phi > \beta \)

\[
Q'_1 = Q'_1 - 1.
\]
\[
\begin{align*}
&\text{If } \phi < \beta \\
&\quad \{ \\
&\quad \quad Q_i' = Q_i' + 1. \\
&\quad \}
\end{align*}
\]
\[
\delta = \phi - \beta.
\]
\[
Q_i(n) = Q_i'.
\]

For understanding equations (5–15 to 20), reader is advised to refer to [Kaufman 81, Ross 95]. Equations (5–21) and (5–22), define the average blocking rate for the two CoSs. In these two equations, the second summation finds the probability of blocking a specific size request and the first summation finds the average probability of blocking any-size request. In (5–23), the effect of the transient periods, approximated in (5–12) and (5–13), is included. Class-\(k\) blocking probability found in (5–21) and (5–22) is meaningful after the end of the transient period, if it exists. The Numerator in (5–23) divides the expected number of blocked class-2 calls throughout the decision period \(t_d\) by the number of all calls expected throughout the same period. The denominator of (5–23) does the same, but for class-1. Note also, in (5–23), the values of \(\zeta_1\) and \(\zeta_2\) belong to one of the following three cases, \((\zeta_1 = 0, \zeta_2 = 0), (\zeta_1 = 0, \zeta_2 > 0), \text{or } (\zeta_1 > 0, \zeta_2 = 0)\). It is clear that the inclusion of the transient response in the calculations of the DBA equations will work as a slowing tool. It will slow down the changes in the allocated bandwidth if a sudden and large change occurs in the demands. This is desirable, since by
this, the network will not make dramatic changes in the allocations unless new demands pattern are long-lived. $\varepsilon$ is a parameter that defines the desired accuracy in forcing the differentiation ratio $\beta$.

In this chapter, we try to emphasize the importance of accurate predictions of demands. We show, by utilizing the above algorithm, that estimating average arrival rates of two CoSs based on local information is not efficient, even if networks have simple structures. To overcome this problem, which is a serious one in CP schemes, we propose a novel algorithm that utilizes signaling messages in connection-oriented networks with reservations to approach global optimality in predicting demands at each router/switch.

5.3 Loss Network Analysis

In this section, we provide a loss network analysis for a multi-class network. For simplicity and without loss of generality we will restrict the analysis to two CoSs. However, this analysis can be extended to any number of CoSs.

Let set $N$ be the set of all nodes in a network that supports two connection-oriented CoSs.

Let $i, j \in N$, where $i \neq j$.

Let $l^i$ be the output port connecting node $i$ to $j$.

A flow is defined as a sequence of output ports visited by the traffic of one or more sessions carrying information between two nodes. $f_{v,n}^{ij}$ is the $n^{th}$ flow moving through $l^i$ and belongs to class $v$, $v = 1, 2$. Two traffics connecting the same two nodes, but each one
moving through different intermediate nodes are considered as two different flows. 

\( a(f_{v,n}^{ij}) \) is the average arrival rate of calls forming the flow \( f_{v,n}^{ij} \), the arrival process is assumed to be Poisson. For each \( I_v^i \) we define a tree \( T_v^i \), which is a set that contains all \( f_{v,n}^{ij}, n = 1, 2, ..., \sigma_v^i \), where \( \sigma_v^i \) is the total number of flows at \( I_v^i \) that belong to class \( v \).

Any flow that belongs to class \( v \) and passes through \( I_v^i \) is a member of \( T_v^i \). For every tree \( T_v^i \), we define roots \( R_v^i \) and branches \( B_v^i \). \( R_v^i \) is a set that contains \( r_{v,n}^{ij}, n = 1, 2, ..., \sigma_v^i \) where \( r_{v,n}^{ij} \) is the part of \( f_{v,n}^{ij} \) that starts at the source of \( f_{v,n}^{ij} \) and ends at node \( i \). \( B_v^i \) is a set that contains \( b_{v,n}^{ij}, n = 1, 2, ..., \sigma_v^i \) where \( b_{v,n}^{ij} \) is the part of \( f_{v,n}^{ij} \) that starts at a node \( j \) and ends at the destination of \( f_{v,n}^{ij} \). We also define \( T^1(T_v^i) \) which is a group of trees. A tree \( T_v^{xy} \in T^1(T_v^i) \) if and only if there exists at least one flow \( f_{v,n}^{ij} \in T_v^i \) such that \( l^{xy} \in f_{v,n}^{ij} \). We call the group \( T^1(T_v^i) \) the group of son-trees (first generation trees) of \( T_v^i \).

Similarly, \( T^2(T_v^i) \) is the group of grandson-trees (second generation trees) of \( T_v^i \). A tree \( T_v^{ab} \in T^2(T_v^i) \) if and only if there exists at least one flow \( f_{v,n}^{ij} \in T_v^i \) such that \( l^{xy} \in f_{v,n}^{ij} \) and there exists at least one flow \( f_{v,k}^{xy} \in T_v^{xy} \) such that \( l^{ab} \in f_{v,k}^{xy} \). In a similar way, we can define \( T^z(T_v^i) \), where \( z \) can take any value between 0 and \( \infty \). Note that \( T^0(T_v^i) = \{ T_v^i \} \).

*Claim:* There exists a bandwidth allocation algorithm that can perform as optimal as a centralized algorithm in a multi-class connection-oriented network while working in a distributed way and requiring an amount of information much less than the global demand matrix.
Proof:

In the following we mean by an output related a tree that there is a flow member of this tree such that this output port is a member of this flow, and by a decision a bandwidth allocation decision. Consider any output port $l^{ij}$ in a connection-oriented network that supports two CoSs.

1) Let $l^s$ be any output port such that $l^s \neq l^{ij}$ where $s, t, i, j \in \mathbb{N}, s \neq t, i \neq j$.

2) Assume there is at least one $f^{ij}_{v,k} \in T^i_j$ such that $l^s \in f^{ij}_{v,k}$. Then, the effect that $T^r (T^v_j)$ may have on the decision at $l^{ij}$ is implied in the state/information at $l^s$, where $r = 0, 1, 2, \ldots, \infty$.

3) If there is no $f^{ij}_{v,k} \in T^i_j$ such that $l^s \in f^{ij}_{v,k}$, then state/information at $l^s$ either has no effect on the decision at $l^{ij}$ or has an indirect effect if and only if there exists a $T^r (T^v_j)$ such that there exists a tree $\in T^r (T^v_j)$ such that $l^s$ is related to this tree, where $r = 1, 2, \ldots, \infty$. Note $r$ starts from 1 and not from 0.

4) For the state/allocation at $l^s$ to have no effect on the decision at $l^{ij}$ then $\forall v, \nu = 1, 2; \forall r, r = 0, 1, 2, \ldots, \infty$, there is no $T^r (T^v_j)$ such that $l^s$ is related to any tree $\in T^r (T^v_j)$. This scenario is shown in Figure 5–2.

5) So, there should exist an algorithm that only needs to know $a(f^{ij}_{v,n}) \forall f^{ij}_{v,n} \in T^v_j, \forall n, n = 1, 2, \ldots, \sigma^v_j; \forall v, \nu = 1, 2$ and the state/information at each output port (say $l^{xv}$) related to any flow $\in (T^v_1 \text{ or } T^v_2)$, in order to issue an optimal decision at $l^{ij}$. Note that, the
state/information at each output port \( l^{xy} \) is in itself a function of all \( f_{v^p}^{xy} \in T^v_{v^p} \) and state/information at each port related to any flow \( e \in (T^1_{xy} \text{ or } T^2_{xy}) \).

The state/information at an output port is a general term that, in the worst case, can be an information package containing the demands of all flows composing the tree at that output port. Note that, when the state/information at an output port is designed for the worst case, we are actually implementing a centralized algorithm with one improvement, that demands on irrelevant flows are not considered by the algorithm. In the best case, state/information can be a brief information about some parameters of interest like utilization and blocking rates of the two CoSs. Worst case and best case here are in terms of complexity.

So far, we claim the existence of an algorithm that runs in a distributed way but still can approach global optimality by collecting an amount of information much smaller than the global demand matrix. Our simulations show that VDD is a very strong candidate to be such an algorithm.

In a practical environment, the demand of any flow is not known a priori. Demands can be measured, based on measured demands over an averaging window of \( \Delta t \) starting at time \( t_0 \). Demands can be predicted throughout \((t_0 +\Delta t, t_0 + 2\Delta t)\). The accuracy of this prediction depends on the nature/correlation of the variation in demands and on \( \Delta t \). As \( \Delta t \) increases, above a certain value, the correlation between successive intervals decreases. On the other hand, as \( \Delta t \) decreases below a certain value, there will not be enough time to estimate the average arrival rate of calls and hence estimate the demand. In [Odlyzko 98], it was shown that the variations in aggregated demands are usually slow
enough such that a decision interval of one hour would even be faster than necessary. Measuring demand needs time and so does broadcasting information. In addition to timing considerations, there are complicated forms of dependency among decisions computed at different output ports throughout the whole multi-class network. With all these considerations, and based on previous discussions, we developed the VDD algorithm. In the following section, we explain and state the VDD algorithm.

5.4 The Virtual Demand Distribution (VDD) Algorithm

Let $t_d$ be the decision interval.

Let $t_u$ be the update interval.

$T_d = mt_u$, where $m \geq$ (maximum number of hops), this is a necessary condition.

The following requirements are necessary for any network for it to be able to support the VDD algorithm:

1- Signaling packets (call-setup packets) must be used to establish any new connection.

2- Packets belonging to a specific session have to follow the same path that was assigned initially by the network.

3- Acknowledgments have to follow the same path -in the reverse direction- that was followed by the call-setup packets. Acknowledgments have to keep all the information collected by the call-setup packets.

4- Call-setup packets belonging to different flows and moving through the same output port must have different labels, or can be identified and assigned different labels.
5- Call-setup packet must continue its journey to the destination edge device, even if the requested bandwidth is rejected by an intermediate node.

The above requirements can be fulfilled by ATM networks [Stallings 97]. Also the framework developed in chapter 4 fulfills these requirements. MPLS architecture can easily support the above requirements too. Each call-setup packet records the following information at each output port it moves through:

1- Whether or not this output port has accepted this new call under the specified CoS.

2- The percentage increase or decrease of the expected blocking rate -based on the current updated decision- with respect to the current measured blocking rate under the specified CoS. We call this the Virtual Percentage Change in Blocking Rate (VPCBR).

This information must be written in the call-setup packet in a way that reflects the order in which the output ports have been visited by. Also, any output port, say \( l^{ij} \), must know its order when it writes information in any call-setup packet and bind this order \( O_{ij,v} \) with the label \( L_{ij,v}^{ij} \) of that call-setup packet. \( L_{ij,v}^{ij} \) is the label assigned to flow \( f_{ij,v}^{ij} \) at output port \( l^{ij} \). This binding needs to be done one time when a new flow starts to move through an output port.

At each output port, the following data is retrieved from each Acknowledgment (ACK) that moves through it. To simplify the notation, we will drop the superscript \( ij \). Also, we define \( l_{v,n}^{ij}(k) \) to be the \( k^{th} \) output port visited by the \( n^{th} \) flow \( f_{v,n} \) at the current
output port at which the algorithm is running, \( k = 1, 2, \ldots, Y_{L,v} \), where \( Y_{L,v} \) is the number of output ports visited by the flow labeled by \( L_{v,n} \).

Assume that the following data is going to be collected and worked with independently, at each output port. A linked-list data structure is to be created, each label \( L_{v,n} \) is assigned one list. Each element of the list assigned to \( L_{v,n} \) contains four counters and two real numbers. \( CA_{L,v,n} (k) \) to count number of accepted \( v \)th class calls, \( CR_{L,v,n} (k) \) to count number of rejected \( v \)th class calls, \( d_{L,v,n} (k) \) to record VPCBR value of the \( v \)th CoS at the \( k \)th output port of the flow labeled \( L_{v,n} \), \( k = 1, 2, \ldots, Y_{L,v} \); \( v = 1, 2 \).

Briefly, the basic idea in the VDD algorithm is to let every output port compute and declares a virtual decision every \( t_u \) sec. The decision is virtual because it will not be implemented physically. It can be interpreted as if the output port is "saying": “I'm planning to implement this decision, so, other output ports try to optimize your decisions based on my decision”. However, we do not distribute the virtual decision itself. Rather, we distribute the expected effect of this virtual decision on the current blocking rate at this specific output port. Note that, by this idea, we give the network a chance to converge to the optimal allocation without physically changing the allocations more frequently, since rapid changes in the allocations will not be helpful in stabilizing QoS routing in a multi-class network. In the following, we will first present a pseudo-code of the algorithm and then we explain how it works in more details.

\[
\text{While ( network is running )}
\]

\{ initialize all counters; \}
\[ x = 0; \text{set clock} = 0; \]

wait \( t_r \text{ sec.} \);

\textbf{for} ( \( x = 1 \) to \( m \) )

\{
\( t = 0; \)

\textbf{while} ( \( t \leq t_u \) )

\{

if ( ACK arrived )

\{ read label \( L_{v,n} \); \}

\( \forall k = 0, 1, 2, \ldots, Y_{l_{v,n}} \) do:

\{ read \( d_{L_{v,n}}(k) \); \}

\( CA_{l_{v,n}}(k) = CA_{l_{v,n}}(k) + 1, \text{if the call was accepted at } l_{v,n}(k); \}

\( CR_{l_{v,n}}(k) = CR_{l_{v,n}}(k) + 1, \text{if the call was rejected at } l_{v,n}(k); \} \}

\textbf{Virtual\_Decision}(1); \}

if ( this link is a DownLink )

\{ compute & declare the semi-final-1 VPCBR // which equals the last computed VPCBR \}

\}

\textbf{else} ( wait until all semi-final-1 VPCBRs are received from all } \( l_{v,n}(O_{l_{v,n}} + 1), n = 1, 2, \ldots, \sigma_v; v = 1, 2 \)

\{ \textbf{Virtual\_Decision}(2) \ \textbackslash this is a semi-final-1 VPCBR \}
wait $t_{pr}$ seconds;

if ( this link is a DownLink )

{ Virtual_Decision( 1 ) \ this is a semi-final-2 VPCBR }

else ( wait until all semi-final-2 VPCBRs are received from all $l_{v,n}(O_{L_{v,n}} + 1), n = 1, 2, \ldots, \sigma_{v}; v = 1, 2$ )

{ Virtual_Decision( 3 ) \ this is a semi-final-2 VPCBR }

if ( this link is an UpLink )

{ Virtual_Decision( 4 ) \ this is a final VPCBR }

else(wait until all final VPCBRs are received from all $l_{v,n}(O_{L_{v,n}} - 1), n = 1, 2, \ldots, \sigma_{v}; v = 1, 2$)

{ Virtual_Decision( 3 ) \ this is a final VPCBR }

when ( clock = $t_{d}$ ){physically implement $Q_1$ }

}

Virtual_Decision( e )

{

switch ( e ){

case 1: { A = 1, B = O_{L_{v,n}} - 1 } 

case 2: { A = O_{L_{v,n}} + 1, B = Y_{v,n} } 

case 3: { A = 1, B = Y_{v,n}; such that $k \neq O_{L_{v,n}}$ } 

case 4: \{ A = 2, B = Y_{v,n} \} \}

\[ A_v = \sum_{n=1}^{\sigma_v} \left( \frac{CA_{L_{v,n}}(l) + CR_{L_{v,n}}(l)}{xt_{l_v}} \prod_{k=A}^{B} (1 - \psi_{L_{v,n}}(k)) \right) ; v = 1, 2. \quad (5-24) \]

Where:

\[ \psi_{L_{v,n}}(k) = \begin{cases} \frac{CR_{L_{v,n}}(k)}{CA_{L_{v,n}}(k) + CR_{L_{v,n}}(k)} (1 + d_{L_{v,n}}(k)) \text{ if } CR_{L_{v,n}}(k) > 0 \\ d_{L_{v,n}}(k) \text{ if } CR_{L_{v,n}}(k) = 0 \end{cases} \quad (5-25) \]

Feed \( A_1 \) and \( A_2 \), computed in (5-24), into Algorithm 1 and find \( Q_1 \); \new virtual bandwidth allocation decision

\[ d^v = \frac{EBR_v - CBR_v}{CBR_v} \quad \text{new VPCBR at this output port} \]

Declare the VPCBR \((d^v)\);

Where:

\( EBR_1 \) = Expected Blocking Rate under 1st CoS due to the current virtual decision. Feed \( A_1 \) and \( Q_1 \) into the recursive algorithm in [Kauffman 81] to get \( EBR_1 \).
\( EBR_2 = \) Expected Blocking Rate under 2\(^{nd}\) CoS due to the current virtual decision. Feed \( A_2 \) and \([W_t - Q_t]\) into the recursive algorithm in [Kaufman 81] to get \( EBR_2 \).

\( CBR_1 = \) Current Blocking Rate measured under 1\(^{st}\) CoS.

\( CBR_2 = \) Current Blocking Rate measured under 2\(^{nd}\) CoS.

The algorithm can be explained as follows:

After every physical implementation of bandwidth allocation, the algorithm waits for \( t_r \) sec. This is to wait for the transient response to approximately die. \( t_r \) can be designed to be an adaptive parameter, but since we are following the slow variations of demands, we can simply set \( t_r \) based on the worst case scenario. As we explained earlier, during a transient period, all incoming calls will be blocked until the utilization drops below the new allocated bandwidth. These rejected calls if accounted for, will feed the algorithm with wrong information, which is why the algorithm must wait until the transient period ends. The time needed for the transient period to end can be approximately estimated as in (5–12) and (5–13).

Throughout every update interval \( t_u \), information is collected at each output port from the returning ACK packets. For example, at output port \( I^i \), the algorithm collects the rate of accepting and rejecting calls at each output port related to \( T_v^i \), \( v = 1, 2 \). At the same time, the algorithm collects all the VPCBRs declared at all output ports related to \( T_v^i \), \( v = 1, 2 \). During the first \( m \) updates, the algorithm considers only information collected from all output ports related to \( R_v^i \), \( v = 1, 2 \). Here, the algorithm does not look
at output ports related to $B^i_j$, $v = 1, 2$; it is clear that, if the algorithm does look at both the roots and branches of the tree $T^i_j$ at the same time, oscillation in the decisions can occur. The idea of looking only at the roots of the tree during the first $m$ updates can be interpreted as providing a chance for the new demands to dig through the network. This is why we required the satisfaction of the condition $m \geq (\text{maximum number of hops})$, so information about new demands originating at the UpLinks will have a chance to reach the most-distanced possible DownLinks.

At $\text{clock} = (x)(t_u)$, where $x \leq m$, algorithm computes the expected demand on both CoSs by simply multiplying the total arrival rate (both accepted and rejected calls) and the end-to-end blocking probability for $r^i_{v,n}$, $n = 1, 2, \ldots$, $\sigma^i_v$; $v = 1, 2$, then adding the results (5–24). Note that, the total arrival rate for a specific flow is the same at all output ports visited by this flow.

At each output port related to any $r^i_{v,n}$, the expected blocking rate is computed based on the measured blocking rate up to $\text{clock} = (x)(t_u)$ and the declared VPCBR at that output port at $\text{clock} = (x-1)t_u$ (5–25). Note that, in (5–25) the expected blocking rate at a specific output port is considered to be the VPCBR itself if the measured blocking rate were zero. The expected demand is then used to compute a new virtual bandwidth allocation $Q_i$, based on the desired objective function. Then, this new virtual allocation will be used to compute and declare the new VPCBR at $\text{clock} = (x)(t_u)$ at $l^i$. The word "declare" means that if any signaling packet moves through $l^i$ after $\text{clock} = x.t_u$, it will carry this new VPCBR to other output ports.
For \( \text{clock} > (m)(t_n) \), the finalization stage starts. A serious problem must be faced in this stage, that is, which output ports will implement their final decisions first and in what order? By looking at the simple case shown in Figure 5–2, one can recognize the dependency that all output ports' decisions have on each other. After careful study and intensive simulations, we reach the following protocols that need to be followed in the final stage.

Once the \( m \)th decision is declared, all DownLinks declare their semi-final-1 decisions, which are basically the \( m \)th decisions themselves at these DownLinks. The semi-final-1 decision is computed in a similar way to the decisions that are made during the first \( m \) updates, with a major difference that, at \( i^v_j \), output ports related to \( T_v^i, v = 1, 2 \), are considered in the computations. So, in the semi-final-1 decision, the algorithm looks at output ports after and before \( i^v_j \). In general, any output port \( i^v_j \) needs to wait until the first output port of every \( b_v^i, n = 1, 2, ..., \sigma_v^i ; v = 1, 2 \), declares its semi-final-1 VPCBR. Note that, when an output port computes the semi-final-1 VPCBR, the VPCBRs collected from \( B_v^i, v = 1, 2 \), are the semi-final-1 VPCBRs. The declaration of semi-final-1 decisions may be viewed as if there is a decision wave propagating from downstream nodes to upstream nodes, we call this wave a semi-final-1 decision wave.

Once a DownLink (say \( i^v_j \)) declares its semi-final-1 VPCBR, it waits for \( T_{pr} \) seconds (propagation time), which is the required/expected time for the semi-final-1 decision wave to reach all the UpLinks. This time depends on the maximum number of hops and on the signaling rate throughout the backbone network which can be easily estimated. So, after \( t_{pr} \) seconds, all DownLinks compute and declare the semi-final-2
decisions which are computed similarly to the semi-final-1 decisions. In general, any output port $l^v$ needs to wait until the first output port of every $b_{\nu,n}^v$, $n = 1, 2, ..., \sigma_v^v$; $v = 1, 2$, declares its semi-final-2 VPCBR. Note that, when $l^v$ computes the semi-final-2 VPCBR, the VPCBRs collected from $B_v^v$, $v = 1, 2$, are the semi-final-2 VPCBRs, while the VPCBRs collected from $R_v^v$, $v = 1, 2$, are the semi-final-1 VPCBRs. The declaration of semi-final-2 decisions may be viewed as if there is a decision wave propagating from downstream nodes to upstream nodes, we call this the semi-final-2 decision wave.

Once an UpLink (say $l^v$) receives all semi-final-2 VPCBRs from the first output port of every $b_{\nu,n}^v$, $n = 1, 2, ..., \sigma_v^v$; $v = 1, 2$, it declares the final decision. In general, any output port $l^v$ needs to wait until the last output port of every $r_{\nu,n}^v$, $n = 1, 2, ..., \sigma_v^v$; $v = 1, 2$, declares its final VPCBR. The final VPCBRs are computed similarly to the semi-final-2 VPCBRs.

Note that, when $l^v$ computes the final VPCBR, the VPCBRs collected from $B_v^v$, $v = 1, 2$, are the semi-final-2 VPCBRs, while the VPCBRs collected from $R_v^v$, $v = 1, 2$, are the final VPCBRs. The declaration of final decisions may be viewed as if there is a decision wave propagating from upstream nodes to downstream nodes, we call this the final decision wave. At a synchronized instance, $clock = t_d$, all output ports physically implement the final decisions that were computed through the final decision wave. $t_d - m_{t_d} - t_r$ should be greater than $3t_{pr}$ to ensure that all output ports compute the final VPCBR.
So, the finalization stage is composed of three decision waves, after pumping virtual demands through the network for \( m \) update intervals, which is enough to ensure that the effects of new demands can reach the most-distanced node in the network. The first decision wave (semi-final-1) starts to provide the output ports with information about how downstream links responded to the offered virtual demands. The second and third decision waves, (semi-final-2) and (final), work as correction waves for some possible wrong information that might be generated by the first decision wave.

One may argue that information is being distributed by the signaling packets and not by independent flooding packets, and that may result in missing some information about specific flows if they are not active and do not have signaling packets during a specific \( t_u \) interval. Actually, this is what the VDD algorithm can perfectly overcome, since the expected demand is computed based on measurements of the average arrival rate of different flows and based on updated information about latest virtual decisions of all related output ports. One can easily realize that, if a specific flow is not active or completely dead through a specific \( t_u \) interval, then this means the contribution of this flow to the total expected demand is insignificant. Hence, even if the algorithm didn't receive updated information about this path, the effect will be absolutely negligible.

5.5 Performance Analysis

In this section, we present the results of testing the performance of the VDD algorithm. We compare the performance of the algorithm to a Simple Distributed (SD) algorithm that uses locally estimated demands. So, both VDD and SDD algorithm utilize
Algorithm 1. However, SD estimates average arrival rate of both CoSs using local measurements, while VDD utilizes signaling messages in a coordinated mechanism to gain wider vision and higher accuracy in estimating average arrival rate of both CoSs every decision interval.

5.5.1 Simulation Settings

Figure 5–1 shows configuration of the simulated network. The sophisticated simulation tool (Bones Designer) was used to build a two-class multi-node network. This network was built to emulate a real backbone network. Each node represents a router, each router is connected to a LAN -not shown in the figure- via a GateWay. Each LAN generates a stream of calls (sessions) based on a Poisson random process. Traffic generated at a specific LAN is distributed on all possible destinations forming a group of flows based on fixed routing (shortest path). The average arrival rate of sessions under each flow varies slowly, following a sinusoidal shape with a period of 48 hours. The phase of this sinusoid is different from one flow to another. Another slower sinusoidal function with a period of 196 hours was used to classify sessions into 1st class and 2nd class under each flow at each LAN, such that the maximum of the sinusoid corresponds to 10% of the sessions being 1st class and 90% being 2nd class. At the minimum of sinusoid, the opposite occurs. Thus, a non-stationary traffic/demand is generated for a period of 48 hours.

Average arrival rate of different flows was set to congest the network, in order to test the performance of different algorithms. The requested bandwidth was generated
following a Binomial distribution with $\bar{b}^1 = \bar{b}^2 = 1.0$ Mbps. The requested bandwidth was quantized with a step of 100 Kbps. Minimum requested bandwidth was limited to 100 Kbps and the maximum to 10 Mbps. Thus, $M_1 = M_2 = 100$. Holding time was modeled as an exponential distribution with $S_1^{-1} = S_2^{-1} = 10$ minutes. The links’ bandwidth, $W_i = 3.0$ Gbps.

For the VDD algorithm, we used the following settings $t_d = 20$ min, $t_a = 2$ min, $t_r = 4$ min, $t_{pr} = 1$ min, and $m = 6$. For the SD algorithm, the decision interval was set to be equal to $t_d$, in order to make the comparison valid. The SD algorithm counts the number of 1st and 2nd CoSs calls arriving throughout the $t_d$ interval, in order to estimate the average arrival rate of both CoSs.

As we mentioned earlier, the goal of VDD algorithm is to provide accurate predictions of demands on different CoSs. It is up to the network designer to choose the objective function to be optimized. For this investigation, we chose a simple objective function to demonstrate how well the algorithm can perform. So the VDD algorithm is not the DBA itself, but rather, it is a tool to provide accurate prediction of demands that can be used by DBA algorithms. In this simulation, we simply chose $\beta = 2$.

### 5.5.2 Simulation Results

Our objective is to force the ratio of the 2nd class blocking rate to the 1st class blocking rate to be two at any output port ($\beta = 2$); we selected $t^{45}$ to be monitored, since it is in the middle of the network. Figure 5–3 shows the blocking rate of both the 1st and
2nd CoS versus time at \( l^{45} \) for the VDD algorithm, Figure 5–4 shows the performance of SD algorithm. It is clear that the VDD algorithm outperforms the SD algorithm, proving the wider vision that VDD enjoys. Figure 5–3 shows the VDD approximately maintaining \( \beta = 2 \) almost all the time, while SD deviated from the value \( \beta = 2 \) most of the time. Figure 5–5 shows utilization and bandwidth allocation decisions for the VDD algorithm at \( l^{45} \), it can be noticed how the algorithm shapes the utilization of both CoSs gradually to enforce the ratio \( \beta = 2 \).

To clarify the importance of global vision, Figure 5–6 shows utilizations and bandwidth allocation decisions for the VDD algorithm at \( l^{57} \). It is interesting to compare this figure to Figure 5–7, which shows the equivalent performance of the SD algorithm. Note, how the bandwidth allocation decisions in the SD algorithm mistakenly congested the 2nd CoS during a specific time interval while there are free resources available, as can be noticed by comparing the 1st class utilization to the allocated bandwidth for the 1st CoS. This can happen simply because the SD algorithm cannot estimate how many of the arriving calls are being accepted by the DownLinks. However, the VDD algorithm is able to evenly distribute the extra resources.

In another experiment, we broke link \( l^{56} \) at a specific instance and intentionally increased the percentage of 1st class calls generated at node 1 and moved through \( l^{56} \) to make the effect of breaking \( l^{56} \) more recognizable. Figure 5–8 shows the performance of the VDD algorithm while Figure 5–9 shows the performance of the SD algorithm. Again, we can see how the SD algorithm failed to notice the effect of breaking \( l^{56} \), while the
VDD algorithm perfectly estimated the effect of this event and allocated resources, accordingly.

5.5.3 Tradeoffs Regarding The VDD Algorithm

The most important feature of VDD algorithm is its ability to have a wider vision when implementing a bandwidth allocation decision. VDD algorithm does not require extra signaling packets, which is a very desirable criterion. It can be considered as a robust algorithm, since it is not affected by failures in some parts of the network, and it will always try to provide the best prediction of demands based on the most recent available information. Dynamic resource allocation based on prediction of demands has been proposed in many previous works, the work in [Gupta 98, Saito 97] is an example on such techniques.

As the Internet increasingly becomes the largest and the most complicated machine in the world, and as the world converges into one community, designing and policing the network based on local geographical information will no longer provide adequate results. The VDD algorithm collects information about demands and congestion from relevant flows that can start and end at any location in the world, thus avoiding this problem.

On the other hand, VDD algorithm adds more overhead on the signaling packets and requires software upgrades at all backbone routers where this algorithm would be implemented. But this overhead is acceptable or even lower than what was proposed in some previous works. Like the frameworks proposed in [Fulp 97, Jordan 95], where
negotiation protocols with much more overhead than what we proposed in this work were proposed. When the network is underutilized, the VDD algorithm could be considered as an over-engineered approach. The complexity of optimizing an objective function may become high if the network supports many CoSs, however the complexity involved in the mechanism of VDD won’t increase significantly as the number of CoSs or VPs increases.

5.6 Conclusion

The importance of accurate predictions of demands in a multi-class connection-oriented network with bandwidth segregation, under non-stationary traffic, led us to develop a novel bandwidth allocation algorithm, the VDD. Simulations showed that the VDD outperforms an equivalent Simple Distributed (SD) algorithm. Results under non-stationary two-class traffic with Poison arrivals in a multi-node network showed that VDD was able to optimally allocate resources to the two CoSs based on the selected objective function. Although we have not been able to mathematically prove that the VDD algorithm can produce globally optimized decisions, we showed by network analysis the existence of an algorithm that can operate in a distributed manner, but can issue globally optimized decisions, by collecting extra information about the network that still would be much less than the information needed by a centralized algorithm. The potential of the VDD algorithm to be used as a benchmark in large network analysis increases its importance.

DBA between CoSs, which can represent different categories of different QoS levels or different topological paths at a common link, is an essential complement of load
balancing techniques between alternative paths. From a traffic engineering point of view, load balancing and QoS-routing can be viewed as a fast process, on the sessions arrival level, while DBA process can be viewed as a slow process that follows the variations in aggregated demands on different CoSs in a shared link.

Figure 5–1: The configuration of the network used in simulating the VDD and other algorithms, (simulation tool: Bones Designer)
Figure 5–2: A schematic view of a network that shows two trees only.
Figure 5–3: Blocking rates under the VDD at link 4-5.
Figure 5–4: Blocking rates under the SD at link 4-5.
Figure 5–5: Utilizations and allocations under the VDD algorithm at link 4-5.
Figure 5–6: Utilizations and allocations under the VDD algorithm at link 5-7.
Figure 5–7: Utilizations and allocations under the SD algorithm at link 5-7.
Figure 5–8: Performance of the VDD when link 5-6 was broken.
Figure 5–9: Performance of the SD algorithm when link 5-6 was broken.
6.1 Review Of MAC Layer in a TDMA Local Area Network (LAN)

Transmission of data from a base station to mobile terminals is organized in frames where synchronization is maintained between a base station and the corresponding mobile terminals. Transmission of data from mobile terminals to a base station may be accomplished through a similar framing structure but using different frequency band in what is called Frequency Division Duplex (FDD). Or by using the same frequency band and dividing the frame-time into uplink-time and downlink-time in what is called Time Division Duplex (TDD). In this section, we will generally describe the architecture without referring to specific industry standards.

In FDD, uplink and downlink frames are on two different frequency channels and thus may be simultaneously transmitted. In TDD, both uplink and downlink frames are on the same frequency channel, and in order to optimize the performance of the TDD system, the boundary (Figure 6–1) between the two frames is dynamically changed based on the demands on both directions [Li 00]. Thus, in TDD, the length of neither frame is fixed.
In both FDD and TDD, frames are partitioned into several time slots, one slot or more at the beginning of the frame is further partitioned into mini-slots [Kubbar 00], as shown in Figure 6–1. In the uplink frame, these mini-slots are used on a contention basis to reserve time slots for data transmission. The mini-slots in the downlink frame are used to respond and organize the requests made by mobile terminals. Mobile terminals use Aloha-like protocols to access the mini-slots by sending request packets. The classical schema is Packet Reservation Multiple Access (PRMA) protocol which was designed for packet voice terminals in cellular systems using FDD [Goodman 89].

PRMA uses packet contention techniques for obtaining a reservation for speech terminals. Each slot in a frame is recognized as ‘reserved’ or ‘available’ according to the acknowledgement message received from the base station at the end of the slot. At each speech terminal, speech activity detection is implemented. So, speech packets are generated and buffered only during talkspurts. Once the first packet of a talkspurt is generated, the terminal attempts to transmit a speech packet with a predetermined probability. If it successfully transmits a speech packet, it reserves that slot in future frames until the end of the talkspurt. Otherwise, it contends again on the next available slot with the same predetermined probability. The contention results and the status of slots are broadcasted via acknowledgment packets on the downlink channel. The permission probability $p_p$ can be predetermined and fixed for all users and all frames, or adapted to system states, which are usually the number of contending terminals and the number of reserved time slots.
Voice terminals can be modeled by a two-state Markov chain as shown in Figure 6–2. Usually this model is used in the literature studying TDMA-LANs, as in [Wu 95]. Following the model in Figure 6–2, the time a talker spends in a talkspurt or silence state has an exponential distribution with a mean of $1/\lambda$ for the talkspurt state and a mean of $1/\alpha$ for the silence state.
PRMA protocol is suitable for systems with voice and data traffic and the protocol is optimized for voice traffic. PRMA protocols do not support mechanisms for accommodating users with changing transmission rates. However, one of the most important features of next generation global networks is the integration of multimedia traffic such as voice, video, computer data, and other traffic types. The current vision is to have this network that supports multimedia traffic extend to the wireless networks, specifically to mobile wireless networks in order to provide mobile access to the global network. Supporting multimedia traffic by mobile wireless networks means these networks must support users with changing transmission rate requirements, such as VBR, real time and non-real time, and CBR. In order to maintain efficiency with wireless networks, specific modifications need to be introduced to the MAC layer. For that reason,

Figure 6–2: A Markov model for voice activities at a voice terminal.
Dynamic Packet Reservation Multiple Access (DPRMA) scheme was introduced [Dyson 97].

In DPRMA, users initially contend for access using Reservation-Aloha-like methods like PRMA. Once a user successfully transmits in a slot it will be allocated the same slot in every frame until the user changes or releases the reservation. As in PRMA, when a user sends an empty slot, the base station understands this as a termination of the reservation and releases this slot. In addition to that, DPRMA defines Reservation Request (RR) bits within the header of each uplink time slot. Users calculate their required slots allocation per frame and set the RR bits, accordingly. At any time, users can increase or decrease their transmission requirements. The base station follows the states of all transmission requests and changes made by mobile terminals and run an algorithm to allocate time slots accordingly. Allocation information is sent back to mobile users via the feedback downlink channel. More details can be found in [Dyson 97].

More proposed modifications on DPRMA and PRMA have been proposed in [Koutsakis 99, Koutsakis 00, Kubbar 00, Qi 99, Wu 95]. However, all these proposals share the same basic idea of DPRMA with different detailed mechanisms, most of the reported work focused on giving priorities for voice packets over data packets.

PRMA and DPRMA are designed to work under FDD, they cannot work in their original structure under TDD. The difference between FDD and TDD is that in FDD the downlink frame is on different frequency from the frequency used by the uplink frame, and thus feedback information about contention results can be transmitted back to mobile
terminals immediately. So, contention and the result of contention can be made on a per-slot basis. In TDD, both uplink and downlink frames use the same frequency, and thus the downlink frame comes in sequence after the uplink frame. Feedback information about contention results can only be sent after the end of the uplink frame, thus contention results can be provided on a per-frame basis. Even though, in [Ren 97] a modified version of PRMA, called F-PRMA was proposed to extend PRMA protocol to TDD systems, the performance was shown to be approximately the same as in the case of PRMA under FDD.

After this brief review, we present in the following section a new type of multi-class MAC protocols. We show how two CoSs of the same application can be supported with a simple mechanism.

6.2 Two Class Packet Reservation Multiple Access, Motivations and Design

6.2.1 Introduction

The Term “Quality-of-Service” (QoS) has become one of the most popular terms that can be found almost in all publications related to telecommunications industry. However, one must distinguish between QoS metrics and QoS scores. Philippopoulos et al concluded in their paper [Philippopoulos 99] that “the primary and most important measure of service quality should be customer satisfaction.” This is true, because it is the improvement in some measurable aspects of the subjective user-perceived quality that
can justify any new design or higher pricing. QoS scores can be used to represent different levels of customer satisfaction.

The most important QoS metrics are average and variance of delay, delay jitter, and packet loss, in addition to throughput and availability [ITU-T-E.800 94]. QoS score is a function of some or all QoS metrics. Goodman et al [Goodman 89] found experimentally the variations in the QoS score versus packet loss rate ($P_{loss}$) for voice-activated speech applications using PRMA technique for wireless LANs. This relation between the QoS score, which is determined in this case by volunteers rating the quality of speech, and $P_{loss}$ defines a QoS sensitivity function, as we demonstrate later.

To clarify the fundamental difference between a QoS score and a QoS metric, one can consider a packetized voice application, specifically the PRMA technique with settings as in [Goodman 89]. In such a system, the OoS score is judged by the satisfaction of human users. In our example, this satisfaction totally depends on $P_{loss}$. It was found [Goodman 89] that the human hearing system does not recognize a noticeable degradation in the perceived QoS if $P_{loss}$ is less than $10^{-2}$. Thus, if for example an engineer proposes a new design for a packetized voice system that improves the performance by lowering $P_{loss}$, say from $10^{-3}$ to $10^{-6}$, while maintaining the same system capacity, then we can tell that such a new design is not attractive. Because, improving some QoS metrics does not necessarily lead to an improvement in the QoS score. The QoS score for a packetized speech session is approximately constant in the interval [0, $10^{-3}]$ of $P_{loss}$. 
In this chapter, with the above understanding, we develop a revenue comparison framework. In addition, we motivate and design a two-class PRMA architecture. Multi-class networks are becoming good candidates to efficiently manage a wide range of QoS metrics required by different applications. Multi-class networks also have the potential to generate more revenue by implementing differential pricing schemes [Honig 94]. We use the revenue comparison framework to prove the usefulness of a simple two-class PRMA architecture. Mechanism and protocols required to upgrade the classical single-class PRMA [Goodman 89] to the proposed Two-Class PRMA (TCPRMA) are explained. Also, performance analysis and simulation results are presented.

6.2.2 Revenue Capacity

Understanding the relation and interaction between QoS, pricing, revenue, and service disciplines is not a new effort. Several initial and successful attempts were done. One can notice, by reviewing previous works in the literature, that different researchers approached the problem from different perspectives. The two major approaches are based on two major objectives. Some researchers consider their objective to be maximizing revenue (private-good network) [Fishburn 98, Honig 94], others, consider their objective to be maximizing the total users’ utility (public-good network) [Cocchi 93, Orda 97, Sairamesh 95, Wang 01]. In both approaches, differential pricing in multi-class networks was found to be very essential.
For example, in [Cocchi 93], the network was assumed to be a public-good. Based on that assumption, authors found that it is always true that a multi-class architecture can be designed to achieve a higher total users’ utility than a single-class architecture, for the same system capacity. While, for example, in [Honig 94], the network was assumed to be a private-good and authors showed that revenues can be maximized with differential pricing.

Most of the work available in the literature, as discussed above, dealt with optimizing pricing vectors for a specific multi-class architecture, in order to maximize users’ utilities or network revenues. Little work has been done to compare different multi-class designs in terms of network revenue. For example, in [Fishburn 98], authors compared between three designs for two types of demands, delay-sensitive and delay-insensitive. The three designs are: two separated networks for the two demand types, one network with one high quality grade of service, and one network that admits the two types of demand but without resource-sharing. This analysis didn’t clearly address the issue of different possible levels of differentiation in the provisioned QoS between the different classes, and the effect of these different levels on network revenue. On the other hand, in [Edell 99] a similar comparison study was conducted through the INDEX project, but the differentiation was done at the level of access speeds, only. In this work, we are interested in comparing different multi-class architectures with different provisioned QoS parameters under different classes, even if all the classes have the same access speed.
Even in the case of a single-class data link, we may ask which is better, to admit 10 live video sessions in a 100Mbps link, with 10Mbps exclusively reserved for each session? Or, to admit 20 of these sessions that statistically share the available 100Mbps capacity? Assume each live video session has a maximum transmission rate of 10Mbps. Such questions become more critical when one compares different multi-class designs, or compares a multi-class to a single-class design.

The main obstacle toward achieving our objective is that it is very difficult to have a complete knowledge about the behavior of users in response to prices and offered levels of service. This behavior is usually characterized in the literature as the willingness to pay functions and users’ utilities in general. Thus, in this work, we try to minimize the dependency on such functions. Our objective is to develop a framework that can help in comparing different multi-class architectures rather than optimizing the design of a specific pricing schema. Since all modern networking architectures are motivated by the potential increase in revenue, we will adopt the private-good network approach in our analysis.

In this subsection, we introduce a new concept in an attempt to contribute in understanding the relation and interaction between the concepts of QoS metric, QoS score, pricing, and revenue. In the following we introduce few definitions:

*QoS Metrics:* measurable data-traffic parameters that may affect the satisfaction of users. Most common metrics are average and variance of delay, delay jitter, and packet loss, in addition to throughput and availability.
**QoS Score** ($q$): numerical value that represents a subjective measure of user satisfaction. User satisfaction can be measured by conducting rating tests using a group of volunteers as in [Goodman 89]. Or, it can be estimated by measuring the mean square error of the received signal compared to the transmitted signal.

**QoS Sensitivity** (QS) Function: a multi-dimensional application-dependent function that defines a relation between the QoS score and the QoS metrics.

**QoS-Dependent Charging Rate** (QDCR) Function ($S(q)$): an application-dependent function that defines a relation between a charging rate, measured in $$/sec., and the QoS scores of a specific application. This charging rate is the maximum rate that can be justified for a given QoS score, from the perspective of all users.

**User-Spending-Power** (USP): the maximum rate, measured in $$/sec., a user can spend on any session.

Note that the QDCR function is different from the Willingness-To-Pay (WTP) function defined in the literature. WTP is defined in a similar way to the QDCR function, but it is defined per-user. Although all WTP functions of all users may have the same general form, each user is assumed to have his/her own specific WTP function.

In this work we decompose users’ behavior into two factors, the QDCR function and the USP level of different users. We argue that all human users have the same QDCR function for a specific application, regardless of their ability to spend money. This means that both, users with high and low USP values agree that a QoS score, say $q_1$, for a specific application justifies a charging rate, say $r_1$. But, users with low USP values may not be able to pay the charging rate $r_1$, which forces them to choose lower quality CoSs
with lower charging rates. This can be clearly seen in real life experiences. For example, both rich and poor persons usually agree that the quality of a luxury car justifies a maximum price of $50,000. The rich person can pay this price, while the poor cannot, despite the fact that he/she believes that the quality of such a car justifies such a price. Also, both rich and poor persons usually do not agree that the quality of an economy car justifies a maximum price of $100,000, for example. The rich person will not pay this price despite the fact that he/she is capable of paying it.

Thus, based on our model, a $WTP$ function of any user can be defined in terms of both the $QDCR$ function of the utilized application and the $USP$ level of this user as follows:

$$WTP^*_i(q) = S_x(q)U[USP_i - S_x(q)] + (USP)U[S_x(q) - USP_i],$$

where $U[.]$ is the unit step function, $WTP^*_i(q)$ is the $WTP$ function of application $x$ for user $i$, $S_x(q)$ is the $QDCR$ function of application $x$, $USP_i$ is the $USP$ level for user $i$. For example, if the two applications, application ‘$a$’ and application ‘$b$’, have linear $QDCR$ functions, $S_a(q)$ and $S_b(q)$, three different users with three different $USP$ levels will have three different $WTP$ functions for each application, as shown in Figure 6–3.
Note that, in [Honig 95], for example, the WTP function was defined against a QoS metric (delay). As we explained earlier, it makes more sense to define it against the QoS score, as we did in this chapter. It is the improvement in the QoS score that justifies higher charging rates, but not necessarily the improvement in some QoS metrics. Also, note that we are not going to deal with the WTP function from now on, but understanding the relation between this function and both the QDCR function and the USP value is important.

Now, consider a data link that supports non-best-effort services, i.e., there is a call admission control unit that limits the number of sessions at each CoS, in order to maintain promised QoS metrics. Assume each CoS carries only one application type. Note, many or all classes may carry the same application type. Assume also that the QDCR function ($S(q)$) is known for all supported application types. We define:

Figure 6–3: Demonstration of the relation between the WTP function, the QDCR function, and the USP value for two applications and three users.
Revenue Capacity (RC): the maximum revenue rate that can be achieved at a data link given a specific capacity, specific application types, QDCR functions of supported applications, and CoSs design.

\[
RC_x = \sum_{i=1}^{M_x} n_i^x S_i(q_i^x) \quad \text{[$/sec]} \tag{6-2}
\]

In (6–2), \(RC_x\) is the Revenue Capacity of multi-class architecture \(x\). \(M_x\) is the number of classes in architecture \(x\). \(n_i^x\) is the maximum number of sessions that can be supported by class \(i\) in architecture \(x\), without violating the promised QoS metrics for class \(i\). \(S_i(.)\) is the QDCR function for the application supported by class \(i\) in architecture \(x\). \(q_i^x\) is the QoS score that will be experienced by a class \(i\) session in architecture \(x\), when the number of sessions in class \(i\) is \(n_i^x\). Note that in (6–2), the QoS score of any session depends on the average throughput, and not only the temporal characteristics of the traffic. Thus, if two sessions of the same application type have the same average and variance for packet loss, delay, and delay jitter, but they have different throughout, then they belong to different CoSs. An example on this is two video sessions, one results in a small screen and the other results in a large screen, but both of them experience the same average and variance for packet loss, delay, and delay jitter. So, capacities of different CoSs are indirectly implied in the QoS scores in the formula of revenue capacity in (6–2).

Theorem 6.1: If the following two assumptions hold true, and if two multi-class architectures, for the same data link capacity, are to be compared in terms of
network revenue, then the architecture that has the higher Revenue Capacity will achieve a higher network revenue.

Assumption 1: design and maintenance costs of the two architectures are the same.

Assumption 2: infinite number of users and all users have the same QDCR function for any specific application type. However, the pool of users is classified into an infinite number of categories, where each category represents a specific USP level.

Proof: assumption 2 means that any CoS in any multi-class architecture will be fully utilized, as long as the charging rate of that CoS complies with the QDCR function. \( RC \) measures the revenue rate when all classes of a specific multi-class architecture are full. If design and maintenance costs for all architectures are the same (assumption 1), then the architecture with the highest \( RC \) value will generate the highest network revenue.

Assumption 1 is valid, because usually designing different multi-class architectures is done at the 2\(^{nd}\), 3\(^{rd}\), and/or 4\(^{th}\) layer, which does not lead to significant differences in the design or maintenance costs. Assumption 2 is a simplification of the real situation, but not far from the reality as we explained earlier. As the Internet becomes a global machine, assumption 2 becomes a reasonable approximation. Also, when comparing different architectures, one is interested in comparing them when they are highly loaded. If architectures are lightly loaded, any architecture may perform very well in terms of QoS scores.
6.2.3 Revenue Comparison Framework

The problem in computing the $RC$ value is its dependency on $QDCR$ functions, which are not usually available. Defining a $QDCR$ function, i.e. a $WTP$ function of a user, requires knowledge beyond engineering. Different researchers made different simplifying assumptions for the characteristics of $WTP$ functions, some examples can be found in [Cocchi 93, Honig 95, Paschalidis 00]. In the following we define general assumptions that do not contradict previously proposed assumptions and allow us to compare between different multi-class architectures without knowing the exact form of $QDCR$ functions.

One assumed characteristic for $QDCR$ functions, which is intuitive and implied in all previous work, is that all $QDCR$ functions must be positive, continuous, and monotonically increasing against the QoS score.

Lemma 6.1: If the following two assumptions hold true, and if two multi-class architectures, $\alpha$ and $\beta$, for the same data link capacity, are to be compared in terms of network revenue, then architecture $\alpha$ is superior to architecture $\beta$ if:

\[ \sum_{i=1}^{M_\alpha} h_i^\alpha q_i^\alpha > \sum_{i=1}^{M_\beta} h_i^\beta q_i^\beta . \]  \hspace{1cm} (6–3)

Assumption 1: all classes of both multi-class architectures support the same application type.

Assumption 2: the $QDCR$ function ($S(.)$) of the supported application type is positive and linearly increasing.

Proof:
\[
\sum_{i=1}^{M_\alpha} n_i^\alpha q_i^\alpha > \sum_{i=1}^{M_\beta} n_i^\beta q_i^\beta \Rightarrow \sum_{i=1}^{M_\alpha} n_i^\alpha S(q_i^\alpha) > \sum_{i=1}^{M_\beta} n_i^\beta S(q_i^\beta)
\]

since \( S(.) \) is linearly increasing (assumption 2).

\[\Rightarrow RC_\alpha > RC_\beta \Rightarrow \text{revenue from architecture } \alpha > \text{revenue from architecture } \beta, \text{ based on theorem 6.1}.\]

Lemma 1 allows us to compare two multi-class architectures without knowing the exact form of the \( QDCR \) function, as long as it is linearly increasing. Assumption 1 means that we are comparing the two multi-class architectures when all users, under all CoSs, run the same application type. In the following, we relax the conditions on the \( QDCR \) function when one is comparing between a single-class and a multi-class architecture.

Lemma 6.2: If the following two assumptions hold true, and if a multi-class architecture \((\theta)\) is to be compared to a single-class architecture \((\rho)\), for the same data link capacity, in terms of network revenue, then architecture \(\theta\) is superior to architecture \(\rho\) if:

\[
\left(\sum_{k=1}^{M_\rho} n_k^\rho\right)^{-1} \sum_{i=1}^{M_\rho} n_i^\rho q_i^\rho > q^\rho \tag{6-4}
\]

and if \(\sum_{k=1}^{M_\rho} n_k^\rho \geq n^\rho. \tag{6-5}\)

Assumption 1: same application type is supported by architecture \(\rho\) and all classes of architecture \(\theta\).
Assumption 2: the QDCR function \( S(.) \) of the supported application is increasing linear or increasing convex.

Proof:

Let \[ \left( \sum_{k=1}^{M_a} n_i^\theta q_i^\theta \right)^{-1} \sum_{i=1}^{M_a} n_i^\theta q_i^\theta > q^\rho \Rightarrow S \left( \left( \sum_{k=1}^{M_a} n_i^\theta q_i^\theta \right)^{-1} \sum_{i=1}^{M_a} n_i^\theta q_i^\theta \right) > S(q^\rho) \] because \( S(.) \) is continuously increasing.

Since \( S(.) \) is increasing linear or increasing convex, the following inequality can be proved easily, where the equality holds when \( S(.) \) is increasing linear:

\[
\left( \sum_{k=1}^{M_a} n_i^\theta q_i^\theta \right)^{-1} \sum_{i=1}^{M_a} n_i^\theta S(q_i^\theta) \geq S \left( \left( \sum_{k=1}^{M_a} n_i^\theta q_i^\theta \right)^{-1} \sum_{i=1}^{M_a} n_i^\theta q_i^\theta \right) \]

\[ \Rightarrow \sum_{i=1}^{M_a} n_i^\theta S(q_i^\theta) \geq \left( \sum_{k=1}^{M_a} c_i^\theta \right) S(q^\rho) \], this comes from (6–6).

\[ \Rightarrow \sum_{i=1}^{M_a} n_i^\theta S(q_i^\theta) > n^\rho S(q^\rho) \], based on the condition in (6–5).

\[ \Rightarrow RC_\theta > RC_\rho \Rightarrow \text{revenue from architecture } \theta > \text{revenue from architecture } \rho, \text{ based on theorem 6.1.} \]

We will utilize lemma 2 to prove the superiority of the proposed TCPRMA architecture over the classical single-class PRMA, when the supported application is voice-activated speech.
6.2.4 Two Class PRMA Architecture

6.2.4.1 Background

PRMA was introduced in [Goodman 89] as an efficient technique to increase the utilization and efficiency of TDMA wireless access technology. Since then, several schemes were proposed to improve the performance of PRMA, which was optimized for speech sessions. New techniques were proposed specifically to handle multimedia traffic requirements [Anstasi 97, Dyson 97, Iera 00, Koutsakis 00, Kubbar 00, Uehara 00].

The concept of multi-class multiple access for TDMA wireless LANs was proposed in several papers [Kubbar 00, Koutsakis 00, Anstasi 97, Capone 98]. The basic idea in most of these proposals is to treat different applications or different traffic types differently, in order to satisfy the QoS requirements of these different applications. For example, the work in [Anastasi 97, Koutsakis 00] classified traffic into speech, video, and data. While the work in [Capone 98, Kubbar 00], for example, classified the traffic into CBR, VBR, and ABR.

In this work, we introduce a mechanism that enables the multiple access protocol of the uplink channel in a TDMA wireless LAN to support two classes or types of service. The system provides two CoSs due to differentiation in the access and reservation mechanism, regardless of the running application. The idea is to allow an application to be supported by any of the two CoSs, but with different experienced QoS at each class. This will justify differential pricing.
6.2.4.2 System Model and Protocols for TCPRMA

In this subsection, we describe the multiple access and reservation protocols that will enable the uplink channel in a start topology wireless LAN to support two CoSs. We propose this system to support voice-activated speech sessions. The objective is to prove that such a simple differentiation mechanism can derive higher revenues. We prove this using the revenue comparison framework we developed in section 5.2.3, especially lemma 6.2.

The system model is as explained in [Goodman 89], i.e., uplink time frame contains 20 time slots, the frame length = 16 msec, and uplink channel rate = 720 kbps. We assume error-free channel with no capture. More details can be found in [Goodman 89].

6.2.4.2.1 Speech model

Each voice source is modeled as an on/off Markov model, like the slow speech activity detector model in [Goodman 91]. The “on” state represents a talk period and the “off” state represents a silence period. We assume an activity detector that can detect states as short as 16 msec. Note, our model assumes state transitions occur at frames boundaries and not at slots boundaries. A source in the “on” state transmits compressed voice at 32 Kbps. All voice sources are independent. For the on/off model, we used the following transition probabilities: \( P(\text{on}|\text{off}) = 0.0119 \), \( P(\text{off}|\text{on}) = 0.016 \), \( P(\text{on}|\text{on}) = 0.9881 \), \( P(\text{off}|\text{off}) = 0.984 \). These values result in an activity factor = 42.6\%. 

6.2.4.2.2 Packet loss

Each mobile source has a buffer that can queue voice packets till they can be successfully transmitted. Any packet that is queued 32 msec or more is dropped.

6.2.4.2.3 Multiple access and contention protocols

1- Protocols for 1st CoS:

the system can be designed to support at most 20 1st class users, which equals to the number of time slots per frame. Each 1st class user is allocated a specific time slot for the whole life of the conversation as in classical deterministic TDMA schemes. When a 1st class source switches from “off” to “on” state, it transmits the first packet immediately in its allocated time slot. It does so without checking the downlink control bit that shows if this time slot is reserved for another 2nd class user.

Since each 1st class user has a dedicated time slot, there is no chance for collision between 1st class users. If the first packet at the beginning of a talk period of a 1st class source collides with a packet from a 2nd class source, the 1st class source retransmits the same packet in the same dedicated time slot in the next frame. Successful transmission in the 2nd attempt for this 1st class source is guaranteed, as we explain later.

2- Protocols for 2nd CoS:

the maximum number of 2nd class users depends on the QoS score the service provider plans to support. When a 2nd class source has a packet at the beginning of a
talk period, it searches for a free time slot by monitoring the downlink control bit. Once it finds a free time slot, say slot $h$, it transmits the packet based on a defined permission probability ($P_p$). If the transmission is successful, the time slot ($h$) will be reserved for this 2\textsuperscript{nd} class user, which keeps sending voice packets in time slot $h$ until one of the following two events occurs:

*Event 1*: the talk period ends and the 2\textsuperscript{nd} class source has no more packets to send. This will lead to the release of time slot $h$.

*Event 2*: a 1\textsuperscript{st} class source that is utilizing time slot $h$ enters a new talk period and sends a packet that collides with the transmission from the 2\textsuperscript{nd} class source. Once the 2\textsuperscript{nd} class source detects a failure in transmission, it stops immediately from sending packets in time slot $h$, which will be declared as reserved.

If event 2 occurs, the 2\textsuperscript{nd} class source must search for another free time slot to retransmit the packet that suffered from collision, and the rest of the voice packets. Since the 2\textsuperscript{nd} class user will refrain from sending packets in time slot $h$ after the detection of collision, and since time slot $h$ will continue to be declared reserved, the only source that can transmits in the next frame is the 1\textsuperscript{st} class source that is utilizing time slot $h$. This ensures continuous transmission for any 1\textsuperscript{st} class user with a maximum possible delay of 16 msec, which is acceptable.

The base station will declare a time slot as reserved, at least for one frame duration, whenever a collision happens. This is to allow the 1\textsuperscript{st} class user utilizing that specific time slot to transmit without losing any packet. However, a collision also may happen between two or more 2\textsuperscript{nd} class sources contending on a specific free time slot,
again this time slot will be declared as reserved, at least for the next frame, which will prevent all 2nd class users from contending on this time slot for one frame. This will waste the time slot if the assigned 1st class user has no packets to send, but at the same time, such protocols will keep the system very simple.

With the previous description of the TCPRMA architecture, 1st class users will enjoy excellent voice quality, without experiencing any packet loss. 2nd class users, on the other hand, will be experiencing packet loss which will degrade the quality of the speech sessions. The fundamental questions are, what are the QoS score levels for 2nd class users that will make the TCPRMA superior to the classical PRMA? Can the protocols described in this section result in an efficient multi-class design? In the following section we address these questions.

6.2.5 Performance Analysis of TCPRMA

In this section, we compare the TCPRMA to the classical single-class PRMA. Thus, we will be using lemma 6.2, where architecture $\theta$ refers to the TCPRMA and architecture $\rho$ refers to the classical PRMA. The same artificial voice sources and the same number of active sessions will be used in the comparison of the two architectures. So, the two conditions in (6-4) and (6-5) can be simplified to one condition, which makes architecture $\theta$ superior to architecture $\rho$ under the assumptions listed in lemma 6.2:
If \[
\left( n_1^\theta + n_2^\theta \right)^{-1} \left( n_1^\theta q_1^\theta + n_2^\theta q_2^\theta \right) > q^\rho
\] (6–7)

The condition in (6-7) simply means that architecture $\theta$ is superior to architecture $\rho$ if the former has average QoS score higher than the later.

Figure 6–4 shows the $QS$ function for a voice-activated speech session. In this case the $QS$ function is only a function of one QoS metric, $P_{loss}$. Figure 6–4 is regenerated from the results reported in [Goodman 89]. Results in [Goodman 89] do not show the QoS score for $P_{loss}$ less than 0.02, so we will force the performance of the two systems into the defined range of $P_{loss}$. In Figure 6–4, score 5 stands for excellent, 4 for good, 3 for fair, 2 for poor, and 1 for unsatisfactory. This rating is based on averaging the opinion of 12 listeners (six male and six females), [Goodman 89].

$P_p$ is a critical control parameter that must be optimally set. It directly affects the probability of collision between contending terminals. Figure 6–5 shows the average $P_{loss}$ experienced by 2$^{nd}$ class users in the TCPRMA architecture when the number of 1$^{st}$ class sessions is 20, the maximum. The results show the sensitivity of performance to $P_p$ value. It is clear that the best value for $P_p$ is 0.4 for approximately any number of 2$^{nd}$ class active sessions ranging from 10 to 20. Remember that the 1$^{st}$ class sessions experience $P_{loss} = 0$.

An encouraging observation from Figure 6–5 is that the TCPRMA can support up to 14 2$^{nd}$ class sessions ($n_2^\theta$) without exceeding $P_{loss} = 0.01$, which is used in [Goodman 89, Goodman 91] to define system capacity, in addition to 20 1$^{st}$ class active sessions ($n_1^\theta$) with $P_{loss} = 0.0$. 
For PRMA, we obtained similar results to the results reported in [Goodman 91]. Our simulations showed that PRMA, with a voice source model defined as in subsection 5.2.4.2.1, can support at most 37 active voice sessions with $P_{loss} = 0.01$. However, in order to benefit from the QS function, Figure 6–4, we compared PRMA to TCPRMA when the total number of users is 40. As we explained earlier, in such a scenario it is enough for TCPRMA to have higher average QoS score in order to have the potential for achieving higher revenue, (6-7). Figure 6–6 shows $P_{loss}$ for both PRMA and TCPRMA when the total number of users is 40 versus $P_p$. As shown, PRMA can achieve a minimum $P_{loss} = 0.025$ which corresponds to QoS score $= 3.85$. TCPRMA can achieve a minimum of $P_{loss} = 0.056$ with $n_1^\theta = 20$ and $n_2^\theta = 20$, which corresponds to an average QoS $= 4.05$. Recall that all 1st class users enjoy a QoS score $= 5$, since they experience $P_{loss} = 0.0$.

An important issue to investigate is the effect of the ratio $(n_1^\theta$ to $n_2^\theta$) on the superiority of TCPRMA over PRMA in terms of network’s revenue. Figure 6–7 shows the performance of TCPRMA with a total number of users $= 40$ and for different numbers of $n_1^\theta$. The average QoS score for all these different scenarios varies between 4.05 and 3.9, with the higher scores for the higher values of $n_1^\theta$. These values still confirm the superiority of TCPRMA over PRMA.

Figure 6–6 shows an interesting observation, that PRMA tends to go unstable before TCPRMA as we increase $P_p$. This suggests that 2nd class users under TCPRMA will experience lower values of $P_{loss}$ than PRMA sessions for high number of users. The
reason for this is that the number of contending sources on a free time slot in TCPRMA is smaller than that number in PRMA. In TCPRMA only 2\textsuperscript{nd} class users may contend on a free time slot. Frequent collisions lead to unstable operation point where every source will have queued packets to be transmitted. Figure 6–8 shows the performance of both PRMA and TCPRMA as the total number of users increases for $P_p = 0.4$, which is a good value in general. For high number of users, $P_{loss}$ for 2\textsuperscript{nd} class sessions of TCPRMA is lower than $P_{loss}$ for sessions under PRMA, despite having 20 1\textsuperscript{st} class sessions with $P_{loss} = 0.0$ under TCPRMA. Although, the QoS score at such high $P_{loss}$ values is extremely low, these results demonstrate the efficiency of TCPRMA in better utilizing available resources by reducing the chance of collisions in general between all users.

It should be noted that such simple multi-class architectures can achieve clearer superiority if other applications that are more sensitive to packet loss rate are considered. In our example, voice-activated speech application, reducing the value of $P_{loss}$ from 0.02 to 0.0 results in an increase from 4.0 to 5.0 only in the QoS score. Despite this, the proposed simple TCPRMA proved to be superior to the classical PRMA in terms of achieving higher network’s revenue.
Figure 6–4: QoS score, based on average of listeners ratings, versus $P_{\text{loss}}$ for voice-activated speech sessions.
Figure 6–5: Average $P_{loss}$ experienced by 2\textsuperscript{nd} class users in the TCPRMA versus the number of 2\textsuperscript{nd} class active sessions. Number of 1\textsuperscript{st} class active sessions = 20.
Figure 6–6: Average $P_{loss}$ versus $P_p$ for the classical PRMA and the $2^{nd}$ class of TCPRMA for a total number of users = 40. In TCPRMA, number of $1^{st}$ class active sessions = 20.
Figure 6–7: 4. Average $P_{loss}$ for 2nd class users in TCPRMA versus $P_p$ for different numbers of 1st class users while total number of users is fixed at 40.
6.3 Conclusion

We developed a framework for comparing different multi-class architectures in terms of network revenue. The difference between QoS metrics and QoS scores was emphasized and the PRMA technique for wireless LANs was used as a demonstrative example. A simple Two-Class PRMA (TCP/RMA) protocol was proposed and compared

*Figure 6–8: $P_{loss}$ experienced by PRMA and 2nd class of TCP/RMA versus total number of users, $P_p = 0.4.$*
to PRMA in terms of network revenue. It was shown that this simple TCPRMA architecture is superior to the classical single-class PRMA. Also, it was shown that TCPRMA approaches instability slower than PRMA when the total number of users is increased. In general, we attempted to contribute for a better understanding of the interaction between quality-of-service, service disciplines, pricing, and revenue.
7.1 Introduction

Dynamic and adaptive pricing schemes have been recently proposed in a response to a rapid and dramatic increase in the volume and diversity of Internet traffic. The motivation behind dynamic pricing is not only to maximize revenues, but also to control network congestion. In [Mackie-Mason 95] there is a fundamental analysis and a description of the basic economic theory of pricing congestible network resources.

There are two approaches for managing and controlling networks via pricing mechanisms, as we briefly mentioned in subsection 6.2.2. Based on these two approaches networks can be classified into private-good networks and public-good networks. In private-good networks, the objective of pricing schemes is to maximize network’s revenue, like the work in [Fishburn 98, Honig 95, Paschalidis 00]. In public-good networks, usually, the objective is either to maximize individual-users’ utility [Cocchi 93, Fulp 98, Korilis 98, Wang 01], or to maximize the social welfare in general [Courcoubetis 00, Sairamesh 95, Varvarigou 98]. Also, auction-based approaches were proposed when little information about demands is available, as in [Semret 99, Semret 00].
For Best Effort services, as we have in the current Internet, static and dynamic usage-based pricing schemes are usually proposed to manage the operation of the Internet. Static schemes cannot be optimal in responding to the dynamic behaviors of users’ traffic. However, it is still debatable whether achieving optimal revenues or optimal users’ utilities is doable or worth the migration from static to dynamic pricing. In [Shenker 96] there is a good discussion and analysis of this debate.

On the other hand, for resource-reservation type networks, like MPLS, ATM, and classical public switched networks, capacity-based pricing schemes can be considered. As is the case in charging users of the public switched network. For example, a user pays for the duration of leasing a T1 line, regardless of the actual usage of this line. Also, a user will pay more for a T3 line if leased for the same duration. So, in such networks, one way to charge users is based on the reserved bandwidth and the holding times of their sessions, rather than based on the actual usage of the reserved resources.

In general, for an arbitrary connection in a communication network, the cost of the connection can be expressed as follows [Kelly 97, Songhurst 97]:

\[
\text{Cost} = z + \beta T + c_p N, \quad (7-1)
\]

where \( z \geq 0 \) is a fixed call-setup charge in $, \( T \) is the duration of the call in sec., \( N \) is the total number of packets or bits transmitted, \( c_p \) is in $/bit, and \( \beta \) is in $/sec. Any of the parameters \( z, c_p, \) and, \( \beta \) can be designed to be fixed or adaptive. For example, in the case of adaptive usage-based pricing schemes proposed for Best Effort services, \( z \) is set to zero, while \( c_p \) and/or \( \beta \) are designed to be dynamic. Some usage-based pricing proposals
assume that $c_p$ and/or $\beta$ can vary even during a session life. Such schemes are unlikely to be implemented, simply because users cannot expect the cost of their sessions a priori. Other proposals allow $c_p$ and/or $\beta$ to change only at the beginning of a new session, which sound to be more doable, like the work in [Courcoubetis 00], where the cost of a session depends on the effective bandwidth and holding time. In general, network service providers do not encourage any complicated pricing scheme because users dislike such schemes. This became obvious for us during several meetings we had with decision makers from Ameritech company (South Western Bell, now).

We return now to capacity-based pricing in resource-reservation type networks. We consider only dynamic pricing where changing the charging rate or cost of a session can happen only at the arrival of the session and not during its duration. For capacity-based pricing we can still utilize the formula in (7–1) by setting $\beta = 0$. Thus the cost of a session can be expressed as:

\[
\text{Cost} = z + (c_p)(\text{reserved bandwidth})(\text{holding time}) = z + (c_p)(N_m), \quad (7–2)
\]

where $N_m$ is the maximum number of bits that can be transmitted during the duration of the session.

In most of the previously proposed schemes, $z$ is fixed at a specific value and $c_p$ is designed to be an adaptive parameter that depends, in general, on the status of the network and the requested bandwidth, as in [Breker 96]. In this work, we choose to fix $c_p$ and let $z$ be adaptive, we then attempt to find an optimal policy for controlling the value of $z$. In [Roberts 98] a similar approach was discussed and a similar formula to the one in
(7–2) was discussed. However, an optimal policy for controlling the value of $z$ was not found. In this work, we also consider networks with guaranteed QoS parameters. In such networks, admitting a user’s request does not affect the QoS enjoyed by other users in the network, a similar model is analyzed in [Orda 97].

Thus, in this chapter, we are interested in resource-reservation type with explicit guaranteed QoS parameters, as can be achieved under specific classes of service in an ATM or an MPLS network. Pricing decisions are assumed to take place at the edge of the network, where several paths are accessed and managed, as is the case in MPLS and ATM architectures. In such architectures, a multiple-link connection between different edge devices (boarder routers/switches) is viewed as one link with dedicated capacity. In this chapter, we study dynamic call-setup pricing for such a point-to-point dedicated channel with variable-size (bandwidth size) arrivals. We adopt a private-good approach, so the objective is to maximize revenue from the perspective of service provider. We model the problem as a discounted Markov Decision Process (MDP). We use the Gauss Seidel value iteration algorithm to achieve an $\varepsilon$-optimal policy for different scenarios. The results are then discussed and analyzed.

7.2 System Description

We assume a point-to-point communication channel with dedicated resources, as is the case in a Label Switched Path (LSP) in Multi-Protocol Label Switching networks or a Virtual Path (VP) in ATM networks. Total capacity of the channel is $C \text{ bps}$. $M$ CoSs are supported. A class-$k$ call consumes $b_k \text{ bps}$, has an exponentially distributed holding
time with a mean value \((1/\mu_k)\) sec., and arrives based on a Poisson arrival process with a mean arrival rate \(\lambda_k\) calls/sec.

If \(U\) is the current utilization in bps, then an arriving class-\(k\) call will be accepted if \(U + b_k \leq C\). At the arrival of each call the system declares a call-setup charge \((z)\). Upon the declaration of the call-setup charge (CSC), a class-\(k\) user will accept this charge with a probability \(f_a^k(z)\), where \(f_a^k(.)\) is a function that defines the probability of accepting a CSC equals to \(z\) when the requesting user belongs to class-\(k\). Note, \(f_a^k(.)\) is not a probability density function.

A fixed usage-based charging rate \((c_p)\) measured in $/sec./bps ($/bit) is assumed to be known by all users. Thus, the expected revenue from an arriving class-\(k\) call is

\[
\frac{1}{1 - \left(1 - f_a^k(z + c_p b_k \mu_k^{-1})\right)} \$. The objective of the dynamic call-setup pricing system is to maximize the revenue generated from the point-to-point dedicated channel by controlling an adaptive state-dependent CSC.

Defining the function \(f_a^k(.)\) needs more consideration. The following issues need to be considered:

1- In a dynamic call-setup pricing system, the decision of rejecting a request even if resources are available must not be ruled out. This decision must be, somehow, added to the action space.

2- \(f_a^k(.)\) is actually an attempt to model users’ behaviors as they interact with declared CSCs.
3- Action space must be a finite group in order to utilize the value iteration algorithm.

In response to issue 1, we assume that any $f_a^k(.)$ must be equal to zero after a specific value, $z_{max}^k$. Thus, choosing $z_{max}^k$ as the optimal decision implies a soft rejection of the current request because $f_a^k(z_{max}^k) = 0$. Issue 2 requires from us to add more specifications on $f_a^k(.)$, in order to approximate the behavior of users in a real situation.

An intuitive observation is that the value $z_{max}^k$ must be related to the expected cost of a user’s request. For example, if a user is expecting to spend $10 on the session he/she intends to run, it is more likely that he/she will accept a CSC of $5 than a user who is expecting to spend $1 on these sessions. Also, it should be intuitive that a user is more likely to accept a smaller value of CSC. Thus, we assume that $f_a^k(.)$ is a monotonically decreasing function with $f_a^k(z_{max}^k) = 0$, where:

$$z_{max}^k = \sigma(c_p b_k \mu_i^{-1}). \quad (7–3)$$

$(c_p b_k \mu_i^{-1})$ is the expected cost of a session requested by a class-$k$ user. $\sigma$ is a performance parameter that quantifies the effect of the expected session’s cost on the behavior of users when responding to CSCs.

Issue 3 forces us to quantify the action space. We assume that CSC has a quantization step size of $1.0. In this study, we will consider three types of $f_a^k(.)$; convex, linear, and concave, as shown in Figure 7–1.
Figure 7–1: The three \( f^k_a (.) \) functions used in this study; concave, linear, and convex.

7.3 Modeling The System As a Discounted Markov Decision Process

For the purpose of mathematical analysis we define the following three vectors:

\[
\bar{B} = [b_1, b_2, ..., b_M].
\] (7–4)
\[
\tilde{H} = [\mu_1^{-1}, \mu_2^{-1}, \ldots, \mu_M^{-1}]. \tag{7-5}
\]

\[
\tilde{I} = [1, 2, \ldots, M]. \tag{7-6}
\]

### 7.3.1 State Space

The state of the system is determined by the number of current active calls of each class and the class of the current arriving call.

\[
S = (\tilde{N}, \tilde{E}). \tag{7-7}
\]

\[
\tilde{N} = [n_1, n_2, \ldots, n_M], \tag{7-8}
\]

where \( n_k \) is the number of class-\( k \) active calls in the system.

\[
\tilde{E} = [e_1, e_2, \ldots, e_M], \tag{7-9}
\]

where \( e_k = \begin{cases} 
1, & \text{if arriving call belongs to class-} \ k \\
0, & \text{otherwise}
\end{cases} \)

As we mentioned earlier, a class-\( k \) call utilizes \( b_k \) bps of the total capacity \( C \).
7.3.2 Decision Epochs

The instance just before the arrival of a new call, belonging to any class, is the decision epoch. Thus, decision epochs are random instances with an inter-decision time exponentially distributed with mean value of $\lambda^{-1}\text{sec.}$, where:

$$\lambda = \sum_{k=1}^{M} \lambda_k.$$  \hspace{1cm} (7–10)

7.3.3 Actions

The action space is defined as follows:

$$A_{(\bar{N},\bar{E})} = \begin{cases} z : 0 \leq z \leq z_{max}^k, \text{if } \bar{N}\cdot\bar{B} + \bar{E}\cdot\bar{B} \leq C \\ \phi, \text{if } \bar{N}\cdot\bar{B} + \bar{E}\cdot\bar{B} > C \end{cases}$$ \hspace{1cm} (7–11)

The action space defined in (7-11) means that if there are no resources available to serve the current request, no pricing decision can be taken. The only available action is to reject this request. Also, in (7-11), $z_{max}^k$ is the solution for the following equation:

$$f_o^k(z) = 0, \text{ for } z > 0.$$ \hspace{1cm} (7–12)
7.3.4 Reward Function

We define the reward to be the expected revenue from the perspective of the network owner:

\[ r(\bar{N}, \bar{E}, z) = \begin{cases} 
    f_a(z) + c_p(\bar{B}.\bar{E})(\bar{N}.\bar{E}), & \text{if } \bar{N}.\bar{B} + \bar{E}.\bar{B} \leq C \\
    -c_p(\bar{B}.\bar{E})(\bar{N}.\bar{E}), & \text{if } \bar{N}.\bar{B} + \bar{E}.\bar{B} > C 
\end{cases} \quad (7–13) \]

The expression in (7-13) defines the reward when the system is at state \((\bar{N}, \bar{E})\) and if a CSC equals to \(z\) will be decided. Note that, when the state of the system is outside the admission region, a negative reward is experienced which equals the expected loss if the current request is rejected due to shortage of resources.

7.3.5 Transition Probabilities

First, we define few parameters to obtain an exact expression for the transition probabilities.

\[ a_x^k = P\{x \text{ class-k calls leave the system in an inter-decision interval}\}. \]

\[ P_b(k) = P\{\text{current arriving call belongs to class-k}\} = \frac{\lambda_k}{\lambda}. \quad (7–14) \]
Based on the system description in section 7.2, transition probabilities can be calculated as follows:

\[
P(\tilde{N}', \tilde{E}'| \tilde{N}, \tilde{E}, z) = \\
\begin{cases} 
  f_a^{I,E}(z)P_h(\tilde{I}, \tilde{E}')(\prod_{k=1}^{M} V_{(0, n_k^k)}^{k}) V_{(n_{i_E}+1, n_{i_E})}^{l,E} \\
  + (1 - f_a^{I,E}(z))P_h(\tilde{I}, \tilde{E}')(\prod_{k=1}^{M} V_{(0, n_k^k)}^{k}), & \text{if } \tilde{N}.\tilde{B} + \tilde{E}.\tilde{B} \leq C \\
  P_h(\tilde{I}, \tilde{E}')(\prod_{k=1}^{M} V_{(0, n_k^k)}^{k}), & \text{if } \tilde{N}.\tilde{B} + \tilde{E}.\tilde{B} > C 
\end{cases} 
\tag{7–15}
\]

where,

\[
0 \leq n_k^i \leq n_k, \forall k \neq \tilde{I}.\tilde{E}, \tag{7–16}
\]

\[
0 \leq n_{i_E}^i \leq n_{i_E} + 1, \tag{7–17}
\]

\[
V_{(x,y)}^{k} = \begin{cases} 
  a_{(x-y)}^k, & \text{for } 1 \leq y \leq x \\
  1 - \sum_{i=1}^{x-1} d_i^k, & \text{for } y = 0 \\
  0, & \text{else} 
\end{cases} \tag{7–18}
\]

Equation (7-15) comes from the undependability of holding times of active sessions in the system, regardless of their classes. Also, the events: class of arriving call, decision of the requesting user to accept or reject the declared CSC, and holding times of active
sessions in the system are independent events. Equations (7-16) and (7-17) define the possible states that can be reached given a specific current state. For a better understanding of (7-18), the reader is advised to refer to [Gross 74].

Now, an exact expression needs to be determined for $a_x^k$. Since decision epochs are the instances at which new calls arrive, regardless of their class, and since holding times are exponentially distributed, then it does not matter how long the active sessions have been in the system. The probability that any active session ends in a time interval $t$ is independent of the instance at which the interval $t$ starts. So, the probability that $x$ sessions end in an inter-decision interval can be determined as shown in (7-19).

$$a_x^k = \int_0^\infty (\mu_t^x)^k e^{-\mu_t^x} \frac{t^x}{x!} \lambda e^{-\lambda t} dt \quad (7-19)$$

$$\Rightarrow a_x^k = \left(\frac{\lambda \mu_t^x}{x!}\right) \int_0^\infty t^x e^{-(\mu_t^x + \lambda) t} dt \quad (7-20)$$

$$\Rightarrow a_x^k = \left(\frac{\lambda \mu_t^x}{x!}\right) \left(D_x \big|_{0}^{\infty}\right), \quad (7-21)$$

where

$$D_x = \frac{t^x e^{-(\mu_t^x + \lambda) t}}{-(\mu_t^x + \lambda)} + \left(\frac{x}{\mu_t^x + \lambda}\right) D_{x-1} \quad (7-22)$$

and $D_m = 0$ for $m < 0$. 
With some algebraic manipulation we can get:

\[
\alpha_s^k = \frac{\lambda \mu^*}{(\mu + \lambda)^{s+1}} = \frac{\rho}{(1 + \rho)^{s+1}}, \quad \rho = \frac{\lambda}{\mu}. \quad (7-23)
\]

With the exact expression in (7-23) we have completely determined the transition probabilities.

### 7.3.6 Optimality Equations

The optimality equations can be written as follows [Puterman 94]:

\[
G_n = \max_{z \in A(s,E)} \left\{ r(\tilde{N}, \tilde{E}, z) + \eta \sum_{E} \sum_{\tilde{N}'} P(\tilde{N}', \tilde{E}'|\tilde{N}, \tilde{E}, z) G_{\tilde{n}'} \left(\tilde{N}', \tilde{E}'|\tilde{N}, \tilde{E}, z\right) \right\}. \quad (7-24)
\]

By using the Gauss Seidel value iteration algorithm [Puterman 94] we can get an \(\varepsilon\)-optimal policy for a small number of classes and medium capacity in a reasonable time.

### 7.4 Results And Analysis For Different Scenarios

We utilized the Gauss Seidel values iteration algorithm to find the \(\varepsilon\)-optimal policy for three CoSs, in order to obtain the policy in a reasonable time for several scenarios. We used the following settings:

\[
\varepsilon = 0.01, \quad \eta = 0.95, \quad C = 12 \text{ Mbps}, \quad M = 3, \quad b_1 = 2.0 \text{ Mbps}, \quad b_2 = 1.0 \text{ Mbps}, \quad b_3 = 0.5 \text{ Mbps}, \quad c_p = (1/6)(10^{-7}) \text{ $/sec/bps (S/bit)$}.
\]
We define the offered load by class-\( k \) \((\theta_k)\) as follows:

\[
\theta_k = \frac{\lambda_k}{\mu_k(C/b_k)}. \tag{7–25}
\]

Presenting the optimal policy completely under any scenario is not possible in this chapter. We will selectively present some partial results, in order to gain some understanding of possible behaviors and trends of the optimal policy.

*Figure 7-2, Figure 7-3, and Figure 7-4* show the optimal policies for the three proposed \( f_a^k(\cdot) \) functions in *Figure 7–1*. In these results, we set \( \mu_1^{-1} = \mu_2^{-1} = \mu_3^{-1} = 600 \) sec., \( \theta_1 = \theta_2 = \theta_3 = 2/3, \sigma = 1 \).

Note that, the units of the vertical axis are \(($/Mbps)\). For example, in *Figure 7-2*, at utilization of 6Mbps, the CSC for a class-1 request is $10, for a class-2 request is $5, and for a class-3 request is $2.5. The reason we present the results in this way is to clarify when a soft rejection decision is issued. In all these results, a value of $10/Mbps represents a soft rejection of the request. This value is translated to $20 for class-1, $10 for class-2, and $5 for class-3. Also, note that a CSC of (-$1.0) represents a physical rejection action.
Figure 7–2: Optimal policy assuming no class-2 or class-1 calls in the system, and concave $f_u^h(\cdot)$. 
Figure 7–3: Optimal policy assuming no class-2 or class-1 calls in the system, and linear $f^3_a(\cdot)$. 
The first observation is that at any class-3 utilization level, CSC is proportional to the amount of requested bandwidth. This is intuitive, because admitting larger requests result in a greater congestion.

However, a very important observation is that the optimal CSC is not always monotonically increasing against utilization. As we can see in Figure 7-2, Figure 7-3, and Figure 7-4, the optimal CSC of a class-3 request has a zigzag shape. The reason for
this phenomenon is related to the fact that a class-3 call requests 0.5Mbps, and thus admitting a class-3 call has a special effect on the admission region.

For example, in any of the above results, when utilization is 10Mbps, the optimal policy rejects a class-3 request, but at a utilization of 10.5Mbps, the optimal policy issues a CSC around $3. This can be explained as follows; at a utilization of 10Mbps, if a class-3 request is admitted, it is possible that the next arrival be a class-1 (which requires 2Mbps) and the utilization continues to be 10.5Mbps, thus this class-1 will be rejected, recall that $C = 12$Mbps. This is why, under the specific settings mentioned above, the optimal policy issues a soft rejection of a class-3 request at utilization of 10Mbps. On the other hand, at utilization of 10.5Mbps, a class-1 would be rejected any way, if it is the next arrival and if utilization continues to be the same. So, admitting a class-3 won’t change the fact that a class-1, if it arrives, will be rejected. Other zigzag regions of the optimal policy can be explained with the same reasoning.

We can also note that the optimal policy tends to issue higher CSC values in the case of convex $f_a^k(\cdot)$ and lower values in the case of concave $f_a^k(\cdot)$. When $f_a^k(\cdot)$ is linear, medium CSC values are observed. A fundamental point here is that users are not informed about the state or utilization of the system, which, most likely, will be the case in a real implementation of a dynamic call-setup pricing policy. Thus, if $f_a^k(\cdot)$, in reality, is convex, the optimal policy will charge a class-2 user a $4$ CSC, as shown in Figure 7-2, even if utilization is 0. Of course, if users are aware of utilization, it is very likely that no one would accept such a policy. So, in this study, the optimal policy is optimal in
terms of maximizing network’s revenue, assuming no knowledge is available for users about link’s state.

To show the effect of $\sigma$ on the optimal policy, Figure 7-5 shows the optimal CSC for class-3 arrivals versus class-3 utilization for three values of $\sigma$, all other parameters were set as mentioned above, $c_p = (1/6)(10^{-7})$ $$/sec/bps$. It is clear that the relation between the optimal CSC values and $\sigma$ is not a linear relation. Figure 7–6 shows the optimal CSC for class-3 arrivals versus class-3 utilization for three values of $c_p$, all other parameters are set as before, $\sigma = 1$. It can be noticed that the optimal CSC is linearly proportional to the $c_p$ value. But, this is not the case with respect to $\sigma$.

Figure 7-7 shows that the optimal CSC values depend on the state of the system, and not on the utilization, only. To clarify this dependency more, we plotted the optimal CSC of class-1 arrivals when the total utilization is fixed at 8Mbps, but the composition of class-1 and class-2 calls in the system varies. Results are shown for different values of $\mu_1$. We set other parameters as before, $\mu_2^{-1} = \mu_3^{-1} = 600$ sec (10 min.), $\lambda_1 = 0.4$ calls/sec., $\lambda_2 = 0.8$ calls/sec., $\lambda_3 = 1.6$ calls/sec., and $c_p = (1/6)(10^{-7})$ $$/sec/bps$. We also fixed $\sigma$ at 1. This means, as in (7–3), for $\mu_1^{-1} = 5$ min., $z_{\text{max}}^1 = 10$; for $\mu_1^{-1} = 10$ min., $z_{\text{max}}^1 = 20$; for $\mu_1^{-1} = 20$ min., $z_{\text{max}}^1 = 40$.

The first observation from Figure 7-7 is that even if utilization is constant, optimal CSC may vary as the state of the system varies for the same $\mu_1^{-1}$ value. Also, as $\mu_1^{-1}$ increases, the optimal CSC also increases, for the same state. This is because
accepting a class-1 call will contribute to a longer congestion period if its holding time is increased.

Figure 7–5: Optimal CSC for class-3 arrivals versus class-3 utilization, for different values of $\sigma$. No class-1 or class-2 calls are in the system.
Figure 7–6: Optimal CSC for class-3 arrivals versus class-3 utilization, for different values of $c_p$. No class-1 or class-2 calls are in the system. $c_o = (1/6)(10^{-7}) \$/sec/bps.
Figure 7–7: Optimal CSC for class-1 arrivals when the total utilization is constant at 8Mbps, but the combination of class-1 and class-2 calls in the system varies. The policy is presented for different values of $\mu_i$. No class-3 calls are in the system.

7.5 Conclusion

In this chapter, we presented an exact model for the problem of Dynamic Call-Setup Pricing for a link with dedicated capacity and arrivals with variable-size requests. We used Gauss
Seidel value iteration algorithm to find the ε-optimal policy for different scenarios. We showed that the optimal call-setup charge is not always monotonically increasing with utilization, but rather it depends on the state of the system. We investigated the effect of usage-based pricing, holding time, and user behavior on such policies. Optimal call-setup charge was found to be linearly proportional to the usage-based pricing rate if we fix $\sigma$, but not linearly proportional to $\sigma$, (7-3), which is related to the maximum call-setup charge that might be accepted by users.
Chapter 8
SUMMARY AND CONCLUSIONS

8.1 Summary of Results

The thesis dealt, in general, with several issues faced in modern broadband networks, mainly Quality-of-Service (QoS) provisioning, performance gain, service disciplines, resource allocation, pricing, and revenue. The thesis attempted to provide a better understanding of the interaction and tradeoffs between these issues.

Among the recent proposed networking techniques; namely Multi-Protocol Over ATM (MPOA), Differentiated Services (DS), and Multi-Protocol Label Switching (MPLS), MPOA was the only technique that was practically implemented. The efficiency of MPOA was investigated and the following results were concluded:

- MPOA technique was proven to have a high performance gain in Emulated Local Area Networks (E-LANs), but this gain disappears if MPOA technique is to be implemented in Wide Area Networks (WANs).
- Cache management operations and cache tables at the Multi-Protocol Clients (MPCs) of an E-LAN are the bottleneck in MPOA technique.
- Performance of MPOA technique highly depends on the traffic composition. Controlling the time-out value in a cache table at an MPC can be much more
efficient if traffic composition is considered, rather than depending only on the length and continuity of new micro-flows.

- Neural Network technique was utilized to build a self-learning cache table management system that extracts several features from the arriving traffic at an MPC and issues optimized time-out values based on its supervised history of operation.

Because of the large spectrum of QoS parameters of current and emerging applications, multi-class networks with multiple grades of services are becoming a very attractive solution. Bandwidth-segregation with controlled-sharing approach was proposed for designing multi-class networks. It was found that in order to guarantee absolute QoS parameters, connection-oriented services must be provided. The multi-class network was proposed to support both, connection-oriented and connectionless services. A new technique for emulating connection-oriented sessions above the IP layer was proposed. This technique was compared to the MPLS technique in terms of packet overhead and per-micro-flow state information at core routers. The new proposed technique produces, on the average, an equivalent packet overhead to that experienced in MPLS architecture. While MPLS requires per-micro-flow or per-aggregates-of-flows state information at core routers, the new proposed technique, which utilizes output ports ID-numbers, does not require such state information at core routers, and thus it has no scalability problems. But, the new technique requires hardware techniques that are not supported by the current routers.
To maintain the efficiency of the proposed multi-class network with bandwidth segregation, accurate bandwidth allocation schemes must be developed. A novel Dynamic Bandwidth Allocation (DBA) algorithm was developed, the Virtual Demand Distribution (VDD) algorithm. The objective was to force a specific differentiation between the blocking rates of different Classes of Service (CoSs) in a non-stationary environment. Arrivals at any CoS are assumed to have variable sizes with known probability mass function. The following results were achieved and supported by extensive simulations:

- A mathematical analysis and a proof was developed to show that there could exist a DBA algorithm that while operating in a distributed way can approach global optimality by collecting more information, which would still be much less than the global demand matrix.

- The VDD algorithm was designed to utilize signaling messages in connection-oriented networks to achieve a much higher accuracy in predicting demands in a non-stationary environment.

- Simulations showed that VDD outperforms an equivalent Simple Distributed (SD) algorithm in enforcing a specific differentiation ratio between two CoSs. Simulations, also, showed that VDD performs near global optimality.

The extension of the multi-class networks to mobile wireless Local Area Networks (LANs) was discussed. An exact protocol description was presented to create a Two Class Packet Reservation Multiple Access (TCPRMA) protocol, assuming error-free channel. A revenue comparison framework was developed to better understand the
interaction between QoS metric, QoS score, service disciplines, willingness-to-pay, and revenues. This framework was developed, also, to compare the efficiency of the proposed TCPRMA to the classical PRMA technique. The following results and contributions were achieved:

- A theory for comparing different multi-class networks in terms of network’s revenue was developed. This theory clarified and integrated the concepts of QoS metric, QoS score, QoS sensitivity function, willingness to pay function, and revenues.
- TCPRMA was found to have a higher potential for achieving higher network’s revenue than the classical PRMA.
- TCPRMA was found to approach instability slower than the classical PRMA as the number of total users increases.
- It was shown that a simple protocol, like TCPRMA, may have the potential to derive more revenues using the same network resources, if the differentiation in QoS is done correctly. Correctly means to take into consideration QoS metrics, QoS scores, and willingness-to-pay functions within the proposed CoSs.

Since the discussion of multi-class networks and efficient usage of resources cannot be complete without considering the pricing issue, we analyzed the call-setup dynamic pricing, which probably is the closest scheme, among other dynamic and usage-based pricing schemes, to be considered for practical implementation. The following contributions and results were achieved:
The problem of dynamic call-setup pricing for a link with variable size arrivals was modeled as a discounted Markov Decision Process. Exact expressions were found for the optimality equations.

An $\varepsilon$-optimal policy was found using the Gauss-Seidel value iteration algorithm for different scenarios.

The main observations were that the optimal call-setup charge is not always monotonically increasing with utilization, but rather, the optimal call-setup charge depends highly on the state of the system. The state here means the exact number of active sessions of each size and the size of the arriving request. Also, the optimal call-setup charge was found to be linearly proportional to the capacity-based pricing rate, as long as users behavior is the same. However, the optimal call-setup charge was found not to be linearly proportional to the maximum acceptable call-setup charge by users (related to users behavior), while usage-based pricing rate is fixed.

8.2 Avenues For Further Research

The most interesting subject that was not fully investigated in this thesis is the performance analysis of a fading-aware multi-class mobile wireless LAN. Because the wireless channel is less reliable and has a time-varying capacity, due to the dynamic nature of fading and shadowing effects, a multi-class scheme becomes very attractive. It would be very useful to characterize the performance of such a multi-class mobile wireless LAN that may utilize the adaptive transmission technique to define different
levels of services. In section 8.2.1 we present some preliminary ideas toward designing a fading-aware TDMA multi-class wireless LAN.

Although we proposed a bandwidth segregation multi-class Internet, complete performance analysis was not conducted. Also, several issues continue to attract further research, like the denial-of-service issue and the multiplexing gain that can be achieved by a controlled sharing mechanism as the one proposed in chapter 4.

With respect to QoS analysis, we found that finding, experimentally, the QoS sensitivity functions for the major known applications would be very helpful to research in this area. QoS sensitivity function was discussed in chapter 6.

For the problem of call-setup dynamic pricing, the Markov model and optimal policies were found based on the assumption that users are not informed about the state of the network. It is possible that an edge-device acts on behalf of users to protect their rights against backbone service providers. In this case, the model needs to be modified to reflect the fact that users are aware of the network’s state.

8.2.1 Fading-Aware TDMA Multi-Class Wireless LAN, Preliminary Ideas

We believe that the philosophy of multi-class networks is very attractive in mobile wireless systems. In previous chapters we discussed the potential of multi-class networks in a wired global network. In mobile wireless networks there is more motivation toward multi-class networks than in the wired networks. In addition to the motivations we discussed for wired networks, mobile wireless channels are not as reliable as wired channels. Actually, sometimes, mobile wireless channels become unreliable
during deep fading [Proakis 95]. So, we can say that the capacity of mobile wireless channels is changing with time. Thus, unlike wired networks, in mobile wireless networks competition over resources is not governed only by demands but also by the status of the channels, because the available “reliable” resources depend on the status of the wireless channels.

It may become possible by technological advancements to provide abundant resources in the wired network, but the time varying characteristics of mobile wireless channels will always exist. Thus, given the reasonable assumption that in the pool of users there are always different categories; QoS sensitive and insensitive users, “rich” and “poor” users, a prioritizing scheme sounds logical.

However, when we talk about prioritizing schemes, we have to emphasize that when we target data/voice integration, we can no longer accept relative QoS parameters. If the next generation global network is to support both data and voice, then when a user asks for a voice session declaring that he or she wants a telephone-quality session, the network must be able to provide the service, once the call is admitted. Otherwise, convergence of both data and voice networks won’t be practically and economically attractive. So, it is becoming more important to find a way to guarantee some absolute QoS parameters.

In most of the proposed solutions to support multi-media traffic via improvements on the PRMA and DPRMA protocols, fading effect was neglected, as is the case in [Dyson 97, Koutsakis 99, Koutsakis 00, Kubbar 00, Qi 99, Ren 97, Wu 95]. The channel was assumed to be error-free, the problem was defined as how to manage the slot-
allocation process in the TDMA frames, and the objective was to give higher priority to voice terminals over data terminals.

It is important now to emphasize the two factors that we are considering when we propose the extension of the multi-class network to TDMA-BMW networks. The first factor is that mobile users can only access the wireless network via contention methods, and that a great deal of efforts need to be exercised in order to achieve high utilization of the resources, which are in this case time slots. This first factor is an essential and non-trivial one, even when we assume an error-free channel. The second factor is that mobile wireless channels are time-varying channels due to environmental variations and due to mobility of mobile terminals leading to what is called fading. This results in variations in the capacity of these wireless channels, so we have to add this situation into consideration when we attempt to guarantee some QoS parameters. If there are no CoSs defined, all users will be equally-affected by deep fading, while with CoSs defined it may be possible to design the system such that resources will be squeezed in order to maintain the QoS parameters for the highest CoS.

In chapter 6 we developed a two-class PRMA protocol, assuming error free channel. Now, we attempt to include the effect of fading in our analysis. It is clear that fading affects the capacity of mobile wireless channels, but because of mobility and environmental differences between different geographical locations, fading may affect some mobile terminals and miss others. However, since in TDMA systems resources are partitioned in the time domain and not in the frequency domain, the effect of fading at any mobile terminal will be integrated in a general effect on the TDMA frame. But, with
the help of adaptive transmission rate technique [Farahvash 01], it will be possible to overcome this problem and control fading effect and limit it to the lowest CoSs, as we explain later. Note that, adaptive error correction codes techniques can also be deployed to respond to variations in the channel characteristics, i.e. variations in the received Signal-to-Noise Ratio (SNR). However, we don’t consider such techniques in this section. We argue that the complexity related to such techniques makes the adaptive transmission rate technique superior.

In the following subsections, we propose preliminary ideas about possible modifications to be done at the MAC layer of the TDMA LAN system and also at the networking layer in order to extend the Multi-Class Network to the TDMA LAN networks.

8.2.1.1 Proposed Modifications On The MAC Layer Of A TDMA Wireless LAN System To Emulate A Multi-Class Network

In this subsection, we will discuss general ideas and principles of our proposed idea. The basic principle for controlling the effect of fading on the system is to implement an adaptive transmission rate technique on a per-user basis.

Three CoSs are proposed, namely: Premium, Assured, and Best Effort. Slots reservation protocol will be developed for both Premium and Assured CoSs, while Best Effort terminals will access the network and send their packets via contention methods only. The difference between the Premium and Assured CoS is that under Premium CoS the system will attempt to guarantee the peak rate declared by Premium terminals, while
under Assured CoS the system will attempt to guarantee a minimum average rate required by Assured terminals.

In our model, it is up to the user to select the CoS regardless of the application that a user is running. For example, a user may choose Best Effort CoS to run a voice session just because he or she is interested in paying the minimum. On the other hand, a user may choose Premium CoS to run an email session just because he or she is interested in transferring an attachment with the shortest possible time and not be concerned about the cost.

In the proposed model, uplink time slots are classified into four categories:

1- *Reserved Slots*: these are the slots that have been reserved and declared so by the base station, only Premium and Assured CoSs can reserve slots.

2- *Best Effort Contention Slots*: these are the slots that are not reserved and declared by the base station as slots available for contention or these are reserved Assured slots but are temporarily declared as available for contention, only Best Effort terminals can contend on this category of time slots.

3- *Assured Contention Slots*: these are slots reserved by the Premium terminals but are declared by the base station as temporarily available for contention by Assured terminals, only.

4- *Access Slots*: these are slots specified by the design as access slots. The location of these slots is fixed. These are the slots that are partitioned into several mini-slots for contention by request packets.
Let us first discuss the proposed protocol for uplink traffic. Under Premium CoS, a terminal requests a peak rate by specifying the number of time slots required per frame in its access request packet. This number of time slots will be calculated based on the estimated peak data rate in packets per second and based on the measured received SNR at the base station. This is because the transmission rate (bits per symbol) in an adaptive transmission rate technique depends on the received SNR assuming perfect feedback channel as discussed in [Farahvash 01]. Once the base station receives the request packet, it consults its admission module, if there are enough resources the request will be accepted and the requesting terminal will be informed via the downlink channel about the location of time slots being reserved for its session. Base station will mark these time slots as reserved, so they will not be available for contention nor to be reserved by other terminals, however some of these time slots may be available for temporary access, as we explain later. During the life of this Premium session, whenever the base station detects changes in the received SNR from that terminal that exceed specific limit, it performs one of two actions. If the SNR goes down and the change is above a specific limit, base station will increase the number of reserved time slots based on the expected reduction in the transmission rate in order to maintain approximately the same peak data rate in packets per second promised for that Premium terminal. For example, if transmission rate is expected to be reduced to a half, then the number of time slots reserved for that Premium terminal will be doubled, and so on. Of course, it is assumed that the feedback channel is perfect such that the measured SNR of the Premium terminal at the base station will be perfectly sent to that Premium terminal, then the Premium terminal will
change its transmission rate based on the agreed upon algorithm. On the other hand, if SNR goes up, transmission rate will be increased and thus the number of time slots reserved for that specific Premium terminal will be decreased.

Under Assured CoS, a terminal requests an access by specifying two parameters, average data rate and minimum accepted data rate expressed in number of time slots per frame. As in Premium CoS, the number of time slots is determined based on estimates of the average and minimum accepted data rates in packets per second and in addition to the measured received SNR at the base station, which determines the current transmission rate.

During the life of an Assured session, it is expected that not all Premium terminals will be transmitting at their peak data rates. When a Premium terminal sees number of packets in the transmission buffer less than the number of slots reserved, after deducting the number of packets to be sent at the current frame, it sends a signal to the base station about the number of slots that will be unused during the next frame. Base station defines the un-used Premium slots as Assured Contention Slots, and informs mobile terminals during the downlink frame about their locations. Assured terminals can contend on these slots if they have extra packets to send above their average data rates. So, during good channel conditions, Assured session may enjoy data rates above the reserved average data rate.

As we have mentioned above, whenever a Premium terminal suffers from fading, it reduces its transmission rate, thus the base station increases the number of time slots reserved for this Premium terminal. The additional required time slots will be first taken
from the Best Effort Contention Slots, if many Premium terminals suffer from fading and all Best Effort Contention Slots have been used, the system will start to take from the reserved Assured slots such that the minimum data rate for Assured sessions is maintained. In the rare situation of having all possible Assured slots used, the system will start to disconnect some Assured sessions and give their time slots to needy Premium terminals. If all Assured sessions have been disconnected and Premium terminals are still in need for time slots, the system will start randomly punish different Premium sessions and try to maintain the connections alive.

It should be noted that the last part of the above scenario is very unlikely to happen if the system is designed well. The design of the system includes many factors, including pricing. If pricing is optimized such that it is always guaranteed that Premium traffic is a small fraction of the total traffic because it is the most expensive, while the rest of the traffic is divided between Assured and Best Effort CoSs. Then, it is fair to assume that the system may never even reach a situation where an Assured session must be disconnected given that the system is well designed with micro-diversity protection and reliable error correction capability.

It may happen that an Assured terminal suffers from fading, in this case the system will not react as long as the achieved data rate is at or above the minimum accepted data rate by this Assured terminal. If transmission rate has been reduced due to fading to a degree at which the delivered data rate is to be under the minimum accepted data rate by that Assured terminal, the system will add time slots to that Assured session to maintain the minimum accepted data rate. The additional time slots will be taken from
other Assured sessions delivering above their minimum data rates. If all Assured sessions deliver at their minimum data rates, the needy Assured terminal may start to take some Best Effort Contention Slots. If all the Best Effort Contention Slots are used by the needy Assured or Premium sessions and an Assured terminal is still in need for more time slots, that Assured session will be disconnected.

Best Effort terminals are allowed to contend on one time slot per frame always, however they may be allowed to contend on more than one time slot if they need, given that estimated measured collision ratio is under a specific threshold.

Note that, during the lightly loaded intervals with good channel conditions, a Best Effort user may run a voice session with an excellent quality since he or she will be able to usually get one time slot per frame, which is enough for a telephone-quality voice session.

The mechanism by which mobile terminals and the corresponding base station will communicate control information can be similar to what is proposed in the D-PRMA protocol [Dyson 97]; which is to define few bits in the MAC frame for control information that can be piggy backed.

Now, let us turn our attention to the downlink channel. The situation here is much simpler because the base station has all the information it needs about the demands on different CoSs and on different sessions. Since the base station is the decision maker in this scenario, it will be easy for it to dynamically allocate time slots to different sessions, following exactly the same protocols and policies that we have suggested for the uplink with a major difference that is there is, no contention on time slots since there is no need
for MAC protocols on the downlink. However, there is something very similar to contention which can be called competition, since Best Effort packets and expanding Assured sessions will compete on the available resources. But, the base station will have control over this competition and will try to fairly distribute time slots on competing sessions after determining the reserved time slots. As in the uplink case, the base station will send control information to mobile terminals to inform them about their allocated slots. Also, the base station will keep watching the received SNR by mobile terminals to adaptively change transmission rate, when necessary.

Details of scheduling procedure and control issues will be left for future works. Before we end this discussion, note that also Best Effort mobile terminals will adopt an adaptive rate transmission technique. Since the main objective of this technique is to maintain reliable transmission of information. So, when fading increases, SNR goes down and consequently probability of error goes up. By reducing the transmission rate and changing the modulation constellation, we can reduce the information transmission rate in bits/symbol to achieve reliable transmission of information.

In order to be able to evaluate the performance of the proposed system, we need to evaluate specific performance measures. The following measures are fundamentals and can be used for performance comparison.

1- *System Throughput* ($\eta_s$): this can be defined as the ratio of the total number of successfully transmitted time slots to the total number of time slots in all frames during the simulation time. We can define an uplink throughput and downlink throughput, since each one uses different access mechanisms.
\[ \eta_s = \frac{\sum_{i=1}^{M} ts(i)}{(N)(T_f)} \quad (8-1) \]

where \( ts(i) \) is the total number of time slots successfully transmitted by the \( i^{th} \) terminal, \( M \) is the total number of terminals accessed the network, \( N \) is the number of time slots per frame, and \( T_f \) is the total number of frames transmitted during simulation time.

2- **Average QoS Score and Revenue Capacity**: As discussed in chapter 6.

3- **System Capacity**: this performance measure defines the maximum number of sessions of a specific application type that can be supported by the network without violating specific requirements on specific QoS metrics.

### 8.2.1.2 Proposed Modifications To The Network Layer Of A TDMA LAN System To Emulate A Multi-Class Network

We are concerned about the handover issue and not about routing and other higher level networking issues. Since in the TDMA-BMW system the cell size will be very small, and thus a moving terminal may switch to several base stations during its session’s life. By implementing a multi-class system like the proposed one, priority can be given to high CoSs in the handover process.

Our philosophy is that it is irrelevant to a user whether his or her session was disconnected due to fading or due to handover. So, it makes sense to give the priority to
satisfy the demand of a handover Premium session over existing Best Effort sessions and over relaxed Assured sessions. The second priority must be given to Assured handover sessions. This will at least guarantee an approximately zero probability for a Premium session to be disconnected because of handover.

An issue similar to what we have discussed in chapter 4 for wired networks arises here. That it may be more accurate not to consider only pure revenue in optimizing the performance of the system. Because, there could be a class of users who can afford the Assured CoS but not the Premium, so blocking them may lead to loosing them completely, the same may also apply to Best Effort users. This leads to the importance of Dynamic Bandwidth Allocation solution, such that specific minimum parts of the resources must be guaranteed for each CoS and this minimum depends on the current demands.

In this multi-class TDMA LAN model, Premium handover sessions can be given priority even over these minimum-reserved resources for Assured and Best Effort CoSs. However, in such cases no new sessions should be accepted until some existing terminals end their sessions. This is again because disconnecting a session has more negative effects than blocking a new call, so we give handover sessions priority over new sessions of the same CoS and over existing sessions of lower CoSs, and this can be done because of the multi-class model we are proposing.
8.3 List of Publications and Reports

Following is a list of submitted or published papers related to the work in this thesis.

Journal papers:


Conference Papers:


Reports:

BIBLIOGRAPHY


# Vita of Samer Marwan Taha

**Current address:**
110 Huntington Dr., Apt# 106
Boalsburg, PA 16827
(814) 466-6015, (814) 360-8534
smt167@psu.edu

**Permanent address:**
Electrical Eng. Department
The Pennsylvania State Univ.
State College, PA 16802
taha_samer@hotmail.com

<table>
<thead>
<tr>
<th>Education</th>
<th>The Pennsylvania State University</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Electrical Engineering, Ph.D. degree, May 2002. GPA = 3.98/4.0</strong></td>
<td>State College, PA</td>
</tr>
<tr>
<td><strong>Jordan University Of Science and Technology (J.U.S.T)</strong></td>
<td>Irbid, Jordan</td>
</tr>
<tr>
<td>Sep. 1993 – May 1998</td>
<td></td>
</tr>
<tr>
<td><strong>B.Sc., Electrical Engineering, Major: Communications and Electronics</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GPA: 3.8/4.0, ranked first on the department.</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Skills</th>
<th>Knowledge gained through research and independent study: ATM Networks, E-LAN, MPOA, MPLS, DiffServ, Neural Networks, 3rd generation TDMA WLANs.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programming Languages:</strong></td>
<td>C++, MatLab, Visual Basic, MC68k Assembly, Fortran-77, Cantata, Bones Designer (sophisticated simulation tool).</td>
</tr>
<tr>
<td><strong>Computer Systems &amp; Tools:</strong></td>
<td>MS DOS, Windows 95/98, Unix OS, MS Office, MS PowerPoint, MathCad.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Experience</th>
<th>Practical Experience:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instructor:</strong></td>
<td>The Pennsylvania State University, Summer 2001. Taught a senior-level course (EE/CSE 458, Data Communications).</td>
</tr>
<tr>
<td><strong>Teaching Assistantship:</strong></td>
<td>The Pennsylvania State University, Spring 2001. Assisted in teaching EE350 (Continuous-Time Linear Systems) and gave recitation sessions.</td>
</tr>
<tr>
<td><strong>Teaching Assistantship:</strong></td>
<td>The Pennsylvania State University, Fall 2000. Assisted in teaching EE/CSE 458 (Data Communications) for senior students of both EE and CSE departments.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Selected Publications</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Awards and Activities</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Five awards of the Outstanding Student in each year of undergraduate study.</td>
<td></td>
</tr>
<tr>
<td>Founded the Club of Developing Scientific Research in Jordan University of Science and Technology and occupied the Chairman position, 1994, Jordan.</td>
<td></td>
</tr>
<tr>
<td>Founded the IEEE J.U.S.T Branch and occupied the Vice Chairman position, 1996, Jordan.</td>
<td></td>
</tr>
<tr>
<td>Member; IEEE/Communications Society, 4 years.</td>
<td></td>
</tr>
<tr>
<td>Member; CICTR, <a href="http://cictr.ee.psu.edu">http://cictr.ee.psu.edu</a>, 3 yaers.</td>
<td></td>
</tr>
</tbody>
</table>